



The Chimera User's Guide has three main parts, which are interconnected:

- [Tutorials](#) - exercises ranging from beginner-level to more advanced
- [Basic Functions](#) - general usage topics, including [commands](#)
- [Tools](#) - descriptions of the Chimera Tools menu entries

The [Chimera Quick Reference Guide](#) (PDF) summarizes command-line usage.

[Chimera documentation](#), including the User's Guide, is bundled with each download. Your local copy of the documentation can be accessed and searched from the Chimera Help menu. We recommend using the bundled documentation because it is synchronized with your installed version of Chimera. Latest production release and development versions of the User's Guide are available from the [Chimera home page documentation index](#). Please see the [Chimera home page](#) for other types of information.

This locally installed Chimera documentation can be searched using "Search Documentation" in the Chimera Help menu.



# Chimera User's Guide

Tutorials

Basic Functions

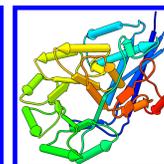
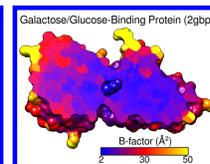
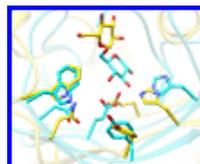
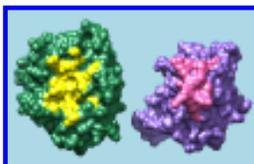
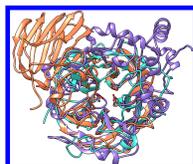
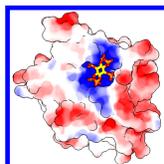
Tools

## Tutorials

[\[Full-Page Index\]](#)
[Getting Started: Menu](#)
[Part 1](#)
[Part 2](#)
[Getting Started: Cmd](#)
[Part 1](#)
[Part 2](#)
[Image Tutorials](#)
[Surface Properties](#)
[Hydrolases](#)
[Opened Interface](#)
[Similar Sites](#)
[B-Factor Coloring](#)
[Density Display](#)
[Pipes and Planks](#)
[Analysis/Comparison](#)
[Setup](#)
[Distances...](#)
[Angles...](#)
[Surfaces...](#)
[Morphing](#)
[Attributes](#)
[Part 1](#)
[Part 2](#)
[Sequences/Structures](#)
[Alignments](#)
[Setup](#)
[Different Proteins](#)
[Same Protein](#)
[Comparative Modeling](#)

## Tutorials Index

1. [Getting Started - Menu Version](#)
  - [Part 1](#) - Manipulation, Selection, and Chains
  - [Part 2](#) - Molecular Representations and Surfaces
2. [Getting Started - Command Version](#)
  - [Part 1](#) - Manipulation, Selection, and Chains
  - [Part 2](#) - Molecular Representations and Surfaces
3. Image Tutorials:



4. [Structure Analysis and Comparison](#)
  - [Background and Setup](#)
  - [Distances, H-bonds, Contacts](#)
  - [Angles, Rotamers, Clashes](#)
  - [Surfaces and Attributes](#)
  - [Superposition and Morphing](#)
5. [Attributes](#)
  - [Part 1](#) - Leucine Zipper
  - [Part 2](#) - GTP-Binding Protein
6. [Sequences and Structures](#)
7. [Superpositions and Alignments](#)
  - [Background and Setup](#)
  - [Different Proteins](#)
  - [Same Protein](#)
8. [Comparative Modeling](#)
9. [The Model Panel and Ensembles](#)
10. [Trajectory and Ensemble Analysis](#)
  - [Part 1](#) - Collagen Peptide
  - [Part 2](#) - Met-Enkephalin
11. [ViewDock](#)

[Model Panel/Ensembles](#)  
[Trajectories/Ensembles](#)

[Part 1](#)

[Part 2](#)

[ViewDock](#)

[More... \(Web\)](#)

## Help Sheets

[Chimera Quick Ref \(PDF\)](#)

[Intro to PDB Format](#)

[More tutorials](#) are available at the Chimera web site.

## Help Sheets

1. [Chimera Quick Reference Guide \(PDF\)](#) - includes a list of commands and several examples of command-line atom specification
2. [Introduction to PDB Format](#) - describes types and formats of data commonly found in PDB files



CHIMERA

# Chimera User's Guide

Tutorials

Basic Functions

Tools

## Basic Functions

[\[Full-Page Index\]](#)

[Chimera Startup](#)

[Input File Types](#)

[Fetch by ID](#)

[Chimera Window](#)

[Menus](#)

[Command Line](#)

[3D Manipulation](#)

[Side View](#)

[Selection](#)

[Picking](#)

[↑ ↓ → ←](#)

[Actions Menu](#)

[Commands](#)

[Atom Spec](#)

[Tools](#)

[Model Panel](#)

[PBond Panel](#)

[Molecule Display](#)

[Atoms/Bonds](#)

[Ribbons](#)

[Surfaces](#)

[Volume Display](#)

[Attribute Inspectors](#)

[Coloring](#)

[Clipping](#)

[Saving Images](#)

[Tips](#)

[Raytracing](#)

[Making Movies](#)

[Sequences](#)

[Superposition](#)

[Building Structures](#)

[Saving Data](#)

[Chimera Sessions](#)

[Preferences](#)

[Help](#)

## Basic Functions Index

1. [Chimera Startup](#)
2. [Input File Types](#)
  - [Fetch by ID](#)
3. [Chimera Window](#)
  - [Menus](#)
  - [Command Line](#)
4. [3D Manipulation](#)
  - [Side View](#)
5. [Selection](#)
  - [Picking](#)
  - [Broadening, Narrowing, Inverting](#)
6. [Actions Menu](#)
7. [Commands](#)
  - [Atom Specification](#)
8. [Tools](#)
  - [Model Panel](#)
  - [PseudoBond Panel](#)
9. [Molecule Display](#)
  - [Atoms/Bonds](#)
  - [Ribbons](#)
  - [Surfaces](#)
10. [Volume \(Density\) Display](#)
11. [Attribute Inspectors](#)
12. [Coloring](#)
13. [Clipping](#)
14. [Saving Images](#)
  - [Tips on Preparing Images](#)
  - [Raytracing](#)
  - [Making Movies](#)
15. [Sequences](#)
16. [Superimposing Structures](#)
17. [Building Structures](#)
18. [Modifying and Saving Data](#)
19. [Chimera Sessions](#)
20. [Preferences](#)
21. [Help](#)
22. [Stereo](#)

[Stereo](#)  
[Keyboard Shortcuts](#)

23. [Keyboard Shortcuts](#)



# Chimera User's Guide

CHIMERA

Tutorials

Basic Functions

Tools

## Tools

[\[List Alphabetically\]](#)

[\[Full-Page Index\]](#)

### General Controls

[Command Line](#)

[Model Panel](#)

[PseudoBond Panel](#)

[Keyboard Shortcuts](#)

[Task Panel](#)

[IDLE](#)

### Viewing Controls

[Side View](#)

[Camera](#)

[Effects](#)

[Lighting](#)

[Shininess](#)

### Depiction

[Color Secondary Structure](#)

[Rainbow](#)

[PipesAndPlanks](#)

[Nucleotides](#)

[Ribbon Style Editor](#)

[Render by Attribute](#)

[Surface Capping](#)

[Per-Model Clipping](#)

[Color Zone](#)

[PseudoBond Reader](#)

### Structure Analysis

[FindHBond](#)

[Find Clashes/Contacts](#)

[Distances](#)

[Angles/Torsions](#)

[Metal Geometry](#)

[Axes/Planes/Centroids](#)

[Thermal Ellipsoids](#)

[Render by Attribute](#)

## Tools Index

1. [2D Labels](#) - create labels with text, [symbols](#), and arrows in 2D
2. [Add Charge](#) - assign partial charges to atoms
3. [AddH](#) - add hydrogens
4. [Add Ions](#) - add monatomic counterions using [AmberTools](#)
5. [Adjust Torsions](#) - rotate bonds (change dihedral angles)
6. [Align Chain Sequences](#) - generate a multiple sequence alignment (MSA) of structure chains using a [Clustal Omega](#) or [MUSCLE](#) web service hosted by the [UCSF RBVI](#)
7. [Angles/Torsions](#) - measure bond angles and dihedral angles
8. [Animation](#) (under development) - save/restore Chimera scenes, arrange them into a timeline, play/record animation
9. [APBS](#) - interface to electrostatics calculations with [APBS](#) (Adaptive Poisson-Boltzmann Solver)
10. [Area/Volume from Web](#) - use the [StrucTools server](#) to calculate surface areas and Voronoi volumes
11. [Attribute Calculator](#) - generate new numerical [attributes](#) from existing ones; calculate totals or averages of a given numerical attribute
12. [AutoDock Vina](#) - interface to single-ligand docking with [AutoDock Vina](#)
13. [Axes/Planes/Centroids](#) - define geometric objects (axes, planes, centroids) based on sets of atoms, perform related measurements
14. [Benchmark](#) - measure hardware performance on standard Chimera rendering tasks
15. [Blast Protein](#) - perform protein BLAST searches using a web service hosted by the [UCSF RBVI](#)
16. [Browser Configuration](#) - configure web browsers to send Chimera [data linked to web pages](#)
17. [Build Structure](#) - create and modify atomic structures
18. [Cage Builder](#) - create polyhedral cages
19. [Camera](#) - control viewing parameters
20. [Change Chain IDs](#) - reassign chain identifiers
21. [Color Editor](#) - create colors interactively
22. [Color Key](#) - create a color key for figures
23. [Color Secondary Structure](#) - color peptides/proteins by secondary structure
24. [Color Zone](#) - color surfaces to match [selected](#) atoms, split volume data by the resulting color zones

[Define Attribute](#)  
[Attribute Calculator](#)  
[ResProp](#)

### Structure Comparison

[MatchMaker](#)  
[Match -> Align](#)  
[Morph Conformations](#)  
[RR Distance Maps](#)  
[Ensemble Cluster](#)  
[Ensemble Match](#)  
[Tile Structures](#)  
[Minrms Plot](#)

### Sequence

[Sequence](#)  
[PDB/UniProt Info](#)  
[Multalign Viewer](#)  
[Match -> Align](#)  
[Blast Protein](#)  
[Align Chain Sequences](#)

### Surface/Binding Analysis

[FindHBond](#)  
[Find Clashes/Contacts](#)  
[Electrostatic Surface Coloring](#)  
[Coulombic Surface Coloring](#)  
[APBS](#)  
[Surface Capping](#)  
[Surface Zone](#)  
[ViewDock](#)  
[Dock Prep](#)  
[AutoDock Vina](#)  
[Measure Volume and Area](#)  
[Area/Volume from Web](#)  
[Measure and Color Blobs](#)  
[Intersurf](#)  
[Surfnet](#)  
[DelPhiController](#)

### Structure Editing

[AddH](#)  
[Add Charge](#)  
[Dock Prep](#)  
[PDB2PQR](#)  
[Rotamers](#)  
[Adjust Torsions](#)  
[Build Structure](#)  
[Model/Refine Loops](#)  
[Movement Mouse Mode](#)  
[Minimize Structure](#)

25. [Command Line](#) - enter Chimera [commands](#)
26. [Constrained Move](#) - perform rotations and translations about specified axes
27. [Coulombic Surface Coloring](#) - color [molecular surfaces](#) by Coulombic electrostatic potential
28. [Crystal Contacts](#) - identify clashes between PDB symmetry copies
29. [Define Attribute](#) - assign [attribute](#) values to atoms, residues, or models
30. [DelPhiController](#) - interface to [DelPhi](#) (obtained separately) for calculating electrostatic potential
31. [Demos](#) and [Demo Editor](#) - create and replay demonstrations in Chimera
32. [Distances](#) - measure distances between pairs of atoms
33. [Dock Prep](#) - add hydrogens, charges, *etc.* to prepare structures for [DOCK](#) or for other calculations
34. [Effects](#) - control visual effects such as depth cueing, shadows, and silhouettes
35. [Electrostatic Surface Coloring](#) - color surfaces using an [electrostatic potential map](#) (from any of several [separate programs](#))
36. [Ensemble Cluster](#) - cluster members of a conformational ensemble
37. [Ensemble Match](#) - match conformations from two ensembles
38. [Find Clashes/Contacts](#) - identify clashes and/or contacts
39. [FindHBond](#) - find possible hydrogen bonds
40. [Fit in Map](#) - fit atoms into a map (volume data) or one map into another
41. [Fit to Segments](#) - fit structures into map segmentation regions
42. [Flatten Icosahedron](#) - rearrange the faces of an icosahedral virus capsid into a plane to create a paper model
43. [Hide Dust](#) - hide small disconnected bits of a surface
44. [Icosahedron Surface](#) - create a hybrid icosahedron/sphere surface
45. [IDLE](#) (Python Interactive DeveLopment Environment) - enter Python commands
46. [Intersurf](#) - generate and display interface surfaces
47. [Keyboard Shortcuts](#) - define and use keyboard shortcuts for Chimera functions
48. [Lighting](#) - adjust and save lighting parameters
49. [Match -> Align](#) - generate a multiple sequence alignment (MSA) from a structural alignment
50. [MatchMaker](#) - superimpose structures by first constructing a sequence alignment (optionally including secondary structure scoring) and then fitting the aligned residue pairs
51. [MD Movie](#) - replay and analyze molecular dynamics trajectories
52. [Measure and Color Blobs](#) - color and measure disconnected parts of a surface
53. [Measure Volume and Area](#) - measure surface area and surface-enclosed volume
54. [Metal Geometry](#) - analyze metal coordination geometry
55. [Minimize Structure](#) - energy-minimize structures
56. [Minrms Plot](#) - examine structural alignments of proteins from [MinRMS](#) (obtained separately, [available](#) as source code)

[Add Ions](#)  
[Solvate](#)  
[Write DMS](#)  
[Change Chain IDs](#)  
[Renumber Residues](#)

**Amber**

[Add Ions](#)  
[Solvate](#)  
[Write Prmtop](#)

**MD/Ensemble Analysis**

[MD Movie](#)  
[Ensemble Cluster](#)  
[Ensemble Match](#)  
[Tile Structures](#)

**Higher-Order Structure**

[Multiscale Models](#)  
[Unit Cell](#)  
[Crystal Contacts](#)  
[Small-Angle X-Ray Profile](#)  
[Scale Bar](#)  
[Icosahedron Surface](#)  
[Flatten Icosahedron](#)  
[Cage Builder](#)

**Volume Data**

[Volume Viewer](#)  
[Fit in Map](#)  
[Surface Color](#)  
[Color Zone](#)  
[Morph Map](#)  
[Volume Tracer](#)  
[Volume Filter](#)  
[Hide Dust](#)  
[Segment Map](#)  
[Fit to Segments](#)  
[MultiFit](#)  
[Volume Eraser](#)  
[Measure Volume and Area](#)  
[Measure and Color Blobs](#)  
[Volume Mean, SD, RMS](#)  
[Values at Atom Positions](#)  
[Volume Series](#)  
[Volume Menu on Menubar](#)

**Demos**

[Demo Editor](#)

57. [Model/Refine Loops](#) - interface to [Modeller](#) for building missing peptide segments or generating alternative conformations of peptide segments already present in a structure
58. [Model Panel](#) - list and act on models
59. [Morph Conformations](#) - create a trajectory that morphs between structures
60. [Morph Map](#) - morph between two related sets of volume data
61. [Movement Mouse Mode](#) - move just part of a structure with the mouse
62. [Movie Recorder](#) - capture image frames and assemble them into a movie file
63. [Multalign Viewer](#) - view sequences, optionally with associated structures; analyze conservation; view [UniProt](#) feature annotations; interface to [Modeller](#)
64. [MultiFit](#) - fit multiple structures into density using a web service hosted by the [UCSF RBVI](#)
65. [Multiscale Models](#) - view macromolecular assemblies at high and low resolution, generate multimers, and navigate structural hierarchies
66. [Notepad](#) - add user notes (text) that can be saved along with [sessions](#)
67. [Nucleotides](#) - create special representations of nucleotide bases and sugars
68. [Palette Editor](#) - create and choose palettes
69. [PDB2PQR](#) - interface to structure cleanup and charge/radius assignment with [PDB2PQR](#)
70. [PDB/UniProt Info](#) - retrieve PDB and [UniProt](#) annotations using a web service provided by the [RCSB PDB](#)
71. [Per-Model Clipping](#) - clip models individually with a plane at any angle
72. [PipesAndPlanks](#) - show protein helices as "pipes," strands as "planks"
73. [PseudoBond Panel](#) - list and act on [pseudobond](#) groups
74. [PseudoBond Reader](#) - create [pseudobonds](#) arbitrarily
75. [Rainbow](#) - use a range of colors, changing the color per residue, chain, or model
76. [ReadStdin](#) - allow communication with Chimera through standard input/output
77. [Render by Attribute](#) - show [attribute](#) values of atoms, residues, and models (with color, *etc.*)
78. [Renumber Residues](#) - reassign residue numbers
79. [Reply Log](#) - show informational, warning, and error messages from Chimera
80. [ResProp](#) - define amino acid categories
81. [RESTServer](#) - allow communication with Chimera through REST interface
82. [Ribbon Style Editor](#) - change ribbon heights/widths (scalings) and cross-sections (styles)
83. [Rotamers](#) - view and evaluate amino acid sidechain rotamers, incorporate them into structures
84. [Rotation](#) - control the center of rotation
85. [RR Distance Maps](#) - generate a protein contact map color-coded by distance (and/or standard deviation, if comparing multiple structures)

**Movement**

[Rotation](#)  
[Undo Move](#)  
[Redo Move](#)  
[Movement Mouse Mode](#)  
[Constrained Move](#)  
[Transform Coordinates](#)

**Utilities**

[Reply Log](#)  
[2D Labels](#)  
[Notepad](#)  
[Movie Recorder](#)  
[Color Key](#)  
[Animation](#)  
[Structure Diagram](#)  
[Browser Configuration](#)  
[Benchmark](#)  
[Color Editor](#)  
[Palette Editor](#)  
[ReadStdin](#)  
[REServer](#)

86. [Scale Bar](#) - draw a scale bar and associated label
87. [Segment Map](#) - partition density maps
88. [Sequence](#) - show amino acid and/or nucleic acid sequence
89. [Shininess](#) - adjust shininess and brightness
90. [Side View](#) - scale the view and move clipping planes interactively
91. [Small-Angle X-Ray Profile](#) - calculate a SAXS profile from a structure, fit with experiment
92. [Solvate](#) - solvate structures using [AmberTools](#)
93. [Structure Diagram](#) - generate 2D chemical diagrams of small molecules
94. [Surface Capping](#) - cap surfaces where they intersect a clipping plane
95. [Surface Color](#) - color surfaces by volume data values or by distance from a point, axis, or plane
96. [Surface Zone](#) - restrict the display of [certain types of surfaces](#) to a zone around [selected](#) atoms
97. [Surfnet](#) - examine cavities and surface indentations
98. [Task Panel](#) - manage jobs started by Chimera
99. [Thermal Ellipsoids](#) - show anisotropic B-factors
100. [Tile Structures](#) - arrange models in a plane
101. [Transform Coordinates](#) - transform a model by specified Euler angles and shifts
102. [Undo Move and Redo Move](#) - go back and forth in the history of model rotational/translational positions
103. [Unit Cell](#) - generate crystallographic unit cell contents from coordinates and transformation matrices
104. [Values at Atom Positions](#) - map volume data to atom positions and assign values as an [attribute](#)
105. [ViewDock](#) - view and prioritize docked molecules
106. [Volume Eraser](#) - interactively zero out parts of volume data
107. [Volume Filter](#) - smooth or transform volume data
108. [Volume Mean, SD, RMS](#) - calculate statistics for [volume data](#)
109. [Volume Series](#) - display a series of [volume data](#) sets
110. [Volume Tracer](#) - place markers and trace paths within volume displays
111. [Volume Viewer](#) - visualize [volume data](#) (3D numerical data such as electron density)
112. [Write DMS](#) - save [molecular surfaces](#) as [DMS files](#)
113. [Write Prmtop](#) - write [Amber](#) parameter/topology files using [AmberTools](#)


[Basic Functions Index](#)
**Commands**
[\[Full-Page Index\]](#)
[Atom Specification](#)
[Command Files](#)
[Quick Ref \(PDF\)](#)
[2dlabels ac addaa](#)
[addcharge addh adjust](#)
[alias align angle aniso apbs](#)
[aromatic background bond](#)
[bondcolor bonddisplay](#)
[bondrepr bondzone cd](#)
[center chain changechains](#)
[chirality clip close cofr](#)
[color colordef colorkey](#)
[combine conic coordset](#)
[copy coulombic](#)
[crystalcontacts defattr](#)
[define delete display](#)
[distance echo export](#)
[fillring findclash findhbond](#)
[fitmap fly focus freeze](#)
[getcrd hbonds help hkcgae](#)
[intersurf invert ksdssp](#)
[label labelopt leap lighting](#)
[linewidth list listen](#)
[longbond mask match](#)
[matchmaker matrixcopy](#)
[matrixget matrixset mclip](#)
[mcopy mda measure](#)
[meshmol minimize](#)
[mmaker modelcolor](#)
[modeldisplay molmap](#)
[morph move movie msc](#)
[msms namesel neon](#)
[nucleotides objdisplay](#)

## Commands Index

Commands are entered into Chimera's [Command Line](#). Past commands can be accessed from the [Command History](#), and commands can be placed in an executable [command file](#). Chimera commands are listed below and in the [Quick Reference \(PDF\)](#).

All commands can be truncated to unique strings, but their keywords cannot be truncated except as noted in the documentation. Entering the contents of the [Command Line](#) (pressing return) executes the contents and updates the display. To hide intermediate stages of processing, multiple commands can be combined into one line with semicolon separators. Certain [text conventions](#) are used to describe usage.

[2dlabels](#) - create labels with text, symbols, and arrows in 2D

[ac](#) - use accelerators ([keyboard shortcuts](#))

[addaa](#) - add an amino acid to a peptide N- or C-terminus

[addcharge](#) - assign Amber partial charges and atom types

[addh](#) - add hydrogens

[adjust](#) - change bond angle or bond length

[alias](#) - create an alias or list the existing aliases

[align](#) - align two atoms or sets of atoms along the line of sight

[angle](#) - measure angles formed by atoms or by axes and planes

[aniso](#) - show thermal ellipsoids

[apbs](#) - interface to electrostatics calculations with [APBS](#) (Adaptive Poisson-Boltzmann Solver)

[aromatic](#) - show ring aromaticity

[background](#) - set background color, gradient, or image

[bond](#) - add/delete bonds

[bondcolor](#) - color bonds independently from atoms

[bonddisplay](#) - control how bond display depends on atom display

[bondrepr](#) - control bond style (wire or stick)

[bondzone](#) - make zoning tools use points along bonds

[cd](#) - change the working directory

[center](#) - center the view on specified atoms

[chain](#) - chain specified atoms, undisplay the others

[changechains](#) - reassign chain identifiers

[chirality](#) - report the R/S configuration of a chiral center

[clip](#) - move global clipping planes

[open](#) [pause](#) [pdb2pqr](#)  
[pdbrun](#) [perframe](#) [play](#)  
[preset](#) [rainbow](#)  
[ramachandran](#) [rangecolor](#)  
[read](#) [represent](#) [reset](#)  
[resrenumber](#) [ribbackbone](#)  
[ribbon](#) [ribclass](#) [ribcolor](#)  
[ribinsidecolor](#) [ribrepr](#)  
[ribscale](#) [ribspline](#) [rlabel](#)  
[rmsd](#) [rna](#) [rock](#) [roll](#) [rotation](#)  
[runscript](#) [save](#) [savepos](#)  
[scale](#) [scene](#) [scolor](#) [section](#)  
[segment](#) [select](#) [sequence](#)  
[set/unset](#) [setattr](#) [shape](#)  
[show](#) [sleep](#) [solvate](#) [sop](#)  
[split](#) [start](#) [stereo](#) [stop](#)  
[struts](#) [surface](#) [surfcat](#)  
[surfcolor](#) [surfrepr](#)  
[surftransparency](#) [swapaa](#)  
[swapna](#) [sym](#) [system](#) [tcolor](#)  
[texture](#) [thickness](#) [tile](#)  
[topography](#) [transparency](#)  
[turn](#) [vdw](#) [vdwdefine](#)  
[vdwdensity](#) [version](#)  
[viewdock](#) [vina](#) [volume](#) [vop](#)  
[vseries](#) [wait](#) [window](#)  
[windoworigin](#) [windowsize](#)  
[write](#) [writesel](#) [zonesel](#)

[Unsupported Midas  
 Commands](#)

[close](#) - close a model  
[cofr](#) - report or change the center of rotation  
[color](#) - color atoms/bonds, ribbons, labels, and surfaces  
[colordef](#) - define a new color  
[colorkey](#) - create a color key  
[combine](#) - combine molecule models into a single model or create another copy of a molecule model  
[conic](#) - create a shadowed space-filling image  
[coordset](#) - play through frames of a trajectory  
[copy](#) - save image files  
[coulombic](#) - color surfaces by Coulombic electrostatic potential  
[crystalcontacts](#) - identify clashes between PDB symmetry copies  
[defattr](#) - assign attribute values to atoms, residues, or models  
[define](#) - calculate axes, planes for sets of atoms  
[delete](#) - delete atoms and bonds  
[display](#) - display and undisplay atoms  
[distance](#) - measure distances between atoms, axes, planes, and/or centroids  
[echo](#) - send text to the status line and Reply Log  
[export](#) - save the graphical scene  
[fillring](#) - show rings as filled  
[findclash](#) - identify clashes and contacts  
[findhbond](#) - identify hydrogen bonds  
[fitmap](#) - fit atoms or map into map  
[fly](#) - smoothly traverse a series of saved positions  
[focus](#) - adjust the view and center of rotation  
[freeze](#) - stop all motion  
[getcrd](#) - report coordinates  
[hbonds](#) - identify hydrogen bonds  
[help](#) - display the manual page for a command  
[hk cage](#) - create a hexagon/pentagon mesh that covers an icosahedron  
[intersurf](#) - generate and display interface surfaces  
[invert](#) - swap substituents of an atom  
[ksdssp](#) - determine secondary structure from protein coordinates  
[label](#) - display atom labels  
[labelopt](#) - control the information in atom labels  
[leap](#) - use a [Leap Motion Controller](#) with Chimera  
[lighting](#) - adjust lighting and shininess  
[linewidth](#) - control the width of wire bonds  
[list](#) - report attribute information  
[listen](#) - report Chimera status  
[longbond](#) - show/hide pseudobonds representing missing segments  
[mask](#) - extract [volume data](#) bounded by surfaces  
[match](#) - perform least-squares fitting of specified atoms  
[matchmaker](#) - align models in sequence, then in 3D  
[matrixcopy](#) - apply the transformation matrix of one model to another  
[matrixget](#) - write the current transformation matrices to a file  
[matrixset](#) - read and apply transformation matrices from a file  
[mclip](#) - control [per-model clipping](#)

[mcopy](#) - copy settings from one molecule model to another

[mda](#) - MultiDomain Assembler: perform several steps toward homology-modeling a multidomain protein

[measure](#) - perform various calculations on structures, surfaces, maps

[meshmol](#) - create a "molecule" to show surface mesh as sticks

[minimize](#) - energy-minimize structures

[mmaker](#) - align models in sequence, then in 3D

[modelcolor](#) - set color at the model level

[modeldisplay](#) - set display at the model level

[molmap](#) - create a density map from atomic coordinates

[morph](#) - morph (interpolate) between different structures

[move](#) - translate models

[movie](#) - capture image frames and assemble them into a movie file

[msc](#) - color [Multiscale Models](#) surfaces to match atoms

[msms](#) - (see [surfcatsurfrepr](#))

[namesel](#) - save and name the current [selection](#)

[neon](#) - create a shadowed stick/tube image (**not available on Windows systems**)

[nucleotides](#) - create special nucleotide representations

[objdisplay](#) - display and undisplay VRML models

[open](#) - read local files or fetch by ID

[pause](#) - pause script execution until the user presses a key

[pdb2pqr](#) - interface to structure cleanup and charge/radius assignment with [PDB2PQR](#)

[pdbrun](#) - send an annotated PDB file to the system shell

[perframe](#) - specify commands to be executed at each display frame

[play](#) - script various complex motions

[preset](#) - apply a predefined combination of display settings

[rainbow](#) - color residues, chains, or models over a range

[ramachandran](#) - show peptide  $\phi, \psi$  distribution ([Ramachandran plot](#))

[rangecolor](#) - color over a range according to attribute values

[read](#) - execute a command file, updating the display at the end

[represent](#) - control atom/bond display style (wire, stick, ball-and-stick, or sphere)

[reset](#) - restore default or saved orientations

[resrenumber](#) - reassign residue numbers

[ribbackbone](#) - allow display of both ribbon and backbone atoms

[ribbon](#) - display ribbon

[ribclass](#) - set ribbon residue class: which atoms control ribbon path and which are hidden by ribbon

[ribcolor](#) - color ribbons

[ribinsidecolor](#) - set a separate color for the insides of protein helix ribbons

[ribrepr](#) - control ribbon style (flat, edged, or rounded)

[ribscale](#) - control ribbon secondary-structure-specific dimensions (Chimera default or licorice)

[ribspline](#) - control ribbon path method (B-spline or cardinal spline)

[rlabel](#) - display residue labels

[rmsd](#) - evaluate the RMSD between specified sets of atoms

[rna](#) - build rough but potentially large-scale models of RNA and DNA

[rock](#) - rotate models back and forth (oscillate)

[roll](#) - rotate models

[rotation](#) - rotate bonds

[runscript](#) - run Python script with command-line arguments

[save](#) - save the current Chimera session

[savepos](#) - save model positions

[scale](#) - scale the view

[scene](#) - save and restore scenes (positions plus styles, colors, labels, *etc.*)

[scolor](#) - color surfaces a single color or by [volume data](#) or geometry

[section](#) - move global clipping planes in parallel

[segment](#) - act on [segmentation](#) models

[select](#) - [select](#) atoms or [activate](#) models for motion

[sequence](#) - show [Sequence](#) for specified chains

[set/unset](#) - set visual effects, make models rotate about individual centers

[setattr](#) - set an attribute to a specified value

[shape](#) - create a surface of a specified geometric shape

[show](#) - display specified atoms, undisplay the others

[sleep](#) - pause script execution for a specified length of time

[solvate](#) - add solvent using [AmberTools](#)

[sop](#) - adjust capping, edit surface models

[split](#) - partition a molecule model into separate submodels

[start](#) - start Chimera tools by name

[stereo](#) - switch amongst stereo options and mono viewing

[stop](#) - exit from Chimera

[struts](#) - add pseudobonds to a molecule to strengthen it for 3D printing

[surface](#) - calculate and display molecular surfaces

[surfcat/surfrepr](#) - create molecular surface categories and control surface style

[surfcOLOR](#) - set molecular surface color source

[surftransparency](#) - adjust surface transparency

[swapaa](#) - mutate amino acids or swap rotamers

[swapna](#) - mutate nucleic acid residues

[sym](#) - generate symmetry-related copies of a structure

[system](#) - send a command to the system shell

[tcolor](#) - color atoms/bonds, labels, and surfaces with a texture color

[texture](#) - define texture maps and associated colors

[thickness](#) - move global clipping planes in opposite directions

[tile](#) - arrange models in a plane

[topography](#) - plot values in a volume data plane as heights in a surface

[transparency](#) - make atoms/bonds, ribbons, and surfaces transparent

[turn](#) - rotate models

[vdw](#) - display van der Waals (VDW) dot surface

[vdwdefine](#) - set VDW radii

[vdwdensity](#) - set VDW surface dot density

[version](#) - show copyright information and Chimera version

[viewdock](#) - start [ViewDock](#) and load docking results

[vina](#) - interface to single-ligand docking with [AutoDock Vina](#)

[volume](#) - display [volume data](#) (3D numerical data such as electron density)

[vop](#) - edit [volume data](#)

[vseries](#) - display an ordered sequence of [volume data](#) sets; can also be used to process and save the data

[wait](#) - suspend command processing a specified number of frames or until motion has stopped

[window](#) - adjust the view to contain the displayed and specified atoms

[windoworigin](#) - set graphics window location

[windowsize](#) - adjust the dimensions of the graphics window

[write](#) - save atomic coordinates (pdb, mol2)

[writesel](#) - write a list of the currently [selected](#) (or unselected) items

[zonesel](#) - [select](#) atoms and/or surfaces within a cutoff distance of specified atoms and/or surfaces

## Chimera Command Files

Chimera command files (scripts) are simply text files containing the same commands that could be entered at the [Command Line](#). Only **plain text** is accepted, not rich text format or Microsoft Word format. Example files:

- [rescol.com](#) for coloring amino acid residues
- [setup.com](#) from the [ViewDock tutorial](#)
- [convergent.com](#) from the [Similar Binding Sites](#) image tutorial
- [movie content examples](#)

Chimera command files can be created manually or by saving the [Command History](#). Rules and tips:

- Lines that begin with a pound sign (#) are interpreted as comments.
- Multiple commands separated by semicolons (;) can be combined into a single line. The last semicolon in a line can precede either a command or a pound sign and in-line comment.
- During execution, a single-frame display update (implicit [wait 1](#)) will be added at the end of each line that:
  - contains one or more commands that could change the display - and -
  - does not already end with an explicit [wait](#) of any length (Exception: display updates are not added when a command file is executed with [read](#).) Multiple commands can be combined into a single line as described above to suppress the display of intermediate states. This allows multiple graphical changes to appear simultaneously instead of one by one.
- Several of the [movie-related commands](#) execute over multiple frames to generate gradual changes such as continuous motion. A multiple-frame command should be followed by a multiple-frame [wait](#) (not the implicit single-frame wait mentioned above) to prevent subsequent commands from executing until it is finished.
- The locations of files opened by a command file can be specified relative to the command file's location, or with absolute pathnames. For example, if the files are in the same directory as the command file, only their names need to be given.

- Structure files can be opened with [open noprefs](#) to circumvent any [New Molecules preferences](#); this prevents inconsistent behavior of command files and [demos](#) potentially caused by the different preferences settings of different users.

Ways to execute a command file:

- by simply [opening](#) the file, which updates the display after each line as needed. The [file type](#) is specified with a suffix (.cmd or .com, part of the filename) or prefix (cmd: or com:, not part of the filename).
- with the command [read](#)
- automatically at [Command Line](#) startup, as specified in the [Command Line preferences](#)

Command script execution can be aborted by pressing the **Esc** (escape) key, or paused/resumed with **Shift-Esc**. See also: [pause](#)

A **midasrc** file is a command file that is executed automatically when the [Command Line](#) is started (see the [Command Line preferences](#)). Placing [aliases](#) and [color definitions](#) in a **midasrc** file is a convenient way to apply them each time Chimera is used.

[Chimera demos](#) are largely [constructed](#) from commands.

Chimera commands do not provide a way to loop through multiple residues or models, but this can be done by combining Chimera commands with Python code or a shell script. The Chimera Programmer's Guide includes a [primer on looping](#).

---

UCSF Computer Graphics Laboratory / October 2014

# UCSF Chimera Quick Reference Guide

April 2014

**Commands** (\*reverse function ~**command** available)

*2dlabels* create labels with text, symbols, and arrows in 2D  
*ac* enable accelerators (keyboard shortcuts)  
*addaa* add an amino acid to a peptide N- or C-terminus  
*addcharge* assign partial charges to atoms  
*addh* add hydrogens  
*adjust* change bond angle or bond length  
*alias\** create an alias or list the existing aliases  
*align* align two atoms or sets of atoms along the line of sight  
*angle* measure angles formed by atoms or by axes and planes  
*aniso\** show thermal ellipsoids  
*aromatic\** show ring aromaticity  
*background* set background color, gradient, or image  
*bond\** add/delete bonds  
*bondzone\** make zoning tools use points along bonds  
*cd* change the working directory  
*center* center the view on specified atoms  
*changechains* reassign chain identifiers  
*chirality* report the R/S configuration of a chiral center  
*clip\** move global clipping planes  
*close* close a model  
*cofr\** report or change the center of rotation  
*color\** color atoms/bonds, ribbons, labels, surfaces  
*colordef* define a new color  
*combine* combine molecule models into a single model  
*coordset* play through frames of a trajectory  
*copy* save image files  
*coulombic* color surfaces by Coulombic electrostatics  
*crystalcontacts* identify clashes between PDB symmetry copies  
*defattr* assign attribute values to atoms, residues, or models  
*define\** calculate and display axes, planes, centroids  
*delete* delete atoms and bonds  
*display\** display specified atoms  
*distance\** measure distances between atoms, axes, planes, centroids  
*echo* send text to the status line and Reply Log  
*export* save the graphical scene  
*filtrng\** show rings as filled  
*findclash\** identify clashes and contacts  
*findhbond\** (*hbonds*) identify hydrogen bonds  
*fitmap* fit atoms or map into map  
*fly* smoothly traverse a series of saved positions  
*focus\** adjust the view and center of rotation  
*freeze* stop all motion  
*getcrd* report coordinates  
*help* display the manual page for a command  
*hkage* create icosahedron as hexagon/pentagon mesh  
*intersurf* generate and display interface surfaces  
*invert* swap substituents of an atom  
*ksdssp* determine secondary structure from protein coordinates  
*label\** display atom labels

*labelopt* control the information in atom labels  
*lighting* adjust lighting and shininess  
*linewidth* control the width of wire bonds  
*longbond\** show/hide pseudobonds representing missing segments  
*mask* extract volume data bounded by surfaces  
*match* perform least-squares fitting of specified atoms  
*matchmaker* (*mnmaker*) align models in sequence, then in 3D  
*matrixcopy* apply the transformation of one model to another  
*matrixget* write the current transformation matrices to a file  
*matrixset* read and apply transformation matrices from a file  
*mclip\** control per-model clipping  
*mcopy* copy settings from one molecule model to another  
*measure* perform calculations on structures, surfaces, maps  
*meshmol* create a "molecule" to show surface mesh as sticks  
*minimize* energy-minimize structures  
*modelcolor* set color at the model level  
*modeldisplay\** set display at the model level  
*molmap* create a density map from atomic coordinates  
*morph* morph (interpolate) between different structures  
*move* translate models  
*movie* capture image frames and assemble them into a movie  
*msc\** color multiscale surfaces to match atoms  
*namesel\** save and name the current selection  
*nucleotides\** create special nucleotide representations  
*objdisplay\** display graphical objects  
*open\** read local files or fetch by ID  
*pause* pause script execution until the user presses a key  
*perframe\** specify commands to be executed at each display frame  
*play* script various complex motions  
*preset* apply a predefined combination of display settings  
*rainbow* color residues, chains, or models over a range  
*ramachandran* show Ramachandran plot of protein residues  
*rangecolor* color over a range according to attribute values  
*read* execute a command file, updating display at the end  
*represent* control atom/bond style (wire, stick, bs, sphere)  
*reset* restore default or saved orientations  
*resrenumber* reassign residue numbers  
*ribbackbone\** allow display of both ribbon and backbone atoms  
*ribbon\** display ribbon  
*ribclass* set ribbon residue class  
*ribinsidecolor\** set a separate color for inside protein helix ribbons  
*ribrepr* control ribbon style (flat, edged, rounded)  
*ribscale* control ribbon scaling (Chimera default, licorice)  
*ribspline* control ribbon path (B-spline or cardinal spline)  
*rlabel\** display residue labels  
*rmsd* evaluate the RMSD between specified sets of atoms  
*rock* rock (rotate back and forth)  
*roll* roll (rotate continuously)  
*rotation\** make a bond rotatable  
*runscript* run Python script with command-line arguments  
*save* save the current Chimera session  
*savepos\** save model positions  
*scale\** scale the view  
*scene\** save/restore scenes (positions, styles, colors, labels, etc.)

*scolor* color surfaces by volume data or geometry  
*section* move global clipping planes in parallel  
*segment* act on segmentation models  
*select\** select atoms, (de)activate models for motion  
*set\** set visual effects, individual model rotation  
*setattr\** set an attribute to a specified value  
*shape* create a surface of a specified geometric shape  
*show\** display specified atoms, undisplay the others  
*sleep* pause script execution for a specified time  
*solvate* add solvent using AmberTools  
*sop* adjust capping, edit surface models  
*split* partition a molecule model into separate submodels  
*start* start Chimera tools by name  
*stereo\** switch amongst stereo options and mono viewing  
*stop* exit from Chimera  
*surface\** calculate and display molecular surfaces  
*surfcat* (*msms cat*) group atoms for surface calculations  
*surfrepr* (*msms repr*) control surface style (solid, mesh, dot)  
*swapaa* mutate amino acids or swap rotamers  
*swapna* mutate nucleic acid residues  
*sym\** generate symmetry-related copies of a structure  
*system* send a command to the system shell  
*thickness* move global clipping planes in opposite directions  
*tile\** arrange models in a plane  
*topography* plot values in a volume data plane as surface heights  
*transparency\** make atoms/bonds, ribbons, and surfaces transparent  
*turn* rotate models  
*vdw\** display van der Waals (VDW) dot surface  
*vdwdefine\** set VDW radii  
*vdwdensity* set VDW surface dot density  
*version* show copyright information and Chimera version  
*viewdock* start ViewDock and load docking results  
*volume* display volume data such as electron density  
*vop* edit volume data  
*vseries* display, process, and save volume series  
*wait* suspend command processing until motion has stopped  
*window* adjust the view to contain the specified atoms  
*windoworigin* set graphics window location  
*windowsize\** adjust the dimensions of the graphics window  
*write* save atomic coordinates (pdb, mol2)  
*writesel* write a list of the currently selected (or unselected) items  
*zonesel* select atoms/surfs within cutoff of specified atoms/surfs

## Miscellaneous Operations (Default Settings)

selection from screen	Ctrl-left mouse button
add/toggle selection	Shift-Ctrl-left mouse button
rotation	left mouse button
XY-translation	middle mouse button
scaling	right mouse button or Side View
preferences	Favorites... Preferences...
searching help	Help... Search Documentation...
reporting a problem	Help... Report a Bug...
mailing list	chimera-users@cgl.ucsf.edu

Specification Symbols		
Symbol	Function	Usage
#	model number	# <i>model</i> (integer)
##	submodel number	## <i>submodel</i> (integer)
:	residue	: <i>residue</i> (name or number)
::	residue name	:: <i>residue</i>
..	chain ID	.. <i>chain</i>
@	atom name	@ <i>atom</i>
@.	alternate location ID	@. <i>alt_loc</i>
-	range	specifies a range of models, submodels, or residues
,	name separator	separates models or residues, ranges of models or residues, or names of atoms
*	whole wildcard	matches whole atom or residue names, e.g., <b>*@CA</b> specifies the alpha carbons of all residues
=	partial wildcard	matches partial atom or residue names, e.g., <b>@C=</b> specifies all atoms with names beginning with C
?	single-char wildcard	used for atom and residue names only, e.g., <b>:G??</b> selects all residues with three-letter names beginning with G
;	command separator	separates multiple commands on a single line
z<	zone specifier	<b>z&lt;zone</b> or <b>zr&lt;zone</b> specifies all residues within <i>zone</i> angstroms, <b>za&lt;zone</b> specifies all atoms (rather than entire residues) within that distance. Using > instead of < gives the complement.
&	intersection	intersection of specified sets
	union	union of specified sets
~	negation	negation of specified set

#### Selected Atom Attributes

Usage	Description
@/altLoc=altloc	alternate location ID
@/areaSAS=sasa	solvent-accessible surface area
@/areaSES=sesa	solvent-excluded surface area
@/bfactor=bfactor	B-factor
@/color=color	atom-level color assignment
@/defaultRadius=rad	default VDW radius
@/display	whether atom display bit is "on"
@/drawMode=mode	<i>mode</i> can be 0 (dot), 1 (sphere), 2 (endcap, as in stick), or 3 (ball)

@/element=atmo	atomic # or element symbol
@/idatmType=type	Chimera atom type
@/label	whether the atom is labeled
@/label=label	text of the atom label
@/labelColor=labcolor	color of the atom label
@/name=name	atom name
@/occupancy=occupancy	crystallographic occupancy
@/radius=radius	current VDW radius
@/serialNumber=n	serial number in the input file
@/surfaceCategory=category	surface calculation category (main, ligand, etc.)
@/surfaceDisplay	per-atom surface display bit (can be true for buried atoms without surface)

#### Selected Residue Attributes

Usage	Description
:/areaSAS=sasa	solvent-accessible surface area
:/areaSES=sesa	solvent-excluded surface area
:/isHet	residues in PDB HETATM records (or the mmCIF equivalent)
:/isHelix	amino acid residues in helices
:/isStrand or /isSheet	amino acid residues in strands
:/kdHydrophobicity=value	Kyte-Doolittle amino acid hydrophobicity
:/phi=angle	protein/peptide backbone phi angle
:/psi=angle	protein/peptide backbone psi angle
:/ssId=N	secondary structure element identifier (1 for first helix and first strand, etc.)
:/uniprotIndex=N	residue number in corresponding UniProt sequence, if any

#### Selected Molecule Model Attributes

Usage	Description
#/ballScale=factor	ball radius relative to VDW radius
#/color=color	model-level color assignment
#/display	model display bit
#/lineWidth=width	linewidth of wire representation
#/numAtoms=N	total number of atoms
#/numResidues=M	total number of residues
#/stickScale=factor	stick radius relative to bond radius

#### Specification Examples

```
#
- all models
#0
- model 0
#3:45-83,90-98
- residues 45-83 and 90-98 in model 3
:lys,arg
- lysine and arginine residues
:12,14@ca
- alpha carbons in residues 12 and 14
:12:14@ca
- all atoms in residue 12 and the alpha carbon in residue 14
:A@ca,c,n,o
- peptide backbone atoms in chain A
:50.B,.D
- residue 50 in chain B and all residues in chain D
:12-15,26-28.a,45.b
- residues 12-15 in all chains (except het/water), 26-28 in chain A, and 45 in chain B
#0.1-3,5
- submodels 1-3 of model 0 and all of model 5
#0.1-3,..5
- submodels 1-3 of model 0 and submodel 5 of all models
ligand
- any/all residues automatically classified as ligand
S | Fe
- all sulfur and iron atoms
@ca!/label and color!=green and color!=red
- atoms named CA which are not labeled, and are not green or red
@/bfactor>=20 and bfactor<=40
- atoms with B-factor values ranging from 20 to 40
:asn & helix
- asparagine residues in helices
#1:asp,glu & #0 z<10
- aspartate and glutamate residues in model 1 within 10 angstroms of model 0
solvent & Ng+ z<3 | solvent & N3+ z<3
- solvent residues within 3 angstroms of guanidinium nitrogens or sp3-hybridized, formally positive nitrogens
@/bfactor>50 & ~ solvent & ~ ions
- atoms with B-factor values over 50, excluding solvent and ions
```

UCSF Chimera is developed by the Resource for Biocomputing, Visualization, and Informatics (RBVI) at the University of California, San Francisco, funded by the National Institutes of Health (NIGMS P41-GM103311). The software is copyrighted and licensed by the Regents of the University of California.

## Getting Started Tutorial - Menu Version

Many tasks in Chimera can be accomplished in multiple ways. For example, colors and [display styles](#) can be changed with the [Actions menu](#) or by entering [commands](#). In general, commands are more concise and powerful, but menus allow easy access to features without knowledge of commands and their syntax.

In this tutorial, many of the same tasks performed with commands in the [Getting Started Tutorial - Command Version](#) are carried out using the menus instead.

To follow along, first [download](#) the PDB files included with this tutorial to a convenient location on your computer:

- [1zik.pdb](#) - leucine zipper
- [1d86.pdb](#) - DNA and netropsin

---

### Menus, Part 1 - Manipulation, Selection, and Chains

On **Windows/Mac**, click the **chimera** icon; on **UNIX**, start Chimera from the system prompt:

```
unix: chimera
```

A splash screen will appear, to be replaced in a few seconds by the main Chimera [graphics window](#) or [Rapid Access](#) interface (it does not matter which, the following instructions will work with either). If you like, resize the Chimera window by dragging its lower right corner.

Now open a structure. Choose **File... Open** from the menu and use the resulting [file browser](#) to locate and open the previously [downloaded](#) file [1zik.pdb](#). The structure is a leucine zipper formed by two peptides.

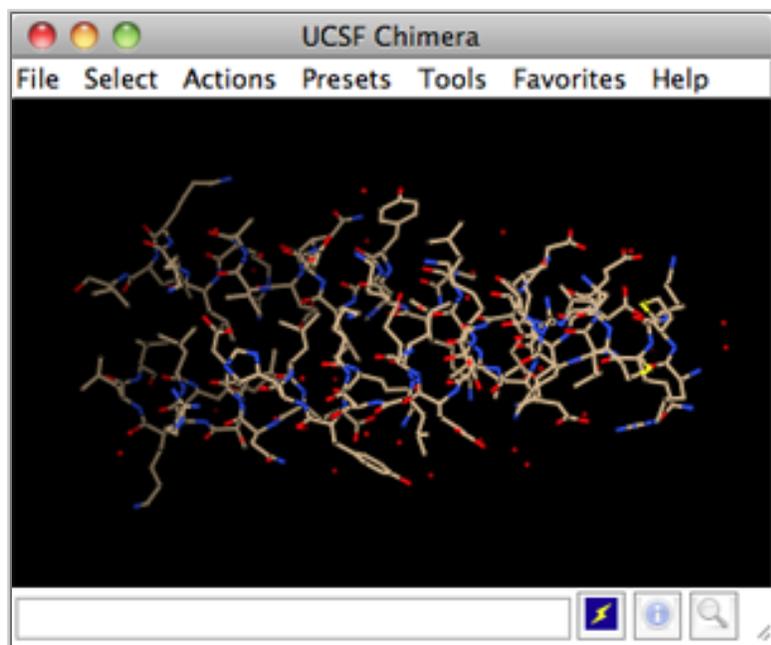
A [preset](#) is a predefined combination of display settings. Apply interactive preset #2:

**Presets... Interactive 2 (all atoms)**

This displays all atoms and color-codes atoms other than carbon [by element](#) (oxygen red, nitrogen blue, etc.); carbons are left in the initial model color, in this case tan.

Try [moving the structure with the mouse](#) in the main graphics window. By default:

- the left mouse button controls rotation

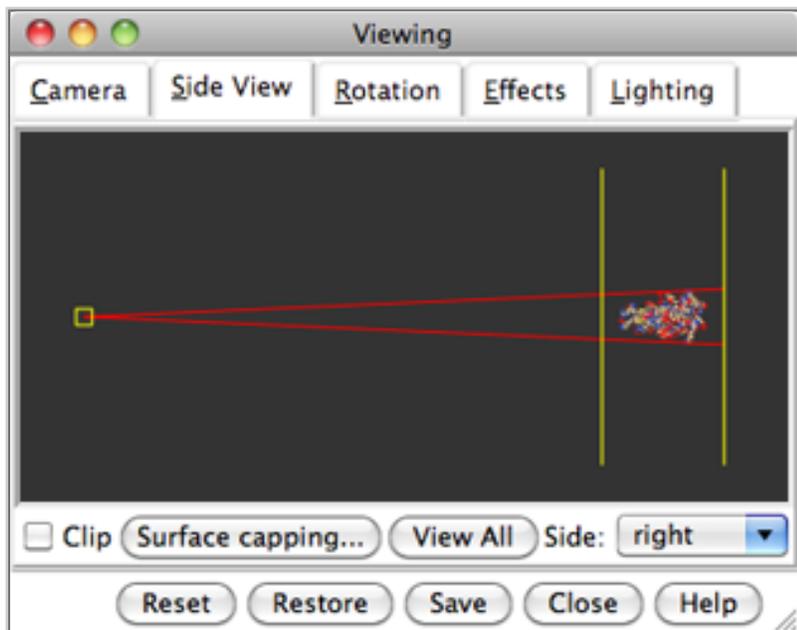


- the middle mouse button controls XY translation
- the right mouse button performs scaling (zooming)

If you are using a touchpad or single-button mouse, modifier keys allow emulating the middle and right mouse buttons. These are **option** and **command** () on Mac keyboards.

Use the **Favorites** menu to show the [Side View](#) for interactive scaling and clipping. It shows a tiny version of the structure. Within the **Side View**, try moving the eye position (the small square) and the clipping planes (vertical lines) with the left mouse button. The **Side View** will renormalize itself after movements, so that the eye or clipping plane positions may appear to “bounce back,” but your adjustments have been applied.

Continue moving and scaling the structure with the mouse in the graphics window and **Side View** as desired throughout the tutorial. When the mouse focus is in the graphics window (you may need to click into it if you have been interacting with a different window), hovering the mouse cursor over an atom or bond (without clicking any buttons) will show identifying information in a pop-up “balloon.” The balloon will disappear when the cursor is moved away.



In Chimera, *selection* specifies atoms, bonds, residues, *etc.* for subsequent operations with the [Actions menu](#). Ways to make a [selection](#) include using the [Select menu](#) or [picking](#) from the screen. The [Actions menu](#) applies to whatever is selected, but when nothing is selected, the [Actions menu](#) applies to everything.

Select the water (red dots):

**Select... Structure... solvent**

Another way to make the same selection is with **Select... Residue... HOH**. The selection is highlighted in green, and the magnifying glass icon near the bottom right of the window is also green: , indicating that something is selected. Hide the selected atoms:

**Actions... Atoms/Bonds... hide**

Even though the water atoms are hidden, they are still selected. Clear the selection, thicken the lines, and display only the chain trace:

**Select... Clear Selection**

**Actions... Atoms/Bonds... wire width... 3**

**Actions... Atoms/Bonds... backbone only... chain trace**

The chain trace includes just the  $\alpha$ -carbons (atoms named CA), connected in the same way that the residues are connected.

By default, [picking](#) from the screen is done by clicking on the atom or bond of interest with the left mouse button while pressing the **Ctrl** key. To add to an existing selection, also press **Shift**. Try picking two atoms, one from each peptide chain (Ctrl-click the first, Shift-Ctrl-click the second).

Label the atoms you have selected, first by atom name and then by residue name and number:

**Actions... Label... name**  
**Actions... Label... off**  
**Actions... Label... residue... name + specifier**

The specifier includes residue number and chain ID. One peptide is chain A and the other is chain B. Use the **Favorites** menu to show the [Preferences](#), change to **Category: Labels**, and adjust the **Label font** and size as you wish. Click **Save** before closing the preferences if you want the settings to apply to later uses of Chimera.

Clear the selection by Ctrl-clicking in empty space, as if [picking](#) "nothing." Turn off the labels:

**Actions... Label... residue... off**

Color the two chains different colors:

**Select... Chain... A**  
**Actions... Color... cyan**

Repeat the process to color chain B yellow.

Select chain A by [picking](#) any atom or bond in the chain, then pressing the up arrow key twice, once to [expand the selection](#) to the entire residue and another time to expand it to the entire chain. Display its full backbone:

**Actions... Atoms/Bonds... backbone only... full**

Display all atoms of chain A only (which is still selected):

**Actions... Atoms/Bonds... show only**

Display all atoms and color them by element:

**Select... Clear Selection**  
**Actions... Atoms/Bonds... show**  
**Actions... Color... by element**

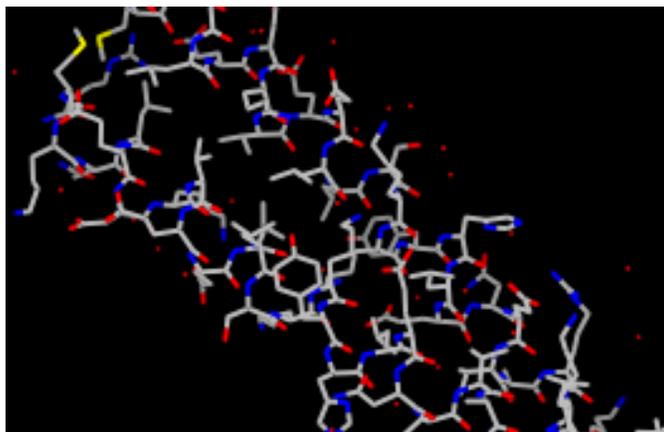
The **by element** coloring is the same as **by heteroatom** except it also color-codes carbons (gray). Heteroatom-only coloring is useful for keeping different structures distinguishable by their different carbon colors.

residue labels



coloring by element

Generally, each file of coordinates opened in Chimera becomes a model with an associated model ID number. Models are assigned successive numbers starting with 0. Models are listed in the left side of the [Model Panel](#) ([Tools... General Controls... Model Panel](#)). A checkbox in the **A**(ctive) column of the **Model Panel** shows that the model is [activated for motion](#); unchecking the box makes it impossible to move the model. Checking the box again restores the movable state. Make sure **1zik.pdb** is highlighted on the left side of the **Model Panel** (if not, click on it) and then click **close** in the list of functions on the right side. Next, use the **Close** button at the bottom to dismiss the **Model Panel**.



Go on to **Part 2** below, **OR** exit from Chimera with **File... Quit**.

## Menus, Part 2 - Molecular Representations and Surfaces

With Chimera started as described at the beginning of [Part 1](#), choose the menu item **File... Open**. Use the resulting [file browser](#) to locate and open the previously [downloaded](#) file [1d86.pdb](#). It contains the molecule netropsin bound to double-helical DNA.

[Move and scale](#) the structure with the mouse in the graphics window and **Side View** as desired throughout the tutorial.

Apply the “all atoms” preset, which will show the DNA as wire and netropsin as spheres:

**Presets... Interactive 2 (all atoms)**

Color carbons white, then undisplay the water:

**Select... Chemistry... element... C**  
**Actions... Color... white**  
**Select... Structure... solvent**  
**Actions... Atoms/Bonds... hide**

Remember that hiding atoms does not deselect them; they remain selected, as indicated by the green magnifying glass icon  near the bottom right of the window, until the selection is cleared or replaced with a new selection.

Color the different nucleotides different colors. For example, color the adenine deoxynucleotides blue:

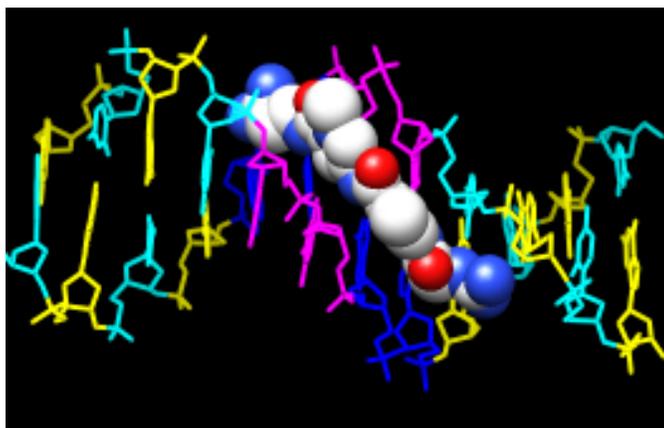
**Select... Residue... DA**  
**Actions... Color... blue**

Analogously, color DC residues cyan, DG residues yellow, and DT residues magenta. Clear the selection with **Select... Clear**

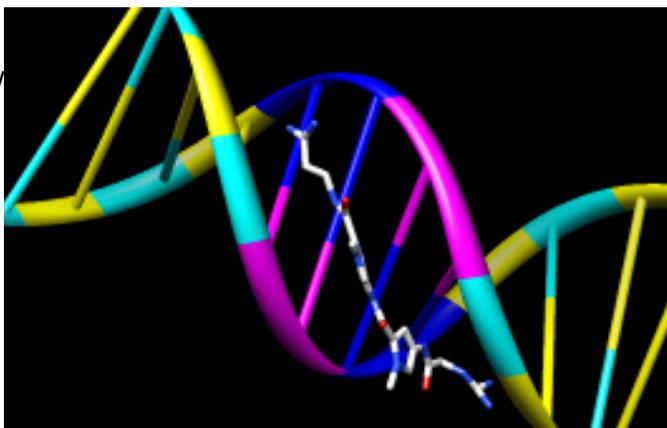
**Selection** or by [picking](#) (Ctrl-click) in empty space.

Next, try some different [display styles](#), or representations.

Actions... Atoms/Bonds... sphere  
Select... Chain... A  
Actions... Atoms/Bonds... ball & stick  
Select... Clear Selection  
Actions... Atoms/Bonds... stick



Showing ribbon automatically hides the [mainchain](#) (backbone) atoms.



Actions... Ribbon... show  
Actions... Ribbon... edged  
Actions... Ribbon... rounded

DNA can be shown with special [nucleotide objects](#). We will show “lollipops,” boxes, and a ladder.

Actions... Atoms/Bonds... nucleotide objects... settings...

In the resulting Nucleotides dialog:

1. set **Show side (sugar/base)** as to **tube/slab**
2. set **Show base orientation** to **false**
3. click **Slab Style** tab, set slab style to **skinny**
4. click **Slab Options** tab, set **Slab object** to **ellipsoid**
5. click **Apply**; these are the “lollipops”

Nucleotide settings can be applied to just the selected residues (not necessarily all of the DNA). One way to select specific residues is in the [Sequence](#) tool:

Favorites... Sequence

Show the sequence of chain A and select one or a few residues in the sequence window with the mouse; this selects the corresponding part of the structure. **Quit** from the sequence window. In the **Nucleotides** dialog (also under **Tools... Depiction** in the menu):

1. set **Show base orientation** to **true**
2. set **Slab object** to **box**
3. click **Apply**; base orientations are shown with “bumps”

Clear the selection (**Select... Clear Selection** or Ctrl-click in empty space) and use **Nucleotides** to show the DNA as a ladder:

1. set **Show side (sugar/base)** as to ladder
2. in the **Ladder Options**, set **Rung radius** to 0.3 Å
3. click **OK** (which will also dismiss the dialog)

To return to more general display styles, turn off the nucleotide objects:

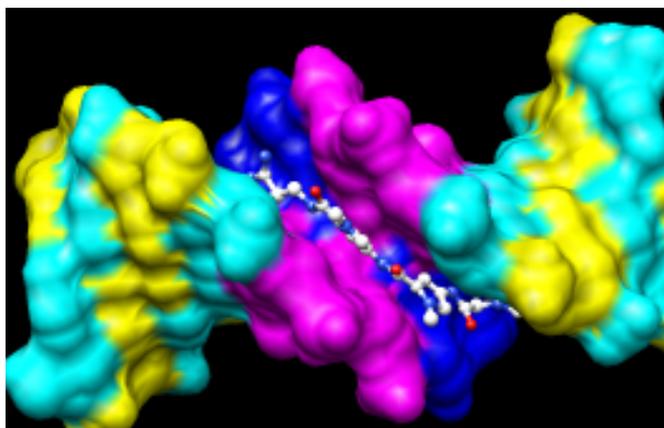
**Actions... Atoms/Bonds... nucleotide objects... off**

Hide the ribbons and show everything as ball-and-stick:

**Actions... Ribbon... hide**  
**Actions... Atoms/Bonds... ball & stick**

Finally, have some fun with surfaces. There are [built-in categories](#) within structures such as **main** and **ligand**; when nothing is [selected](#), **Actions... Surface... show** displays the surface of **main**.

**Actions... Surface... show**  
**Actions... Surface... hide**  
**Select... Structure... ligand**  
**Actions... Surface... show**  
**Actions... Surface... mesh**



Surface color can be specified separately from the colors of the underlying atoms. The ligand surface is tan and white because the original model color (tan) is used for surfaces of atoms not explicitly recolored by the user, and above, only the carbon atoms were changed to white. With the ligand still selected, choose **Actions... Color... all options...** to open the [Color Actions](#) dialog. In that dialog,

1. change the **Coloring applies to** (target) setting to **surfaces**
2. click **red**
3. click **Close** (which will automatically reset the coloring target back to **all of the above**)

Clear the selection, change back to a solid surface, and then undisplay the surface:

**Select... Clear Selection**  
**Actions... Surface... solid**  
**Actions... Surface... hide**

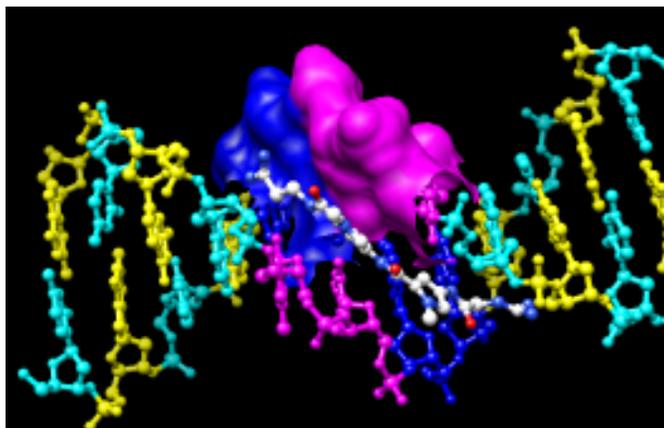
As an example of a more complicated selection process, show the surface of the adenine and thymine deoxynucleotides in chain B only:

1. change the selection mode: **Select... Selection Mode... append**

2. **Select... Residue... DA**
3. **Select... Residue... DT**
4. change the selection mode: **Select... Selection Mode... intersect**
5. **Select... Chain... B**
6. **Actions... Surface... show**

To prepare for any subsequent operations, restore the selection mode and clear the selection:

**Select... Selection Mode... replace**  
**Select... Clear Selection** (or Ctrl-click in empty space)



The [command](#) equivalent is much more concise, but requires some knowledge of the [atom specification syntax](#):

*Command:* [surf](#) :da.b,dt.b

Sometimes it is helpful to make a surface transparent:

**Actions... Surface... transparency... 50%**

Choose **File... Quit** from the menu to terminate the Chimera session.

## Getting Started Tutorial - Command Version

Many tasks in Chimera can be accomplished in multiple ways. For example, colors and [display styles](#) can be changed with the [Actions menu](#) or by entering [commands](#). In general, commands are more concise and powerful, but menus allow easy access to features without knowledge of commands and their syntax.

In this tutorial, many of the same tasks performed with menus in the [Getting Started Tutorial - Menu Version](#) are carried out using commands instead.

To follow along, first [download](#) the PDB files included with this tutorial to a convenient location on your computer:

- [1zik.pdb](#) - leucine zipper
- [1d86.pdb](#) - DNA and netropsin

---

### Commands, Part 1 - Manipulation, Selection, and Chains

On **Windows/Mac**, click the **chimera** icon; on **UNIX**, start Chimera from the system prompt:

```
unix: chimera
```

A splash screen will appear, to be replaced in a few seconds by the main Chimera [graphics window](#) or [Rapid Access](#) interface (it does not matter which, the following instructions will work with either). If you like, resize the Chimera window by dragging its lower right corner.

Use the **Favorites** menu to show the [Command Line](#). Now open a structure. Enter the command:

```
Command: open
```

and use the resulting [file browser](#) to locate and open the previously [downloaded](#) file [1zik.pdb](#). The structure is a leucine zipper formed by two peptides.

A [preset](#) is a predefined combination of display settings. Apply interactive preset #2:

```
Command: preset apply int 2
```

This displays all atoms and color-codes atoms other than carbon [by element](#) (oxygens red, nitrogens blue, *etc.*); carbons are left in the initial model color, in this case tan.

Try [moving the structure with the mouse](#) in the main graphics window. By default:

- the left mouse button controls rotation
- the middle mouse button controls XY translation
- the right mouse button performs scaling (zooming)

If you are using a touchpad or single-button mouse, modifier keys allow emulating the middle and right mouse buttons. These are **option** and **command** () on Mac keyboards.

Show the [Side View](#) for interactive scaling and clipping:

*Command:* [start Side View](#)

It is also listed in the **Favorites** menu by default. It shows a tiny version of the structure. Within the **Side View**, try moving the eye position (the small square) and the clipping planes (vertical lines) with the left mouse button. The **Side View** will renormalize itself after movements, so that the eye or clipping plane positions may appear to “bounce back,” but your adjustments have been applied.

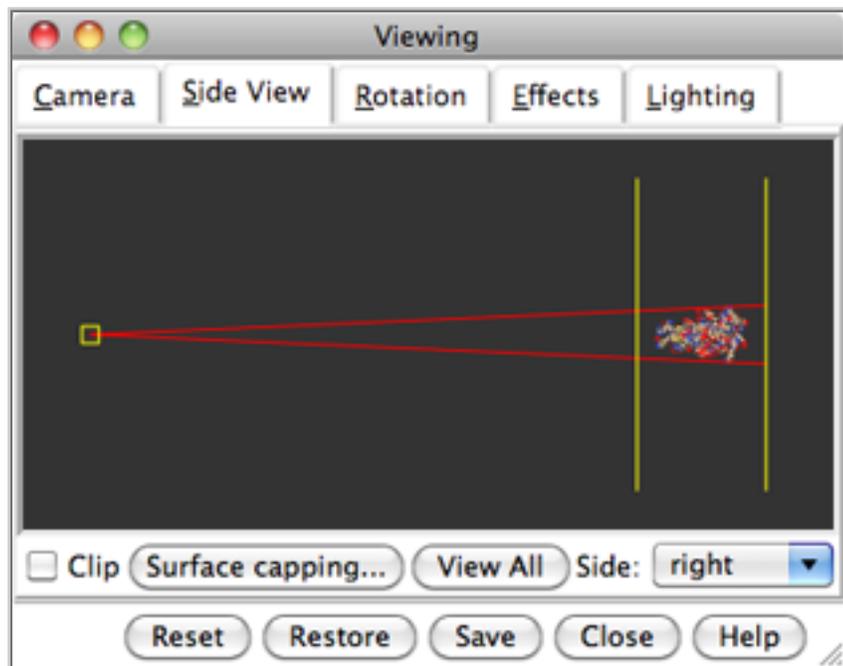
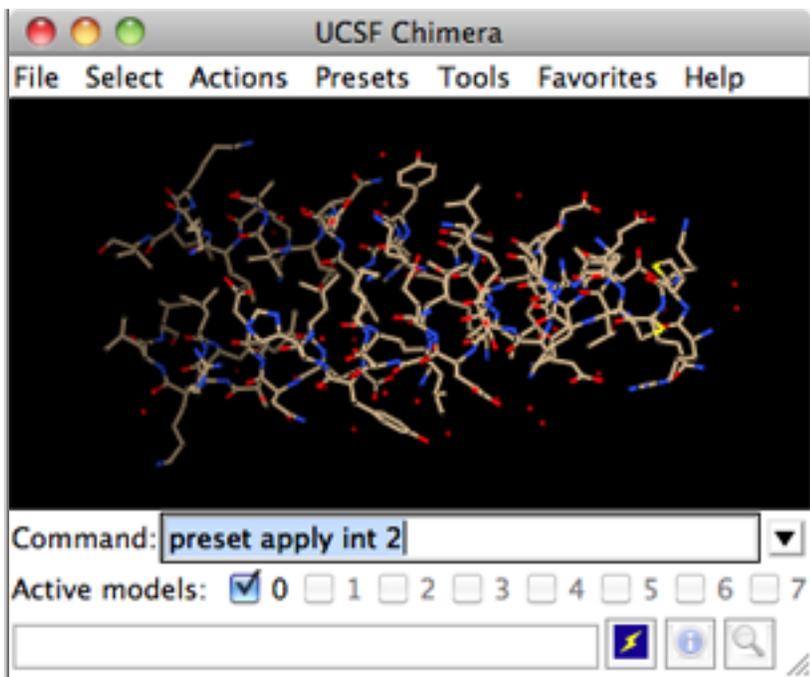
Continue moving and scaling the structure with the mouse in the graphics window and

**Side View** as desired throughout the tutorial. When the mouse focus is in the graphics window (you may need to click into it if you have been interacting with a different window), hovering the mouse cursor over an atom or bond (without clicking any buttons) will show identifying information in a pop-up “balloon.” The balloon will disappear when the cursor is moved away.

A Chimera command may include *arguments* and/or an [atom specification](#). For example, in the following,

*Command:* [color hot pink :lys](#)

the [color name hot pink](#) is an argument of the command [color](#), and [:lys](#) specifies all residues named LYS.



A blank specification is interpreted as all applicable items. For example,

Command: [color](#) hot pink

makes all atoms (and their labels, surfaces, etc.) hot pink.

Many commands have “~” versions that perform the opposite function. The following will change a structure back to its default color:

Command: [~color](#)

The command [help](#) can be used to show the manual page for any command. For example,

Command: [help](#) color

shows the manual page for the command [color](#). The [Chimera Quick Reference Guide \(PDF\)](#) lists most [commands](#) and gives some examples of command-line [atom specification](#).

Thicken the lines and display only the atoms named CA ( $\alpha$ -carbons):

Command: [linewidth](#) 3

Command: [show](#) @ca

By default, [picking](#) from the screen (a type of [selection](#)) is done by clicking on the atom or bond of interest with the left mouse button while pressing the **Ctrl** key. To add to an existing selection, also press **Shift**. Try picking two  $\alpha$ -carbons, one from each peptide chain (Ctrl-click the first, Shift-Ctrl-click the second). The selection is highlighted in green, and the magnifying glass icon near the bottom right of the window is also green: , indicating that something is selected.

The word **selected**, **sel**, or **picked** can be used in commands to specify the current selection. Label the atoms you have selected:

Command: [label](#) sel

The [label](#) command shows atom information (atom name, by default). Undisplay the atom labels, then show labels for the *residues* containing the selected atoms:

Atom Specification Symbols		
Symbol	Meaning	Usage
#	model	# <i>model</i> (model ID number)
:	residue	: <i>residue</i> (residue name or number)
::	chain	:: <i>chain</i> (chain ID)
@	atom	@ <i>atom</i> (atom name)
=	partial wildcard	matches partial atom or residue name, e.g., @C= specifies all atoms with names beginning with C
?	single-character wildcard	matches single character in atom or residue name, e.g., :G?? specifies all residues with three-letter names beginning with G

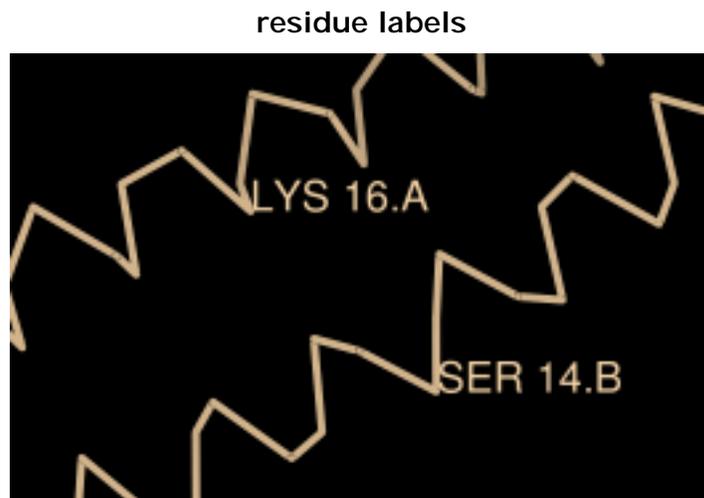
Command: [~label](#)

Command: [rlabel](#) sel

Each residue label is of the form:

*res\_name res\_number.chain*

One peptide is chain A and the other is chain B. Use the **Favorites** menu to show the [Preferences](#), change to **Category: Labels**, and adjust the **Label font** and size as you wish. Click **Save** before closing the preferences if you want the settings to apply to later uses of Chimera.



Clear the selection by Ctrl-clicking in empty space, as if [picking](#) “nothing.” Turn off the labels:

Command: [~rlabel](#)

Color the two chains different colors; note that commands can be truncated to unique strings:

Command: [color](#) cyan :a

Command: [col](#) yellow :b

Residues and atoms can also be specified, along with or independent of chain:

Command: [col](#) orange :5-9.a,12.a,8.b

Command: [col](#) magenta :14-18

Command: [disp](#) :leu.b

Command: [col](#) green :leu.b@cb

The structure also includes water, which can be shown with:

Command: [disp](#) solvent

-OR- (equivalent)

Command: [disp](#) :hoh

Display the full backbone of chain A:

Command: [disp](#) :a@n,ca,c,o

Display all atoms in chain A only:

Command: [show](#) :a

Display all atoms and color them by element:

Command: [disp](#)

Command: [col](#) byelement

Coloring **byelement** is the same as **byhet** except it also color-codes carbons (gray). Heteroatom-only coloring is useful for keeping different structures distinguishable by their different carbon colors.

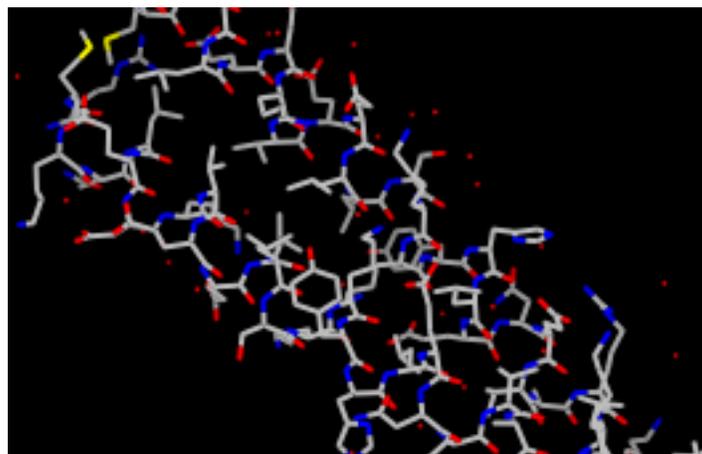
Generally, each file of coordinates opened in Chimera becomes a model with an associated model ID number. Models are assigned successive numbers starting with 0. The **Active models** line right under the **Command Line** shows which models are [activated for motion](#).

The checkbox for **0** is activated. Unchecking the box makes it impossible to move model 0. Checking the box again restores the movable state.

Command: [close](#) 0

closes the model. Go on to **Part 2** below, **OR** exit from Chimera with the following command:

Command: [stop](#)




---

## Commands, Part 2 - Molecular Representations and Surfaces

With Chimera started and the [Command Line](#) opened as described at the beginning of [Part 1](#), choose the menu item **File... Open**. Use the resulting [file browser](#) to locate and open the previously [downloaded](#) file [1d86.pdb](#). It contains the molecule netropsin bound to double-helical DNA.

[Move and scale](#) the structure with the mouse in the graphics window and **Side View** as desired throughout the tutorial.

Apply the “all atoms” preset, which will show the DNA as wire and netropsin as spheres:

Command: [preset](#) apply int 2

Color carbons white, then undisplay the water:

Command: [color](#) white C

Command: [~disp](#) solvent

Residue names can be identified by looking in the **Select... Residue** menu or by hovering the cursor over an atom or bond to see information in a pop-up

“balloon.” Color the different nucleotides different colors, specifying them by residue name:

Command: `color blue :da`  
 Command: `color cyan :dc`  
 Command: `color yellow :dg`  
 Command: `color magenta :dt`

Next, try some different [display styles](#), or representations.

Command: `represent sphere`  
 Command: `repr bs :a`  
 Command: `rep stick`

Notice that commands (but not necessarily their keyword arguments) can be truncated to unique strings. For example, the command `represent` can be shortened to `repr` or `rep` but not `re` (because other commands also start with `re`), whereas its keywords `stick`, `sphere`, etc. cannot be truncated. If the truncation is not unique, one of the corresponding commands will be executed, but it may not be the one intended.

Showing ribbon automatically hides the [mainchain](#) (backbone) atoms.

Command: `ribbon`  
 Command: `ribrep edged`  
 Command: `ribr rounded`

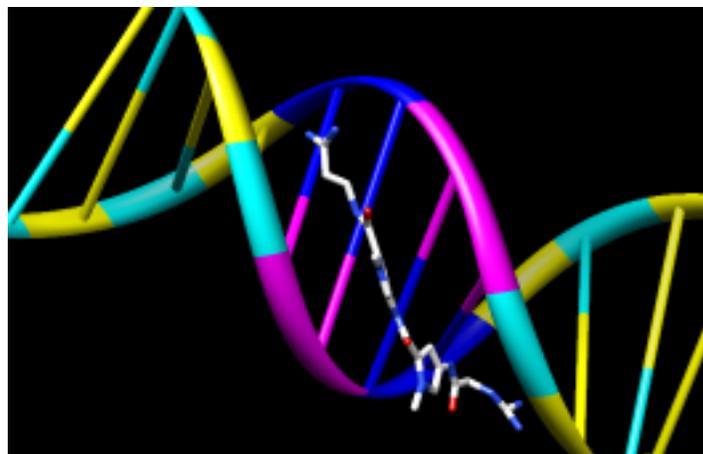
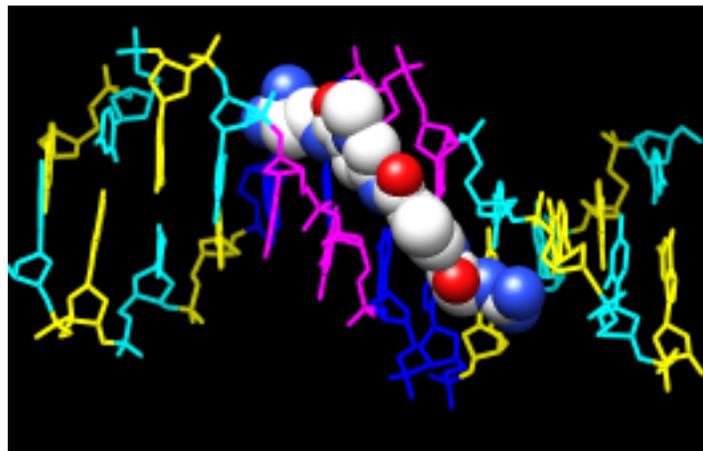
DNA can be shown with special [nucleotide objects](#). We will show “lollipops,” boxes with orientation bumps, and then a ladder. You can copy and paste into the **Command Line**. The command-line contents can be edited, and past commands can be accessed using the up and down arrow keys or **Ctrl-p** (previous) and **Ctrl-n** (next).

Command: `nuc side tube/slab shape ellipsoid orient false style skinny`  
 Command: `nuc side tube/slab shape box orient true style skinny :8-10.a`  
 Command: `nuc side ladder radius 0.3`

To return to more general display styles, turn off the nucleotide objects:

Command: `~nuc`

Hide the ribbons and show everything as ball-and-stick:



Command: [~ribbon](#)

Command: [rep bs](#)

Finally, have some fun with the **surface** command. There are [built-in categories](#) within structures such as **main** and **ligand**; when nothing is specified, [surface](#) shows the surface of **main**.

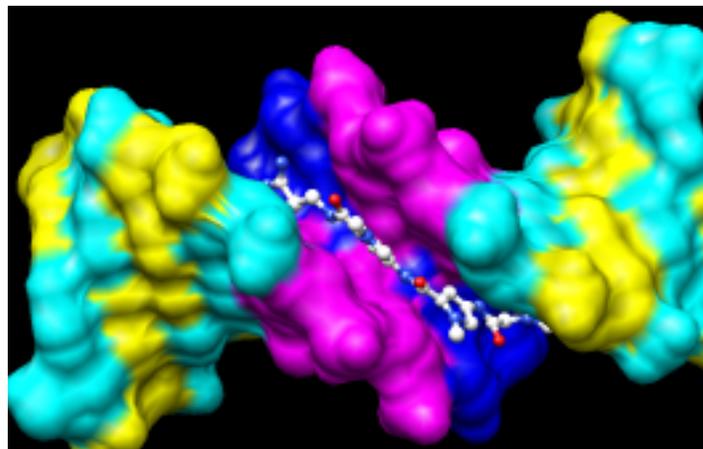
Command: [surface](#)

Command: [~surf](#)

Command: [surf](#) ligand

-OR- (equivalent)

Command: [surf](#) :nt



Surface color can be specified separately from the colors of the underlying atoms. The ligand surface is tan and white because the original model color (tan) is used for surfaces of atoms not explicitly recolored by the user, and above, only the carbon atoms were changed to white. Show the ligand surface as red mesh:

Command: [surfrep mesh](#)

Command: [color](#) red,s ligand

Command: [surfrep solid](#)

Parts of a surface can be shown:

Command: [~surf](#)

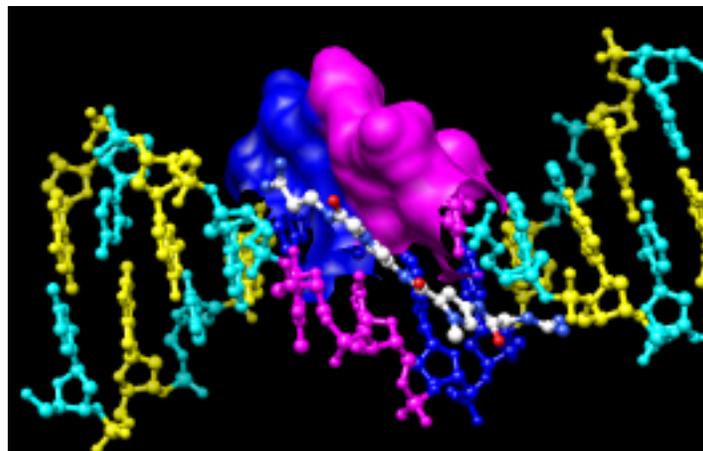
Command: [surf](#) :da,dt

Command: [~surf](#)

Command: [surf](#) :da,b,dt,b

Sometimes it is helpful to make a surface transparent:

Command: [transp](#) 50,s



When finished, exit from Chimera:

Command: [stop](#) now

## Image Tutorial: Surface Properties

This tutorial describes how to make high-quality images of a protein surface colored by hydrophobicity and electrostatic potential. Note there are several routes to the same or similar results. See also: [presets](#), [tips on preparing images](#)

- [Background and Setup](#)
- [Amino Acid Hydrophobicity](#)
- [Electrostatic Potential: Coulombic](#)
- [Electrostatic Potential: Poisson-Boltzmann](#)

### ← Background and Setup

[Start Chimera](#) and show the [Command Line](#) (for example, with [Favorites...](#) [Command Line](#)). Fetch Protein Data Bank entry [3eeb](#):

*Command:* [open](#) 3eeb

[Move and scale](#) the structure as you wish throughout the tutorial. It contains two copies of a protease domain bound to a highly negatively charged small molecule, inositol hexakisphosphate (IHP). Delete one of the copies, chain A, and solvent:

*Command:* [delete](#) :a

*Command:* [del](#) solvent

The ligand IHP looks somewhat like a distorted snowflake. IHP binds to a conserved site lined with positively charged groups. The ligand heteroatoms and a nearby sodium ion are [colored by element](#). Show the molecular surface, hide the ion, make sticks fatter, and make ligand carbons yellow:

*Command:* [surface](#)

*Command:* [~disp](#) ions

*Command:* [setattr](#) m stickScale 2

*Command:* [color](#) yellow ligand & C

Resize the window as desired, either by dragging its lower right corner with the mouse or by using the command [window size](#). The window dimensions define the aspect ratio (width:height) of output images, but image resolution (pixel dimensions) can be specified independently when an image is saved.

### ← Amino Acid Hydrophobicity

After [setup](#), color the molecular surface by amino acid hydrophobicity on the [Kyte-Doolittle scale](#):

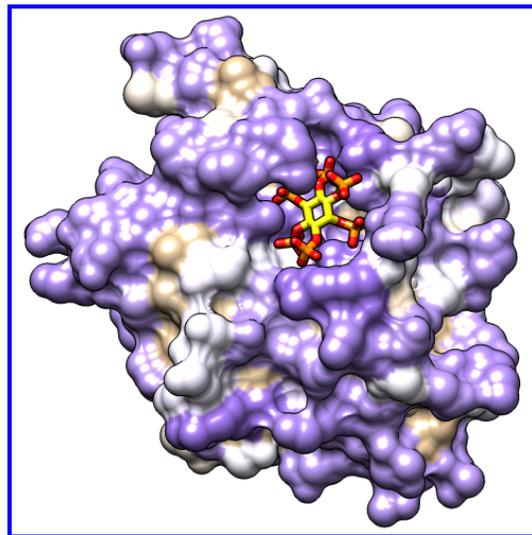
*Command:*

[range color](#) kdHydrophobicity min dodger blue 0 white max orange red

This gives the same coloring as the “hydrophobicity surface” preset: from **dodger blue** for the most hydrophilic, to **white**, to **orange red** for the most hydrophobic. The IHP-binding pocket is primarily blue, indicating its hydrophilic character.

However, any colors can be used. The figure shows the result of the following command:

*Command:* [range color](#) kdHydrophobicity min medium purple 0 white max tan



Now the most polar residues are **medium purple** and the most hydrophobic are **tan**. Coloring by residue hydrophobicity and other attributes (B-factor, sequence conservation, *etc.*) can also be done with the [Render by Attribute](#) graphical interface, as in the [B-Factor Coloring](#) image tutorial. The graphical interface is also helpful in that it shows the names and value ranges of

the available attributes.

The built-in colors can be viewed [here](#) or by choosing **Actions... Color... all options** from the menu and in the resulting dialog, checking the option to **Show all colors**. Additional colors can be created with the command [colordef](#).

Use a publication preset for nice image settings, including white background, black outlines, and increased smoothness:

*Command:* [preset](#) apply pub 1

The example image was saved from a 500x500-pixel window using **File... Save Image** with default settings.

### ← Electrostatic Potential: Coulombic

After [setup](#), use a publication preset (if not already done):

*Command:* [preset](#) apply pub 1

Start [Coulombic Surface Coloring](#) (under **Tools... Surface/Binding Analysis**). This tool calculates electrostatic potential according to Coulomb's law. The colors and associated values can be changed, but for most cases the default settings are suitable. Simply click **OK** to color from red for negative potential, to white near neutral, to blue for positive potential.

The publication presets provide a nice starting point, but you might want to adjust some settings, such as the thickness of the black outlines:

*Command:* [set silhouetteWidth](#) 3

If you prefer a simple "line drawing" appearance, try ambient-only lighting:

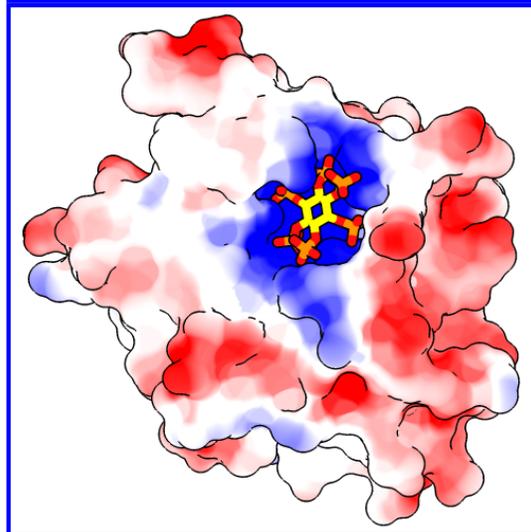
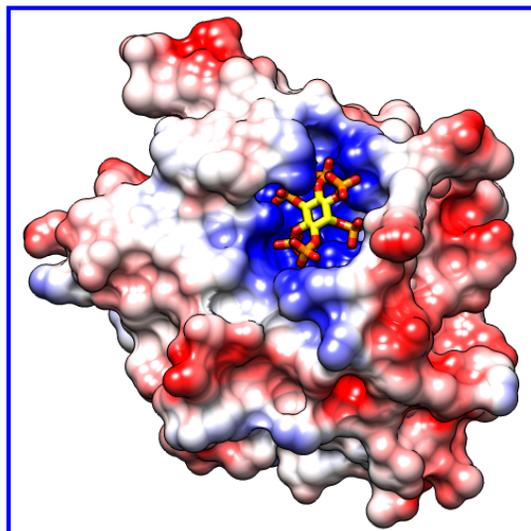
*Command:* [light](#) mode ambient

To restore the default lighting mode:

*Command:* [light](#) mode two-point

The example images were saved from a 500x500-pixel window using **File... Save Image** with default settings.

There is a limit to how thick lines such as silhouettes, wire, and mesh can be drawn while rendering an image. If supersampling is done, the image is initially drawn at a larger size than requested (3x3 by default) and then sampled back down to the final size. In large and/or supersampled images, lines may be thinner than expected. The **File... Save Image** dialog reports the effective maximum linewidth. It may be possible to achieve the desired thickness by reducing the supersampling level and/or the pixel dimensions of the image.



### ← Electrostatic Potential: Poisson-Boltzmann

After [setup](#), use a publication preset (if not already done):

*Command:* [preset](#) apply pub 1

Poisson-Boltzmann (PB) electrostatics calculations take into account spatial variations in the dielectric according to the shape of the molecule. Programs that solve the PB equation include [APBS](#) and [DelPhi](#). Chimera can read grid files or "maps" of electrostatic potential from running such programs separately, but it also includes the following interfaces:

- the [APBS](#) tool runs an APBS web service provided by the [NBCR](#)

- [DelPhiController](#) runs a local (user-installed) copy of DelPhi

To use web services, start the [APBS](#) interface (under **Tools... Surface/Binding Analysis**). Otherwise, [download](#) and open the provided map file [3eebB.phi](#) previously calculated with DelPhi, and skip to [coloring](#).

APBS requires first assigning atomic partial charges. Chimera's [PDB2PQR](#) interface uses another web service to assign charges and radii appropriate for PB calculations. Choose **Tools... Structure Editing... PDB2PQR** from the Chimera menu. Look at the **Options** if you wish, but for this tutorial, simply click **OK** to assign PARSE parameters and dismiss the dialog. The assignment may take several seconds. The resulting structure with PARSE charges and radii is automatically opened as a new model (#1) containing only the protein atoms from the original structure, #0.

In the **APBS** dialog, choose the new model, 3eeb PDB2PQR (#1), as the **Molecule** of interest. If you like, reveal the **Options** to see what they are, but then click **OK** to run the APBS web service with default settings. The calculation may take a few minutes. When it finishes, the resulting ".dx" map file will be opened automatically as model #2.

Opening an electrostatic potential map starts the [Electrostatic Surface Coloring \(Surface Color\)](#) tool. The surface coloring dialog allows changing the colors and associated potential values, but for most cases the default settings are suitable. Make sure the dialog is set to color the molecular surface (#0) and click **Color** to use the standard coloring scheme of red for negative, white for neutral, and blue for positive electrostatic potential. It may take several seconds to read the potential file before coloring occurs. As with [Coulombic coloring](#), the positive potential in the IHP-binding pocket should be clearly evident.

---

meng-at-cgl.ucsf.edu / July 2013



[Tutorials Index](#)

**Images: Glycoside Hydrolases**

[Setup](#)

[Important Residues](#)

[Matching](#)

[Ribbons](#)

[Positions](#)

[Coloring](#)

[Effects](#)

[Sessions](#)

[Saving Images](#)

## Image Tutorial: Glycoside Hydrolases

This tutorial describes superimposing structures, saving positions and sessions, and creating publication-quality images. The process described here is only one of many possibilities. Internet connectivity is required to fetch the structures **1uyp**, **1gyd**, and **1oyg**.

See also: [presets](#), [tips on preparing images](#), and on the Chimera web site, the [Image Gallery](#)

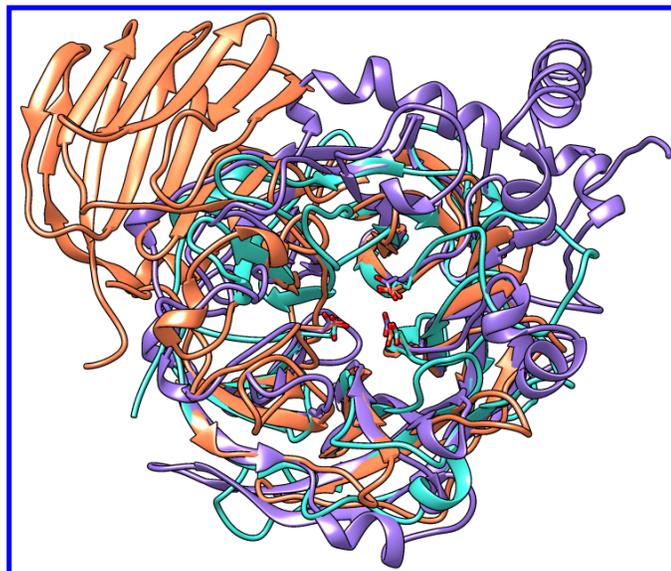
### Background and Setup

Glycoside hydrolases (GH) are a large, heterogeneous set of enzymes that hydrolyze bonds between sugars and other groups. Based on sequence comparisons, these enzymes have been categorized into many families; see the [CAZy web site](#) for details.

This tutorial describes the creation of images to show structural similarities among members of GH families 32, 43, and 68. These enzymes were grouped into different families because their similarities were not evident from sequence comparisons.

[Three acidic residues are at the active site of a beta-propeller architecture in glycoside hydrolase families 32, 43, 62, and 68.](#) Pons T, Naumoff DG, Martínez-Fleites C, Hernández L. *Proteins*. 2004 Feb 15;54(3):424-32.

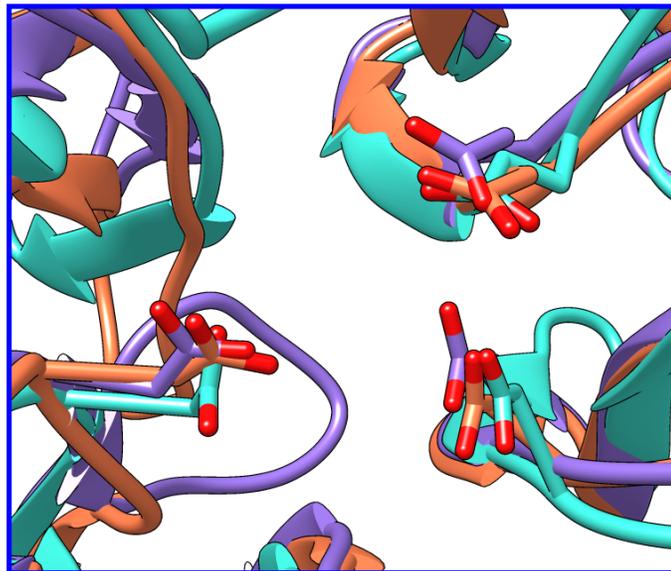
[The three-dimensional structure of invertase \(beta-fructosidase\) from \*Thermotoga maritima\* reveals a bimodular arrangement and an](#)



[evolutionary relationship between retaining and inverting glycosidases](#). Alberto F, Bignon C, Sulzenbacher G, Henrissat B, Czjzek M. *J Biol Chem*. 2004 Apr 30;279(18):18903-10.

[Start Chimera](#) and enlarge the window as desired. Show the [Command Line](#) (for example, with [Favorites... Command Line](#)).

Structures can be obtained directly from the [Protein Data Bank](#). Choose [File... Fetch by ID](#) from the Chimera menu. In the resulting dialog, choose the **PDB** database and check the option to **Keep dialog up after Fetch**. Fetch the PDB structures **1uyp**, **1gyd**, and **1oyg**, in that order, and then **Close** the dialog.



Invertase (GH 32) is coral, arabinanase A (GH 43) is turquoise, and levansucrase (GH 68) is purple. Despite significant divergence, the enzymes share similar overall folds (top) and highly conserved active site residues (bottom).

Some salient features of these structures:

PDB ID	enzyme	family	chains	conserved residues			model # in Chimera
				Asp17	Asp138	Glu190	
<b>1uyp</b>	invertase, <i>T. maritima</i>	<a href="#">GH32</a>	A-F	Asp17	Asp138	Glu190	0
<b>1gyd</b>	arabinanase A, <i>C. japonicus</i>	<a href="#">GH43</a>	B	Asp38	Asp158	Glu221	1
<b>1oyg</b>	levansucrase, <i>B. subtilis</i>	<a href="#">GH68</a>	A	Asp86	Asp247	Glu342	2

Simplify the situation by deleting unwanted (for our purposes) extra chains:

*Command:* [delete](#) #0:b-f

Choose [Presets... Interactive 1 \(ribbons\)](#) from the menu; depending on your settings, this may not change anything in the display. [Move and scale](#) the structures as desired throughout the tutorial.

## Important Residues

Alias the [conserved residues](#) to some short name for convenience. To avoid typing long commands, you can copy text from this page and paste it into the **Command Line**:

*Command:* [alias](#) myres #0:17,138,190#1:38,158,221#2:86,247,342

Show only the atoms/bonds of these residues:

Command: `~display`

Command: `display myres`

Even though all atoms of the residues are “displayed,” backbone atom display is suppressed by the ribbon (display of both at the same time can be enabled with [ribbackbone](#)).

## Matching the Structures

Initially, the relative positions of the structures are not meaningful and merely reflect their coordinates as deposited in the Protein Data Bank. Chimera includes a few different ways of [superimposing structures](#).

First, try [MatchMaker](#) (under **Tools... Structure Comparison** in the menu). In the resulting dialog, choose **1uyp** as the reference and the other two (or all three) as the structures to match. Leave the other settings as defaults and click **OK**. This uses the protein sequences and secondary structure assignments to generate pairwise sequence alignments, then fits the  $\alpha$ -carbons of the aligned pairs of residues. The numbers of points used for fitting and the resulting RMSD values are reported in the [Reply Log](#) (under **Favorites** in the menu). When the structures have been superimposed, focus the view:

Command: `focus`

Manually zoom and reorient to see how well the conserved residues are superimposed. Save the position to facilitate comparison with the results of a different matching method, used below. A *position* includes the scale, locations, and orientations of the structures.

Command: `savepos pos1`

Since structurally equivalent residues are known, another way to superimpose the structures is by [specifying atoms](#) to use in a least-squares fit. The `match` command requires equal numbers of atoms from the two models being matched. The following specifies using the backbone atoms of the [conserved residues](#):

Command: `match #1:38,158,221@n,ca,c,o #0:17,138,190@n,ca,c,o`

Command: `match #2:86,247,342@n,ca,c,o #0:17,138,190@n,ca,c,o`

Command: `savepos pos2`

Again, **1uyp** (model 0, invertase) was used as the reference structure for both pairwise matches, and RMSD values are given in the [Reply Log](#). Chain identifiers were not needed in the commands since each model has only one chain (chains B-F were deleted from model 0).

Review the two superpositions to decide which is better:

Command: `reset pos1 30`

Command: `reset pos2 30`

Repeat as desired. The number at the end of the [reset](#) command is the number of frames over which to gradually restore the position. A gradual rather than abrupt transition between positions may make them easier to compare.

In this case, the light blue structure (model 1, arabinanase) is better superimposed with the others in the result from **MatchMaker** (position **pos1**).

## Ribbons

You may have noticed that some of the bonds from sidechains to the ribbon look very long. This is because the ribbon path (by default) is a smooth B-spline, which can diverge from the true positions of the backbone atoms. A cardinal spline allows tracking the backbone more closely. Without smoothing, a cardinal spline follows the  $\alpha$ -carbons exactly, or it can be combined with some “compromise” smoothing of strand and/or coil:

*Command:* [ribspline](#) cardinal

*Command:* [ribsp](#) card smooth strand

*Command:* [ribsp](#) default

Ribbon styles include **edged** and **rounded**; different styles can be used for different parts of the same ribbon, as in the first example below:

*Command:* [ribrep](#) edged strand

*Command:* [ribrep](#) edged

*Command:* [ribrep](#) rounded

Ribbon scalings are secondary-structure-specific heights and widths. Try the “licorice” scaling and then go back to the default.

*Command:* [ribscale](#) licorice

*Command:* [ribscale](#) "Chimera default"

Custom ribbon [styles](#) and [scalings](#) can be created with the [Ribbon Style Editor](#) (Tools... Depiction... Ribbon Style Editor).

Although the [examples](#) show ribbons for the entire chains, each residue's ribbon segment can be shown or hidden. For example, the following hides ribbons for the C-terminal domain of the first structure (model 0):

*Command:* [~ribbon](#) #0:296-end

A command without residues specified affects all residues:

*Command:* [ribbon](#)

## Positions for Images

Resize the Chimera window to the desired aspect ratio and find positions suitable for the images.

A window size of 750x636 pixels and two positions were used for the [example images](#). A view of the overall structures was generated by interactive manipulation, then saved (with [savepos](#)) as a position named **overall**. Another position zoomed into the active site was generated from **overall** by scaling and translation only (to maintain a consistent view) and saved as the position **closeup**.

Saved positions are included in saved [sessions](#) and can be restored before [saving images](#).

## Coloring the Structures

The current model colors could be used, but typically users will want to apply their own scheme. Several factors should be considered in [choosing colors](#), including what the colors are meant to indicate, their distinguishability, and whether viewers may have color vision deficiencies. The colors named **coral**, **turquoise**, and **medium purple** were used for models 0, 1, and 2, respectively, in the [example images](#).

Open the [Color Actions](#) dialog (**Actions... Color... all options**) and near the bottom, check **Show all colors** to expand it. Select a model:

*Command:* [sel](#) #0

and click the desired color in the dialog. Repeat for models 1 and 2, then close the coloring dialog. Clear the selection:

*Command:* [~sel](#)

To use heteroatom color-coding so that sidechain oxygens are red:

*Command:* [color](#) byhet

## Effects and Lighting

The background can be any color, but white is often best for publication images.

*Command:* [background](#) solid white

Depth cueing is front-to-back shading of the scene, which looks like “fog” when the depth cueing color is pale. By default, the depth cueing color automatically matches the background. Turn off depth cueing to remove the fog effect:

*Command:* [~set depthCue](#)

Ribbons, sticks, and other spacefilling [display styles](#) are made up of planar facets. The coarseness of the facets is controlled by the subdivision parameter, which ranges from 1 to 20 (default 1.5). Increase smoothness by increasing the subdivision:

*Command:* [set subdivision](#) 5

Silhouettes are outlines that highlight boundaries and discontinuities. Turn on silhouettes:

*Command:* [set silhouette](#)

The silhouettes look rough in the Chimera window, but they will be smooth in the saved images due to [supersampling](#).

Instead of the four preceding commands, publication preset 1 could be used to achieve the same result (command [preset apply pub 1](#) or menu item **Presets... Publication 1**). Silhouettes and increased subdivision may degrade interactive response; if so, adjust these settings last before saving [sessions](#) and [images](#).

The contrast, or darkness of shading, is another important contributor to the clarity of figures. Decrease the contrast (default 0.83):

*Command:* [light contrast 0.55](#)

Graphical interfaces to [Effects](#) and [Lighting](#) are available under **Tools... Viewing Controls** in the menu.

## Sessions

A [session](#) file can be saved with **File... Save Session As** in the Chimera menu. It is generally prudent to save sessions for publication images, as this decreases the labor necessary if the figures have to be redone. Often minor adjustments such as changing a color or displaying a different set of side chains will be required.

Later, the session can be restarted with **File... Restore Session**, and any saved [positions](#) within that session restored with the command [reset](#).

## Saving Images

Previously saved positions can be restored, for example:

*Command:* [reset overall](#)

Choosing **File... Save Image** brings up the [Save Image panel](#) for specifying the output file location, name, and type (PNG was used for the [examples](#)), and other options.

Supersampling refers to generating an initial image larger than requested and then sampling it down to the final size. This smooths edges within the image with little effect on file size. The default [Supersample](#) setting of 3x3, used for the [examples](#), is generally sufficient for publication.

The **Image Size** can be specified directly in pixels, or in units of length (when [Use print units](#) is on). If units of length, the output pixel dimensions are calculated using the specified [Print resolution](#). The graphics window can be resized manually or with the command [window size](#).

To make an image for a single-column width of 85 mm (3.346 in) with a resolution of at least 300 dpi, possible approaches include:

- with **Units** set to **pixels**, setting the **Image width** to at least 1004 pixels (3.346 x 300)
- activating [Use print units](#), and with **Units** set to **inches**, setting the **Image width** to at least 3.346 inches and the [Print resolution](#) to 300 dpi
- activating [Use print units](#), and with **Units** set to **millimeters**, setting the **Image width** to at least 85 mm and the [Print resolution](#) to 300 dpi

Clicking **Save** dismisses the dialog and initiates saving the image.

---

meng-at-cgl.ucsf.edu / October 2014

## Image Tutorial: Opened Interface

This tutorial describes how to reveal an interface by splaying it open like a book. The structures are shown as surfaces with interactive shadows. See also: [Intersurf](#), [presets](#), [tips on preparing images](#)

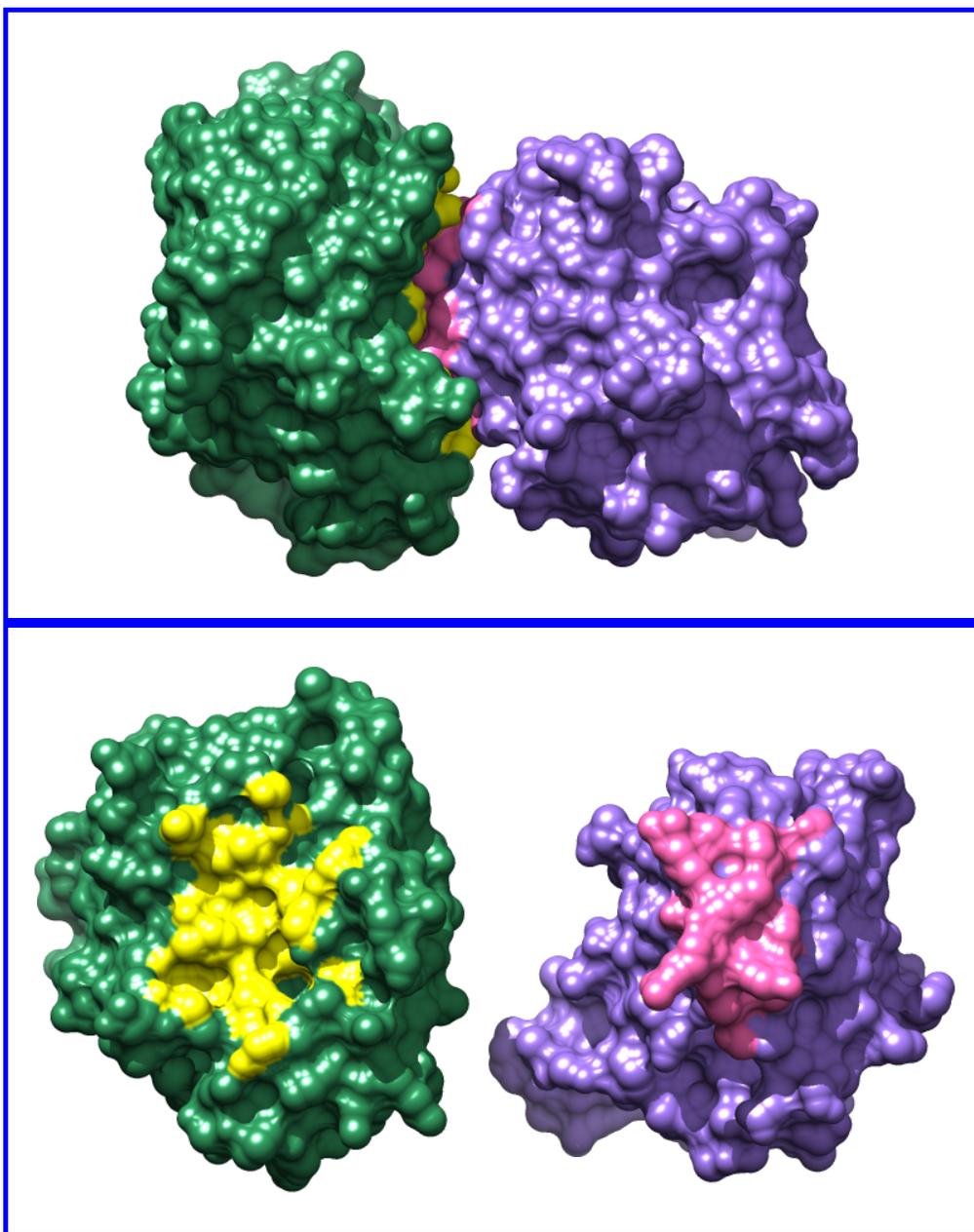
- [Background and Setup](#)
- [Opening/Closing the Interface](#)
- [Coloring the Interface](#)
- [Orientations for Figures](#)
- [Finishing Touches](#)
- [Alternative Colorings](#)

### ← Background and Setup

[Start Chimera](#) and show the [Command Line](#) (for example, with [Favorites... Command Line](#)). Fetch Protein Data Bank entry [1avx](#):

*Command:* [open](#) 1avx

The structure contains porcine pancreatic trypsin, chain A, complexed with a trypsin inhibitor from soybean, chain B. [Move and scale](#) (zoom) with the mouse to see how the two proteins fit together. Resize the window as desired, either by dragging its lower right corner with the mouse or by using the command [window size](#). The window dimensions define the aspect ratio (width:height) of output images, but image resolution can be specified independently when an image is saved. An 800x500-pixel window was used for the example images at right.



In chain B, there is a break in the ribbon and a dashed line where atomic coordinates could not be resolved. Hide this dashed line “pseudobond” and split the model to facilitate moving chains independently:

*Command:* [~longbond](#)

*Command:* [split](#)

The original model #0 is now split into model #0.1 containing chain A and model #0.2 containing chain B. Models are listed in the [Model Panel](#) (under [Favorites](#) in the menu).

Use the all-atoms preset, change to sticks, delete water, and color the two chains differently:

Command: [preset](#) apply int 2  
Command: [repr](#) stick  
Command: [delete](#) solvent  
Command: [color](#) sea green #0.1  
Command: [color](#) medium purple #0.2

Splitting is also one way to indicate that the molecular surface should enclose each chain separately instead of collectively. Show the surface:

Command: [surface](#)

## ← Opening/Closing the Interface

One key to making figures is saving and restoring [positions](#). For example, the following saves the current position and names it **p1**:

Command: [savepos](#) p1

After the view or individual structures have been moved around, the position can be restored with:

Command: [reset](#) p1

... or even restored gradually, over a specified number of frames:

Command: [reset](#) p1 80

You can save/restore multiple positions using different names, and positions are included in subsequently saved [session](#) files. To save your work at any point, save a [session](#) (see the **File** menu).

To open the interface, we will:

1. orient the complex so that trypsin (green) is on the left and the inhibitor (purple) is on the right, with the interface approximately edge-on
2. rotate each protein 90° so that its interface surface is facing the viewer
3. translate the proteins apart horizontally

The first step will be done [with the mouse](#), but we will create command aliases for the other two since they may be done many times starting from different initial orientations. To avoid typing long commands, you can copy text from this page and paste it into the **Command Line**:

Command: [alias](#) ^face-me [set independent](#); [turn](#) y -  
90 model #0.1; [turn](#) y 90 model #0.2; [~set independent](#)  
Command: [alias](#) ^separate [move](#) x \$1 model #0.2; [center](#)

\*\*These aliases assume the complex is oriented so that #0.1 (trypsin, green) is on the left and #0.2 (inhibitor, purple) is on the right.\*\* Aliases can be used in the **Command Line**, and **face-me** also appears in the **Aliases** menu; **separate** does not, because it requires also entering the distance of separation (Å). The “^” symbol means

the alias should only be expanded when it appears at the beginning of a command. In the [independent rotation mode](#), each model rotates about its own center rather than a single collective center.

Before using the aliases, generate initial positions by moving the complex [with the mouse](#). If you have already separated the proteins and/or rotated them independently, first put the complex back together, for example:

*Command:* [reset](#) default

The **default** position is that without user-applied rotations and translations, as when the structures were first opened. For using the aliases, a good initial position of the complex has trypsin on the left and the inhibitor on the right, with the interface approximately edge-on (similar to the top image [above](#), but could be rotated differently about the horizontal axis). When you have what seems to be a good initial position, save it, use the aliases to open the interface, and save the open position. For example:

*Command:* [savepos](#) closed

*Command:* [face-me](#)

*Command:* [separate](#) 18

*Command:* [savepos](#) open

You can start over by resetting to the **closed** position (or whatever you named it). An existing position can be overwritten by saving to the same name. However, don't worry about optimizing the positions for figures yet. The goal at this stage is to simplify opening/closing the interface to evaluate colorings.

## ← Coloring the Interface

Any one of the following three methods could be used to identify and color the binding interface. Remember to put the structures in the bound state (closed-interface position) before running calculations, and if trying different methods or parameter values, recoloring the proteins between trials as in the [setup](#). Previous commands can be re-executed using the [command history](#). (Advanced users may be interested in some additional [aliases](#) used during tutorial development.) Reset to an open-interface position to see the coloring results.

**Method 1** (used for the [figures](#)) identifies atom-atom contacts with [Find Clashes/Contacts](#) or its command equivalent, [findclash](#).

With the proteins in the bound state:

*Command:* [findclash](#) #0.1 test #0.2 intersub true overlap -

1 hb 0 make false select true

*Command:* [namesel](#) contacts

*Command:* [~select](#)

*Command:* [color](#) yellow contacts&#0.1

*Command:* [color](#) hot pink contacts&#0.2

In the [findclash](#) command, the **overlap** and **hb** parameters are adjustable, with values of 0.0-(-1.0) Å and 0.0 Å, respectively, recommended for finding *contacts*. An **overlap** cutoff of -1.0 identifies pairs of atoms with VDW surfaces up to 1.0 Å apart. When the command is instead used to find only *clashes* (unfavorable, too-close contacts), **hb** values > 0.0 help to exclude H-bonding atom pairs. The two sets of atoms are specified with model numbers (e.g. #0.1), but chain identifiers could have been used instead (e.g. :a), and if water had not been deleted, the calculation could have been limited explicitly to protein (e.g. #0.1&protein or :a&protein).

**Method 2** first calculates buried surface area, then uses the resulting per-atom values (assigned as atom [attributes](#)) to identify interface atoms.

With the proteins in the bound state:

Command: [measure buriedArea](#) #0.1 #0.2  
 Command: [color](#) yellow #0.1@/buriedSESArea>1  
 Command: [color](#) hot pink #0.2@/buriedSESArea>1

The total buried area and details of the calculation are given in the [Reply Log](#). Different cutoff values could be used, but in this case, atoms with  $> 1.0 \text{ \AA}^2$  of solvent-excluded surface area buried in the interface are similar to the set of atoms found in the [method 1](#) example. Although solvent, ions, and ligands are not enclosed in the displayed surfaces, the buried-area calculation will include all specified atoms. Thus it is important to specify only the intended atoms; for example, if nonprotein atoms were present:

Command: [measure buriedArea](#) #0.1&protein #0.2&protein

**Method 3** identifies where surfaces are close to one another and does not involve atoms.

With the proteins in the bound state and surfaces shown:

Command: [measure contactArea](#) #0.1 #0.2 2.5 color yellow offset 0  
 Command: [measure contactArea](#) #0.2 #0.1 2.5 color hotpink offset 0

These commands identify where the surfaces are within 2.5 Å of each other. Again, different cutoffs could be used, but 2.5 gave a result roughly similar to the preceding examples. The specifications in the contact-area command (e.g. #0.1) refer to the surface models, which happen to have the same model numbers as the corresponding atomic structures.

## ← Orientations for Figures

Start the [Side View](#) (under **Favorites** in the menu) and place it beside the Chimera window so that you can see the structure from two directions at once. By default, Chimera uses perspective, in which nearer parts of structures appear larger. This enhances 3D perception, but subtly distorts structures as the eye proceeds from the center of the Chimera window towards the edges. The orthographic projection (no perspective) may be better for certain images such as side-by-side comparisons:

Command: [set projection](#) ortho

Continue translating, rotating, and zooming [with the mouse](#), saving and resetting positions, and using the aliases to generate final views. Starting from a position with the intact complex, the **face-me** alias should only be used once, since it applies a 90° rotation to open the interface. However, it is fine to use **separate** multiple times, or scale or translate the view as a whole [using the mouse](#) (carefully!) and/or commands, for example:

Command: [move](#) x -8  
 Command: [scale](#) 1.4  
 Command: [move](#) y 2

It is prudent to save your favorite views as positions, then save a session (and save a session again after incorporating any [finishing touches](#)). You can overwrite an existing position or session by using the same name.

## ← Finishing Touches

Many different visual effects can be applied, and choosing which to apply depends on both personal taste and what the image is meant to illustrate. The example images [above](#) were made with smooth molecular surfaces, a white background, and shadows:

*Command:* [setattr](#) s density 8

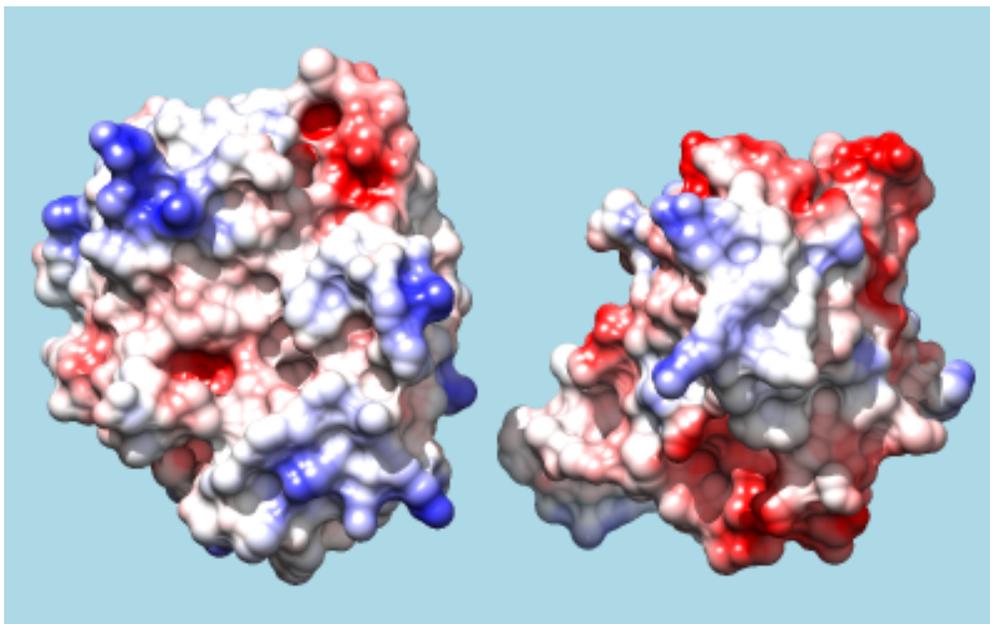
*Command:* [back](#) solid white

*Command:* [set](#) shadows

The vertex density of a molecular surface (default 2) controls the fineness of triangulation, with higher values giving a greater number of smaller triangles and a smoother-looking surface.

## ← Alternative Colorings

Of course, many other combinations of effects and coloring schemes are possible. Just to name a few, surfaces could be colored to show hydrophobic/hydrophilic compatibility or electrostatic complementarity as in the [Surface Properties tutorial](#), conservation within associated sequence alignments as in the [Sequences and Structures tutorial](#), or concavity/convexity as in the [Attributes tutorial, part 2](#). A color key and text could be added, as in the [B-factor Coloring tutorial](#).



If a coloring scheme includes white, it may be helpful to make the background a different color:

*Command:* [back](#) solid light blue

To color by Coulombic electrostatic potential:

*Command:* [coulombic](#) -  
10 red 0 white 10 blue

The result shows a predominantly positive protuberance on the inhibitor (right) interacting with a

primarily negative pocket on the enzyme (left), with the opposite charge pattern on surrounding areas.

To color by per-residue concavity/convexity, open the [Attribute Calculator](#) (Tools... Structure Analysis... Attribute Calculator). Calculate a new attribute named **convexity** for residues using the **Formula**

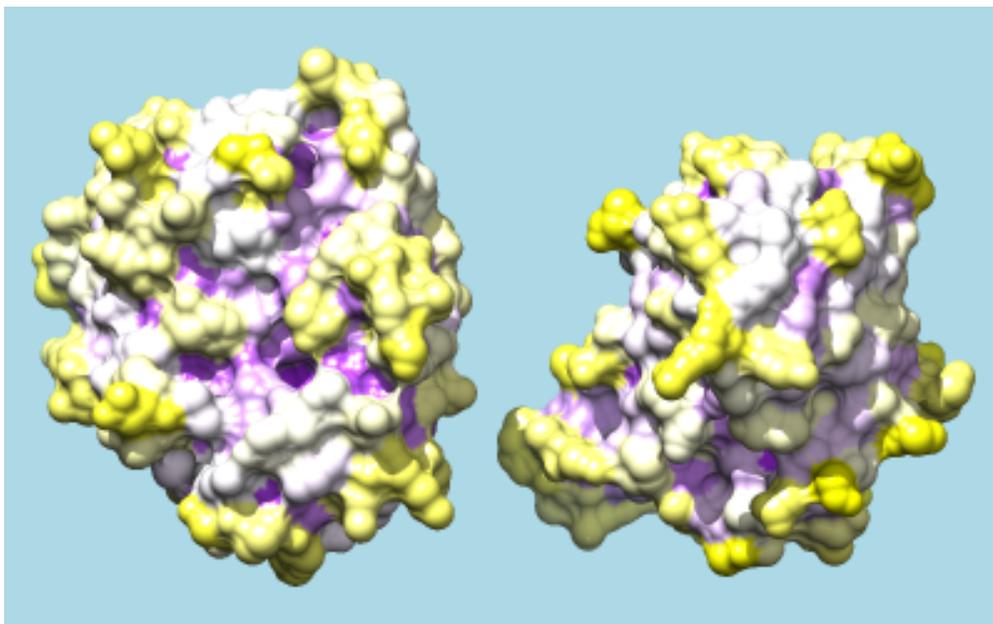
```
residue.areaSAS/residue.  
areaSES
```

Values of **convexity** > 1 represent convex areas, while values < 1 represent concave areas. Click **OK** to

perform the calculation and assignment. A warning message will appear because some residues have an **areaSES** of zero, resulting in a divide-by-zero error. However, just close the warning dialog; values have been assigned correctly to the remaining residues. The coloring command could be something like:

*Command:* [rangecolor](#) convexity min purple 1 white max yellow

The result mainly serves to emphasize what is already evident from the shadowed surfaces: that a knobby protuberance on the inhibitor plugs into a socket on trypsin. The [Attributes tutorial, part 2](#) describes showing per-atom rather than per-residue convexity, but the results are similar.



## Image Tutorial: Similar Binding Sites

### Background

$\beta$ -1,4-glycanase (gold, Protein Data Bank entry [1exp](#)) and cellobiohydrolase I (cyan, Protein Data Bank entry [1cel](#)) have different folds but a similar pattern of amino acids near a glucose residue. This possible example of convergent evolution was identified in Table 5 of:

[Detection of protein three-dimensional side-chain patterns: new examples of convergent evolution.](#) Russell RB. *J Mol Biol.* 1998 Jun 26;279(5):1211-27.

### Image How-To

The recipe here is just an example; there are usually many routes to the same or similar results. Not all of the steps can be done with commands – for example, there is no substitute for interactively finding the best views – but commands for much of the setup below are collected into the Chimera command file [convergent.com](#).

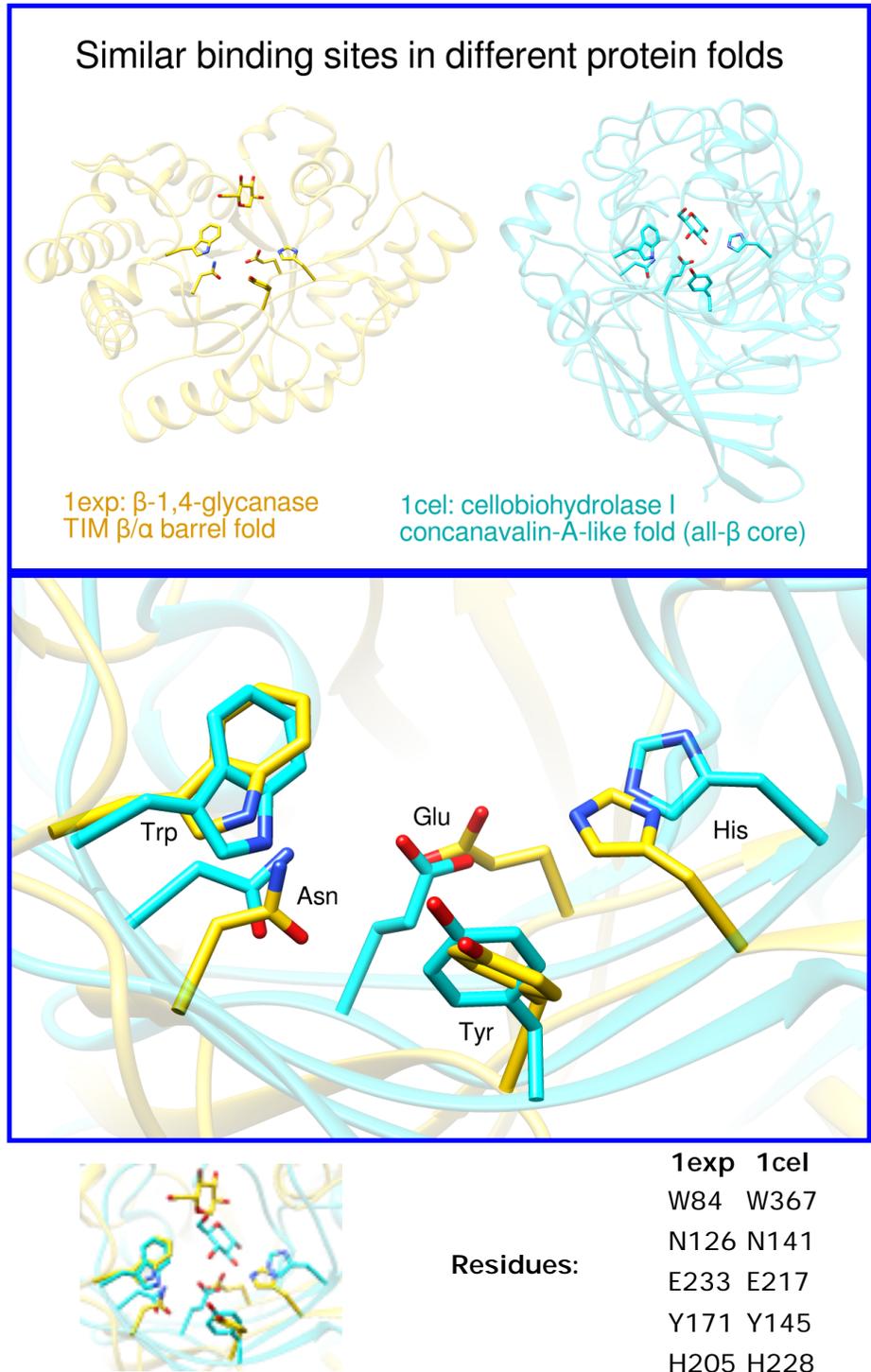
See also: [presets](#), [tips on preparing images](#)

The images show (A) the two proteins side by side, (B) a closeup of the superimposed sites without glucose, and (C) a thumbnail of the superimposed sites. Often it is best to create a set of views that, while different, are consistent or related to each other by only simple transformations. We will generate the view shown in B first and then base the others upon it.

[Start Chimera](#) and enlarge the window as desired. Show the [Command Line](#) (for example, with Favorites... [Command Line](#)). Fetch [1exp](#) and then [1cel](#) from the [Protein Data Bank](#):

Command: [open](#) 1exp

Command: [open](#) 1cel



Command: [focus](#)

Move and scale the structures [using the mouse](#) and the [Side View](#) as you wish throughout the tutorial.

**1cel** includes two copies of the protein, chains A and B; one copy can be deleted:

Command: [delete](#) :.b

Command: [focus](#)

Use the ribbons preset, and for now, hide all atoms and bonds:

Command: [preset](#) apply int 1

Command: [~disp](#)

Set up the basic coloring scheme and white background:

Command: [color](#) gold #0

Command: [color](#) cyan #1

Command: [background](#) solid white

Display the residues identified by Russell and use them to superimpose the structures:

Command: [alias](#) site1 #0:84,126,233,171,205

Command: [alias](#) site2 #1:367,141,217,145,228

Command: [alias](#) both site1 | site2

Command: [disp](#) both

Command: [match](#) iterate 2.0 site2 site1

Command: [focus](#) both

Command: [color](#) byhet

The aliases in the above [match](#) command specify entire residues. The match command requires equal numbers of atoms from the two structures, specified in the same order. In this case, the residues are of the same types (thus containing the same numbers of atoms) and the atoms of each residue will be in the same order because they are automatically sorted by name.

Make the sticks and ribbons smoother by increasing subdivision:

Command: [set subdivision](#) 10

## De-Emphasizing the Ribbons

The display is still quite “busy,” in part because the two proteins have different folds. We will de-emphasize the ribbons by making them transparent and slimmer. Make the ribbons 75% transparent (25% opaque):

Command: [transparency](#) 75,r

The ,r part indicates that only ribbons should be affected, not atoms, bonds, surfaces, *etc.*

By default, transparent surfaces (including objects as well as surfaces proper) appear more opaque when viewed edge-on as compared to face-on. The transparent ribbons will be less obtrusive with *non*-angle-dependent transparency:

Command: [set flatTransparency](#)

The sidechain sticks terminate as rounded “nubbins” inside the ribbon. To make these ends flat instead:

Command: [repr wire @ca](#)

The ribbon dimensions can be adjusted in [Ribbon Style Editor](#). Start that tool (under **Tools... Depiction** in the menu) and adjust the **Ribbon Scaling** values:

2° Structure	Width	Height
<b>Coil</b>	0.20	0.20
<b>Helix</b>	0.60	0.20
<b>Sheet</b>	0.60	0.20
<b>Arrow (base)</b>	1.0	0.20
<b>Arrow (tip)</b>	0.20	0.20

The **Nucleic** values only pertain to DNA and RNA. Click **Apply** to adjust the ribbon scaling (secondary-structure-specific dimensions). Click **Save As...** to name and save the scaling for later use. For example, if the scaling is named **slim**, it could also be applied with:

Command: [ribscale slim](#)

## Generating Positions (Views)

The displayed sidechains are somewhat obscured by other parts of the proteins. Rotate, translate, and scale the structures, and in the [Side View](#), adjust the front and back clipping planes to simplify the view. You may also want to adjust the window dimensions. After generating a good view of the set of displayed residues (such as in [figure B](#)), save the position:

Command: [savepos closeup](#)

It does not have to be perfect; the same command can be used again later to re-save the position after further adjustments. You may also wish to [save the session](#) (see the **File** menu) at this stage and whenever you are pleased with your previous work and want to ensure it will not be lost.

Next, generate and save positions for the other two figures. The intent is to preserve the orientations in the **closeup** position while possibly translating up/down and left/right and changing the scale.

1. position for [figure C](#), closeup with glucose molecules:

Display the glucose residues, both named BGC (**Note**: older versions of these PDB entries used the name GLC instead):

Command: [disp](#) :bgc

If you have changed the view from the **closeup** position, restore it:

Command: [reset](#) closeup

Scale and translate everything (avoiding rotation) so that the glucose molecules are visible, optionally adjust the clipping planes in the [Side View](#), and then save another position:

Command: [savepos](#) closeup2

Scaling can be done [with the mouse](#), or with the command [scale](#), or by moving the small square in the [Side View](#). Translations (up/down, left/right) can be performed [with the mouse](#) or with the command [move](#). If you make a mistake, you can simply revert to the **closeup** position and start over.

2. position for [figure A](#), the two proteins side by side:

Again we wish to leave the orientation of the structures the same as in the **closeup** position, but separate them and zoom out so that all parts are visible. Restore the **closeup** position:

Command: [reset](#) closeup

Zoom out (scale down) and turn off clipping by unchecking the **Clip** option in the [Side View](#). Scale and vertically translate the structures as needed to leave room for a title at the top and other text at the bottom. The proteins are still superimposed, in the middle of the window. Translate one structure to the left and the other to the right:

Command: [move](#) x -25 models #0

Command: [move](#) x 30 models #1

The exact translation distances needed will differ according to your starting position. When a satisfactory side-by-side view has been achieved, save the position:

Command: [savepos](#) sidebyside

## Adding Labels and Saving Scenes

Chimera [scenes](#) are similar to positions but contain more information such as colors, display styles, and which atoms and labels are displayed. If scenes are saved and restored, it will not be necessary to manually display and undisplay the glucose residues and the appropriate labels between saving the different images. However, a drawback of scenes is that they increase session file size.

Start [2D Labels](#) (under **Tools... Utilities**) to annotate the images.

1. [figure B](#):

Command: [reset](#) closeup

Command: [~disp](#) :bgc

and then use [2D Labels](#) to label the residues. Click to start a label and type to enter its contents. Click the [color well](#) to set label color, in this case **black**. The appropriate font size depends on the size of the window; use what you think looks best. Each label can be repositioned by dragging. When done creating the labels for this view, save the scene:

Command: [scene](#) sceneB save

Next, hide all of these labels using the **Shown** checkboxes in the [2D Labels](#) dialog.

## 2. [figure C](#):

Command: [reset](#) closeup2

Command: [disp](#) :bgc

Command: [scene](#) sceneC save

We will not make the window thumbnail-sized until right before saving the image.

## 3. [figure A](#):

Command: [reset](#) sidebyside

and then use [2D Labels](#) to add a **black** title above the structures and color-coded descriptions below them. The label color for the gold structure was first set to gold (by clicking the [color well](#) and entering the name **gold** into the [Color Editor](#)) and then darkened until suitably visible on the white background. The label color for the cyan structure was similarly derived from **cyan**. Symbols such as Greek letters can be inserted using the menu in the [2D Labels](#) dialog. When done creating the labels for this view, save the scene:

Command: [scene](#) sceneA save

Now all of the work other than saving the images is done. You should definitely [save the session](#) (see the **File** menu) at this point; it will include the labels, saved positions, and saved scenes. The images can be saved now or later, after quitting from Chimera, restarting Chimera, and [restoring the session](#).

## Saving the Images

### • [figure A](#):

1. as needed, [start Chimera](#) and [restore the session](#) (for example, using **File... Restore Session**)
2. restore the corresponding scene:

Command: [scene](#) sceneA reset

3. choose **File... Save Image**, specify file location, name and type (**PNG** was used), adjust size as

desired (width 1200 pixels was used), and click **Save**

- [figure B:](#)

1. restore the corresponding scene:

*Command:* [scene](#) sceneB reset

2. choose **File... Save Image**, specify file location, name and type (**PNG** was used), adjust size as desired (width 1200 pixels was used), and click **Save**

- [figure C:](#)

1. restore the corresponding scene:

*Command:* [scene](#) sceneC reset

2. it was decided that this thumbnail image for the [Chimera Image Gallery](#) should be 110 by 87 pixels, a different aspect ratio than the window in the session. Resize the window to the planned thumbnail size, then scale as needed:

*Command:* [windowsize](#) 110 87

*Command:* [scale](#) 1.2

Of course, the scaling could have been done interactively with the mouse, but instead a scale factor was determined empirically (by resetting and trying different values) and written down so that the thumbnail could also be reproduced exactly from the session that had already been saved.

3. choose **File... Save Image**, specify file location, name and type (**PNG** was used), and click **Save**; in this case, the size was kept the same as the Chimera graphics window (width 110 pixels, height 87 pixels)

## Image Tutorial: B-Factor Coloring

The recipe here is just one possibility; there are usually several ways to get the same or similar results. See also: [presets](#), [tips on preparing images](#)

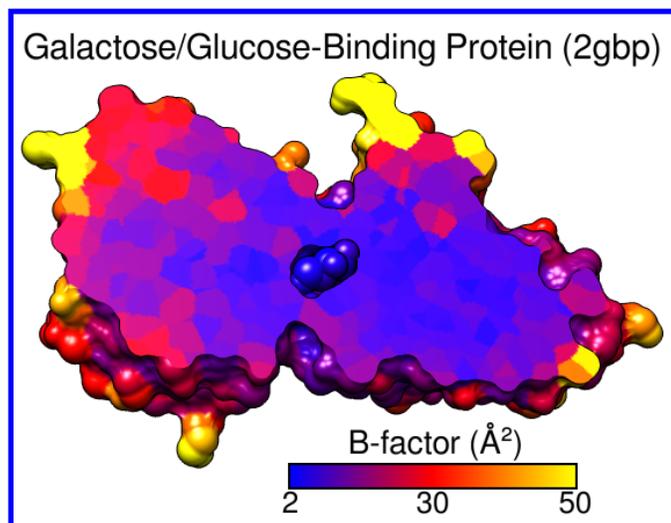
[Start Chimera](#) and show the [Command Line](#) (for example, with **Favorites... Command Line**). Fetch Protein Data Bank entry [2gbp](#):

*Command:* [open](#) 2gbp

Use the “all atoms” preset and delete the water:

*Command:* [preset](#) apply int 2

*Command:* [delete](#) solvent



Resize the window as desired, either by dragging its lower right corner with the mouse or by using the command [window size](#). The window dimensions define the aspect ratio (width:height) of output images, but image resolution (pixel dimensions) can be specified independently when an image is saved.

The [example image](#) was saved with pixel dimensions 573 x 443 from a window of that same size:

*Command:* [window size](#) 573 443

Move and scale the structure [using the mouse](#) and the [Side View](#) as you wish throughout the tutorial.

### Coloring by B-Factor

Start [Render by Attribute](#) (under **Tools... Depiction**) and make sure it is set to show the [attributes](#) of **atoms**. Look in the **Attribute** list to see what is available: **bfactor** and **occupancy**, which were read from the input PDB file. Choose **bfactor**; a histogram of the values will appear, with blue, white and red vertical bars (or [thresholds](#)).

You can control how colors will map to B-factor values by dragging the thresholds along the histogram and/or changing their colors. The **Value** and **Color** are shown for the most recently clicked or moved threshold. The **Value** changes when the threshold is moved, or the position can be changed by entering a value and pressing return. The **Color** can be changed by clicking the square [color well](#) and using the [Color Editor](#). The slider positions in the editor can be adjusted, or a [color name](#) simply entered in the **Color name** field.

To make the [example image](#), a **blue** threshold at value 2, a **red** threshold at 30, and a **yellow** threshold at 50 were used. Clicking **Apply** shows the currently defined color scheme on the structure. Once you have

the desired coloring, click **Create corresponding color key**, but then just **Close** both the [Color Key](#) and [Render by Attribute](#) dialogs. Although the color key has now been set up to match the coloring scheme, we will not draw it in the Chimera window until later.

Alternatively, instead of the graphical interface, a command could have been used to perform the coloring:

*Command:* [rangecolor](#) bfactor key 2 blue 30 red 50 yellow

The **key** keyword sets up the [Color Key](#) dialog; as above, just close that dialog for now. While the command approach is more concise, the [Render by Attribute](#) graphical interface is friendlier in that it does not require previous knowledge of the attribute names, and it provides visual (histogram) information on the value distribution.

The coloring can be limited to certain representations, but in this example, both the atoms and the molecular surface (not yet shown) have been colored by **bfactor** value.

## Clipping the Surface

Display the molecular surface:

*Command:* [surf](#)

Start [Per-Model Clipping](#) (under **Tools... Depiction**). In that dialog, set **Model** to the surface (not **2gbp** itself, which contains the atoms) and turn on **Enable clipping**. Now part of the surface but not the atoms should be chopped away. The planar “cap” is not yet colored by B-factor; that will have to be done in a separate step. Keep the clipping dialog open.

Hide all atoms except for the ligand, glucose:

*Command:* [show](#) ligand

Next, adjust the clipping of the surface to the desired angle and position relative to the structure. In the [Per-Model Clipping](#) dialog, turn on **Adjust clipping with mouse as below** and use the specified mouse buttons to translate and rotate the clipping plane. A transparent green disk will be shown while you are making these adjustments. The mouse button assignments can be changed if you prefer. To return the mouse buttons to their normal functions, turn off **Adjust clipping with mouse as below**. It may be necessary to turn it on and off a few times to position the clipping plane to your satisfaction. Once that has been done, either turn off **Adjust clipping with mouse as below** or just **Close** the clipping dialog to return the mouse buttons to their normal functions.

The atoms were colored by **bfactor** value in the previous section. Select protein atoms and use [Color Zone](#) to make the planar cap match these atoms:

*Command:* [select](#) protein

*Command:* [start](#) Color Zone

This tool is also under **Tools... Depiction** in the menu. It is important to select only the intended atoms so as to avoid coloring the surface by other nearby atoms such as water and ions, which may be undisplayed and easy to forget. Click the **Color** button and increase the **Coloring radius** as needed to color the whole cap. A radius of ~3-4 Å should suffice. Although the only part of the surface that needed coloring was the cap, **Color Zone** applies to the whole surface. Whereas **Render by Attribute** (and most other per-atom coloring of molecular surfaces) uses surface-to-atom associations from the molecular surface calculation, **Color Zone** (which works on both nonmolecular and molecular surfaces) simply associates a surface point with the closest atom center, and may thus change the coloring slightly from the **Render by Attribute** result.

Close the **Color Zone** tool and clear the selection:

*Command:* [~select](#)

The color patches on the planar cap look somewhat fuzzy because of coarse surface triangulation. Start [Surface Capping](#):

*Command:* [start](#) Surface Capping

and change the **Mesh subdivision factor** to **3** to make the triangles smaller and the color patches on the cap more distinct. Close that tool.

## Finishing Touches, Labels, Color Key

Use a publication preset to make the background white, turn on black silhouettes, and generate a smoother molecular surface:

*Command:* [preset](#) apply pub 1

This preset also turns off depth-cueing (fog).

Place and orient the structure for the final image, leaving room for the title and color key. If you want to, save the position:

*Command:* [savepos](#) p1

Then if you accidentally move the structure later, the position can be restored with:

*Command:* [reset](#) p1

If you decide a later position is better, you can save it to a new name, or the same name to overwrite the previously saved position:

*Command:* [savepos](#) p1

Open the [Color Key](#) dialog (under **Tools... Utilities**). In that dialog, set the **Label color** to **black** by clicking

the [color well](#) and using the [Color Editor](#). The appropriate **Font size** depends on the window size; **26** was used with a 573 x 443 window.

Make sure that **Use mouse for key placement** is turned on and then click-drag in the main window to draw the key. It may take a few tries; click-drag to start over, or click on the middle of the key to drag it around. To rotate the structure instead (and/or to prevent erasing the existing color key), turn off **Use mouse for key placement**. After drawing the key to your satisfaction:

- set **Numeric label separation** to **equal** in order to space the value labels 2, 30, and 50 evenly rather than based on their values
- change **Border color** to **black**
- adjust **Border width** to **2**

To add additional text to the image, click the tab to go to the **Labels (2D Labels)** section of the same dialog. Set the **Color** to **black**, then click in the graphics window to start a label and type to enter its contents. Each label can be repositioned by dragging and its font size can be edited to a different value. What font sizes are appropriate depends on the window size; use what you think looks best. The [example image](#) was created using window size 573 x 443, font size 30 for the main title, and font size 28 for the text above the color key. The Å symbol and superscript <sup>2</sup> can be included by using the **Insert symbol** menu. Alternatively, symbols can be copied from text displays (such as a [symbols page](#) or even the preceding sentence!) and pasted into the **Text** area of the [2D Labels](#) dialog. **Close** the dialog as soon as the labels are done, to allow rotating the structure and to avoid accidentally creating more labels.

## Saving Your Work

[Saving a session](#) (see the **File** menu) right before or after saving the image is recommended if you might need to regenerate the same or a similar figure later, say at a different resolution or with modified coloring.

The [image](#) was saved at the same size as the window (573 x 443 pixels) using **File... Save Image** with default settings.

## Image Tutorial: Density Display

### Background

These images show part of Protein Data Bank entry [2fma](#), the Alzheimer's amyloid precursor protein (APP) copper-binding domain, along with its electron density map (2fo-fc) available from the Electron Density Server.

### Image How-To

The recipe here is just an example; there are usually many routes to the same or similar results. See also: [presets](#), [tips on preparing images](#), and on the Chimera web site, the [Image Gallery](#) and [Guide to Volume Display](#)

[Start Chimera](#) and enlarge the window as desired. Show the [Command Line](#) (for example, with [Favorites... Command Line](#)).

Fetch [2fma](#) from the [Protein Data Bank](#), change its model color to white, then apply interactive preset #2 to display all atoms with heteroatom color-coding:

Command: [open](#) 2fma

Command: [modelcol](#) white

Command: [preset](#) apply int 2

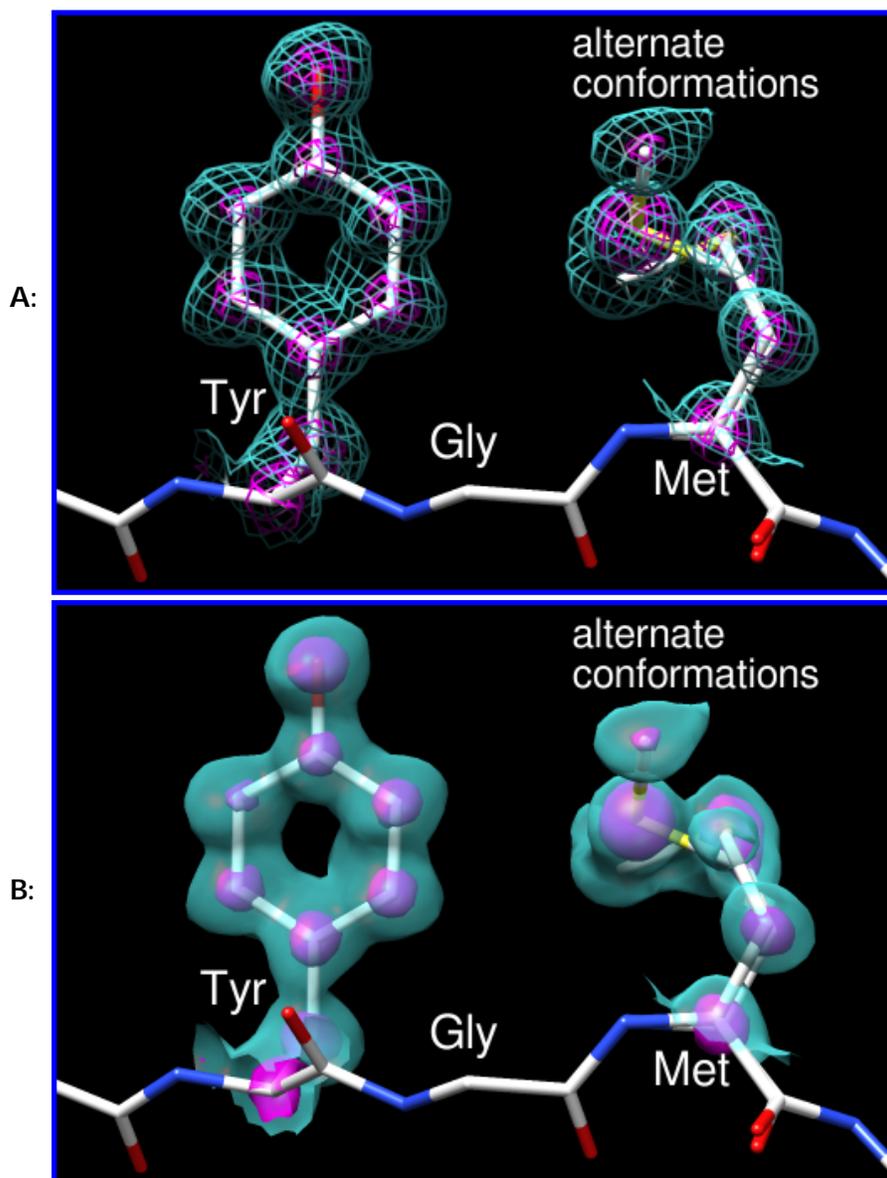
Fetch the density map for this structure from the [Electron Density Server](#):

Command: [open](#) edsID:2fma

Click the little eye icon on the [Volume Viewer](#) dialog to hide the electron density for now. Move and scale [using the mouse](#) and the [Side View](#) as you wish throughout the tutorial.

Since one goal was to show an example of alternate conformations, residues with atoms in alternate conformations were identified by labeling:

Command: [rlab](#) @.a



Met-170 and Glu-183 have alternate conformations; a stretch of residues near Met-170 was chosen for the figure. Show just residues 168-170 along with the backbone of the flanking residues:

Command: [~rlab](#)

Command: [show](#) :168-170

Command: [focus](#)

Command: [disp](#) :167,171@n,ca,c,o

Change to the stick representation and make the sticks thinner:

Command: [rep](#) stick

Command: [setattr](#) m stickScale 0.5

Select the sidechains only of Tyr-168 and Met-170; this selection will be used to limit the density display to a zone.

Command: [sel](#) :168,170 & without CA/C1'

The **without CA/C1'** part is shorthand for the menu entry **Select... Structure... side chain/base... without CA/C1'**. All terminal menu entries under **Select... Structure** can be used as [command-line specifiers](#).

If [Volume Viewer](#) has gotten buried under other windows, one way to resurrect it is by clicking the lightning-

bolt icon  near the bottom of the Chimera window to show the [Rapid Access](#) interface, then clicking the **Volume Viewer** button under "Active Dialogs." In the [Volume Viewer](#) dialog,

1. click the eye icon to turn density display back on
2. choose **Features... Zone** from the menu and click the **Zone** button that appears within the dialog. This limits the density display to a zone near the selected atoms. After that, the selection can be cleared (for example, by Ctrl-clicking in empty space). The zone will continue to center on those atoms, and its **Radius** can be adjusted by moving the slider or entering a new value. A zone radius of 1.96 Å was used for the [figures](#).
3. the vertical bar (*threshold*) on the histogram represents a contour level;
  - o to change the level, drag the threshold to the left or right with the mouse or type a different value into the **Level** field below the histogram; to see more density, move the threshold to the left (use a lower level)
  - o to change the **Color**, click the square [color well](#) below the histogram and use the [Color Editor](#): check the **Opacity** option in that tool to adjust transparency (1-A) as well as color components (R red, G green, B blue); the sliders can be dragged, or values can be entered directly
  - o to add another contour level, Ctrl-click on the histogram; changes in **Level** and **Color** apply to the threshold most recently dragged or clicked
4. choose **Features... Surface and Mesh Options** from the menu to reveal additional display settings

Volume display settings (other than defaults) for [figure A](#):

- **Style** mesh
- two contour levels:
  1. **Level** 0.426, **Color** (RGBA) 0.36 1.0 1.0 0.5
  2. **Level** 2.06, **Color** (RGBA) 1.0 0.0 1.0 1.0 (**magenta**)

- turned on **Smooth mesh lines**
- **Mesh line thickness 1.5 pixels**

By default, Chimera shows only the topmost layer of transparency. This is recommended in most situations because it simplifies the view and serves to de-emphasize the transparent parts. However, for both of these figures, single-layer transparency was turned off so that multiple transparent layers would be visible:

*Command:* [~set singleLayer](#)

The setting can be toggled (**set singleLayer** and then **~set** again, as above) if you want to review the difference.

For [figure B](#), the same contour levels and RGB colors are used, but different transparencies and style. Volume display settings (other than defaults):

- **Style surface**
- two contour levels:
  1. **Level 0.426, Color (RGBA) 0.36 1.0 1.0 0.4**
  2. **Level 2.06, Color (RGBA) 1.0 0.0 1.0 0.6**
- turned on **Surface smoothing iterations 2 factor 0.3**

Use [2D Labels](#) (under **Tools... Utilities**) to add label text. The labels are the same for the two [figures](#). Click in the graphics window to start a label and type to enter its contents. Each label can be repositioned by dragging. Each label's color can be adjusted by clicking the [color well](#) and using the [Color Editor](#), and its font size can be edited to a different value. What font sizes are appropriate depends on the window size; use what you think looks best. In this case, the labels are all **white**, and the residue labels are in a slightly larger font than "alternate conformations." The latter was typed with return (Enter) between the two words so that they would appear on different lines.

The [figures](#) were saved as PNG files with the same pixel dimensions as the Chimera window by using **File... Save Image** with default settings.

[Saving a session](#) (see the **File** menu) right before or after saving the image is recommended if you might need to regenerate the same or a similar figure later, say at a different resolution or with modified coloring. A [position](#) named **session-start** is created automatically when a session is restored. This is useful when you [restore a session](#) and move things around but then want to restore its initial position:

*Command:* [reset session-start](#)

## Image Tutorial: Pipes and Planks

This tutorial describes showing protein helices as “pipes” (cylinders) and strands as “planks” (rectangular boxes) with [PipesAndPlanks](#), and adjusting the lighting to decrease contrast. See also: [Axes/Planes/Centroids](#), [presets](#), [tips on preparing images](#)

[Start Chimera](#) and show the [Command Line](#) (for example, with **Favorites... Command Line**). Fetch Protein Data Bank entry [2mnr](#):

*Command:* [open](#) 2mnr

The structure is the enzyme mandelate racemase, containing an N-terminal  $\alpha+\beta$  domain and a C-terminal  $\beta/\alpha$ -barrel domain. [Move and scale](#) the structure as desired throughout the tutorial.

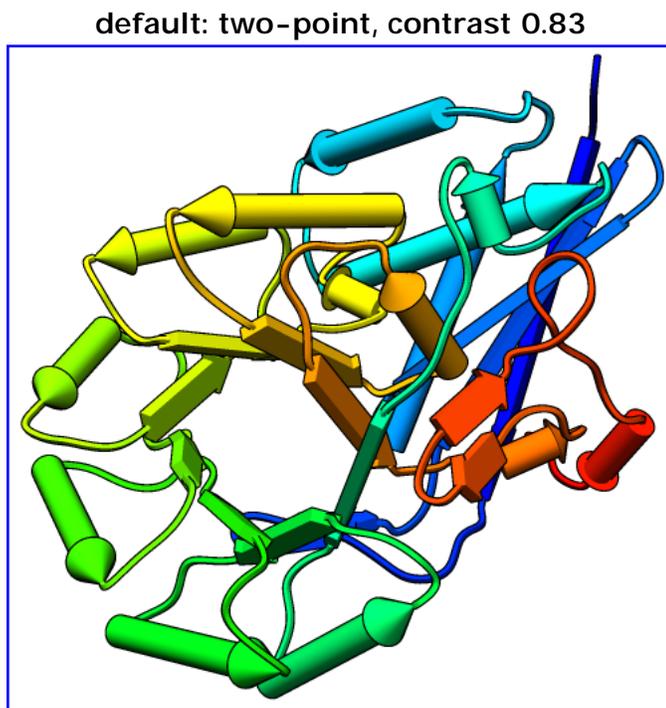
The window can be resized by dragging its lower right corner with the mouse or by using a command, for example:

*Command:* [windowsize](#) 500 500

Rainbow-color-code the protein chain from blue at the N-terminus to red at the C-terminus:

*Command:* [rainbow](#)

Start [PipesAndPlanks](#) (under **Tools... Depiction** in the menu), and click **Apply** to generate the pipes-and-planks representation with default settings. **Helix color**, **Strand color**, and **Coil color** settings of “No color” indicate that each pipe, plank, and stretch of coil should be colored using the ribbon color of its first residue.



The pipes-and-planks representation is shown in addition to the ribbon, rather than replacing it. Hide the ribbon and atoms:

*Command:* [~ribbon](#)

*Command:* [~disp](#)

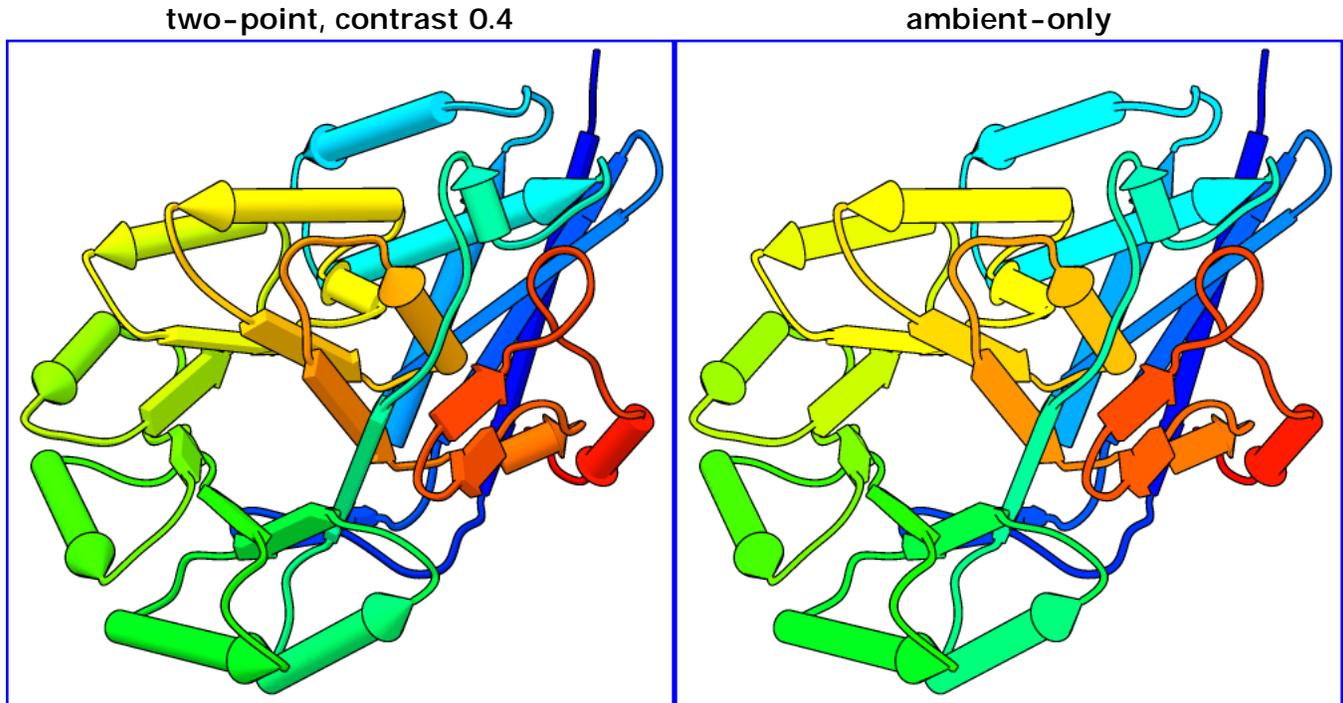
Apply publication preset 1, which sets the background to white and turns on black outlines (silhouettes):

*Command:* [preset](#) apply pub 1

The first example image was saved with the current settings, including default lighting.

## Decreasing the Contrast

While shading can enhance the perception of depth, shapes, and orientations, it can also make colors muddy and images harder to interpret. One way to lighten the shading is to decrease the contrast, for example:



Command: [light](#) contrast 0.4

The default lighting mode, **two-point**, includes two directional lights and ambient (nondirectional) light. Decreasing the contrast increases the ambient light and gives a flatter appearance.

A yet flatter appearance, similar to a line drawing, can be achieved by changing the lighting mode to ambient-only:

Command: [light](#) mode ambient

The lighting mode and contrast can also be adjusted in the [Lighting](#) tool (under **Tools... Viewing Controls** in the menu), along with other related settings.

## Additional Outlines

In the ambient-only lighting mode, some of the object contours are lost. However, more black outlines can be added using **edge color** settings in the [PipesAndPlanks](#) dialog. The last example image was saved in the ambient-only lighting mode after setting the **Helix edge color** and **Strand edge color** to black, but leaving the **Coil edge color** as "No color."

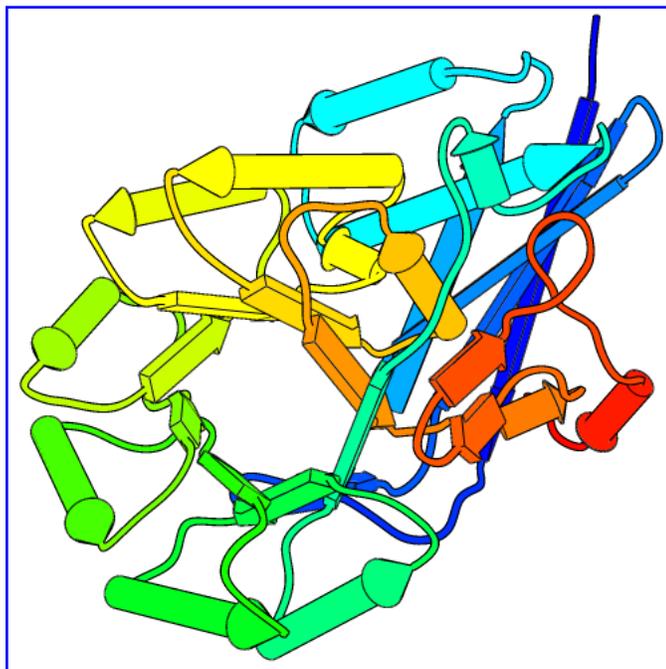
ambient-only, more outlines

A color setting can be changed by clicking the square [color well](#) and using the resulting [Color Editor](#). The current

color can be changed by moving the sliders or by entering a [color name](#) (for example, **black**) in the **Color name** field. The active color well should change concurrently, but drag-and-drop from the **Color Editor** to a color well or between two color wells will also work.

Click **Apply** in the [PipesAndPlanks](#) dialog to apply your changes. The edges may look quite fuzzy in the Chimera window, but image [supersampling](#) (on by default) will make them smoother in saved images. All images in this tutorial were saved using **File... Save Image** with default settings.

Try other changes if you wish; adjustable settings include pipe radius, plank width, and whether to show helix and strand N→C directionality with arrowheads.



## Structure Analysis and Comparison Tutorial

This tutorial includes binding site analysis and comparison of related structures by superposition and morphing. Internet connectivity is required to fetch the structures **3w7f** and **2zco**.

- [Background and Setup](#)
- [Distances, H-Bonds, Contacts](#)
- [Angles, Rotamers, Clashes](#)
- [Surfaces and Attributes](#)
- [Superposition and Morphing](#)

### ← Background and Setup

The pathogenic organism *Staphylococcus aureus* makes a pigment called staphyloxanthin. The pigment imparts a golden color (hence *aureus*), but more importantly, contributes to virulence by protecting the bacteria from being killed by the host immune system. The *S. aureus* enzyme CrtM may be a good drug target because it catalyzes a key step in staphyloxanthin synthesis:

[A cholesterol biosynthesis inhibitor blocks \*Staphylococcus aureus\* virulence.](#) Liu CI, Liu GY, Song Y, Yin F, Hensler ME, Jeng WY, Nizet V, Wang AH, Oldfield E. *Science*. 2008 Mar 7;319(5868):1391-4.

We will view and compare different structures of this enzyme.

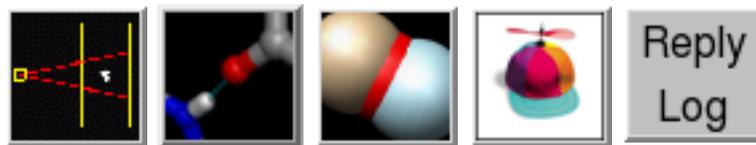
On **Windows/Mac**, click the **chimera** icon; on **UNIX**, start Chimera from the system prompt:

```
unix: chimera
```

A splash screen will appear, to be replaced in a few seconds by the main Chimera [graphics window](#) or [Rapid Access](#) interface (it does not matter which, the following instructions will work with either). If you like, resize the Chimera window by dragging its lower right corner.

Show the [Command Line](#) (Tools... General Controls... Command Line). Choose Favorites... Add to Favorites/Toolbar to place some icons on the toolbar. This opens the Preferences, set to Category: [Tools](#). In the **On Toolbar** column, check the boxes for:

- [Side View](#) (under Viewing Controls)
- [FindHBond](#) (Structure Analysis)
- [Find Clashes/Contacts](#) (Structure Analysis)
- [Rotamers](#) (Structure Editing)
- [Reply Log](#) (Utilities, the last section)



If you want these icons to appear in later uses of Chimera, click **Save** before closing the preferences.

Fetch a structure from the [Protein Data Bank](#) and use the ribbons [preset](#):

*Command:* [open](#) 3w7f

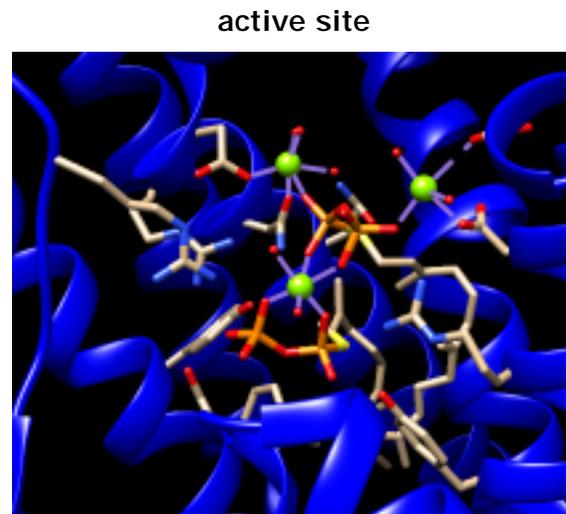
*Menu:* Presets... Interactive 1 (ribbons)

This preset displays the two protein chains (A and B) as red and blue ribbons; bound molecules and nearby sidechains are shown as sticks. The two chains are two copies of the enzyme. Delete chain B:

*Command:* [delete](#) :.b

Move and scale structures [using the mouse](#) and [Side View](#) as desired throughout the tutorial.

The enzyme combines two 15-carbon molecules of farnesyl pyrophosphate to form a 30-carbon lipid. This structure contains farnesyl *thi*opyrophosphate, which differs from the substrate by having a sulfur in the place of one oxygen. Sulfur atoms are shown in yellow, phosphorus orange, oxygen red, and nitrogen blue. Label the ligand residues:



*Command:* [rlabel](#) ligand

In this structure, the farnesyl thiopyrophosphate molecules are named FPS. Three magnesium ions help to offset the negative charges on the phosphates. The ions are shown as greenish spheres; clicking into the Chimera window and hovering the mouse cursor over each shows information in a pop-up balloon. Metal coordination bonds from FPS, water, and the protein are shown with dashed purple lines. Lines drawn to indicate interactions other than covalent bonds are called *pseudobonds*. Hovering the cursor over a pseudobond or bond shows balloon information about its end atoms and length.

Delete the water, label residues with displayed sidechains, and place the labels near  $\alpha$ -carbons instead of residue centroids:

*Command:* [del](#) solvent

*Command:* [rlab](#) @/display

*Command:* [setattr](#) m residueLabelPos 2

Water could be included in the various analyses, but is removed here to simplify the tutorial.

## ← Distances, H-Bonds, Contacts

It looks like several sidechains could be donating hydrogen bonds to phosphate oxygens. (Although the structure does not include hydrogens, we know they are there!)

One of the displayed residues is Ser 21. To measure a distance:

1. Ctrl-click to pick the sidechain oxygen of Ser 21
2. Shift-Ctrl-doubleclick on the nearest phosphate oxygen
3. click **Show Distance** in the resulting context menu

Similarly, measure the distance between the sidechain oxygen of Tyr 248 and the same phosphate oxygen.

The distances seem consistent with hydrogen bonds. However, rather than measuring many distances and trying to remember the appropriate hydrogen-bonding distances for different types of atoms, just use [FindHBond](#). We will limit the search to H-bonds involving the FPS residues:

1. Select the FPS residues (for example, with **Select... Residue... FPS** in the menu).
2. Start [FindHBond](#) by clicking its [icon](#): 
3. In that dialog, turn on the options:
  - o **Only find H-bonds with at least one end selected**
  - o **Write information to reply log**
4. Set **Line width** to 3.
5. Click **OK**.

The H-bonds are shown as pseudobonds of the specified color and line width. Both measured distances were found to be consistent with hydrogen bonding. Details can be viewed in the **Reply Log**, which can be opened by clicking its [icon](#). (In your own work, you may prefer to write the information to a file instead.) Remove the “hydrogen bond” pseudobonds:

*Command:* [~hbond](#)

[Find Clashes/Contacts](#) is similar to [FindHBond](#) but can also identify nonpolar interactions:

- *clashes* - unfavorable interactions where atoms are too close together; close contacts
- *contacts* - all kinds of direct interactions: polar and nonpolar, favorable and unfavorable (including clashes)

We will identify contacts of the FPS residues:

1. Start [Find Clashes/Contacts](#) by clicking its [icon](#): 
2. With the FPS residues still selected, click **Designate**. Now, 48 atoms should be designated for checking against **all other atoms**.
3. Set the **Clash/Contact Parameters** to the default **contact** criteria (by clicking the button marked **contact**).
4. Set the **Treatment of Clash/Contact Atoms** to:
  - o **Select**
  - o **If endpoint atom hidden, show endpoint residue**
  - o **Write information to reply log**
  - o (turn off any other treatment options)
5. Click **Apply**.

In the **Reply Log**, the atom-atom contacts are listed in order of decreasing VDW *overlap*, the amount of intersection of atomic van der Waals spheres: 0 if just touching, positive if interpenetrating, negative if separated by space. Slightly separated spheres are still considered contacts by default. The distances between atomic centers are also given.

One might simply want a list of the interacting residues rather than the details of each atomic contact. A list of the selected residues can be saved. First, deselect the ligand residues and ions, leaving the protein residues selected:

Command: `~sel ligand`

Command: `~sel ions`

Open the [Selection Inspector](#) by clicking the magnifying glass icon  near the bottom right corner of the main window. It reports that 26 residues are selected. Click its **Write List...** button (or choose **Actions... Write List** from the main menu). In the resulting dialog, indicate that **selected residues** should be written. Click **Log** to write the list to the **Reply Log** instead of to a file.

## ← Angles, Rotamers, Clashes

Amino acid torsion angle values can be viewed in the [Selection Inspector](#). Focus on Tyr 248:

Command: `focus :248`

Ctrl-click to pick any atom or bond in Tyr 248. This will replace (clear) any pre-existing selection. Open the [Selection Inspector](#) again if it is not already open. Inspecting **Residue** (instead of **Atom**) shows that the sidechain angles chi1 and chi2 in this residue are approximately  $-107^\circ$  and  $-142^\circ$ , respectively. The backbone angles phi and psi are also given. **Close the Selection Inspector.**

We will set up clash checking and rotate the sidechain interactively.

1. Display residues within 4 Å of Tyr 248:

Command: `disp :248 z<4`

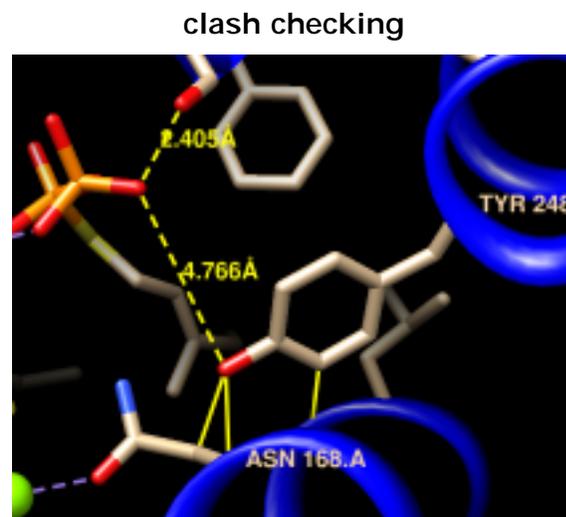
2. Click into the main graphics window, then press the up arrow key on the keyboard to promote the atom or bond selection within Tyr 248 to the whole residue.

3. Start [Find Clashes/Contacts](#) by clicking its [icon](#): 

4. In that dialog, click **Designate**. The 12 atoms of Tyr 248 should be designated for checking against **all other atoms**.

5. Set the **Clash/Contact Parameters** to the default **clash** criteria (by clicking the button marked **clash**).

6. Set the **Treatment of Clash/Contact Atoms** to **Draw pseudobonds**; turn off any other treatment options.



## 7. Set the **Frequency of Checking** to **after relative motions**.

No clashes are found in the current position. To rotate the sidechain of Tyr 248 interactively:

1. Ctrl-doubleclick its CA-CB bond (the stick segment right next to the ribbon)
2. click **Rotate Bond** in the resulting menu

The rotatable bond will be listed in the [Adjust Torsions](#) dialog. In that dialog, if the **Near** atom is set to **N**, the value reported is the chi1 angle. There are several ways to rotate a bond, including dragging the pointer on the dial, clicking the black arrowheads, and editing the angle value directly. If the dialog becomes obscured by other windows, it can be raised again from the menu (**Tools... Structure Editing... Adjust Torsions**).

Use whichever method you prefer to rotate the bond. As the sidechain moves, yellow pseudobonds show any clashes, and the distance monitor updates automatically. The sidechain is fairly constrained; only chi1 angles of approximately  $-(100-120)^\circ$  avoid clashes if only that bond is rotated. The sidechain can be frozen in a new position, but for tutorial purposes, simply restore it to its original position: in the **Adjust Torsions** dialog, click the entry under **Bond** to show a menu. In that menu, choose **Revert** (move back to original position) and then **Deactivate** (make the bond no longer rotatable). **Close** the torsion dialog.

Click the **Find Clashes/Contacts** [icon](#) to bring the dialog to the front and **Close** it to halt clash checking.

Next, we will compare the conformation of Tyr 248 in the structure to tyrosine rotamers from a library. This will indicate whether the sidechain is in a frequently observed conformation and whether other conformations might also fit into the structure.

With part of Tyr 248 still selected (if not, Ctrl-click to select any of its atoms or bonds), start [Rotamers](#)

by clicking its [icon](#) . Click **OK** to show **TYR** rotamers from the [Dunbrack backbone-dependent library](#). The rotamers are shown in the wire representation and listed in another dialog.

Clicking a line in the rotamer dialog displays just that rotamer. Clicking the lines one by one shows that none of the rotamers match the conformation in the structure.

The rotamer dialog reports chi1 and chi2 angles and probabilities from the library based on the residue's backbone conformation. Other than the backbone angles, the probabilities do not take into account any interactions specific to this structure. However, the rotamer dialog is integrated with clash and H-bond detection. Choose **Columns... Add... Clashes** and click **OK** to use the default settings; repeat with **Columns... Add... H-Bonds**.

The new columns show that each rotamer forms several clashes but no hydrogen bonds. We showed above that Tyr 248 H-bonds with the ligand and avoids clashes, which may compensate for its nonrotameric (presumably strained) conformation.

A sidechain can be replaced with a chosen rotamer. With a single rotamer shown, either click **OK** in the rotamer dialog to replace the sidechain with that rotamer and remove the others, or simply **Close** the dialog to remove the rotamers without replacing the sidechain. Although not done in this case, rotamers can be shown for a different amino acid type than in the structure, allowing for virtual mutations.

Unlike Tyr 248, Tyr 41 closely resembles the highest-probability tyrosine rotamer given its backbone angles. If you like, focus on Tyr 41, select one of its atoms or bonds, and use [Rotamers](#) to show and evaluate rotamers of tyrosine or some other amino acid at that position.

Clear the selection, unlabel the residues, zoom back out, and remove the distance pseudobonds:

Command: [~sel](#)

Command: [~rlab](#)

Command: [focus](#)

Command: [~dist](#)

## ← Surfaces and Attributes

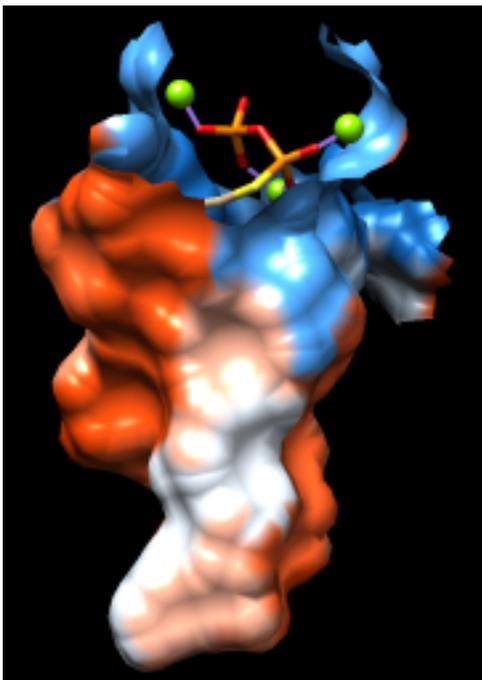
FPS and the natural substrate both have a highly polar/charged “head” and a long hydrocarbon “tail.” The enzyme pocket is very large and deep, as needed to accommodate two of these molecules.

A surface representation is best for showing the shape of the pocket. Choose **Presets... Interactive 3 (hydrophobicity surface)** from the menu to display a molecular surface color-coded by amino acid hydrophobicity. This [preset](#) colors the surface ranging from dodger blue for the most polar residues to orange red for the most hydrophobic, with white in between.

As one might expect, the mouth of the pocket near the phosphates and ions is mostly polar, while the rest of the pocket is largely hydrophobic. However, the pocket is so deep that it is hard to see when the whole surface is shown. Restricting the surface display to the pocket region allows it to be viewed from the outside, as shown in the [figure](#).

Hide the ribbon and the protein atoms, then show only surface near the ligand:

pocket surface



Command: [~ribbon](#)

Command: [~disp protein](#)

Command: [sop zone #0 ligand 5.0 max 1](#)

The [sop](#) (“surface operations”) command limits the display of surface #0 to a 5.0-Å zone around the ligand and shows only the largest resulting fragment. Without **max 1**, a few additional disconnected shards of surface would be shown. If you like, try different cutoff values; the [figure](#) shows results for a cutoff of 4.7 Å.

Amino acid hydrophobicity is one example of an *attribute*. Another attribute commonly used in structure analysis is atomic B-factor. B-factor values indicate which parts of the structure are more or less flexible or disordered.

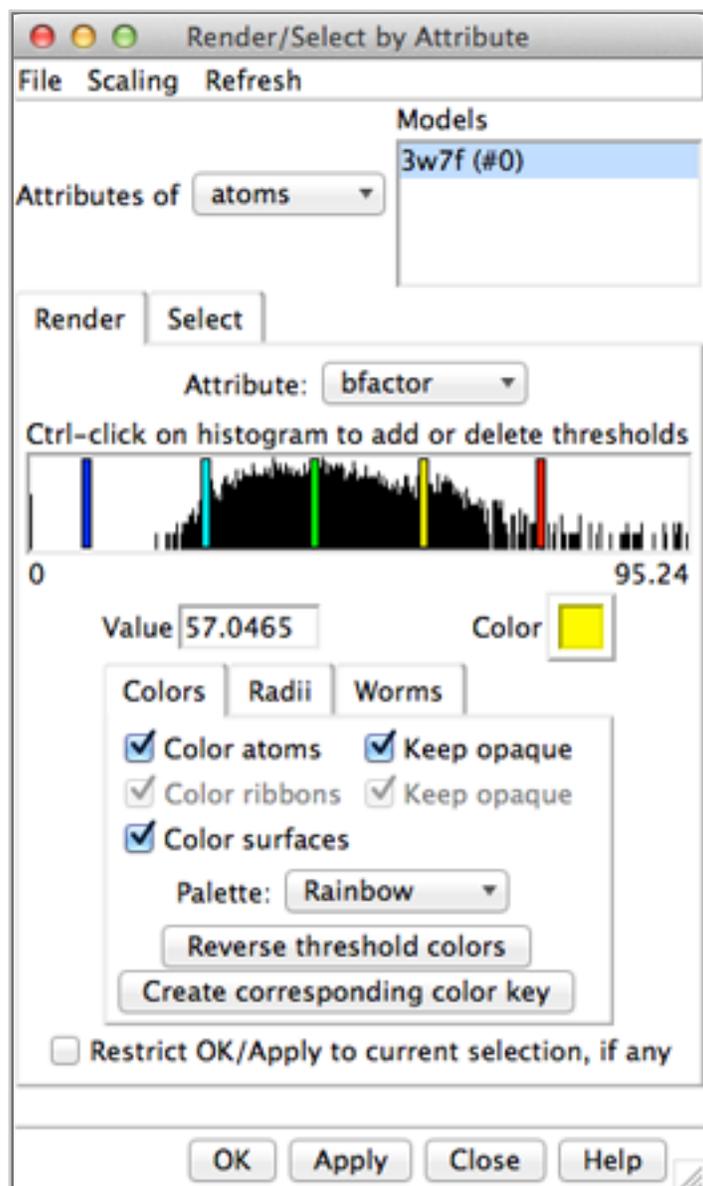
Display all atoms by choosing **Presets... Interactive 2 (all atoms)** from the menu. Now the ligand atoms are shown as spheres and the protein as wires. Show the protein as sticks:

Command: [repr](#) stick protein

The [Render by Attribute](#) tool shows attribute values with colors and other visual cues. Choose **Tools...**

**Depiction... Render by Attribute** from the menu to start this tool. The dialog is set to **atoms** when it appears. Choose **bfactor** from the attribute menu. The B-factor values are shown in a histogram, and the vertical bars on the histogram define a color mapping. These bars or *thresholds* can be dragged along the histogram with the mouse, and their colors can be changed. Clicking the square [color well](#) allows changing the **Color** of the threshold that was most recently clicked or moved. The colors can also be changed collectively by choosing a different **Palette** or clicking the button to **Reverse threshold colors**. (The [figure](#) shows the dialog after a user has added more thresholds, chosen the **Rainbow** palette, and then reversed the colors.) Adjust the color mapping as desired and **Apply** it to the structure. The lowest B-factors are in the protein core, the highest in a loop over the active site and the C-terminus on the opposite side. If any sidechain has been replaced with a rotamer, the new atoms will not have B-factor values, but they can be assigned a **No-value color**.

In the **Render** dialog, switch from **atoms** to **residues** and show the histogram for the attribute **kdHydrophobicity**, the [Kyte-Doolittle hydrophobicity](#) of amino acids. This is the same attribute as used by the surface [preset](#), but different coloring schemes can be applied. Larger positive values correspond to more hydrophobic residues, negative values to hydrophilic residues. The **No-value color** pertains to residues without Kyte-Doolittle hydrophobicities, namely any that are not amino acids (such as the ligands in this structure). The colors can be changed as described above. Apply coloring by hydrophobicity if you like, then **Close** the dialog.



## ← Superposition and Morphing

Comparing different structures of a protein is another way to evaluate flexibility. We have been viewing a structure bound to substrate analogs, **3w7f**. A structure of the same enzyme without ligands is also available. Fetch the “empty” structure and apply the ribbons preset:

Command: [open](#) 2zco

Command: [preset](#) apply int 1

Now the first structure is tan and the new structure is sky blue, as shown in the [Model Panel](#) (**Tools... General Controls... Model Panel**).

The structures are in completely different positions, so the next step is to superimpose them. This is easily accomplished with [MatchMaker](#) or its [command equivalent](#):

*Command:* `mm #0 #1 show true`

**MatchMaker** first generates a sequence alignment using residue types and secondary structure (tries to align helix with helix and strand with strand), then fits the sequence-aligned residues in 3D by pairing their CA atoms. The **show true** option displays the sequence alignment. By default, the fit is iterated so that far-apart pairs are not included in the final match. This allows the most similar parts to superimpose more tightly and the dissimilar parts to stand out. Final match statistics are reported in the **Reply Log**, and the pairs used in the final fit are shown with light orange boxes on the sequence alignment. The RMSD histogram above the sequences shows spatial variability, calculated as the root-mean-square deviation among the CA atoms of residues associated with each column. In this case (only two structures), the RMSD value is the same as the CA-CA distance.

The structures are highly similar except for the loop at approximately residues 52-56; you can see residue numbers by clicking into the sequence alignment window and hovering the cursor over the corresponding one-letter codes. Click any part of the light orange boxes to select all the residues used in the final fit; these form the common core, the most conformationally similar parts of the structures. The selection can be inverted if you want to do something to the less similar parts, for example:

*Command:* `sel invert`

*Command:* `disp sel & protein`

*Command:* `~sel`

Morphing involves calculating a series of intermediate, interpolated structures between the original input structures. In Chimera, the series of structures is treated as a trajectory that can be replayed, saved to a coordinate file, or saved as a movie using [MD Movie](#).

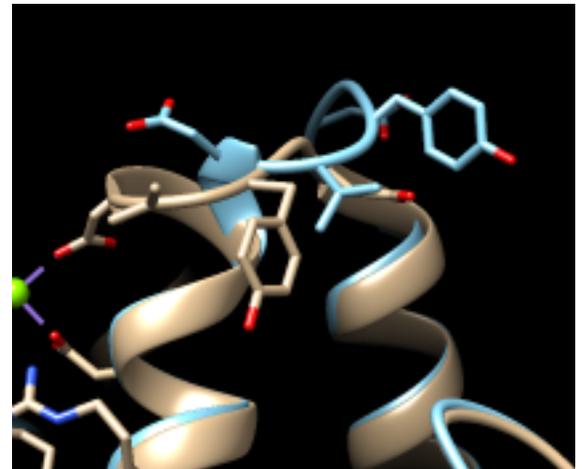
Start the [morphing tool](#):

*Menu:* **Tools... Structure Comparison... Morph Conformations**

Click **Add...** and in the resulting list of models, doubleclick to choose #0, #1, and #0 again, corresponding to a morph trajectory from the ligand-bound structure to the empty structure and back. **Close** the model list. In the main **Morph Conformations** dialog, set the **Action on Create** to **hide Conformations**, and then click **Create**.

The progress of the calculation is reported in the [status line](#). When all the intermediate structures have been calculated, the input structures are hidden, the trajectory is opened as a third model (#2), and the [MD Movie](#) tool appears. The trajectory can be played continuously or one step at a time by clicking buttons on the tool, or a frame number can be entered directly. If the **MD Movie** playback dialog becomes obscured by other windows, it can be raised using its [instance in the Tools menu](#) (near the bottom of the menu, below the horizontal line).

different loop conformations



The initial display of the morph trajectory is as ribbons rather than all atoms, but all atoms that are present in both input structures are also present in the morph. Their display can be controlled just as in any other structure, for example:

*Command:* [disp](#) #2:52-56

Ligand residues were only present in one of the structures, so they are not included in the morph trajectory. However, part or all of the original structures can be displayed along with the trajectory. Display of the other models can be re-enabled by checking the **S**(hown) boxes in the [Model Panel](#) (**Tools... General Controls... Model Panel**) or with a command, for example:

*Command:* [modeldisp](#) #0

Then, to show only the ligand residues and ions from that model:

*Command:* [~ribbon](#) #0

*Command:* [show](#) :fps,mg

In this case, the secondary structure does not change much between the input structures. When conformational differences are larger and ribbons will be displayed, one may want to re-evaluate the secondary structure at each step of the trajectory. That can be done by putting the command [ksdssp](#) in a [per-frame Chimera script](#) in [MD Movie](#). Such scripts also allow performing actions (such as showing/hiding a ligand) at specific steps in the trajectory.

The [Morph Conformations](#) page lists a few additional [example systems](#).

When finished enjoying the morph trajectory, choose **File... Quit** from the menu to exit from Chimera.

## Attributes Tutorial

This tutorial demonstrates many uses of [attributes](#), or properties of atoms, residues, and molecule models. Attributes can be numerical (such as atomic number), boolean (e.g., whether a residue is in a helix), or string-valued (such as [atom type](#)). Attribute values can be rendered visually and used in [selection](#) and command-line [atom specification](#). See also the [Mapping Sequence Conservation](#) tutorial on the Chimera web site.

[Part 1](#) uses a leucine zipper structure, and [Part 2](#) uses the structure of a GTP-binding protein.

---

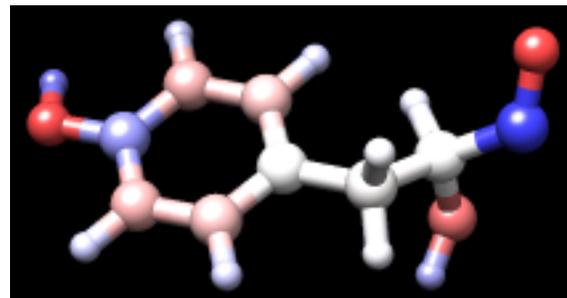
### Attributes, Part 1 - Leucine Zipper

On **Windows/Mac**, click the **chimera** icon; on **UNIX**, start Chimera from the system prompt:

```
unix: chimera
```

A splash screen will appear, to be replaced in a few seconds by the main Chimera [graphics window](#) or [Rapid Access](#) interface (it does not matter which, the following instructions will work with either). If you like, resize the Chimera window by dragging its lower right corner. Show the [Command Line](#) by choosing it from the **Favorites** menu, and optionally, the [Side View](#) for clipping and scaling.

tyrosine colored by charge



If you have internet connectivity, structures can be obtained directly from the [Protein Data Bank](#). Choose **File... Fetch by ID** from the Chimera menu and use the resulting dialog to fetch **1zik** from the **PDB**. If you do not have internet connectivity, [download](#) the file [1zik.pdb](#) included with this tutorial and use **File... Open** to open it.

The structure is a leucine zipper formed by two peptides. Apply the “all atoms” [preset](#), change to stick representation, and hide water:

```
Command: preset apply interactive 2
```

```
Command: repr stick
```

```
Command: ~disp solvent
```

[Move and scale](#) the structure as desired throughout the tutorial.

Open the [Render by Attribute](#) tool (**Tools... Structure Analysis... Render by Attribute**). Make sure it is set to show the attributes of **atoms**. Look in the **Attribute** list to see the attributes available: **bfactor** and **occupancy**, which were read from the input PDB file. Choose **bfactor**; a histogram of the values will

appear, with colored vertical bars (or [thresholds](#)) that control how the values will be mapped to color. Click **Apply** to color the atoms by B-factor. As expected, the atoms with higher B-factors are on the outside of the structure.

The default coloring scheme is blue to white to red, but the color mapping is easy to change. Thresholds can be dragged along the histogram with the mouse, and added or deleted with Ctrl-click. The **Value** and **Color** are shown for the most recently clicked or moved threshold. Colors can be changed individually. For example, drag the middle threshold to a lower value, click the square [color well](#), and use the resulting [Color Editor](#) to change the color to yellow. **Close** the editor, and in the **Render by Attribute** dialog, click **Apply**. Drag the red threshold to the left of the yellow one and then **Apply** again.

The colors can also be changed collectively, by choosing a **Palette** in the **Render by Attribute** dialog. Choose the **Palette** named **Cyan-Maroon** (cyan to white to maroon), then click **Reverse threshold colors** for coloring from maroon to white to cyan. Move the thresholds if you wish and **Apply** again.

Note that the histogram includes the B-factor values of the waters even though they are not displayed. Display waters with B-factors less than 75:

*Command:* [disp solvent](#) & [@/bfactor<75](#)

Show ribbons:

*Command:* [ribbon](#)

Next, change to the attributes of **residues** in the **Render by Attribute** dialog. Available residue attributes include **kdHydrophobicity** and average **bfactor** and **occupancy**. The latter two are residue averages over the atomic values. Choose **kdHydrophobicity**, the [Kyte-Doolittle hydrophobicity scale](#) for amino acids. The values will be displayed in the histogram: negative for polar residues, positive for hydrophobic residues.

Hydrophobicity could be shown with color, but for variety, we will show it with “worms,” modified ribbons that vary in radius. In the **Render by Attribute** dialog, change from **Colors** to **Worms**. The values are still shown in a histogram, but now the thresholds have a **Worm radius** instead of a color. The thresholds can be added, deleted, and moved like before, and the **Worm radius** can be changed. By default, the more hydrophobic residues will be shown with a fatter worm (a larger worm radius). Change the mapping as desired and click **Apply**.

The worms show that the most hydrophobic residues tend to face the interior of the structure. To return to a normal ribbon instead of a worm, change the **Worm style** to **non-worm** and click **OK** (which is equivalent to **Apply** plus **Close**).

Additional [hydrophobicity scales](#) are available as [attribute assignment files](#).

Some Chimera tools create new attributes. For example, the [Add Charge](#) tool (also implemented as the command [addcharge](#)) assigns partial charges as an atom attribute named **charge**. Charge values from [Amber](#) are used for standard amino acids and nucleotides.

Use the “all atoms” preset again:

Command: [preset](#) apply int 2

Next, add hydrogens and assign atomic partial charges:

Command: [addh](#)

Command: [addcharge](#)

Above, attribute values were shown with color and worms; another approach is to display the values as atom labels. Since it would be too confusing to show all of these labels at once, only show them for the atoms in a single residue:

Command: [alias](#) myres :tyr.a

Command: [show](#) myres

Command: [focus](#)

Command: [labelopt](#) info charge

Command: [label](#) myres

The number of decimal places and whether plus signs should be used for positive values can be specified, for example:

Command: [labelopt](#) info %(charge)+.2f

Command: [lab](#) myres

New attributes can be used in the command line just like built-in attributes. For example, show the atoms with **charge** less than -0.4 as balls:

Command: [repr](#) bs @/charge<-0.4

Finally, remove the labels, change to ball-and-stick, and color atoms by their **charge** values:

Command: [~lab](#)

Command: [repr](#) bs

Command: [rangecol](#) charge -0.8 red 0 white 0.8 blue

The result should look something like the [figure](#). [Rangecolor](#) is the command alternative to **Render by Attribute** for coloring.

Close the model:

Command: [close](#) 0

Go on to [Part 2](#) below, **OR** terminate the Chimera session with the following command:

Command: [stop](#)

---

## Attributes, Part 2 - GTP-Binding Protein

Begin with Chimera started and the [Command Line](#) (and optionally the [Side View](#)) opened as described at the beginning of [Part 1](#).

If you have internet connectivity, use a command to fetch the PDB structure **121p**:

Command: [open](#) 121p

If you do not have internet connectivity, [download](#) the file [121p.pdb](#) included with this tutorial and use **File... Open** to open it.

The structure is H-ras, a small GTP-binding protein, along with a bound GTP analog, a Mg<sup>++</sup> ion, and some water. [Move and scale](#) the structure as desired throughout the tutorial.

Use the ribbons [preset](#):

Command: [preset](#) apply int 1

This shows a ribbon rainbow-color-coded from blue at the N-terminus to red at the C-terminus, plus atoms in or near binding sites. Delete the water (most of which is hidden) and label the residues near the ion:

Command: [delete](#) solvent

Command: [focus](#) ions z<3.5

Command: [rlabel](#) ions z<3.5

The active site Mg<sup>++</sup> ion is coordinated by Ser 17, Thr 35, and phosphonate oxygens in the GTP analog, which is named GCP. Color the ligand GCP yellow, remove the residue labels, and zoom back out:

Command: [color](#) yellow ligand

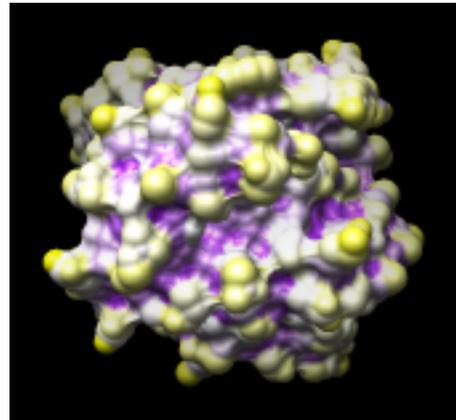
Command: [~rlabel](#)

Command: [focus](#)

In [HOMSTRAD](#), this protein is classified in the [GTP-binding protein](#) family. [Download](#) the alignment file [homstrad-gtp.pir](#) (originally from HOMSTRAD, now included with this tutorial) and open it with **File... Open**.

The alignment is shown in a separate [Multalign Viewer](#) window. A colored rectangle under the sequence name **5p21** indicates association of that sequence with the structure. If the sequence window later

121p colored by convexity



becomes obscured, it can be raised by choosing **MAV - homstrad-gtp.pir...** **Raise** from near the bottom of the **Tools** menu.

When a structure is associated with a sequence alignment in [Multalign Viewer](#), its residues are assigned [conservation attributes](#). From the **Multalign Viewer** menu, choose **Structure... Render by Conservation**. This opens the same [Render by Attribute](#) tool used in [Part 1](#), except that now it shows the attribute of residues named **mavConservation**. The “mav” part of the name is shorthand for **Multalign Viewer**, and “Conservation” indicates that the values correspond to what is shown in the **Conservation** line of the [sequence window](#). By default, the value for a column is the fraction of sequences with the most common residue type at that position, for example: 0.7 where 7 of 10 sequences have the same type of residue, 0.1 where each of the 10 has a different type. (Different calculation methods are available, see [Preferences... Headers](#) in the **Multalign Viewer** menu.)

In [Render by Attribute](#), use **Colors** and adjust the color mapping as desired before clicking **Apply**. The coloring shows that many positions in or near the binding site are highly conserved.

The **Render by Attribute** dialog also lists the residue attribute **mavPercentConserved**. Whereas **mavConservation** changes along with the **Conservation** calculation method, **mavPercentConserved** is always the percentage of sequences with the most common residue type at that position (same as the **Conservation** default, except expressed as a percentage rather than a fraction).

Select residues within 6 Å of the ligand:

*Command:* [select](#) ligand z<6

The selection is shown with green boxes on the sequence alignment.

Like other attributes, **mavConservation** and **mavPercentConserved** can be used in commands:

*Command:* [select](#) ligand z<6 & :/mavPercentConserved>80

This selects a smaller set of residues than before, only those both within 6 Å of the ligand and meeting the conservation criterion. Clear the selection (**Select... Clear Selection** is one way) and **Quit** from **Multalign Viewer**.

Show the molecular surface of the protein:

*Command:* [surface](#)

By default, the surface colors match the underlying atom colors. It is again evident that highly conserved residues line the binding pocket. Generating a molecular surface automatically creates the atom attributes **areaSAS** and **areaSES**, solvent-accessible and solvent-excluded surface areas in the context of the structure. For example, the values of both attributes are zero for atoms buried in the protein interior. What Chimera shows is the *solvent-excluded* molecular surface, composed of probe contact, toroidal, and reentrant surface. The *solvent-accessible* surface (not displayed) is farther out from the molecule, the surface traced out by the probe center.

If **Render by Attribute** is not already open, open it (**Tools... Structure Analysis... Render by Attribute**) and see what attributes are present for **atoms**. If **areaSAS** and **areaSES** are not yet listed, show them with **Refresh... Menus**. The histograms show the ranges of values obtained for the structure.

Open the **Attribute Calculator** (**Tools... Structure Analysis... Attribute Calculator**). Calculate a new attribute named **convexity** for **atoms** using the **Formula**

```
atom.areaSAS/atom.areaSES
```

Values of **convexity** > 1 represent convex areas, while values < 1 represent concave areas. Click **OK** to perform the calculation and assignment. A warning message will appear because some atoms have an **areaSES** of zero, resulting in a divide-by-zero error. However, just close the warning dialog; the attribute has been assigned correctly for the atoms with nonzero areas.

Finally, show the **convexity** values with color on the protein surface. In the **Render by Attribute** dialog, make sure that the histogram is showing the new attribute of **atoms** named **convexity**. In the **Colors** section, use three thresholds:

- drag the lowest-value (leftmost) threshold to the far left and make it **purple** (click the [color well](#) and then enter the **Color name** in the [Color Editor](#))
- drag the highest-value threshold to the far right and make it **yellow**
- click the middle threshold, make it **white**, and set its value to 1 by simply entering that number in the **Value** field and pressing return

Click **Apply**; the result should resemble the [figure](#).

The surface can be smoothed by increasing its vertex density from the default of 2.0, for example:

*Command:* [setattr s density 5](#)

When finished, end the Chimera session:

*Command:* [stop really](#)

## Sequences and Structures Tutorial

This tutorial is an introduction to working with sequence alignments and associated structures using [Multalign Viewer](#). Data are taken from the enolase superfamily:

[The enolase superfamily: a general strategy for enzyme-catalyzed abstraction of the alpha-protons of carboxylic acids](#). Babbitt PC, Hasson MS, Wedekind JE, Palmer DR, Barrett WC, Reed GH, Rayment I, Ringe D, Kenyon GL, Gerlt JA. *Biochemistry*. 1996 Dec 24;35(51):16489-501.

To follow along, first [download](#) the sequence alignment file [super8.msf](#) to a convenient location. This sequence alignment contains the barrel domains of eight enolase superfamily members.

On **Windows/Mac**, click the **chimera** icon; on **UNIX**, start Chimera from the system prompt:

```
unix: chimera
```

A splash screen will appear, to be replaced in a few seconds by the main Chimera [graphics window](#) or [Rapid Access](#) interface (it does not matter which, the following instructions will work with either).

Choose **File... Open**, then locate and open [super8.msf](#). Opening a [sequence alignment file](#) automatically starts **Multalign Viewer** and uses it to display the alignment.

**Multalign Viewer** has its own set of preferences, including display options. From the alignment window menu, choose **Preferences... Appearance** and adjust settings as desired for **Multiple alignments**, then **Close** the preferences dialog.

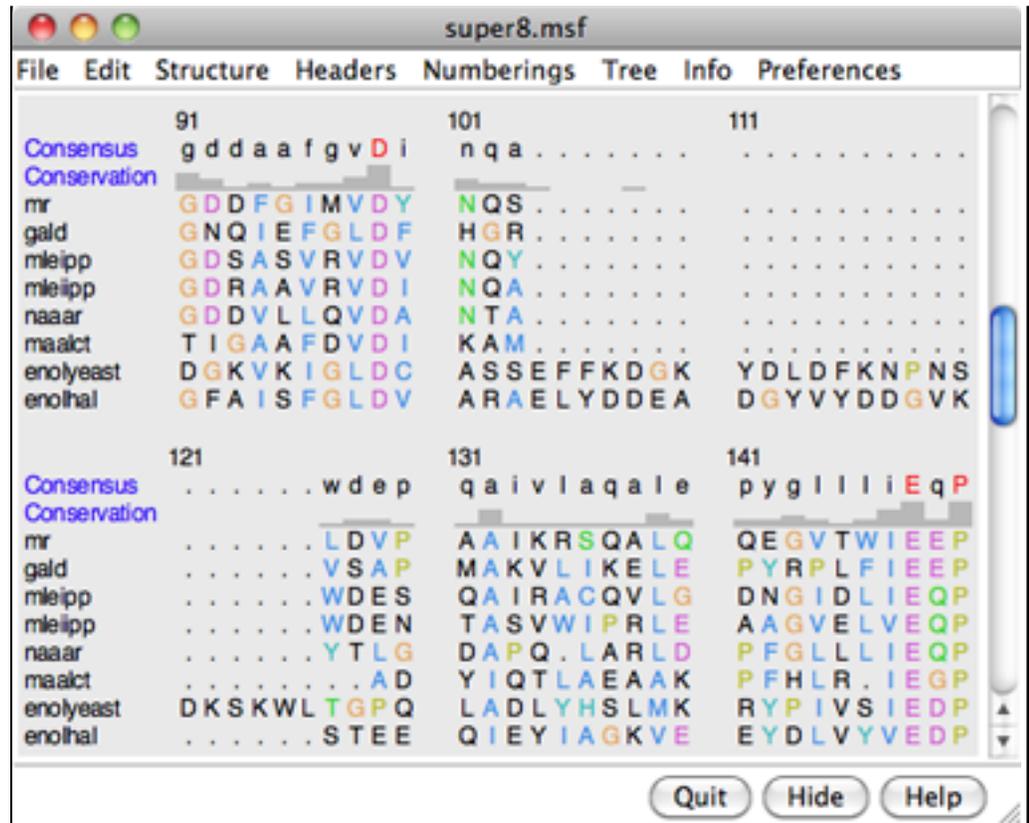
Size and place the sequence window and main Chimera window so that both are visible. If the sequence window later becomes obscured, it can be raised by choosing **Tools... MAV - super8.msf... Raise** from the menu, or by clicking [super8.msf](#) in the lefthand **Active Dialogs** section of [Rapid Access](#) (shown/hidden by clicking the lightning-bolt icon at the bottom of the Chimera window).

### Multalign Viewer sequence window



A [Consensus sequence](#) and [Conservation histogram](#) are shown above the sequences. They can be hidden/shown using the **Headers** menu. Red capital letters in the consensus sequence indicate a few positions in which all of the sequences have the same type of residue.

The first sequence in the alignment, named **mr**, corresponds to the barrel domain of mandelate racemase from *Pseudomonas putida*. The next-to-the last sequence in the alignment, named **enolyeast**, corresponds to the barrel domain of enolase from *Saccharomyces cerevisiae*. There are multiple structures in the Protein Data Bank for each of these sequences; **2mnr** (mandelate racemase) and **4enl** (enolase) are used in this tutorial.



If you have internet connectivity, structures can be obtained directly from the [Protein Data Bank](#). Use [Fetch by ID](#) (**File... Fetch by ID** in the Chimera menu) to get **2mnr** and then **4enl** from the **PDB** database. If you do not have internet connectivity, instead [download](#) the files [2mnr.pdb](#) and [4enl.pdb](#) included with this tutorial and open them in that order with **File... Open**.

The view is initially centered on the first structure opened, mandelate racemase. Off to the side and possibly out of view is the second structure, enolase. Use the menu to bring everything within view:

### Actions... Focus

Apply the ribbons [preset](#) (which may or may not change the appearance, depending on your preference settings):

### Presets... Interactive 1 (ribbons)

[Move and scale](#) the structures as desired throughout the tutorial.

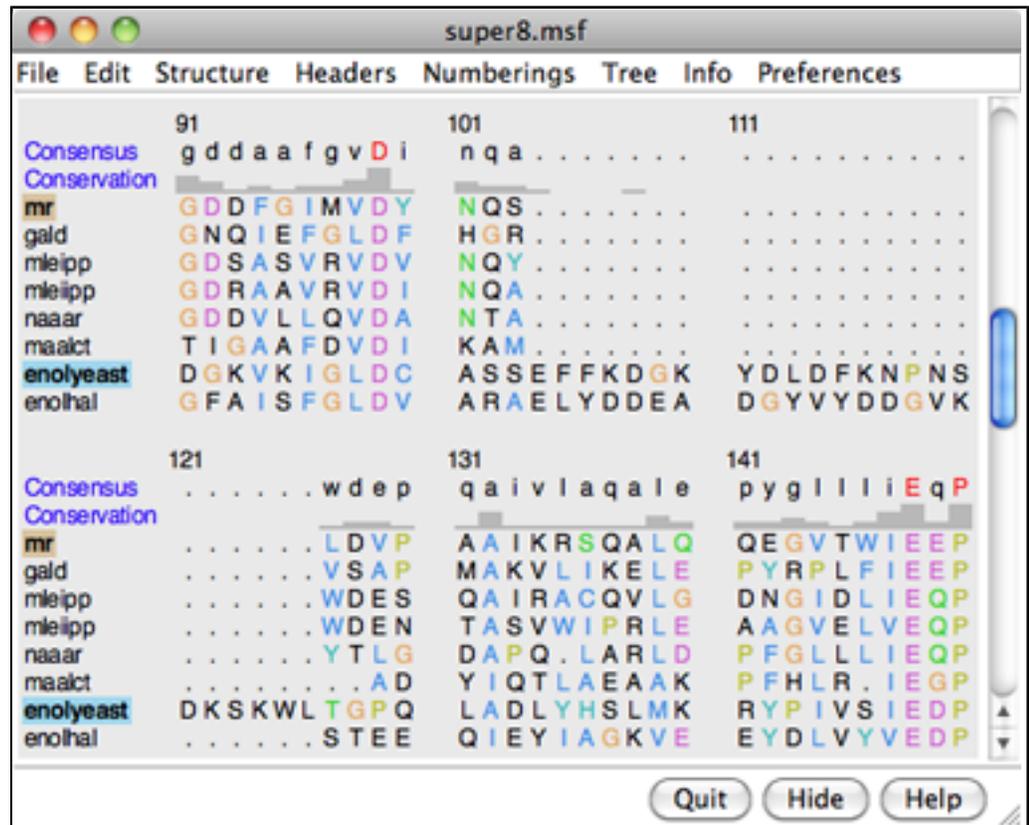
Notice that in the sequence window, the sequence names **mr** and **enolyeast** are now shown in bold within colored boxes. Each structure was [automatically associated](#) with its best-matching sequence, and the colors indicate the pairing.

### sequence window after structure association

Association enables several types of crosstalk, which will be explored in more detail below:

- selecting in the sequence window selects residues in the structure(s) and *vice versa*
- information from the sequences can be shown on the structures and *vice versa*
- the structures can be superimposed using the sequence alignment

Association tolerates some mismatches, which is important for working with mutated or partial sequences and/or structures. As a practical matter, sequence data from common sources are rarely perfect matches to the sequences of structures, even if both are available for the same species. Further, structures may not be available for the exact proteins in the sequence alignment, only their homologs.



First, we will use the sequence alignment to guide a structural match. From the **Multalign Viewer** menu, choose **Structure... Match** and click **Apply** without checking any boxes. This superimposes the structures using the alpha-carbons of all pairs of residues aligned in the sequence alignment. Readjust the view to focus on the active sites, in this case, the metal ions and surrounding atoms:

**Select... Structure... ions**  
**Select... Zone...** (try an 8-Å cutoff)  
**Actions... Focus**

Clear the selection by Ctrl-clicking in an empty part of the graphics window.

Match statistics are reported briefly in the status line at the bottom of the Chimera window and written to the [Reply Log](#) (**Favorites... Reply Log**). The match is fairly rough, giving an RMSD of 8.4 Å, and the active sites are somewhat offset. Apparently, not all 175 pairs of residues in the same columns of the sequence alignment superimpose well in space. Try matching again, but this time, turn on the option to **Iterate by pruning...** and edit the angstrom value to **1.0** before clicking **OK**. This improves the superposition of the active sites by omitting dissimilar parts from the fit.

Zoom back out to view the overall structures:

**active sites**

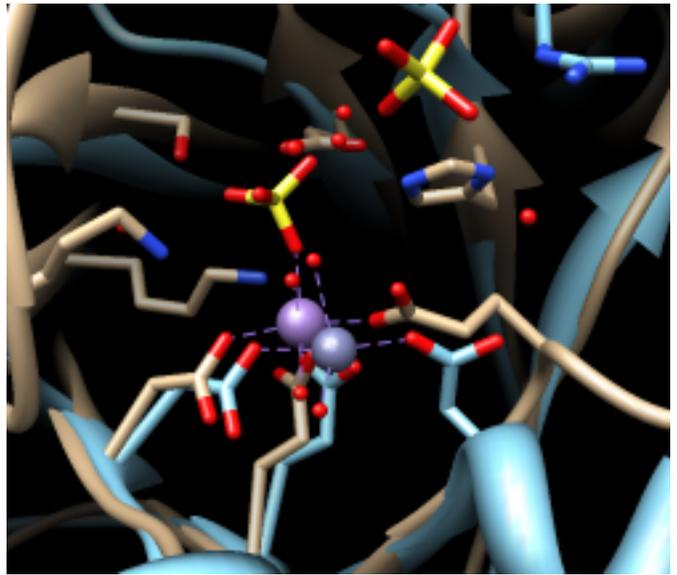
**Actions... Focus**

The structures have N-terminal and barrel domains, but

only the barrel domains are included in the sequence alignment. With the mouse, drag a box to highlight the entire sequence alignment. That will [select](#) the corresponding parts of the proteins. Invert the selection to the parts of the proteins *not* included in the sequence alignment, then hide the ribbon for those parts:

**Select... Invert (all models)**  
**Actions... Ribbon... hide**  
**Select... Clear Selection**  
**Actions... Focus**

Boxes drawn on the alignment are called [regions](#). A single region may consist of several disconnected boxes. The active region is shown with a dashed outline. Delete the active region by clicking in some blank area of the sequence window (not on the sequences) to make sure this window has the mouse focus, then pressing the keyboard **delete** key. If the region is not active (the border is not dashed), first click on the region to activate it, then press **delete**.



Communication also goes in the opposite direction: a [selection](#) in the structures is shown on the sequence alignment as a region. For example, select all aromatic residues in the structures:

**Select... Residue... amino acid category... aromatic**

The green boxes in the sequence alignment are the selection region. Clear the selection by Ctrl-clicking in an empty part of the graphics window.

Placing the cursor over any structure-associated residue in the sequence alignment shows the corresponding structure residue number near the bottom of the sequence window. For example, clicking into the sequence window and then placing the cursor over the first residue in the **mr** sequence shows that it is associated with Val 134 in chain A of model #0 (**2mnr**, mandelate racemase).

Next, see where some of the conserved residues are within the structures. In the sequence window, find the first completely conserved residue in the alignment. It is an aspartic acid, D, at alignment position 99. Drag with the mouse to create a region containing just that column of the alignment. The corresponding structure residues will be [selected](#).

These aspartic acid residues are already displayed because the ribbons [preset](#) (used above) automatically displays sidechains near any bound ions or ligand molecules.

At alignment position 150, there is a completely conserved proline (P). Create another region for this column (this time, press **Ctrl** along with the mouse button to start a *new* region instead of replacing the first). Only ribbons are shown for these residues; display the sidechain atoms too:

**Actions... Atoms/Bonds... show**

Alignment regions can also be created automatically. From the **Multalign Viewer** menu, choose

**Structure... Secondary Structure... show actual.** This creates regions in the structure-associated sequences named **structure helices** (light yellow with gold outline) and **structure strands** (light green with green outline).

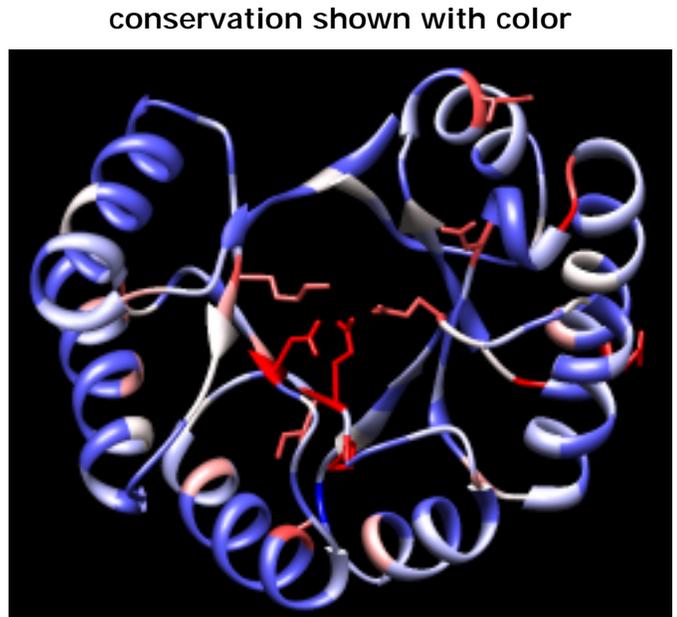
The region names are listed in the [Region Browser](#) (**Info... Region Browser** in the **Multalign Viewer** menu). Clicking the **Active** checkbox for a region will [select](#) the corresponding residues in any associated structures. Only one region can be active at a time. **Close** the **Region Browser**.

Choose **Tools... General Controls... Command Line** from the main Chimera menu and use a command to close the enolase structure:

*Command:* [close](#) 1

Clear the selection (if any) by Ctrl-clicking in an empty part of the graphics window.

A structure can be colored according to the conservation in an associated sequence alignment. Choose **Structure... Render by Conservation** from the **Multalign Viewer** menu. The resulting [Render by Attribute](#) tool shows a histogram of the [residue attribute](#) named **mavConservation**. The “mav” part of the name is shorthand for **Multalign Viewer**, and “Conservation” indicates the values correspond to what is shown in the **Conservation** line of the [sequence window](#). By default, the value for a column is the fraction of sequences with the most common residue type at that position, for example: 1.0 where all 8 sequences have the same type of residue, and  $2/8 = 0.25$  where only 2 (but any 2) share a type.



The **Render by Attribute** histogram contains vertical bars (or [thresholds](#)) for mapping the attribute values to colors, radii, or worms. Use the **Colors** section. You can define your own color mapping by dragging the thresholds along the histogram and/or changing their colors. The **Value** and **Color** are shown for the most recently clicked or moved threshold. The **Value** changes when the threshold is moved, or the position can be changed by entering a value and pressing return. The **Color** can be changed by clicking the square [color well](#) and using the [Color Editor](#). Adjust the color mapping as desired before clicking **Apply**. Coloring the structure shows the high conservation of the metal-binding residues and the low conservation of most residues around the outside of the barrel.

Attributes can also be used in the command line. For example, show only the residues with **mavConservation** values greater than 0.7:

*Command:* [show](#) :/mavConservation>.7

The conservation can be calculated in different ways (entropy, variability, *etc.*) using the AL2CO program included with Chimera. Choose [Preferences... Headers](#) from the **Multalign Viewer** menu, set **Conservation style** to **AL2CO**, and adjust additional AL2CO parameters as desired. The **Conservation** line in the sequence window will update automatically. In **Render by Attribute**, it may be necessary to use [Refresh... Values](#) to update the histogram before recoloring the structure.

[Multalign Viewer](#) includes many features, only a few of which are sampled here. Users are encouraged to explore the menus and [documentation](#) for more information. See also the [Mapping Sequence Conservation](#) tutorial on the Chimera web site.

End the Chimera session:

*Command:* [stop now](#)

---

meng-at-cgl.ucsf.edu / August 2014

## Superpositions and Alignments Tutorial

In this tutorial, [MatchMaker](#) is used to align protein structures (create a superposition), [Match -> Align](#) is used to generate a multiple sequence alignment from the structural superposition, and [Morph Conformations](#) is used to morph between related structures.

Sequence alignments are displayed in [Multalign Viewer](#), which is covered in more detail in the [Sequences and Structures tutorial](#), and the morphing trajectory is displayed in [MD Movie](#), which is covered in more detail in the [Trajectory and Ensemble Analysis tutorial](#).

Internet connectivity is required to fetch the structures used in this tutorial: **1tad, 121p, 1r2q, 1j2j, 1puj, 1tnd, 1tag**

- [Background and setup](#)
- [Different but related proteins](#)
  - [Superposition](#)
  - [Structure-based sequence alignment](#)
- [Different conformations of the same protein](#)
  - [Morphing](#)

### ← Background and Setup

Protein structures are classified within databases such as [SCOP](#), [CATH](#), and [HOMSTRAD](#). Classifications range from groups of highly similar and closely related proteins to larger, more diverse sets. For analysis and comparison, it is often useful to superimpose related structures. Although it is not always clear whether proteins with the same fold are evolutionarily related (homologous), they should still be superimposable. In general, more closely related proteins are easier to superimpose.

*G proteins* (guanine nucleotide-binding proteins) are used as examples. G proteins are important in signal transduction. They act as molecular switches, changing conformation and interaction partners depending on whether GTP or GDP is bound. Many diverse structures are known. The two main subsets are the small monomeric G proteins, such as Ras, and the larger heterotrimeric G proteins, which act immediately downstream of G-protein-coupled receptors. The  $\alpha$  subunits of heterotrimeric G proteins are homologous to the small G proteins.

On **Windows/Mac**, click the **chimera** icon; on **UNIX**, start Chimera from the system prompt:

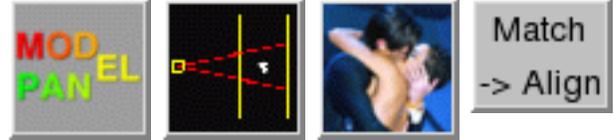
```
unix: chimera
```

A splash screen will appear, to be replaced in a few seconds by the main Chimera [graphics window](#) or [Rapid Access](#) interface (it does not matter which, the following instructions will work with either). If you like, resize the Chimera window by dragging its lower right corner.

Show the [Command Line](#) (Tools... General Controls... Command Line). Choose Favorites... Add to

**Favorites/Toolbar** to place some icons on the toolbar. This opens the **Preferences**, set to **Category: Tools**. In the **On Toolbar** column, check the boxes for:

- [Model Panel](#) (under General Controls)
- [Side View](#) (Viewing Controls)
- [MatchMaker](#) (Structure Comparison)
- [Match -> Align](#) (Structure Comparison)



You can also specify tools such as the [Command Line](#) to **Auto Start** (start when Chimera is started). If you want these settings to apply to subsequent uses of Chimera, click **Save** before closing the preferences.

Fetch a structure from the [Protein Data Bank](#):

*Command:* `open 1tad`

The structure contains three copies of the  $\alpha$  subunit of transducin, a heterotrimeric G protein. Delete solvent and two of the copies, chains B and C:

*Command:* `del solvent`

*Command:* `del :.b-c`

Move and scale the structures [using the mouse](#) and [Side View](#) as desired throughout the tutorial.

## ← Different but Related Proteins

We will [superimpose structures](#) of a sample of G proteins, then [create a sequence alignment](#) from the superposition.

The  $\alpha$  subunit of the heterotrimeric G protein transducin was already opened in the [setup](#). Fetch structures for the monomeric G proteins H-Ras, Rab5a, and ADP-ribosylation factor 1, respectively:

*Command:* `open 121p`

*Command:* `open 1r2q`

*Command:* `open 1j2j`

Use the ribbons [preset](#) (which may or may not change the appearance, depending on your preference settings):

*Menu:* **Presets... Interactive 1 (ribbons)**

This preset displays ribbons plus ions, ligands, and nearby sidechains.

## ← Superposition

superimposed G proteins

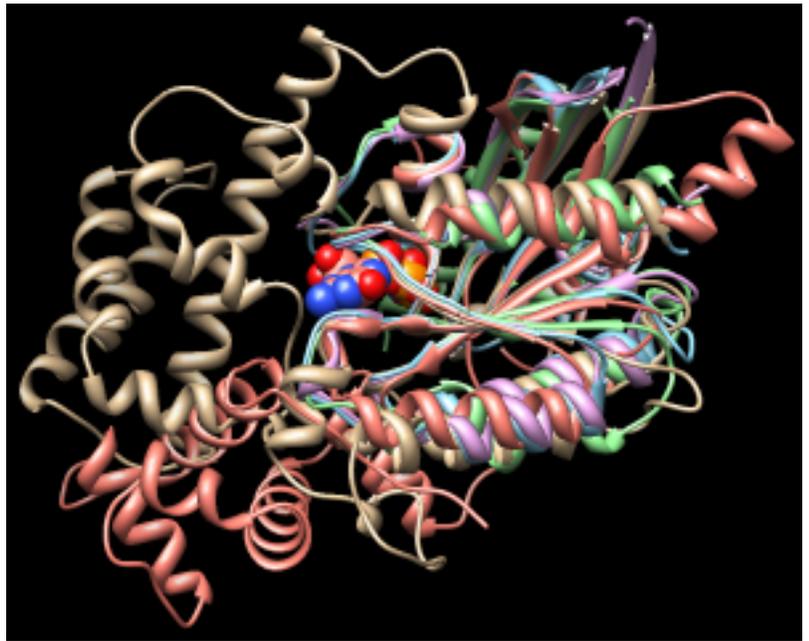
The structures need to be superimposed so that they can be compared. Start [MatchMaker](#) by

clicking its [icon](#): 

[MatchMaker](#) superimposes structures pairwise by first aligning their sequences and then fitting the  $\alpha$ -carbons of residues in the same columns of the sequence alignment. Usually the fit is iterated so that residue pairs aligned in sequence but far apart in space are not used in the final 3D match.

Several parameters control the sequence alignment step:

- algorithm - Needleman-Wunsch (global; default) or Smith-Waterman (local)
- scoring function -
  - residue similarity matrix (default BLOSUM-62)
  - whether secondary structure information should be used (default yes)
  - weighting of the secondary structure and residue similarity terms (default 30% and 70%, respectively)
  - gap penalties



Click **Reset to defaults** (near the bottom of the dialog) to ensure that the default settings will be used. All of the structures should already be chosen as the **Structure(s) to match**; keep that the same, but choose **1tad** as the **Reference** and click **Apply** to match all the others to it.

The number of  $\alpha$ -carbon pairs and RMSD in the final iteration of each pairwise fit are reported in the **Reply Log** (in the menu under **Favorites**). However, simple visual inspection of the overall structures is often the most useful indicator of success.

Another visual indicator is how well similar ligands superimpose. Show only residues classified as [ligand](#), and label them:

*Command:* [show](#) ligand

*Command:* [rlab](#) ligand

Each of these structures includes GTP or an analog of GTP in the binding site. However, some other ligands were simply present in the crystallization solution and are not biologically relevant. GOL is glycerol and can be removed:

*Command:* [del](#) :gol

*Command:* [~rlab](#)

Try using different reference structures in **MatchMaker** (click a line in the **Reference structure** list, click **Apply**). With the default alignment parameters, the superposition is similar and basically correct no matter

which structure is used as the reference. Detailed examination of the match statistics and guanine nucleotide positions suggests results may be slightly better with **1r2q** as the reference.

Next, try a structure that is harder to superimpose, and display its ligand in the sphere representation:

Command: [open](#) **1puj**

Menu: Presets... Interactive 1 (ribbons)

Command: [show](#) ligand

Command: [repr](#) sphere ligand & #4

Besides lacking sequence similarity, this protein is *circularly permuted* compared to the others: its N-terminal part structurally matches the C-terminal part of other G proteins and vice versa.

In the **MatchMaker** dialog, change the **Structure to match** to only **1puj** and try the others in turn as the reference. Again, ligand positions can be used to help gauge the match.

Trials with the default alignment parameters are not successful. When proteins are very distantly related, it may be useful to switch to a lower-number BLOSUM matrix and/or increase the proportion of secondary structure scoring. Usually a range of parameters will give similar results. For example, with **121p** as the reference structure, **1puj** can be superimposed as shown in the [figure](#) with any of BLOSUM 30-75 if secondary structure weighting is raised to 90% and the Smith-Waterman algorithm is used (leaving other settings as defaults). Keep in mind that when proteins are very distantly related, their backbones may diverge even in the best possible superposition.

## ← Structure-Based Sequence Alignment

When all five proteins are superimposed to your satisfaction, **Cancel** the **MatchMaker** dialog. We will generate a structure-based alignment of the five sequences using [Match -> Align](#); start that tool by

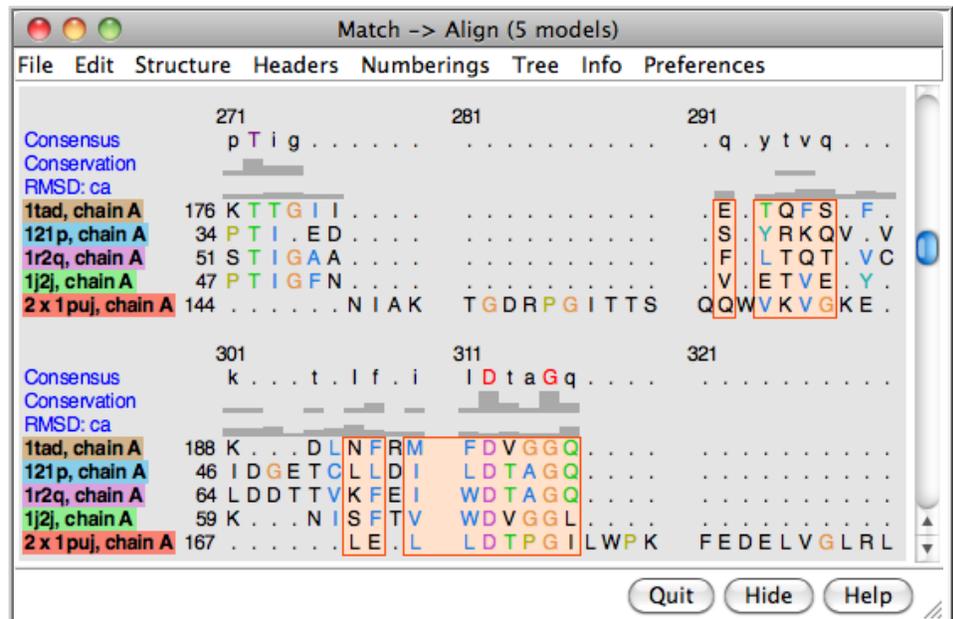
clicking its [icon](#): 

[Match -> Align](#) uses only the distances between  $\alpha$ -carbons to create an alignment. Residue types and how the structures were superimposed are not important. All of the A chains should already be chosen in the dialog; the B chain of **1j2j** is an unrelated peptide and should not be chosen. Use a cutoff of 5.0 Å, specify **Residue aligned in column if within cutoff of [at least one other]**, and turn on **Allow for circular permutation**. Click **OK** to start the calculation.

It may take a minute or two to create the alignment; progress is reported in the [status line](#). When the calculation is finished, the new alignment will be displayed in [Multalign Viewer](#) and can be [saved to a file](#) from that tool.

The output multiple sequence alignment (example: [5gees.afa](#)) shows that **1puj** was correctly recognized as a circular permutation relative to the

others. **Match** -> **Align** doubled its sequence to allow C-terminal residues (in the first copy of the sequence) to appear before more N-terminal residues (in the second copy) within the alignment. The columns with residues from all five structures are highlighted as a region in light orange with dark orange outline. Clicking the region will select the corresponding parts of the structures, in effect their common cores. The [alignment header](#) named "RMSD: ca" shows the spatial variation per column ( $\alpha$ -carbon root-mean-square deviation) as a histogram.



Keep the sequence alignment, but close most of the structures:

1. start the [Model Panel](#) by clicking its [icon](#): 
2. in the **Model Panel**, choose all of the models except **1tad** on the left side and click the **close** button on the right (not at the bottom of the dialog!)
3. **Close the Model Panel**

If [Multalign Viewer](#) (the sequence alignment window) is hidden behind other windows, it can be resurrected by choosing **MAV - alignment-name...** **Raise** from near the bottom of the **Tools** menu. In **Multalign Viewer**:

- choose **Preferences... Appearance** and adjust settings for **Multiple alignments** as desired
- use **Info... Percent Identity** to compare **all sequences with all sequences**, confirming that the pairwise identities are <30% for these examples
- use **Edit... Delete Sequences/Gaps** to delete the sequence named **2 x 1puj, chain A** and any resulting all-gap columns

Now the alignment clearly shows the large insertion in  $\alpha$ -transducin (**1tad**) relative to the small monomeric G proteins. Select and display residues that are completely conserved in the sequence alignment:

Command: [sel](#) :/mavPercentConserved=100

Command: [disp](#) sel

Some of the conserved residues are Gly (no sidechain). Clear the selection by Ctrl-clicking in an empty area of the graphics window.

## ← Different Conformations of the Same Protein

(To jump to this section right after performing the [setup](#), open the sequence alignment file [4gees.afa](#) included with this tutorial.)

Now we will compare **1tad** with different structures of the same protein, transducin- $\alpha$ :

Command: `open 1tnd`

Command: `open 1tag`

Delete solvent and chains B-C (extra copies in **1tag**):

Command: `del solvent`

Command: `del :.b-c`

If [Multalign Viewer](#) (the sequence alignment window) is hidden, bring it to the front by choosing **MAV - alignment-name...** **Raise** from near the bottom of the **Tools** menu.

[Multalign Viewer](#) displays lines of information called [headers](#) above the sequences in the alignment. Use the **Headers** menu to hide **Consensus** and **Conservation** and to show **RMSD: ca**, if not already shown. The sequence name **1tad, chain A** has a dashed green line around it, indicating that the sequence is associated with multiple structures. The RMSD header shows the spatial variability of residues associated with each column ( $\alpha$ -carbon root-mean-square deviation); currently, it contains high values everywhere because the structures are not all superimposed.

To superimpose the structures using the sequence alignment, choose **Structure... Match** from the **Multalign Viewer** menu. One structure (it does not matter which) should be designated as the reference, and all three can be designated as the structures to match. Check the option to **Iterate by pruning...** using a **2.0-Å** cutoff and click **OK**. The RMSD header is automatically recomputed, showing much lower values.

Superposition of proteins with the same or nearly the same sequence is generally trivial. We used [Multalign Viewer](#) since we already had a sequence alignment, but [MatchMaker](#) (or its [command equivalent](#)) or the command `match` could have been used instead. These other methods are used and discussed in the [Structure Analysis and Comparison tutorial](#).

Use the ribbons preset (which may or may not change the appearance, depending on your preference settings) and focus on the ligand residues:

Command: `preset apply int 1`

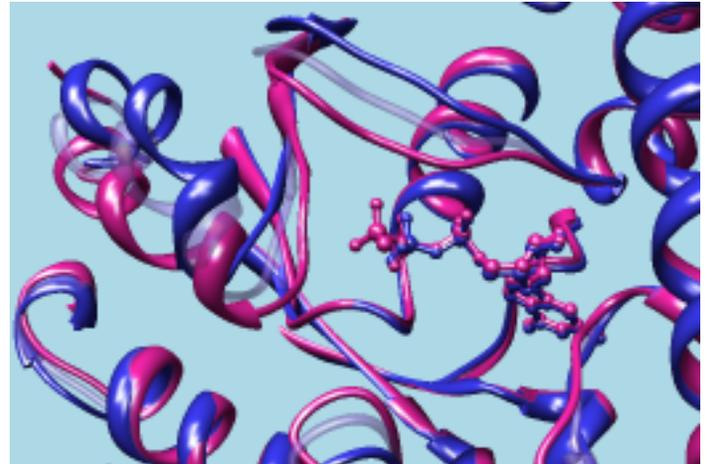
Command: `focus ligand`

Command: `r1ab ligand`

Open the [Model Panel](#) and use the **S(hown)** checkboxes to view the structures individually.

The **1tad** structure (tan) represents the activated form of a G protein; even though it includes GDP, the GDP and ALF (AlF<sub>4</sub><sup>-</sup>) residues together mimic the transition state of GTP hydrolysis. **1tnd** (light blue) contains the GTP analog GSP and also represents the activated form. The third structure, **1tag** (purplish

**GTP-binding switch**  
(**1tagA**, **1tndA**, morph intermediate)



pink), includes GDP and represents the nonactivated form.

Use the [Model Panel](#) checkboxes to show all three structures together. Remove the labels and focus on the overall structures:

Command: [~rlab](#)

Command: [focus](#)

Although the structures are mostly similar, the nonactive conformation (pink) differs from the activated ones (tan and light blue) in specific areas, termed *switch regions*.

In the sequence alignment window, the three most prominent “humps” in the RMSD header correspond to the known G protein switch regions at approximately residues 173-183, 195-215, and 227-238 of transducin- $\alpha$ . The third switch region is unique to heterotrimeric G proteins; it is an insertion relative to the monomeric G proteins. Placing the cursor over a position in the **1tad** sequence lists the associated structure residues near the bottom of the sequence window, and drawing a box around residues in the sequence alignment (click to start, drag to expand) [selects](#) the associated parts of the structures.

Close **1tad**:

Command: [close 0](#)

The RMSD histogram looks much the same; now it simply shows the CA-CA distances between the two remaining structures, **1tnd** representing the activated form and **1tag** representing the nonactivated form.

## ← Morphing

Finally, morph between the two structures. Morphing involves calculating a series of intermediate structures. In Chimera, the series of structures is treated as a trajectory that can be replayed, saved to a coordinate file, or saved as a movie using [MD Movie](#).

Start the [morphing tool](#):

Command: [start Morph Conformations](#)

Click **Add...** and in the resulting list of models, doubleclick to choose #2, #1, and #2 again, corresponding to a morph trajectory from the nonactivated structure to the activated and back. **Close** the model list. In the main **Morph Conformations** dialog, set the **Action on Create** to **hide Conformations**, and then click **Create**.

The progress of the calculation is reported in the [status line](#). When all the intermediate structures have been calculated, the input structures are hidden, the trajectory is opened as model #0, and the [MD Movie](#) tool appears.

The trajectory can be played continuously or one step at a time using the buttons on the tool. If the player dialog becomes obscured by other windows, it can be resurrected by choosing **MD Movie - trajectory-**

*name...* **Raise** from near the bottom of the **Tools** menu. If you want to see the original structures again, use the **S(hown)** checkboxes in the [Model Panel](#).

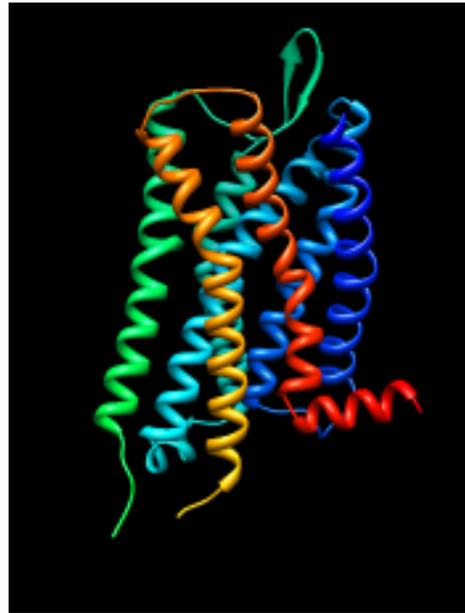
When you have finished viewing the morph trajectory, choose **File... Quit** from the menu to exit from Chimera.

---

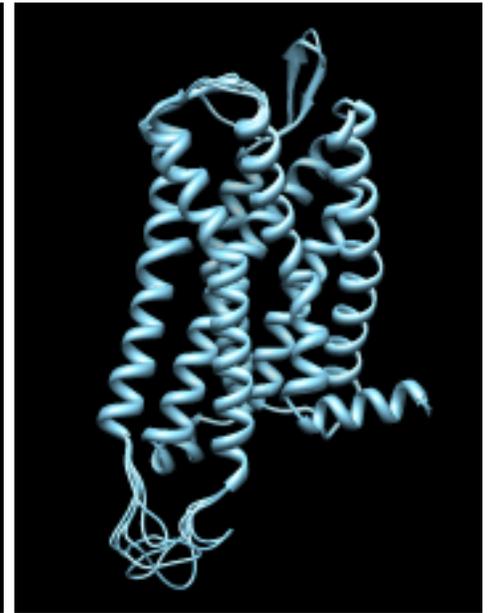
meng-at-cgl.ucsf.edu / May 2014

## Comparative Modeling Tutorial

This tutorial includes running [Blast sequence search](#) and [Modeller comparative modeling](#) calculations from Chimera. Internet connectivity is required to fetch data and to access Blast, Modeller, and other web services. Although no software installation (other than Chimera itself) is needed to follow the tutorial, Modeller use requires a license key. Academic users can obtain a license key free of charge by [registering](#) at the Modeller website.



template:  $\mu$ -opioid receptor



target:  $\delta$ -opioid receptor  
(5 comparative models)

- [Background and Caveats](#)
- [Blast Search for Templates](#)
- [Verifying the Alignment](#)
- [Running Modeller](#)

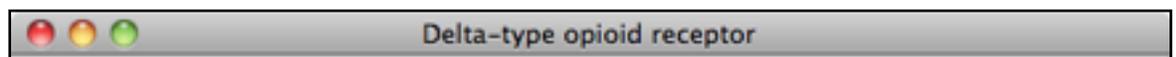
### ← Background and Caveats

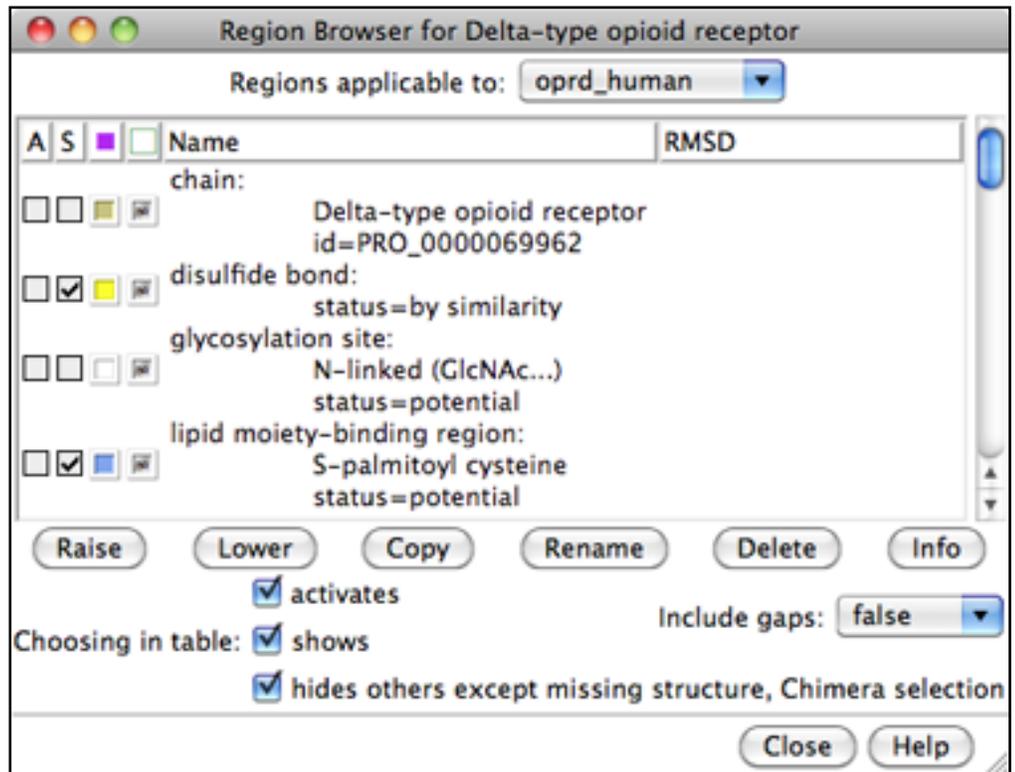
In comparative (homology) modeling, theoretical models of a protein are built using at least one known related structure and a sequence alignment of the known and unknown structures. The protein to be modeled is the *target*, and a related known structure used for modeling is a *template*.

The target in this tutorial is the human  [\$\delta\$ -opioid receptor](#), a G-protein-coupled receptor (GPCR). GPCRs are transmembrane proteins and have been relatively resistant to structure determination. Although recent advances have allowed solving the structures of several members of this large and biomedically important class of proteins (see the [Protein Structure Initiative GPCR network](#)), at the time of creating this tutorial (May 2012), no structure was available for the  $\delta$ -opioid receptor.

Tutorial caveats:

- As database contents and web services are updated, the results of calculations are likely to differ from what is described here. However, this tutorial is meant to illustrate the general process rather than any specific result.
- This tutorial is not meant to indicate the optimal parameter settings for comparative modeling, as these will vary depending on the system of interest and the information available at the time.
- This tutorial describes only one of several equally valid approaches. For example, the sequence of the target could be read from a FASTA file instead of fetched from the UniProt database.
- Modeller quality scores were developed for globular (soluble) proteins and may be less effective indicators of model quality for transmembrane proteins such as GPCRs.





## ← Blast Search for Templates

[Start Chimera](#). A splash screen will appear, to be replaced in a few seconds by the main Chimera [graphics window](#) or [Rapid Access](#)

interface (it does not matter which, the following instructions will work with either).

GI	PDB	Evalue	Score	Resolution	Chain names
query		0	0		
380765131	4DKL_A	2e-86	317	2.8	A: Mu-type opioid receptor, lysozyme chimera SEE REMARK 999
380765109	4DJH_B	5e-81	299	2.9	AB: Kappa-type opioid receptor, Lysozyme UNP P41145 residues 43-261, UNP P00720 residues 2-161, UNP P41145 residues 362-358
288965576	3KJ6_A	4e-24	110	3.4	A: Beta-2 adrenergic receptor L: Fab light chain H: Fab heavy chain R: Lysozyme, Beta-2 adrenergic receptor
340780682	3SN6_R	6e-24	110	3.2	R: Lysozyme, Beta-2 adrenergic receptor

Choose **File... Fetch by ID** from the menu and use the resulting dialog to fetch the sequence of the target, the human  $\delta$ -opioid receptor: **UniProt ID oprd\_human**. If you want to verify the ID before fetching, click the **Web Page** button on the fetch dialog to see the corresponding page at [UniProt](#). (One way to determine the ID in the first place is by searching at the [UniProt](#) site.)

The sequence is displayed in [Multalign Viewer](#), and its UniProt feature annotations listed in the [Region Browser](#). The **S** column checkboxes in the **Region Browser** can be used to show feature annotations as colored boxes in the sequence window. Close the **Region Browser**; it can be accessed any time from the sequence window **Info** menu.

The next step is to find a known protein structure suitable for use as a modeling template. We will use Chimera's [Blast Protein](#) tool to search the [Protein Data Bank](#) (PDB; a database of known structures) for sequences similar to the target. From the sequence window menu, choose **Info... Blast Protein**, click **OK** to use **oprd\_human** as the query, and **OK** again to perform the search using default settings, including **pdb** as the database to search. Searching the **pdb** sequences should take only a few seconds. Searching the **nr** database, which also contains a huge number of sequences without known structures, would take much longer.

In the Blast results dialog, the hits are listed from best to worst. Click the **Columns** button to reveal several checkboxes for controlling which columns of information are shown. Hide (uncheck) **Description**, then show **Resolution** and **Chain names**. As shown in the figure, the two best hits are opioid receptors, followed by several other types of GPCRs. One technique for GPCR structure determination is to express the receptor as a fusion with some other protein that favors crystallization. The two best hits, PDB entries

[4DKL](#) and [4DJH](#), contain structures of opioid receptors fused with lysozyme.

It is possible to use multiple templates, but we will use just 4DKL\_A (PDB entry 4DKL, chain A). The **Chain names** information says to “SEE REMARK 999,” which we will do after opening the structure. In the Blast results dialog, click to highlight the corresponding row, then at the bottom of the dialog:

1. click **Show in MAV** to display the query-hit sequence alignment from Blast in another [Multalign Viewer](#) (MAV) window
2. click **Load Structure** to fetch 4DKL from the PDB and open it in Chimera
3. click **Quit** to dismiss the Blast results dialog

To see the “REMARK 999” lines in the PDB file of the structure:

- choose **Favorites... Model Panel** from the Chimera menu
- in the [Model Panel](#), click **attributes...** to show the attributes of model 4DKL
- at the bottom of the attributes dialog, click **PDB Headers...**
- in the resulting dialog, scroll down to see the **REMARK 999** lines:

```
REMARK 999
REMARK 999 SEQUENCE
REMARK 999 CHAIN A IS AN INTERNAL FUSION OF LYSOZYME (RESIDUES 2-161 OF UNP
REMARK 999 P00720) BETWEEN RESIDUES 52-263 AND RESIDUES 270-352 OF MU-TYPE
REMARK 999 OPIOID RECEPTOR (UNP P42866). AN OFFSET OF 1000 HAS BEEN ADDED TO
REMARK 999 LYSOZYME RESIDUE NUMBERS WITHIN THE COORDINATES TO DISTINGUISH THAT
REMARK 999 PORTION OF CHAIN A. LYSOZYME RESIDUES ARE THEREFORE NUMBERED 1002-
REMARK 999 1161.
```

From this, we note:

- the  $\mu$ -opioid receptor portion has UniProt ID [p42866](#) (equivalent to **oprm\_mouse**)
  - the lysozyme residues are numbered 1002-1161
- Close the headers dialog, attributes dialog, and **Model Panel**

Show the Chimera [Command Line](#) (for example, with **Favorites... Command Line**), hide atoms, rainbow-color the ribbon, and make the lysozyme portion dark:

*Command:* [~display](#)

*Command:* [rainbow](#)

*Command:* [alias](#) lyso :1002-1161

*Command:* [color](#) dim gray lyso

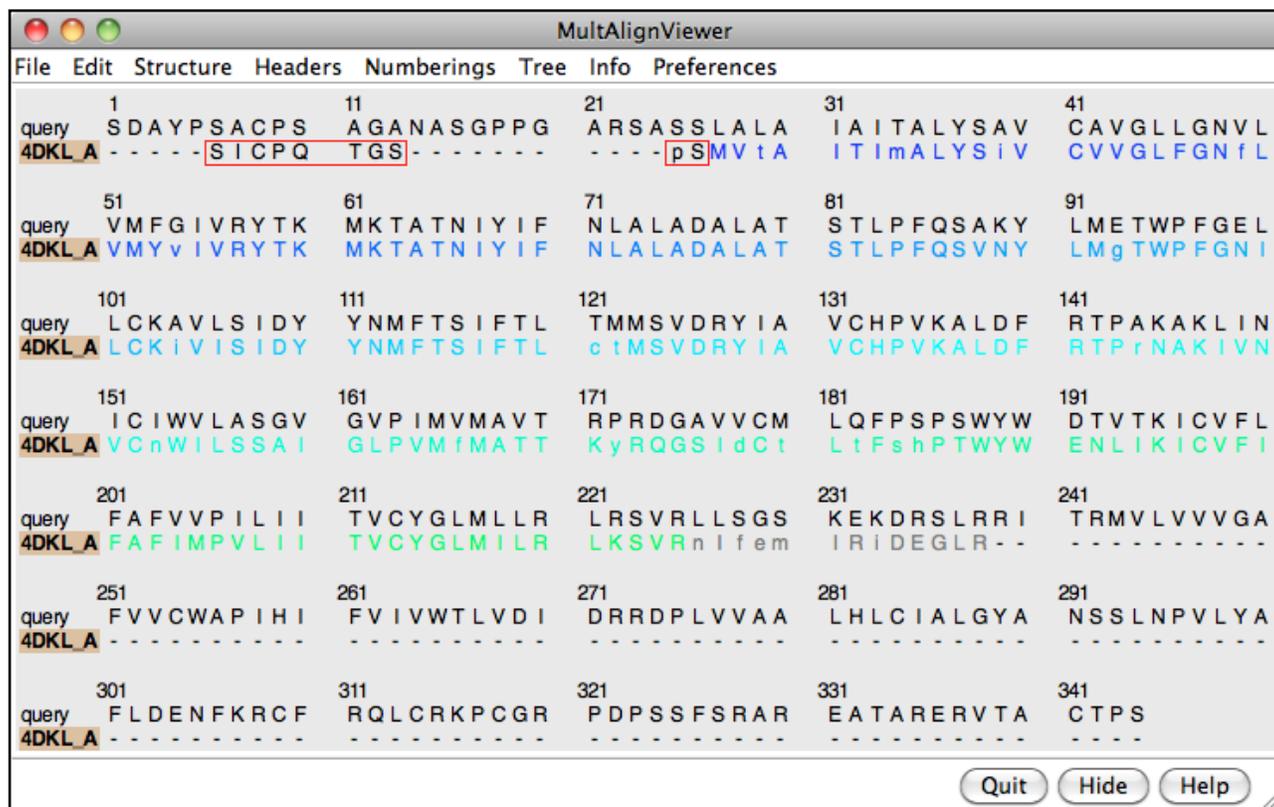
GPCRs have an extracellular N-terminus, seven transmembrane helices, and an intracellular C-terminus. Visually tracing the structure from N-terminus (blue) to C-terminus (red) reveals that lysozyme is inserted between the fifth and sixth transmembrane helices, in the third intracellular loop.



## ← Verifying the Alignment

Comparative modeling requires a template structure and a target-template sequence alignment. The sequence alignment is important; it controls which residues in the template are used to model which residues in the target, and any inaccuracies in the alignment will result in the application of incorrect constraints during 3D modeling. Regardless of how the sequence alignment was obtained, it should be examined and adjusted as needed before initiating the more computationally intensive 3D modeling calculations.

An alignment was generated [above](#) as a by-product of the Blast similarity search. However, Blast is meant to identify local similarities quickly rather than to give accurate full-length alignments. View the Blast alignment to see if it looks reasonable.



In general, if a Chimera window or dialog is obscured by other windows, it can be accessed using its [instance](#) near the bottom of the **Tools** menu, or from the Active Dialogs section of the [Rapid Access](#) interface (itself shown by clicking the lightning bolt icon near the bottom of the main Chimera window).

Use the sequence window **Headers** menu to hide the **Consensus** and **Conservation** lines, then scroll or resize the window to show the whole alignment. A large part of the query (target) sequence is not aligned. To understand what happened, try coloring the structure sequence to match the ribbon display: from the sequence window menu, choose **Preferences... Appearance** and in the resulting dialog, change the multiple alignments **Color scheme** to **ribbon**. As shown in the figure, the coloring reveals that the alignment includes the first five transmembrane helices but cuts off in the lysozyme insertion. The rest of the structure sequence is omitted, leaving the rest of the query unaligned.

Thus the alignment from Blast is not adequate for modeling purposes, and the target-template sequence alignment must be generated in some other way. Click **Quit** to close the sequence alignment from Blast.

To generate the target-template sequence alignment, we will return to the original **oprd\_human** (target) sequence and use the Needleman-Wunsch global alignment algorithm to add the sequence of the  $\mu$ -opioid receptor (template). If the sequence window was closed, not to worry, the target sequence can be fetched

again as described [above](#) or using a command:

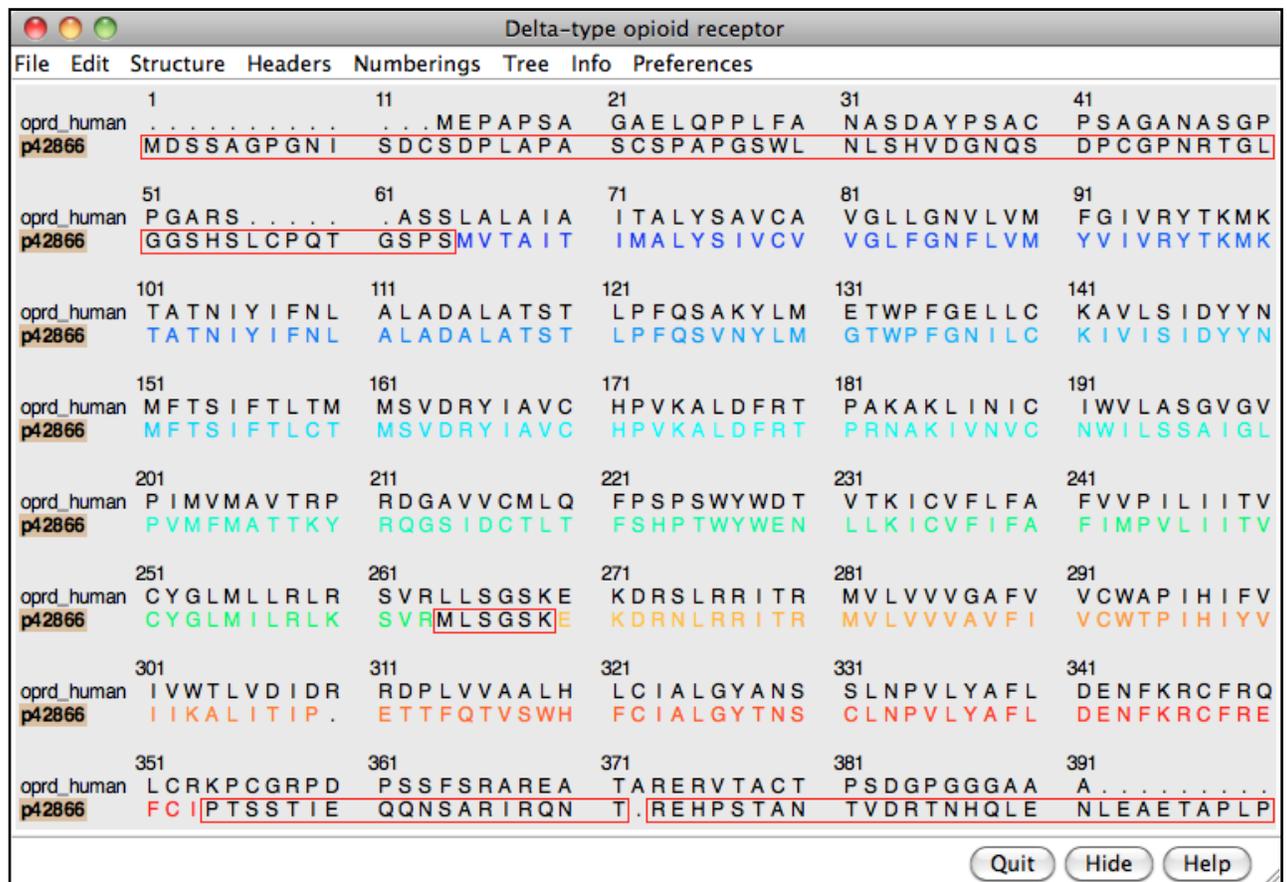
Command: [open](#) uniprot:opr\_d\_human

From the sequence window menu, choose **Edit... Add Sequence**. The resulting dialog contains tabs for different ways of obtaining the sequence.

In similar situations, it would usually be fine to add the template sequence **From Structure**. However, in this special case of a chimeric template protein, the structure sequence includes residues from another protein (lysozyme) that should not be in the alignment. Even deleting the residues from the structure, as will be done below, does not remove them from the structure sequence.

Instead, use the **From UniProt** tab and add ID **p42866** (noted [above](#) for the  $\mu$ -opioid receptor part of the structure) using default alignment parameters. The **Region Browser** will appear and can be closed.

Delete the lysozyme part of the template structure since only the  $\mu$ -opioid receptor part is useful for modeling the  $\delta$ -opioid receptor:



Command: [delete](#) lyso

The template structure still needs to be associated with the corresponding sequence in the alignment. Often sequences and structures will associate automatically, but in this case it is necessary to do it manually: from the sequence window menu, choose **Structure... Associations** and associate the structure **4DKL** with the sequence **p42866**.

In the sequence window, the association is indicated with a tan box around the sequence name (tan is the default color of the structure). Red outline boxes enclose residues that are in the sequence but not in the

associated structure. There are quite a few missing residues: stretches at both ends and a few in the third intracellular loop, which had been partly replaced by lysozyme. However, the residues that are present in the template structure, including all seven transmembrane helices, are well-aligned with the target.

If you find the sequence coloring difficult to view, it can be changed to some other scheme (such as all black) using **Preferences... Appearance** in the sequence window menu. This also allows changing sequence wrapping, font size, *etc.* Coloring the sequence to match the structure ribbon is only one of several approaches for sequence-structure mapping. For example:

- highlighting residues in the sequence with the mouse [selects](#) them in the structure
- [selecting](#) residues in the structure highlights them in the sequence (green boxes)
- structure helix and strand assignments can be shown on the sequence with **Structure... Secondary Structure... show actual**

## ← Running Modeller

From the sequence window menu, choose **Structure... Modeller (homology)** to open the Chimera interface to [comparative modeling with Modeller](#). The target should be set to **opr\_d\_human**. Click **p42866** in the dialog to choose it as the template.

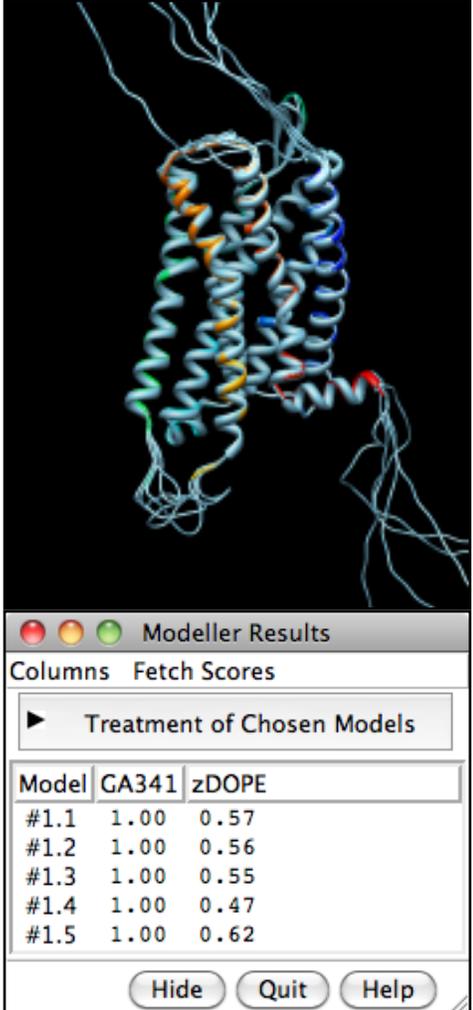
Click the **Advanced Options** button to reveal additional settings. **Run Modeller via web service** indicates using a web service hosted by the [UCSF RBVI](#). No local installation is required to run the web service, but it is necessary to enter a **Modeller license key**, available free of charge to academic users upon [registration](#) at the Modeller website. After entering the license key, click **OK** to launch the calculation with default settings. Five comparative models will be generated.

The Modeller run may take several minutes and is handled as a background task. Clicking the information icon  near the bottom of the Chimera window will bring up the [Task Panel](#), in which the job can be canceled if desired.

When the five models have been generated, they will be opened in Chimera and their evaluation scores shown in a **Model List** dialog. The models can be viewed individually or collectively by choosing rows in the dialog with the mouse. The different scores from Modeller use different criteria and will not necessarily agree on which models are best:

- **GA341** - model score derived from statistical potentials; a value > 0.7 generally indicates a reliable model, >95% probability of having the correct fold
- **zDOPE** - normalized Discrete Optimized Protein Energy (DOPE), an atomic distance-dependent statistical score; negative values indicate better models

Although Modeller scores were developed for globular proteins and thus have limited applicability to the



Model	GA341	zDOPE
#1.1	1.00	0.57
#1.2	1.00	0.56
#1.3	1.00	0.55
#1.4	1.00	0.47
#1.5	1.00	0.62

transmembrane protein in this tutorial, another reason for the poor (positive) zDOPE scores is that the termini extend far beyond the template structure and cannot be modeled reliably. Displaying all the models at once shows little conformational variability except in the termini, and to a lesser extent, the untemplated part of the third intracellular loop. This conclusion is reinforced by the **RMSD** histogram in the sequence window, where bar heights indicate root-mean-square distances among the  $\alpha$ -carbons of the residues associated with a column.

Although there is also a Chimera interface to Modeller for [untemplated building and refinement](#), in this tutorial we will simply remove the termini and rescore the models.

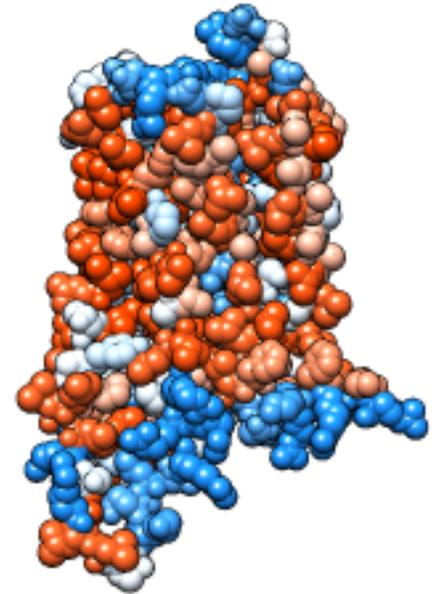
Click into the sequence window, then move the cursor over the residues to see the corresponding structure residue numbers near the bottom of the window. In the comparative models (#1.1-5), Leu-46 is aligned with the first residue in the template structure and Arg-334 is aligned with the last residue in the template structure. Delete the termini in the comparative models that extend beyond the template:

*Command:* `del #1:start-45,335-end`

To rescore the models, choose **Fetch Scores... zDOPE and Estimated RMSD/Overlap** from the **Model List** menu. Rescoring uses a web service provided by the [Sali lab](#) at UCSF. After a minute or few, more favorable zDOPE values are obtained, along with the additional scores:

- [Estimated RMSD](#) - TSVMMod-predicted C $\alpha$  root-mean-square deviation (RMSD) of the model from the native structure
- [Estimated Overlap \(3.5 Å\)](#) - TSVMMod-predicted native overlap (3.5 Å), fraction of C $\alpha$  atoms in the model within 3.5 Å of the corresponding atoms in the native structure after rigid-body superposition

Modeller Results				
Columns		Fetch Scores		
▶ Treatment of Chosen Models				
Model	GA341	zDOPE	Estimated RMSD	Estimated Overlap (3.5Å)
#1.1	1.00	-0.20	6.931	0.739
#1.2	1.00	-0.24	5.542	0.774
#1.3	1.00	-0.22	6.743	0.764
#1.4	1.00	-0.29	7.065	0.700
#1.5	1.00	-0.17	7.572	0.763



The comparative models are atomically detailed and can be subjected to various analyses in Chimera. In the figure, the model with the best (lowest) estimated RMSD score is shown as spheres colored by [amino acid hydrophobicity](#), from **dodger blue** for the most hydrophilic to **white** to **orange red** for the most hydrophobic (see [color names](#)). The **Model List** dialog was used to show only this model of the five, then the following commands were used to hide the template and adjust the model's appearance:

*Command:* `~modeldisp #0`

*Command:* `disp`

*Command:* `~ribbon`

*Command:* `rangeocol kdHydrophobicity min dodger blue mid white max orange red`

*Command:* `preset apply pub 1`

*Command:* `repr sphere`

## The Model Panel and Ensembles Tutorial

This tutorial focuses on using the [Model Panel](#) and handling ensembles of structures (such as those determined by NMR).

We will view solution structures of a toxin that binds to sodium channels. Separate ensembles were determined for *cis*- and *trans*-proline conformations of this toxin:

[NMR solution structures of  \$\delta\$ -conotoxin EVIA from \*Conus ermineus\* that selectively acts on vertebrate neuronal Na<sup>+</sup> channels.](#) Volpon L, Lamthanh H, Barbier J, Gilles N, Molgó J, Ménez A, Lancelin JM. *J Biol Chem*. 2004 May 14;279(20):21356-66.

On **Windows/Mac**, click the **chimera** icon; on **UNIX**, start Chimera from the system prompt:

```
unix: chimera
```

A splash screen will appear, to be replaced in a few seconds by the main Chimera [graphics window](#) or [Rapid Access](#) interface (it does not matter which, the following instructions will work with either). If you like, resize the Chimera window by dragging its lower right corner. Show the [Command Line](#) by choosing **Tools... General Controls... Command Line** from the menu.

If you have internet connectivity, structures can be obtained directly from the [Protein Data Bank](#):

```
Command: open 1g1p
```

```
Command: open 1g1z
```

If you do not have internet connectivity, you can [download](#) the files [1g1p.pdb.gz](#) and [1g1z.pdb.gz](#) into your working directory and then open them in that order as local files (with **File... Open**). It is not necessary to uncompress the files.

Use the ribbons [preset](#):

```
Menu: Presets... Interactive 1 (ribbons)
```

This may or may not change the appearance, depending on your preference settings.

[Rotate, translate, and scale](#) the structures as desired throughout the tutorial. There are [several ways](#) to start tools, including from the menu or with a command. Open the [Model Panel](#):

```
Command: start Model Panel
```

The **Model Panel** lists models on the left and [functions](#) on the right.

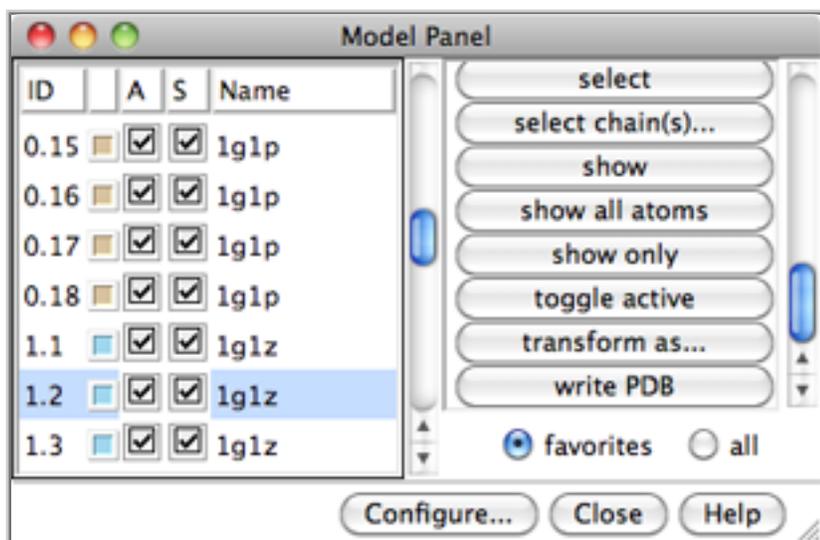
A file of coordinates opened in Chimera becomes a *model* with an associated ID number and [model color](#). Some PDB files are further subdivided into multiple structures designated with MODEL and ENDMDL records; these are assigned *submodel* numbers. Since each structure can be handled independently, the general term “model” can refer to a submodel as well as a model that is not subdivided.

Each of these PDB files is an NMR ensemble of several structures (submodels). At first, the submodels in a model are collapsed into a single row. Expand the listing to individual submodels:

1. *choose* one model by clicking its ID or name in the left side of the panel
2. click the **group/ungroup** function button on the right

Repeat the process for the other model. **1g1p** (*trans*-proline conformations, white) has been opened as models 0.1–0.18 and **1g1z** (*cis*-proline conformations, magenta) as models 1.1–1.18.

By default, the list of [functions](#) on the right side of the **Model Panel** includes only those classified as **favorites**. If you do not see some function that is mentioned, try changing from **favorites** to **all** functions using the checkbox below the list.



Scroll down the list of models in the left side of the **Model Panel** and choose model 1.2. Try various functions:

- show only** - hide the other models
- show all atoms** - display all atoms
- select** - [select](#) the entire model for further operations

By default, ribbons suppress backbone atom display. Hide the ribbon to reveal the backbone atoms:

*Menu:* **Actions... Ribbon... hide**

Select and delete hydrogens in *all* models:

*Menu:* **Select... Chemistry... element... H**  
*Menu:* **Actions... Atoms/Bonds... delete**

When atoms might be needed later, **hide** should be used instead of **delete**.

Back to the **Model Panel**:

**sequence...** (it may be necessary to change from **favorites** to **all** to see this function in the **Model Panel**) opens a [sequence window](#) for the model. Two very short  $\beta$ -strands (positions

24-25 and 29-30) are highlighted in light green. Placing the mouse cursor over a residue in the sequence shows the corresponding structure residue number at the bottom of the sequence window. The  $\beta$ -strand locations were read from the input file along with the coordinates. Highlight a string of residues in the sequence with the mouse and see how they become [selected](#) in the structure. **Quit** from the sequence window, then act on the selection with the menu: **Actions... Color... orange**

Clear the selection (Ctrl-click in an empty area of the graphics window) and go back to showing only ribbons:

*Menu: Presets... Interactive 1 (ribbons)*

Note the orange coloring is gone; [interactive presets](#) reset the coloring, whereas [publication presets](#) do not adjust colors, aside from making the background white.

The ribbon shows the  $\beta$ -strands as arrows. Although the input file specifies the strand locations as 24-25 and 29-30, the [paper](#) describes three  $\beta$ -strands, comprised of residues 8-10, 23-26, and 28-31. Such differences are common because secondary structure assignments are method- and parameter-dependent. Secondary structure assignments could be recomputed with [ksdssp](#), but if the desired assignments are already known, it is much more efficient to change them directly:

*Command: [setattr](#) r isStrand true :8-10,23-26,28-31*

This command assigns values of the [residue attribute](#) named **isStrand**. In this case, it is not necessary to set **isStrand** (or **isHelix**) to false for any residues, since the original strand residues are still in strands and none of the new strand residues were in  $\alpha$ -helices. Secondary structure assignments can also be changed by [selecting](#) residues and using the [Selection Inspector](#).

The strands can be emphasized with color:

*Command: [color](#) blue #1.2*

*Command: [color](#) yellow #1.2 & strand*

(For those who prefer a graphical interface, there is also a [Color Secondary Structure](#) tool and associated **color by SS** function in the **Model Panel**.)

In the **Model Panel**, the **S (Shown)** checkboxes toggle [model display](#) without changing the display settings of individual atoms, bonds, and ribbon segments.

uncheck the **S** box for model 1.2

check the **S** box for model 1.2

The [modeldisplay](#) command does the same thing. Show all of the models:

*Command: [modeldisp](#)*

The **A (Active)** checkboxes in the **Model Panel** control what can be moved:

uncheck the **A** box for model 1.2 and try moving the structures [with the mouse](#); now only the other models can be rotated and translated

check the **A** box for model 1.2 and try moving the structures again

Reset to the original model positions:

*Command:* [reset](#)

We will use [Ensemble Cluster](#) to cluster each ensemble and identify representative structures, then [Ensemble Match](#) to compare the representatives.

Start [Ensemble Cluster](#) (under **Tools... MD/Ensemble Analysis**) and choose **1g1p** as the ensemble to cluster. Leave the **Parts to Match** blank to use all atoms and click **OK**. Results are shown in a cluster list dialog; three clusters were found. In that dialog,

if options are not shown under **Treatment of Chosen Clusters**, click the black arrowhead to reveal them

set the treatment options to **Color [all members]** and **Choose [representatives]** in **Model Panel**

choose all three cluster lines in the dialog with the mouse

Now the submodels of **1g1p** (#0) are shown in three different colors for the three clusters, and only the three cluster representatives are chosen in the left side of the **Model Panel**. In the right side of the **Model Panel**, click **show only** to hide all models except those three representatives. **Quit** from the cluster dialog, then delete the undisplayed members of the **1g1p** ensemble:

*Command:* [delete](#) #0/!display

This command deletes models with ID number 0 and model display turned off.

Start [Ensemble Cluster](#) again and cluster **1g1z** using all atoms. This time, four clusters are found. Keeping the treatment settings the same as used before, choose just the two clusters with more than one structure, and again click **show only** in the **Model Panel**. **Quit** from the cluster dialog and delete the undisplayed members of the **1g1z** ensemble:

*Command:* [delete](#) #1/!display

In the left side of the **Model Panel**, drag with the mouse to choose all five remaining models. Use **Model Panel** functions to show them, assign them unique colors, and spread them out into a plane:

**show** - show all five models

**rainbow...** click **OK** to rainbow-color the models using default settings

**tile...** click **OK** to spread the models out into a plane

Finally, compare the structures with [Ensemble Match](#) (under **Tools...** **MD/Ensemble Analysis**). Choose one ensemble as the reference and the other as the alternative. For **Parts to Match** just [specify](#) the backbone atoms:

@n,ca,c,o

Click **OK** to calculate the matches. Results are shown as a 3 x 2 (or 2 x 3) table with entries for all pairwise comparisons between the ensembles. The **A** and **D** buttons control model [activation for motion](#) and [model display](#), respectively. The numbers in the table are pairwise RMSDs using the atoms that were specified as **Parts to Match**.

	#1.1	#1.9
#0.9	3.828	2.131
#1.1	2.576	2.128
#1.9	2.396	3.065

The structures are not yet superimposed. Clicking a button next to an RMSD value performs the corresponding match and reports in the [status line](#) the number of atom pairs used. Superimpose each *cis*-proline conformation in model 1 on the most similar (lowest-RMSD) *trans*-proline conformation in model 0.

In this case, *cis* and *trans* refer to the peptide bond between leucine-12 and proline-13. Display these residues as ball-and-stick:

Command: [disp](#) :12-13

Command: [repr](#) bs

The proline rings look somewhat distorted, especially in the *cis*-proline structures (currently red and yellow). By default, the ribbon path is a smoothed B-spline that does not pass exactly through the true positions of the backbone atoms. Bonds to backbone atoms are drawn to the ribbon and thus may appear to be stretched. Try a cardinal spline instead, which gives a messier-looking ribbon but follows the backbone atom positions more closely, even when some smoothing is applied:

Command: [ribspline](#) cardinal

Command: [ribsp](#) card smooth strand

To return to the default B-spline ribbon:

Command: [ribsp](#) bspline

When finished viewing the structures, choose **File... Quit** from the menu to exit from Chimera.

# Trajectory and Ensemble Analysis Tutorial

This tutorial focuses on visualization and analysis of molecular dynamics (MD) trajectories and other structural ensembles with the [MD Movie](#) tool. [Part 1](#) uses an MD trajectory of a collagen peptide, and [Part 2](#) uses an NMR ensemble of Met-enkephalin.

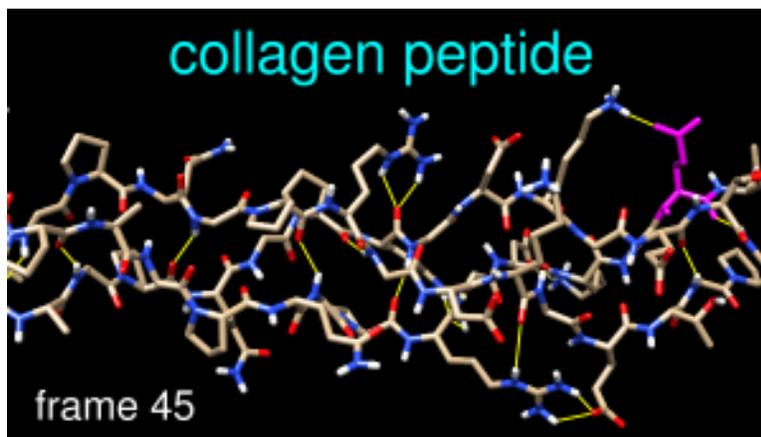
## Part 1 - Collagen Peptide

We will view an MD trajectory of the nonmutant collagen peptide described in:

[Severity of osteogenesis imperfecta and structure of a collagen-like peptide modeling a lethal mutation site](#). Radmer RJ, Klein TE. *Biochemistry*. 2004 May 11;43(18):5314-23.

(Thanks to the authors for providing the data!) To follow along, [download](#) the data files:

- [leap.top](#) - **Amber** parameter/topology file
- [md01.crd](#) - **Amber** trajectory file
- [collagen.meta](#) - metafile specifying these input files for [MD Movie](#)



On **Windows/Mac**, click the **chimera** icon; on **UNIX**, start Chimera from the system prompt:

```
unix: chimera
```

A splash screen will appear, to be replaced in a few seconds by the main Chimera [graphics window](#) or [Rapid Access](#) interface (it does not matter which, the following instructions will work with either). If you like, resize the Chimera window by dragging its lower right corner.

Show the [Command Line](#) (Tools... General Controls... Command Line) and start [MD Movie](#) (Tools... MD/Ensemble Analysis... MD Movie). In the resulting dialog, the inputs can be specified in two different ways:

- by setting the **Trajectory format** to **Amber** and browsing to the **Prmtop** file [leap.top](#) and the **Trajectory** file [md01.crd](#)
- by setting the **Trajectory format** to **metafile** and browsing to the file [collagen.meta](#) (it must be in the same directory or folder as the other two files). It contains the following lines, which simply specify the options and filenames that would otherwise be entered into the dialog:

```
amber
leap.top
md01.crd
```

Once the inputs have been specified, click **OK**. The first set of coordinates will be displayed and the **MD Movie** controller will appear. If the controller becomes obscured by other windows, it can be raised using its [instance in the Tools menu](#) (near the bottom of the menu, below the horizontal line).

Apply interactive preset #2 to display all atoms with heteroatom color-coding, hide hydrogens on carbons ([atom type](#) HC), and change to the stick representation:

Command: [preset](#) apply int 2

Command: [~disp](#) HC

Command: [repr](#) stick

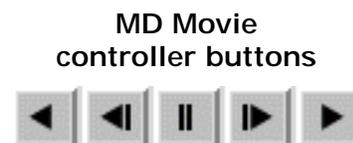
Show the backbone as ribbons instead of atoms, and change from the default B-spline ribbon to cardinal spline, which more closely follows the  $\alpha$ -carbon positions:

Command: [ribbon](#)

Command: [ribspline](#) cardinal

[Move and scale](#) the structure as desired throughout the tutorial. The structure contains three chains. Each chain is in a left-handed polyproline II helix conformation, and together the chains form the right-handed triple helix characteristic of fibrillar collagen. The ribbons are narrow like spaghetti because the peptides are not in a standard  $\alpha$ -helix or  $\beta$ -strand conformation.

Use the **MD Movie** controller to play the trajectory. From left to right, the buttons mean: play backward continuously; go back one step; stop; go forward one step; and play forward continuously. The rate of continuous play can be adjusted with the **Playback speed** slider. The **Frame** number is reported and can also be entered directly to view a specific frame. **Frame number** and **Step size** changes take effect when return (**Enter**) is pressed.



Show the amino acid sequence with **Tools... Sequence... Sequence** (pick any one of the three chains). Fibrillar collagen typically contains many -Gly-X-Y- repeats, where X is often Pro (proline) and Y is often Hyp (hydroxyproline). Both Pro and Hyp are shown as P in the sequence panel. Selecting residues highlights both the sequences and the structures:

Command: [sel](#) :gly

Command: [sel](#) :pro

Command: [sel](#) :hyp

**Quit** from the sequence panel.

It may be useful to hold certain atoms steady during trajectory playback. For example, hold Glu-86 steady to view its interactions:

Command: [sel](#) :86

(from the **MD Movie** controller menu) **Actions... Hold selection steady**

Command: [color](#) magenta sel

Command: [~sel](#)

Even though it is no longer selected, residue 86 will be held steady during playback (as possible; there will still

be internal motions) until **Hold selection steady** with a different selection or **Stop holding steady** is used. The structure can still be moved with the mouse, however. Try playing the trajectory with residue 86 held steady and then without holding any atoms steady (from the controller menu, choose **Actions... Stop holding steady**).

We will create a short movie of several frames in the trajectory. The following procedure is just one example; there are many possibilities of what to show, how to show it, whether to use a script, and so on.

Adjust the Chimera window to the dimensions desired for the movie. Turn off the ribbon to reveal the backbone atoms:

*Command:* [~ribbon](#)

Use [2D Labels](#) (**Tools... Utilities... 2D Labels**) to add a title. When **Use mouse for label placement** is checked, the left mouse button (button 1) is reassigned to labeling: clicking starts a new 2D label and previously created 2D labels can be repositioned by dragging. Click in the Chimera window where you would like to start a title and type in the title text; drag the text if you want to reposition it. Adjust the **Font size**, **Font style**, and **Color** (click the [color well](#), use the [Color Editor](#)) to your liking. Unchecking **Use mouse for label placement** returns the left mouse button to its previous function (by default, [rotation](#)).

Create another 2D label, this time using the [2dlabels](#) command so that the label will have a name:

*Command:* [2dlabels](#) create mylabel text temp

This label will be used to display the frame number. In the **2D Labels** dialog, make sure that **Use mouse for label placement** is checked, then drag the temporary text to near the lower left corner of the Chimera window. Adjust the settings of that label to your liking, then **Close** the **2D Labels** dialog.

Next, define a script to execute at each frame. Halt any playback. From the **MD Movie** controller menu, choose **Per-Frame... Define script**. Enter a script to be interpreted as **Chimera commands**:

```
findhbond linewidth 2 color yellow
2dlabels change mylabel text "frame <FRAME>"
```

Uncheck the option to **Use leading zeroes...** This script will calculate the hydrogen bonds in each frame, show them as yellow lines, and display the current frame number in the label named **mylabel**. Click **OK** to dismiss the dialog with the script. Play a few steps by clicking the button to go forward or backward one step at a time. The number and arrangement of H-bonds vary somewhat from step to step. (Although the number of H-bonds cannot be accessed in **Chimera commands**, a **Python** script could be used to display this information. For example, [hbcoun.py](#) would show the H-bond count instead of the frame number in **mylabel**.)

Finally, [record a movie](#). Halt any playback, but move the **Playback speed** slider all the way to the right. From the controller menu, choose **File... Record movie**. If a dialog with an MPEG license agreement appears, click **Accept** since the movie will not be used for commercial purposes. In the dialog for recording,

- specify a convenient name and location for the output movie file
- choose a **File type** you will be able to play back on your computer (the choices are MPEG-1, MPEG-2, MPEG-4, AVI MSMPEG-4v2, and Quicktime)
- change the **Ending frame** to 25
- click **Record**

Frames 1-25 will then be played, saved as images, and automatically assembled into a movie file. View the resulting 1-second movie with the appropriate application on your computer.

Click **Quit** on the controller to close the trajectory and exit from **MD Movie**. An easy way to delete all of the 2D labels is with **File... Close Session**. Go on to [Part 2](#) below, **OR** terminate the Chimera session:

Command: [stop](#)

---

## Part 2 - Met-Enkephalin

We will view an NMR ensemble of Met-enkephalin in negatively charged bicelles, as described in:

[A multidimensional 1H NMR investigation of the conformation of methionine-enkephalin in fast-tumbling bicelles](#). Marcotte I, Separovic F, Auger M, Gagné SM. *Biophys J*. 2004 Mar;86(3):1587-600.

To follow along, [download](#) the data file [1plx.pdb](#).

With Chimera started and the [Command Line](#) shown (as in [Part 1](#)), choose **Tools... MD/Ensemble Analysis... MD Movie**. In the resulting dialog, choose **PDB** as the **Trajectory format** and indicate that the frames are contained in a **single file**. Browse to the file [1plx.pdb](#), set the input location, and then click **OK**.

The first set of coordinates will be displayed and the **MD Movie** controller will appear. If the controller becomes obscured by other windows, it can be raised using its [instance in the Tools menu](#) (near the bottom of the menu, below the horizontal line).

Use a preset to show the structure as sticks color-coded by heteroatom:

Command: [preset](#) apply int 1

(The structure may have been shown that way already, depending on your preference settings.) Only the polar hydrogens are shown; hydrogens on carbons are present but undisplayed. [Move and scale](#) the structure as desired throughout the tutorial.

This structure is Met-enkephalin, with the sequence Tyr-Gly-Gly-Phe-Met. Enkephalins are neuropeptides that activate opioid receptors. Different subtypes of opioid receptors mediate different but overlapping responses in the body. For example, molecules that selectively activate  $\mu$ -opioid receptors are more effective for treating severe pain than molecules that selectively activate  $\delta$ -opioid receptors, but are also more likely to cause constipation. The conformations of molecules that bind opioid receptors (enkephalins, morphine, *etc.*) are of interest because they influence the selectivity of receptor binding and thus the physiological response.

Use the **MD Movie** controller to flip through the different conformations, as described [above](#). The frames do not reflect time ordering, as this is an NMR ensemble rather than a trajectory.

It is thought (see the [reference](#)) that a conformation of enkephalin in which the Tyr and Phe rings point in

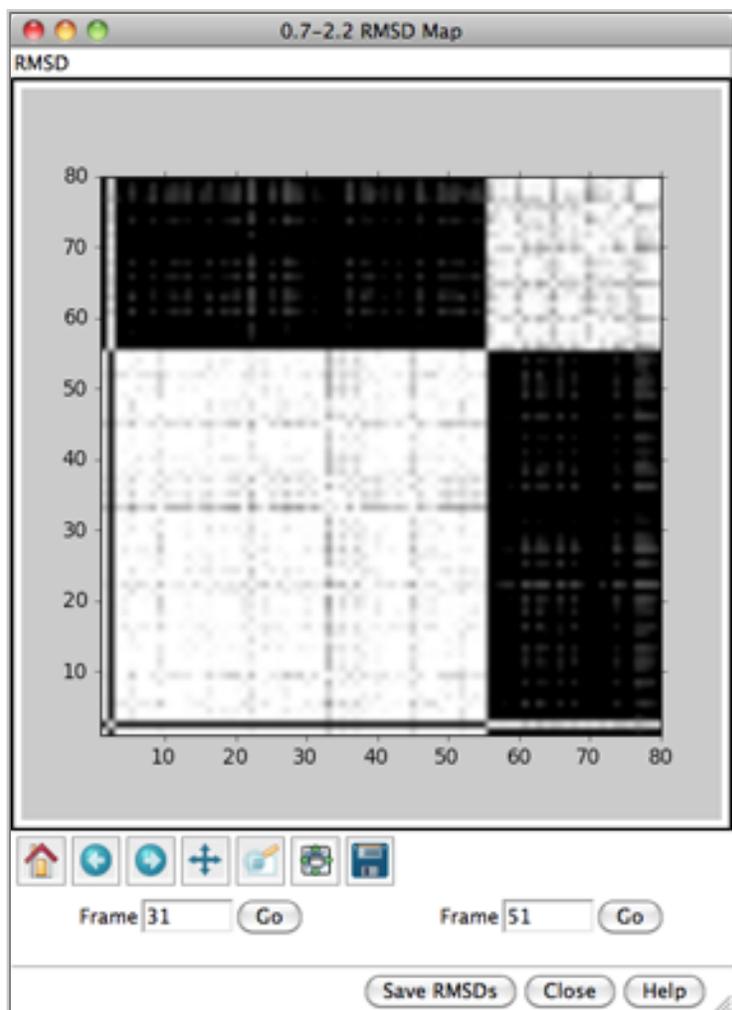
different directions (like frames 1 and 25) binds to  $\mu$ -opioid receptors, and a conformation in which they point in roughly the same direction (like frames 2 and 80) binds to  $\delta$ -opioid receptors.

One way to analyze the ensemble is to calculate root-mean-square deviations (RMSDs) between pairs of frames. From the controller menu, choose **Analysis...**

**RMSD map.** Click **Apply** on the RMSD parameters dialog to perform the calculation without closing the dialog. This will compute all pairwise RMSDs between frames and show the result as a map in grayscale. After the initial calculation, the map will be recolored to enhance contrast.

In the RMSD map, the axes are frame numbers; lighter squares reflect pairs of frames with lower RMSDs (more similar structures) and darker squares reflect pairs of frames with higher RMSDs (less similar structures). Mousing over the map shows the RMSD values and the numbers of the frames being compared. Clicking on the map enters the corresponding pair of **Frame** numbers below the map. Clicking **Go** then displays that frame in the main Chimera window.

Roughly, the lower left block of white in the map represents conformations more similar to a  $\mu$ -binding conformation, and the upper right block of white represents conformations more similar to a  $\delta$ -binding conformation. Similar conformations are mostly grouped together in this ensemble, but will not necessarily be grouped together in ensembles in general. Also, ensembles usually contain many more groups of conformations, especially for larger structures with more degrees of freedom.



All nonhydrogen atoms were used in the RMSD calculation, because although the parameter **Restrict map to current selection, if any** was set to **true**, nothing was selected. Select the backbone atoms,

*Command:* [sel](#) @n,ca,c,o

and this time click **OK** to dismiss the RMSD parameters dialog and perform the calculation. Although the two maps span different ranges in RMSD, they reveal essentially the same groups of conformations. **Close** both maps.

Another way to analyze the ensemble is to calculate spatial occupancy maps for atoms of interest. From the controller menu, choose **Analysis... Calculate occupancy**. The resulting dialog will show the warning message “No atoms being held steady.” This means you have not previously [selected](#) a set of atoms and chosen **Actions... Hold selection steady** from the controller menu for this ensemble.

Why might it be useful to hold atoms steady while calculating occupancy? The contents of different frames may move around enough to obscure certain spatial patterns. Even if the structure as a whole is held fairly steady, one may want to hold a particular set of atoms (such as a sidechain) steady to examine local interactions. However, if the structure or region of interest is already sufficiently steady, the “hold steady” step can be omitted.

One approach to analyzing the relative positions of the aromatic rings is to hold the Phe ring atoms steady and map the occupancy of the Tyr ring atoms:

1. [Select](#) the six Phe ring atoms. One way is to pick them from the screen (Ctrl-click on one, Shift-Ctrl-click on each of the other five). Another way is with a command:

*Command:* [sel](#) :phe & aromatic ring

2. Choose **Actions... Hold selection steady** from the controller menu.
3. [Select](#) the six Tyr ring atoms, by picking or with a command:

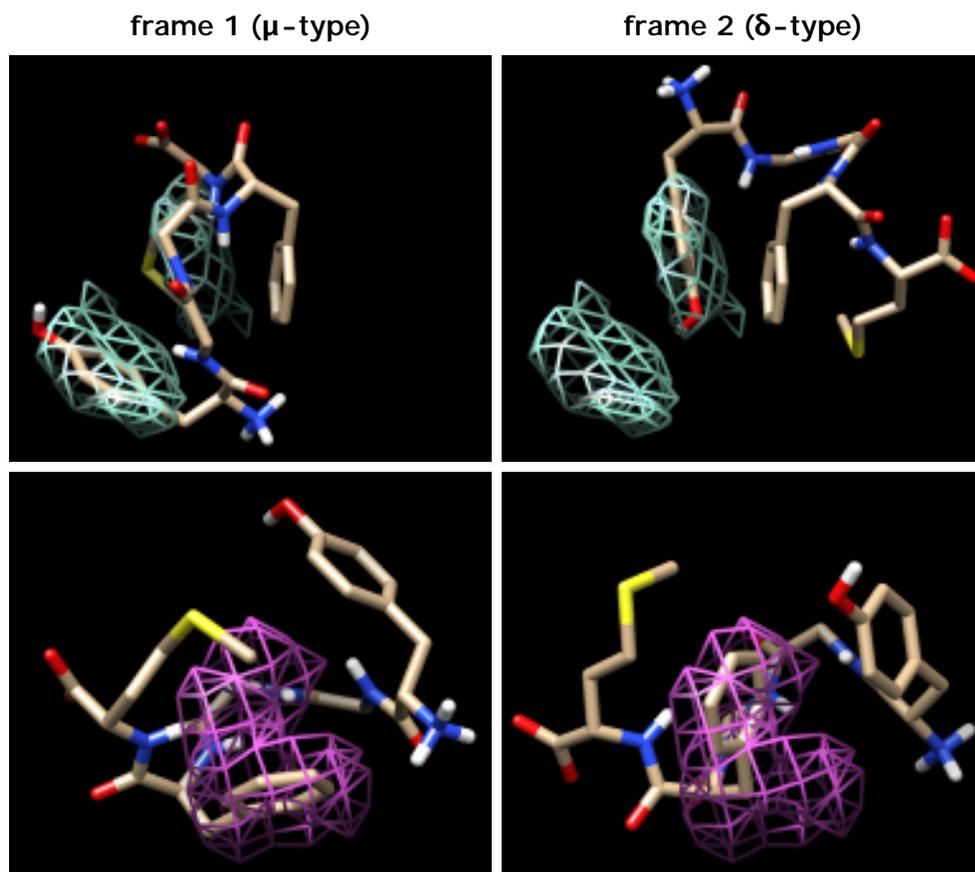
*Command:* [sel](#) :tyr & aromatic ring

4. If the occupancy dialog is not already up, choose **Analysis... Calculate occupancy** from the controller menu.
5. Click **OK** on the occupancy dialog.

When the map has been computed, the [Volume Viewer](#) tool will appear. This tool shows a histogram of the values in the map; the slider controls what contour level is displayed.

Clear the selection in Chimera ([Select... Clear Selection](#)). In the [Volume Viewer](#) tool, change the **Style** from **surface** to **mesh** and move the slider to a **Level** of approximately 2. Optionally, change the color of the mesh surface by clicking the [color well](#) below the histogram and using the [Color Editor](#).

You will see two main blobs or volumes representing probable positions of the Tyr ring relative to the Phe ring, similar to the upper set of images at right. The larger volume represents  $\mu$ -type conformations and the smaller represents  $\delta$ -type conformations.



For the images, the mesh lines were smoothed and their thickness increased. This can be done by choosing **Features... Surface and Mesh options** from the [Volume Viewer](#) menu and using the resulting settings in the dialog (see the [Volume Viewer documentation](#) for details on its many features), or by using the [volume](#) command.

If desired, flip through the ensemble (still holding the Phe ring steady) to verify that the volumes show areas occupied by the Tyr ring. To delete the volume display, choose **File... Close map** from the [Volume Viewer](#) menu.

Finally, calculate a map without holding any atoms steady. Choose **Actions... Stop holding steady** from the controller menu. Since this ensemble maintains the Tyr ring in roughly the same place, simply map the occupancy of the Phe ring atoms. Select the Phe ring atoms (as [above](#)), choose **Analysis... Calculate occupancy**, and click **OK**. This time, the conformations are not as well separated; two lobes of occupancy are apparent at a contour level of approximately 2, similar to the lower set of images. One lobe represents  $\mu$ -type conformations and the other represents  $\delta$ -type conformations.

When finished with the Met-enkephalin ensemble, quit from Chimera (**File... Quit**).

---

meng-at-cgl.ucsf.edu / January 2012

## ViewDock Tutorial

Given the structures of ligand and receptor molecules, docking programs calculate possible binding modes. In virtual screening, small organic compounds (typically from a database of many thousands) are treated as possible ligands, and a target macromolecule is treated as the receptor. Various scoring methods are used to identify the most favorable binding modes of a given compound and then to rank the compounds. Researchers then screen the results interactively to decide which compounds should be tested in the real world.

The [ViewDock](#) tool facilitates interactive analysis of receptor-ligand docking results. This tutorial uses the results of searching a very small database against H-ras (Protein Data Bank entry 121P) with the program [DOCK](#).

To follow along with the tutorial, first [download](#) the following files to a convenient location (all should be placed in the same folder or directory):

- [ras.mol2](#) - the docked molecules output by DOCK 4, in Mol2 format
- [receptor.pdb](#) - the structure of the receptor, H-ras, from PDB entry 121p
- [GCP.pdb](#) - the co-crystallized ligand GCP (a GTP analog) from PDB entry 121p, for comparison with docked molecules
- [setup.com](#) - a file containing commands to set up the viewing context

On **Windows/Mac**, click the **chimera** icon; on **UNIX**, start Chimera from the system prompt:

```
unix: chimera
```

A splash screen will appear, to be replaced in a few seconds by the main Chimera [graphics window](#) or [Rapid Access](#) interface (it does not matter which, the following instructions will work with either). If you like, resize the Chimera window by dragging its lower right corner.

Start **ViewDock (Tools... Surface/Binding Analysis... ViewDock)**, and with the resulting dialog, locate and open `ras.mol2`, the file of docked ligands.

The **ViewDock ListBox** will appear, and the first ligand in the file will be displayed in the graphics window. Move the **ListBox** aside if it is obstructing the graphics window or any of the other tools.

Next, we will open the structures of the receptor and its co-crystallized ligand and display them in a way that is convenient for evaluating the docked molecules. Often many different files of docked molecules will need to be evaluated in the context of the same receptor. It can be tedious to set up the same view over and over. One approach is to save a [session](#) with the target protein displayed as desired, and then repeatedly restart that

session before opening different files of docked ligands with **ViewDock**. Another approach (used in this tutorial) is to put the necessary commands in a file and simply execute the [command file](#) as needed.

The command file ([setup.com](#)) contains:

```
open receptor.pdb
open GCP.pdb
preset apply interactive 1
color aquamarine #1
disp #1 & #0 z<5
color orange,a #1@o=
color medium blue,a #1@n=
color magenta #2
repr bs #2
```

Simply opening a command file will execute its contents. Choose **File... Open**, make sure the file type is **all (guess type)** or **Chimera commands**, and locate and open `setup.com`.

The lowest available model number is used for each successive structure opened, so the docked molecules (opened before the command file) are model number 0, the receptor is model 1, and GCP is model 2. Besides opening structures, the command file applies a ribbon/sticks [preset](#), colors the receptor aquamarine, and displays receptor residues within 5 Å of any docked molecule. Oxygen atoms in the receptor are colored orange, nitrogens medium blue. The co-crystallized ligand GCP is shown in magenta ball-and-stick.

Throughout the tutorial, adjust the view as desired [with the mouse](#) and [Side View](#) (**Tools... Viewing Controls... Side View**).

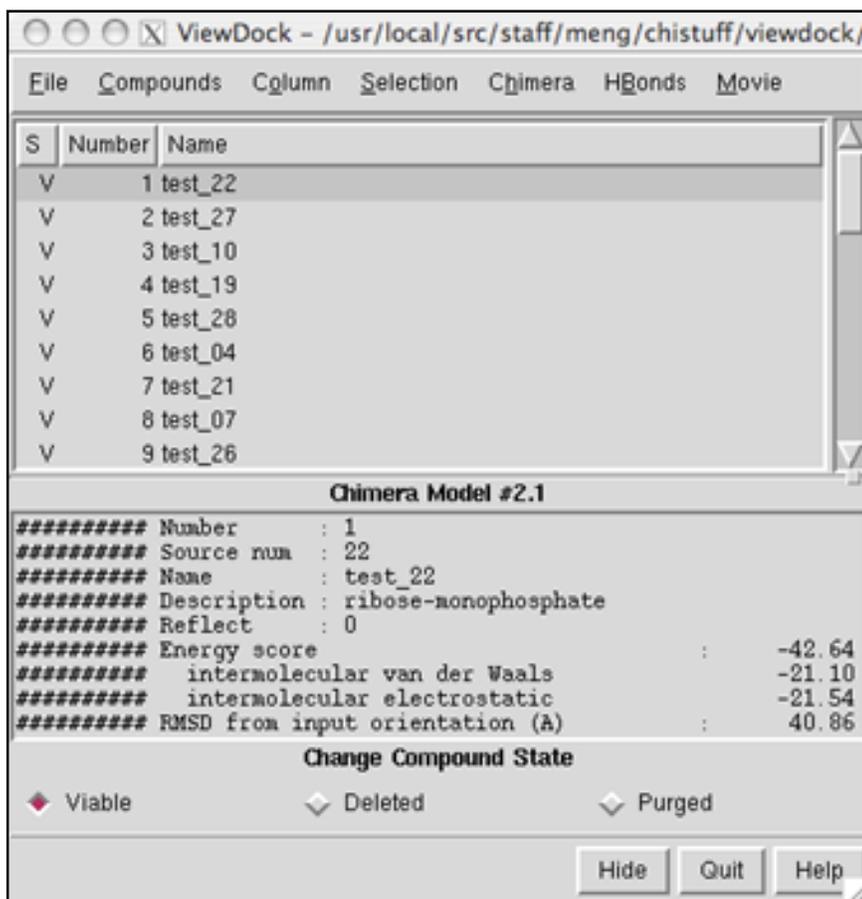
The co-crystallized ligand GCP (shown in magenta) indicates the location of the active site. Ctrl-click to select any atom in GCP, press the keyboard up arrow to promote the selection to the whole residue, and then hide it:

*Menu: Actions... Atoms/Bonds... hide*

*Menu: Select... Clear Selection*

The docked compounds are enumerated in the top part of the **ViewDock ListBox**. If the **ListBox** has become obscured by other windows, it can be resurrected with **Tools... ViewDock** (near the bottom of the menu, below the horizontal line)... **Raise**. Since in this case **Name** is not very informative, it may be helpful to add other descriptors to the listing. Use the **Column** menu to show **Description** and **Energy score**, and to hide **Name** and **Number**.

Clicking on a line *chooses* the corresponding compound: the line is highlighted, just the chosen compound is shown in the main graphics window, and more detailed information is shown in the lower part of the **ListBox**. Try clicking various lines in the **ListBox** to choose different docked molecules. Multiple compounds may be chosen at once. **Ctrl**-click adds to an existing choice rather than replacing it. To choose a block of compounds without having to hold down the mouse button, click on the first (or last) and then **Shift**-click on the last (or



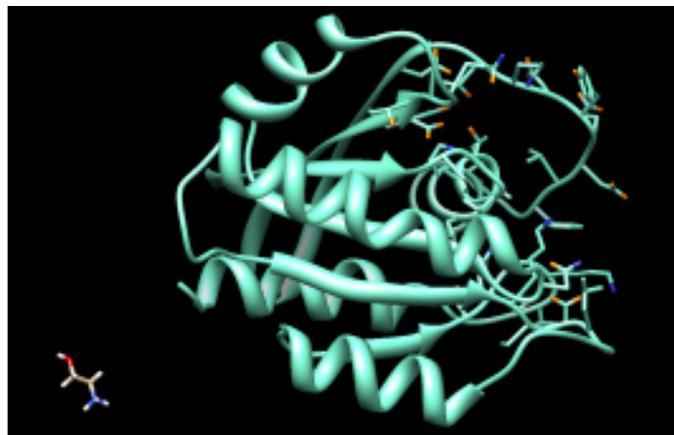
first) in the desired block.

The listing can be sorted by any column, by clicking on the header. Make sure the list is sorted by **Energy score**, with the most negative values (which are the most favorable) at the top. Scroll down to the lowest line in the top panel of the **ListBox** and click on it to choose the worst-scoring molecule.

As shown in the figure, this compound is not docked in the active site like the others. Its docking scores are zero.

There are three mutually exclusive states that can be assigned to docked compounds. **Viable** compounds are interesting (or have not been looked at yet), **Deleted** compounds are less interesting but may deserve another look, and **Purged** compounds are definitely not interesting. The **S** column shows **V**, **D**, and **P** to indicate these states. Viable and deleted but not purged molecules are included when **File... Rewrite** is used. Change the status of the worst-scoring molecule to purged by clicking the **Purged** checkbox near the bottom of the **ListBox**. Note that its listing disappears; make it reappear by checking the box next to **List Purged** in the **Compounds** menu.

worst-scoring molecule



Normally, a user will click on successive lines, examine the compounds in the binding site, and change the status of less interesting compounds to deleted or purged. Compounds can also be [chosen by descriptor values](#) and then changed in status collectively. As an example, we will calculate hydrogen bonds between the compounds and the receptor, then mark compounds with only 0-1 hydrogen bonds as purged.

**HBonds... Add Count to Entire Receptor** brings up the [FindHBond](#) tool. In that dialog, make sure the **inter-model** mode is set. The hydrogen bonds will be shown as lines; increase the **Line width** to 3 and change the **H-bond color** to yellow (clicking the [color well](#) opens the [Color Editor](#), in which a new color can be chosen; one way to change the color is to enter a new **Color name**, in this case **yellow**). Click **OK**. When the calculation is finished, new columns of descriptors will appear in the **ListBox**. Again, individual compounds can be examined by clicking on their respective lines in the **ListBox**. Use the **Column** menu to hide the descriptors **HBond Ligand Atoms** and **HBond Receptor Atoms** (the numbers of ligand and receptor atoms, respectively, participating in the detected ligand-receptor hydrogen bonds).

**Compounds... Choose by Value** opens an interface with several sections. Choose from **Viable** compounds and uncheck the boxes next to **Description** and **Energy score** to collapse the corresponding sections. In the **HBonds (all)** section, move the sliders to include 0-1 hydrogen bonds. A message near the top of the **Choose by Value** dialog will report that 17 of the 29 viable compounds meet the criteria. Click **OK** to choose the compounds and dismiss the dialog. The 17 viable compounds with 0-1 hydrogen bonds to the receptor will be chosen in the **ListBox** and displayed in the main Chimera window. Change these compounds to purged by clicking the **Purged** checkbox near the bottom of the **ListBox**. Uncheck the box next to **List Purged** in the **Compounds** menu to remove the purged compounds from the listing.

binding site surface and H-bonds

Finally, flip through the remaining listed compounds with the **Movie** feature. First, show the surface of the protein and make it transparent:

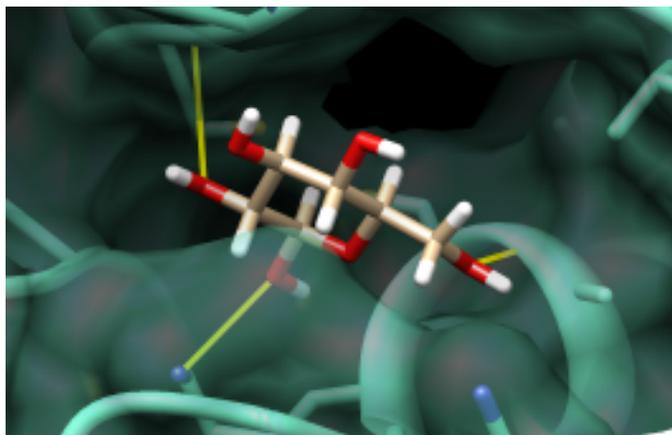
*Menu: Select... Structure... protein*

*Menu: Actions... Surface... show*

*Menu: Actions... Surface... transparency... 60%*

*Menu: Select... Clear Selection*

In the **ListBox** menu, choosing **Movie... Play** flips through all of the listed compounds, in the order in which they are listed, regardless of status. It is possible to change the view and move the molecules around while the movie is playing. The movie will loop continuously through the list until halted with **Movie... Stop**. The length of time each compound is shown can be controlled with **Movie... Options**. If molecules are “unlisted” using the checkboxes in the **Compounds** menu, they will not be included in the movie; in addition, the order of display depends on how the molecules are sorted. No matter how the molecules are sorted in the **ListBox**, however, they remain in the original order (minus any purged compounds) in output files created with **File... Rewrite**. Once you have seen enough, stop the movie and exit from Chimera:



*Menu: File... Quit*

---

meng-at-cgl.ucsf.edu / January 2012



# Chimera User's Guide

CHIMERA

Tutorials

Basic Functions

Tools

[Tutorials Index](#)
[Intro to PDB Format](#)
[Background](#)

- [Atomic Coordinates](#)
- [Secondary Structure](#)

[Examples](#)
[Common Errors](#)
[Hydrogen Atoms](#)
[PQR Format](#)

## Introduction to Protein Data Bank Format

Protein Data Bank (PDB) format is a standard for files containing atomic coordinates. It is used for structures in the [Protein Data Bank](#) and is read and written by many programs. While this short description will suffice for many users, those in need of further details should consult the [definitive description](#). The complete PDB file specification provides for a wealth of information, including authors, literature references, and the method of structure determination.

PDB format consists of lines of information in a text file. Each line of information in the file is called a *record*. A PDB file generally contains several different types of records, arranged in a specific order to describe a structure.

Selected Protein Data Bank Record Types	
Record Type	Data Provided by Record
ATOM	atomic coordinate record containing the X,Y,Z orthogonal Å coordinates for atoms in standard residues (amino acids and nucleic acids).
HETATM	atomic coordinate record containing the X,Y,Z orthogonal Å coordinates for atoms in nonstandard residues. Nonstandard residues include inhibitors, cofactors, ions, and solvent. The only functional difference from ATOM records is that HETATM residues are by default not connected to other residues. Note that water residues should be in HETATM records.
TER	indicates the end of a chain of residues. For example, a hemoglobin molecule consists of four subunit chains that are not connected. TER indicates the end of a chain and prevents the display of a connection to the next chain.
HELIX	indicates the location and type (right-handed alpha, <i>etc.</i> ) of helices. One record per helix.
SHEET	indicates the location, sense (anti-parallel, <i>etc.</i> ) and registration with respect to the previous strand in the sheet (if any) of each strand in the model. One record per strand.
SSBOND	defines disulfide bond linkages between cysteine residues.

The formats of these record types are given in the tables below. Older PDB files may not adhere completely to the specifications. Some differences between older and newer files occur in the fields following the temperature factor in ATOM and HETATM records; these fields are omitted from the [examples](#). Some fields are frequently blank, such as the alternate location indicator when an atom does not have alternate locations.

Protein Data Bank Format: Coordinate Section				
Record Type	Columns	Data	Justification	Data Type
ATOM	1-4	"ATOM"		character
	7-11 <sup>#</sup>	Atom serial number	right	integer
	13-16	Atom name	left <sup>*</sup>	character
	17	Alternate location indicator		character
	18-20 <sup>§</sup>	Residue name	right	character
	22	Chain identifier		character
	23-26	Residue sequence number	right	integer
	27	Code for insertions of residues		character
	31-38	X orthogonal Å coordinate	right	real (8.3)
	39-46	Y orthogonal Å coordinate	right	real (8.3)
	47-54	Z orthogonal Å coordinate	right	real (8.3)
	55-60	Occupancy	right	real (6.2)
	61-66	Temperature factor	right	real (6.2)
	73-76	Segment identifier <sup>¶</sup>	left	character
	77-78	Element symbol	right	character
79-80	Charge		character	
HETATM	1-6	"HETATM"		character
	7-80	same as ATOM records		
TER	1-3	"TER"		character
	7-11 <sup>#</sup>	Serial number	right	integer
	18-20 <sup>§</sup>	Residue name	right	character
	22	Chain identifier		character
	23-26	Residue sequence number	right	integer
	27	Code for insertions of residues		character

<sup>#</sup>Chimera allows (nonstandard) use of columns 6-11 for the integer atom serial number in ATOM records, and in TER records, only the "TER" is required.

<sup>\*</sup>Atom names start with element symbols right-justified in columns 13-14 as permitted by the length of the name. For example, the symbol FE for iron appears in columns 13-14, whereas the symbol C for carbon appears in column 14 (see [Misaligned Atom Names](#)). If an atom name

has four characters, however, it must start in column 13 even if the element symbol is a single character (for example, see [Hydrogen Atoms](#)).

§Chimera allows (nonstandard) use of four-character residue names occupying an additional column to the right.

†Segment identifier is obsolete, but still used by some programs.

Protein Data Bank Format: Protein Secondary Structure and Disulfides				
Record Type	Columns	Data	Justification	Data Type
HELIX	1-5	"HELIX"		character
	8-10	Helix serial number	right	integer
	12-14	Helix identifier	right	character
	16-18 <sup>§</sup>	Initial residue name	right	character
	20	Chain identifier		character
	22-25	Residue sequence number	right	integer
	26	Code for insertions of residues		character
	28-30 <sup>§</sup>	Terminal residue name	right	character
	32	Chain identifier		character
	34-37	Residue sequence number	right	integer
	38	Code for insertions of residues		character
	39-40	Type of helix <sup>†</sup>	right	integer
	41-70	Comment	left	character
	72-76	Length of helix	right	integer
SHEET	1-5	"SHEET"		character
	8-10	Strand number (in current sheet)	right	integer
	12-14	Sheet identifier	right	character
	15-16	Number of strands (in current sheet)	right	integer
	18-20 <sup>§</sup>	Initial residue name	right	character
	22	Chain identifier		character
	23-26	Residue sequence number	right	integer
	27	Code for insertions of residues		character
	29-31 <sup>§</sup>	Terminal residue name	right	character

33	Chain identifier		character
34-37	Residue sequence number	right	integer
38	Code for insertions of residues		character
39-40	Strand sense with respect to previous <sup>‡</sup>	right	integer
<p>The following fields identify two atoms involved in a hydrogen bond, the first in the current strand and the second in the previous strand. These fields should be blank for strand 1 (the first strand in a sheet).</p>			
42-45	Atom name (as per ATOM record)	left	character
46-48 <sup>§</sup>	Residue name	right	character
50	Chain identifier		character
51-54	Residue sequence number	right	integer
55	Code for insertions of residues		character
57-60	Atom name (as per ATOM record)	left	character
61-63 <sup>§</sup>	Residue name	right	character
65	Chain identifier		character
66-69	Residue sequence number	right	integer
70	Code for insertions of residues		character
SSBOND	1-6	"SSBOND"	character
	8-10	Serial number	right integer
	12-14	Residue name ("CYS")	right character
	16	Chain identifier	character
	18-21	Residue sequence number	right integer
	22	Code for insertions of residues	character
	26-28	Residue name ("CYS")	right character
	30	Chain identifier	character
	32-35	Residue sequence number	right integer
	36	Code for insertions of residues	character
	60-65	Symmetry operator for first residue	right integer

	67-72	Symmetry operator for second residue	right	integer
	74-78	Length of disulfide bond	right	real (5.2)

†Helix types:

1	Right-handed alpha (default)	6	Left-handed alpha
2	Right-handed omega	7	Left-handed omega
3	Right-handed pi	8	Left-handed gamma
4	Right-handed gamma	9	2/7 ribbon/helix
5	Right-handed 3/10	10	Polyproline

‡Sense is 0 for strand 1 (the first strand in a sheet), 1 for parallel, and -1 for antiparallel.

For those who are familiar with the FORTRAN programming language, the following format descriptions will be meaningful. Those unfamiliar with FORTRAN should ignore this gibberish:

**ATOM** Format ( A6,I5,1X,A4,A1,A3,1X,A1,I4,A1,3X,3F8.3,2F6.2,10X,A2,A2 )  
**HETATM**

**HELIX** Format ( A6,1X,I3,1X,A3,2(1X,A3,1X,A1,1X,I4,A1),I2,A30,1X,I5 )

**SHEET** Format ( A6,1X,I3,1X,A3,I2,2(1X,A3,1X,A1,I4,A1),I2,2(1X,A4,A3,1X,A1,I4,A1) )

**SSBOND** Format ( A6,1X,I3,1X,A3,1X,A1,1X,I4,A1,3X,A3,1X,A1,1X,I4,A1,23X,2(2I3,1X),F5.2 )

## Examples of PDB Format

Glucagon is a small protein of 29 amino acids in a single chain. The first residue is the amino-terminal amino acid, histidine, which is followed by a serine residue and then a glutamine. The coordinate information (entry **1gcn**) starts with:

```

ATOM      1  N   HIS  A   1         49.668  24.248  10.436  1.00 25.00           N
ATOM      2  CA  HIS  A   1         50.197  25.578  10.784  1.00 16.00           C
ATOM      3  C   HIS  A   1         49.169  26.701  10.917  1.00 16.00           C
ATOM      4  O   HIS  A   1         48.241  26.524  11.749  1.00 16.00           O
ATOM      5  CB  HIS  A   1         51.312  26.048   9.843  1.00 16.00           C
ATOM      6  CG  HIS  A   1         50.958  26.068   8.340  1.00 16.00           C
ATOM      7  ND1 HIS  A   1         49.636  26.144   7.860  1.00 16.00           N
ATOM      8  CD2 HIS  A   1         51.797  26.043   7.286  1.00 16.00           C
ATOM      9  CE1 HIS  A   1         49.691  26.152   6.454  1.00 17.00           C
ATOM     10  NE2 HIS  A   1         51.046  26.090   6.098  1.00 17.00           N
ATOM     11  N   SER  A   2         49.788  27.850  10.784  1.00 16.00           N
ATOM     12  CA  SER  A   2         49.138  29.147  10.620  1.00 15.00           C
ATOM     13  C   SER  A   2         47.713  29.006  10.110  1.00 15.00           C
ATOM     14  O   SER  A   2         46.740  29.251  10.864  1.00 15.00           O
ATOM     15  CB  SER  A   2         49.875  29.930   9.569  1.00 16.00           C
ATOM     16  OG  SER  A   2         49.145  31.057   9.176  1.00 19.00           O
ATOM     17  N   GLN  A   3         47.620  28.367   8.973  1.00 15.00           N

```

ATOM	18	CA	GLN	A	3	46.287	28.193	8.308	1.00	14.00	C
ATOM	19	C	GLN	A	3	45.406	27.172	8.963	1.00	14.00	C

Notice that each line or record begins with the record type ATOM. The atom serial number is the next item in each record.

The atom name is the third item in the record. Notice that the first one or two characters of the atom name consists of the chemical symbol for the atom type. All the atom names beginning with C are carbon atoms; N indicates a nitrogen and O indicates oxygen. In amino acid residues, the next character is the remoteness indicator code, which is transliterated according to:

$\alpha$  A  
 $\beta$  B  
 $\gamma$  G  
 $\delta$  D  
 $\epsilon$  E  
 $\zeta$  Z  
 $\eta$  H

The next character of the atom name is a branch indicator, if required.

The next data field is the residue type. Notice that each record contains the residue type. In this example, the first residue in the chain is HIS (histidine) and the second residue is a SER (serine).

The next data field contains the chain identifier, in this case A.

The next data field contains the residue sequence number. Notice that as the residue changes from histidine to serine, the residue number changes from 1 to 2. Two like residues may be adjacent to one another, so the residue number is important for distinguishing between them.

The next three data fields contain the X, Y, and Z coordinate values, respectively. The last three fields shown are the occupancy, temperature factor (B-factor), and element symbol.

The spacing of the data fields is crucial. If a data field does not apply, it should be left blank.

The glucagon data file continues in this manner until the final residue is reached:

ATOM	239	N	THR	A	29	3.391	19.940	12.762	1.00	21.00	N
ATOM	240	CA	THR	A	29	2.014	19.761	13.283	1.00	21.00	C
ATOM	241	C	THR	A	29	0.826	19.943	12.332	1.00	23.00	C
ATOM	242	O	THR	A	29	0.932	19.600	11.133	1.00	30.00	O
ATOM	243	CB	THR	A	29	1.845	20.667	14.505	1.00	21.00	C
ATOM	244	OG1	THR	A	29	1.214	21.893	14.153	1.00	21.00	O
ATOM	245	CG2	THR	A	29	3.180	20.968	15.185	1.00	21.00	C
ATOM	246	OXT	THR	A	29	-0.317	20.109	12.824	1.00	25.00	O
TER	247		THR	A	29						

Note that this residue includes the extra oxygen atom OXT on the terminal carboxyl group. Other than OXT and the rarely seen HXT, atoms in standard nucleotides and amino acids in version 3.0 PDB files are named according to the IUPAC recommendations (*Pure Appl Chem* **70**:117 (1998) [[abstract](#)] [[PDF](#)]). The TER record terminates the amino acid chain.

A more complicated protein, hemoglobin, consists of four amino acid chains, each with an associated heme group. There are two alpha chains (identifiers A and C) and two beta chains (identifiers B and D). The first ten lines of coordinates for this molecule (entry **3hhb**) are:

```

ATOM      1  N   VAL A   1      6.452  16.459   4.843   7.00  47.38      N
ATOM      2  CA  VAL A   1      7.060  17.792   4.760   6.00  48.47      C
ATOM      3  C   VAL A   1      8.561  17.703   5.038   6.00  37.13      C
ATOM      4  O   VAL A   1      8.992  17.182   6.072   8.00  36.25      O
ATOM      5  CB  VAL A   1      6.342  18.738   5.727   6.00  55.13      C
ATOM      6  CG1 VAL A   1      7.114  20.033   5.993   6.00  54.30      C
ATOM      7  CG2 VAL A   1      4.924  19.032   5.232   6.00  64.75      C
ATOM      8  N   LEU A   2      9.333  18.209   4.095   7.00  30.18      N
ATOM      9  CA  LEU A   2     10.785  18.159   4.237   6.00  35.60      C
ATOM     10  C   LEU A   2     11.247  19.305   5.133   6.00  35.47      C

```

At the end of chain A, the heme group records appear:

```

ATOM    1058  N   ARG A  141     -6.466  12.036 -10.348   7.00  19.11      N
ATOM    1059  CA  ARG A  141     -7.922  12.248 -10.253   6.00  26.80      C
ATOM    1060  C   ARG A  141     -8.119  13.499  -9.393   6.00  28.93      C
ATOM    1061  O   ARG A  141     -7.112  13.967  -8.853   8.00  28.68      O
ATOM    1062  CB  ARG A  141     -8.639  11.005  -9.687   6.00  24.11      C
ATOM    1063  CG  ARG A  141     -8.153  10.551  -8.308   6.00  19.20      C
ATOM    1064  CD  ARG A  141     -8.914   9.319  -7.796   6.00  21.53      C
ATOM    1065  NE  ARG A  141     -8.517   9.076  -6.403   7.00  20.93      N
ATOM    1066  CZ  ARG A  141     -9.142   8.234  -5.593   6.00  23.56      C
ATOM    1067  NH1 ARG A  141    -10.150   7.487  -6.019   7.00  19.04      N
ATOM    1068  NH2 ARG A  141     -8.725   8.129  -4.343   7.00  25.11      N
ATOM    1069  OXT ARG A  141     -9.233  14.024  -9.296   8.00  40.35      O
TER     1070          ARG A  141
HETATM  1071  FE   HEM A   1      8.128   7.371 -15.022  24.00  16.74      FE
HETATM  1072  CHA HEM A   1      8.617   7.879 -18.361   6.00  17.74      C
HETATM  1073  CHB HEM A   1     10.356  10.005 -14.319   6.00  18.92      C
HETATM  1074  CHC HEM A   1      8.307   6.456 -11.669   6.00  11.00      C
HETATM  1075  CHD HEM A   1      6.928   4.145 -15.725   6.00  13.25      C

```

The last residue in the alpha chain is an ARG (arginine). Again, the extra oxygen atom OXT appears in the terminal carboxyl group. The TER record indicates the end of the peptide chain. It is important to have TER records at the end of peptide chains so a bond is not drawn from the end of one chain to the start of another.

In the example above, the TER record is correct and should be present, but the molecule chain would still be terminated at that point even without a TER record, because HETATM residues are not connected to other residues or to each other. The heme group is a single residue made up of HETATM records.

After the heme group associated with chain A, chain B begins:

```

HETATM  1109  CAD HEM A   1      7.618   5.696 -20.432   6.00  21.38      C
HETATM  1110  CBD HEM A   1      8.947   5.143 -20.947   6.00  29.03      C
HETATM  1111  CGD HEM A   1      9.047   5.155 -22.461   6.00  30.08      C
HETATM  1112  OLD HEM A   1     10.139   5.458 -22.959   8.00  33.72      O
HETATM  1113  O2D HEM A   1      8.096   4.833 -23.177   8.00  33.55      O
ATOM    1114  N   VAL B   1      9.143 -20.582   1.231   7.00  48.92      N
ATOM    1115  CA  VAL B   1      8.824 -20.084  -0.109   6.00  52.26      C
ATOM    1116  C   VAL B   1      9.440 -20.964  -1.190   6.00  57.72      C
ATOM    1117  O   VAL B   1      9.768 -22.138  -0.985   8.00  55.05      O

```

```

ATOM  1118  CB  VAL  B  1          9.314 -18.642 -0.302  6.00 58.48          C
ATOM  1119  CG1 VAL  B  1          8.269 -17.606  0.113  6.00 59.43          C
ATOM  1120  CG2 VAL  B  1         10.683 -18.373  0.331  6.00 45.96          C

```

Here the TER card is implicit in the start of a new chain.

Protein Data Bank format relies on the concept of *residues*:

- Each atom in a residue must be uniquely identifiable. Two atoms in the same residue can only have the same name if they have different alternate location identifiers.
- Residue names are a maximum of three characters long<sup>S</sup> and uniquely identify the residue type. Thus, all residues of a given name should be the same type of residue and have the same structure (contain the same atoms with the same connectivity).

## Common Errors in PDB Format Files

If a data file fails to display correctly, it is sometimes difficult to determine where in the hundreds of lines of data the mistake occurred. This section enumerates some of the most common errors found in PDB files.

### Program-Generated PDB Files

#### Spurious Long Bonds

A couple of common errors in program-generated PDB files result in the display of very long bonds between residues:

- Missing TER cards - Either a TER card or a change in the chain ID is needed to mark the end of a chain.
- Improper use of ATOM records instead of HETATM records - HETATM records should be employed for compounds that do not form chains, such as water or heme. The first *six* columns of the ATOM record should be changed to HETATM so that the remaining columns stay aligned correctly.

Apart from any format errors, Chimera also uses long bonds to indicate the underlying connectivity across chain segments that lack coordinates (*e.g.*, regions of missing density due to crystallographic disorder). Regardless of their cause, long bonds in Chimera can be hidden with the command [~longbond](#).

#### Misaligned Atom Names

Incorrectly aligned atom names in PDB records can cause problems. Atom names are composed of an atomic (element) symbol *right*-justified in columns 13-14, and trailing identifying characters *left*-justified in columns 15-16. A single-character element symbol should not appear in column 13 unless the atom name has four characters (for example, see [Hydrogen Atoms](#)). Many programs simply left-justify all atom names starting in column 13. The difference can be seen clearly in a short segment of hemoglobin (entry **3hhb**):

*Correct:*

```

HETATM 1071 FE   HEM  A  1          8.128  7.371 -15.022  24.00 16.74          FE
HETATM 1072  CHA HEM  A  1          8.617  7.879 -18.361   6.00 17.74          C
HETATM 1073  CHB HEM  A  1         10.356 10.005 -14.319   6.00 18.92          C
HETATM 1074  CHC HEM  A  1          8.307  6.456 -11.669   6.00 11.00          C

```

```
HETATM 1075 CHD HEM A 1 6.928 4.145 -15.725 6.00 13.25 C
```

*Incorrect:*

```
HETATM 1071 FE HEM A 1 8.128 7.371 -15.022 24.00 16.74 FE
HETATM 1072 CHA HEM A 1 8.617 7.879 -18.361 6.00 17.74 C
HETATM 1073 CHB HEM A 1 10.356 10.005 -14.319 6.00 18.92 C
HETATM 1074 CHC HEM A 1 8.307 6.456 -11.669 6.00 11.00 C
HETATM 1075 CHD HEM A 1 6.928 4.145 -15.725 6.00 13.25 C
```

## Hand-Edited PDB Files

### Duplicate Atom Names

One possible editing mistake is the failure to uniquely name all atoms within a given residue. In the following example, two atoms in the same residue are named CA:

```
ATOM 185 N VAL A 23 13.455 17.883 10.517 1.00 7.00 N
ATOM 186 CA VAL A 23 12.574 17.403 11.589 1.00 7.00 C
ATOM 187 C VAL A 23 11.283 18.205 11.729 1.00 7.00 C
ATOM 188 O VAL A 23 10.233 17.600 12.052 1.00 7.00 O
ATOM 189 CA VAL A 23 13.339 17.278 12.906 1.00 10.00 C
ATOM 190 CG1 VAL A 23 12.441 17.004 14.108 1.00 13.00 C
ATOM 191 CG2 VAL A 23 14.455 16.248 12.794 1.00 13.00 C
ATOM 192 N GLN A 24 11.255 19.253 10.941 1.00 8.00 N
ATOM 193 CA GLN A 24 10.082 20.114 10.818 1.00 8.00 C
ATOM 194 C GLN A 24 9.158 19.638 9.692 1.00 8.00 C
```

Depending on the display program, the residue may be shown with incorrect connectivity, or it may become evident only upon labeling that the residue is missing a CB atom.

### Residues Out of Sequence

In the following example, the second residue in the file is erroneously numbered residue 5. Many display programs will show this residue as connected to residues 1 and 3. If this residue was meant to be connected to residues 4 and 6 instead, it should appear between those residues in the PDB file.

```
ATOM 1 N HIS A 1 49.668 24.248 10.436 1.00 25.00 N
ATOM 2 CA HIS A 1 50.197 25.578 10.784 1.00 16.00 C
ATOM 3 C HIS A 1 49.169 26.701 10.917 1.00 16.00 C
ATOM 4 O HIS A 1 48.241 26.524 11.749 1.00 16.00 O
ATOM 5 CB HIS A 1 51.312 26.048 9.843 1.00 16.00 C
ATOM 6 CG HIS A 1 50.958 26.068 8.340 1.00 16.00 C
ATOM 7 ND1 HIS A 1 49.636 26.144 7.860 1.00 16.00 N
ATOM 8 CD2 HIS A 1 51.797 26.043 7.286 1.00 16.00 C
ATOM 9 CE1 HIS A 1 49.691 26.152 6.454 1.00 17.00 C
ATOM 10 NE2 HIS A 1 51.046 26.090 6.098 1.00 17.00 N
ATOM 11 N SER A 5 49.788 27.850 10.784 1.00 16.00 N
ATOM 12 CA SER A 5 49.138 29.147 10.620 1.00 15.00 C
ATOM 13 C SER A 5 47.713 29.006 10.110 1.00 15.00 C
ATOM 14 O SER A 5 46.740 29.251 10.864 1.00 15.00 O
ATOM 15 CB SER A 5 49.875 29.930 9.569 1.00 16.00 C
ATOM 16 OG SER A 5 49.145 31.057 9.176 1.00 19.00 O
ATOM 17 N GLN A 3 47.620 28.367 8.973 1.00 15.00 N
ATOM 18 CA GLN A 3 46.287 28.193 8.308 1.00 14.00 C
```

## Common Typos

Sometimes the letter l is accidentally substituted for the number 1. This has different repercussions depending on where in the file the error occurs; a grossly misplaced atom may indicate the presence of such an error in a coordinate field. These errors can be located readily if the text of the data file appears in uppercase, by invoking a text editor to search for all instances of the lowercase letter l.

## Hydrogen Atoms

In brief, conventions for hydrogen atoms in version 3.0 PDB format are as follows:

- Hydrogen atom records follow the records of all other atoms of a particular residue.
- A hydrogen atom name starts with H. The next part of the name is based on the name of the connected nonhydrogen atom. For example, in amino acid residues, H is followed by the remoteness indicator (if any) of the connected atom, followed by the branch indicator (if any) of the connected atom; if more than one hydrogen is connected to the same atom, an additional digit is appended so that each hydrogen atom will have a unique name. Hydrogen atoms in standard nucleotides and amino acids (other than the rarely seen HXT) are named according to the IUPAC recommendations (*Pure Appl Chem* **70**:117 (1998) [[abstract](#)] [[PDF](#)]). Names of hydrogen atoms in HETATM residues are determined in a similar fashion.
- If the name of a hydrogen has four characters, it is left-justified starting in column 13; if it has fewer than four characters, it is left-justified starting in column 14.

In the following excerpt from entry **1vm3**, atom H is attached to atom N. Atom HA is attached to atom CA; the remoteness indicator A is the same for these atoms. Two hydrogen atoms are connected to CB, one is connected to CG, three are connected to CD1, and three are connected to CD2.

ATOM	10	N	LEU	A	2	4.595	6.365	3.756	1.00	0.00		N
ATOM	11	CA	LEU	A	2	4.471	5.443	2.633	1.00	0.00		C
ATOM	12	C	LEU	A	2	5.841	5.176	2.015	1.00	0.00		C
ATOM	13	O	LEU	A	2	6.205	4.029	1.755	1.00	0.00		O
ATOM	14	CB	LEU	A	2	3.526	6.037	1.578	1.00	0.00		C
ATOM	15	CG	LEU	A	2	2.790	4.919	0.823	1.00	0.00		C
ATOM	16	CD1	LEU	A	2	3.803	3.916	0.262	1.00	0.00		C
ATOM	17	CD2	LEU	A	2	1.817	4.196	1.769	1.00	0.00		C
ATOM	18	H	LEU	A	2	4.169	7.246	3.704	1.00	0.00		H
ATOM	19	HA	LEU	A	2	4.063	4.514	2.992	1.00	0.00		H
ATOM	20	HB2	LEU	A	2	2.804	6.675	2.065	1.00	0.00		H
ATOM	21	HB3	LEU	A	2	4.099	6.623	0.873	1.00	0.00		H
ATOM	22	HG	LEU	A	2	2.234	5.353	0.004	1.00	0.00		H
ATOM	23	HD11	LEU	A	2	4.648	4.447	-0.148	1.00	0.00		H
ATOM	24	HD12	LEU	A	2	3.334	3.331	-0.516	1.00	0.00		H
ATOM	25	HD13	LEU	A	2	4.137	3.260	1.052	1.00	0.00		H
ATOM	26	HD21	LEU	A	2	0.941	3.892	1.216	1.00	0.00		H
ATOM	27	HD22	LEU	A	2	1.522	4.860	2.568	1.00	0.00		H
ATOM	28	HD23	LEU	A	2	2.296	3.323	2.188	1.00	0.00		H

## PQR Variant of PDB Format

Several programs use a modified PDB format called PQR, in which atomic partial charge (Q) and radius (R) fields follow the X,Y,Z coordinate fields in ATOM and HETATM records. An excerpt:

ATOM	1	N	ALA	1	46.457	12.189	21.556	0.1414	1.8240			
------	---	---	-----	---	--------	--------	--------	--------	--------	--	--	--

ATOM	2	CA	ALA	1	47.614	11.997	22.448	0.0962	1.9080
ATOM	3	C	ALA	1	47.538	12.947	23.645	0.6163	1.9080
ATOM	4	O	ALA	1	46.441	13.476	23.962	-0.5722	1.6612
ATOM	5	CB	ALA	1	48.911	12.134	21.650	-0.0597	1.9080
ATOM	6	H2	ALA	1	45.672	11.684	21.917	0.1997	0.6000
ATOM	7	H3	ALA	1	46.235	13.163	21.506	0.1997	0.6000
ATOM	8	H	ALA	1	46.683	11.849	20.642	0.1997	0.6000
ATOM	9	HA	ALA	1	47.603	11.052	22.786	0.0889	1.1000
ATOM	10	HB1	ALA	1	49.041	11.319	21.087	0.0300	1.4870
ATOM	11	HB3	ALA	1	48.855	12.941	21.064	0.0300	1.4870
ATOM	12	HB2	ALA	1	49.679	12.231	22.281	0.0300	1.4870
ATOM	13	N	ASP	2	48.702	13.128	24.279	-0.5163	1.8240
ATOM	14	CA	ASP	2	48.826	13.956	25.493	0.0381	1.9080
ATOM	15	C	ASP	2	48.614	15.471	25.323	0.5366	1.9080
ATOM	16	O	ASP	2	49.292	16.362	24.807	-0.5819	1.6612
ATOM	17	CB	ASP	2	50.156	13.635	26.226	-0.0303	1.9080
ATOM	18	CG	ASP	2	49.984	12.419	27.136	0.7994	1.9080
ATOM	19	OD1	ASP	2	50.595	12.308	28.221	-0.8014	1.6612
ATOM	20	OD2	ASP	2	49.198	11.502	26.778	-0.8014	1.6612
ATOM	21	H	ASP	2	49.511	12.637	23.845	0.2936	0.6000
ATOM	22	HA	ASP	2	48.104	13.630	26.146	0.0880	1.3870
ATOM	23	HB3	ASP	2	50.392	14.413	26.773	-0.0122	1.4870
ATOM	24	HB2	ASP	2	50.832	13.431	25.545	-0.0122	1.4870

PQR format is rather loosely defined and varies according to which program is producing or using the file. For example, [APBS](#) requires only that all fields be whitespace-delimited.

If an ATOM or HETATM record being read by Chimera is not in PDB format, Chimera next tries to read it as PQR format. In that case, all fields up to and including the coordinates are still expected to adhere to the [standard format](#), but the next two eight-column fields are each expected to contain a floating-point number: charge is read from columns 55-62 and radius is read from columns 63-70. The values are assigned as the atom [attributes](#) **charge** and **radius**, respectively.

[PDB2PQR](#) is a program for structure cleanup, charge/radius assignment, and PQR file generation. The [PDB2PQR](#) tool in Chimera uses a web service provided by the [National Biomedical Computation Resource \(NBCR\)](#).

---

UCSF Computer Graphics Laboratory / May 2014

## Chimera Startup

At graphical startup, a Chimera splash screen appears, to be replaced in a few seconds by the [Chimera graphics window](#) or the [Rapid Access](#) interface.

On **Windows**,

Chimera can be started by clicking the **chimera** icon. To specify [options](#) or [input files](#) at startup:

1. drag the **chimera** icon with the right mouse button and select **Copy Here** from the menu that appears; this creates another shortcut icon called "copy of chimera"
2. right-click the new **copy of chimera** icon, select **Properties** from the menu that appears
3. in the resulting panel, click the **Shortcut** tab and then append the desired [option\(s\)](#) and [input\(s\)](#) to the command in the **Target** field. The path to the Chimera executable, arguments of options, and input file pathnames should each be enclosed in double quotes if they contain any spaces, for example:

```
"C:\Program Files\Chimera\bin\chimera.exe" --stereo seq "f:\PDB Files\protease.pdb"
```

4. click the the new **copy of chimera** icon

On **UNIX**,

Chimera can be started from the system command line:

```
> chimera [options] [input1 input2 ...]
```

Bracketed arguments are optional. The user's execution path should include *chimera\_install\_dir/bin* (the default *chimera\_install\_dir* is */usr/local/chimera*).

On **Mac OS X**,

Chimera can be started by clicking the **chimera** icon or by dragging and dropping certain file types on the **chimera** icon. The drag-and-drop approach currently works for the [Chimera web data](#) (\*.chimerax) and PDB (\*.pdb) [file types](#).

Chimera can also be started from the system command line:

```
> chimera_install_dir/Contents/MacOS/chimera [options] [input1 input2 ...]
```

Bracketed arguments are optional. The default *chimera\_install\_dir* is */Applications/Chimera.app*.

Chimera for Mac OS X (X Windows) requires an X server to be installed and running. Starting this version of Chimera will automatically start the X server if it is not already running.

## Startup Files

If a Chimera [preferences file](#) is [found](#), it is read at startup. If a **midasrc** file is found (see the [Command Line preferences](#)), it will be executed when the [Command Line](#) is shown.

## System Command-Line Input

Input files may contain structures to be displayed, commands or code to be executed, or other data. If Chimera is started in [nogui mode](#) without command/script file input, a prompt will be supplied in the system shell for entering [Chimera commands](#) on standard input.

Any of the [registered file types](#) can be opened from the system command line at the time of [Chimera startup](#). File type can be specified by a [suffix](#) that is part of the filename or by [prefix:filename](#), where [prefix](#) is not part of the filename. If a prefix and a suffix are both given, the prefix overrides the suffix. Filenames, prefixes, and suffixes are case-sensitive. Unrecognizable prefixes are assumed to be part of the filename. For many of the [registered file types](#), files that are gzipped (as indicated by .gz following the regular filename) are recognized and opened. Similarly, compressed files (\*.Z) can be recognized and opened for many input types if *gzip* is on the user's execution path (can be run by entering *gzip* at the system command line).

Input is generally specified as a pathname to a file or the name of a file in the current working directory. In some cases ([indicated with certain prefixes](#)), the filename can be the database identifier of a file to be retrieved and opened. Internet connectivity is required to fetch files over the Web.

## Input within Chimera

Any of the [registered file types](#) can also be opened from within Chimera. The most general ways are:

- with [File... Open](#) (local files), [Fetch by ID](#) (files to be fetched from databases)
- from the Chimera [Command Line](#), using the command [open](#) (default type [PDB](#))

In addition, many [tools](#) bring up dialogs to open specific file types.

On a Mac, dragging and dropping known [file types](#) on the Chimera icon will start Chimera (if not already started) and open the files.

Models can be closed with the command [close](#), the [Model Panel](#), or [File... Close Session](#).

## Exit from Chimera

A Chimera session can be terminated by choosing [File... Quit](#) from the menu, or by entering the command [stop](#) in the [Command Line](#). Whether the user should be asked to confirm exiting from Chimera is set in the [General preferences](#).

## Input File Types

Ways to open registered file types in Chimera:

- with [File... Open](#) (local files), [Fetch by ID](#) (files to be fetched from databases)
- from the Chimera [Command Line](#), using the command [open](#) (default type [PDB](#))
- from the system command line at the time of [Chimera startup](#)

In the Chimera and system command lines, the file type can be specified by a *suffix* that is part of the filename or by *prefix:filename*, where *prefix* is not part of the filename. Suffixes (filename extensions) also control which files are listed in the [Open File dialog](#) when the **File type** is set to a specific type.

The file types are grouped by data type below:

- [Molecular Structures and Related Data](#) (see also [MD Movie](#) and [ViewDock](#))
- [Electrostatic Potential](#) (associated tool: [Electrostatic Surface Coloring](#))
- [Volume Data](#) (associated tool: [Volume Viewer](#))
- [Sequence](#) (associated tool: [Multalign Viewer](#))
- [Command Scripts](#)
- [Other 3D Objects](#)

When a tool-specific type of file is opened, the associated tool will execute or start.

### Molecular Structures and Related Data

file type	prefix	suffix	contents
<a href="#">CASTp</a> ID fetch	CASTp: castp:		structure and pocket measurements (data to fetch from the from the <a href="#">Computed Atlas of Surface Topography of proteins (CASTp) database</a> specified by 4-character PDB ID with chain ID optionally appended, for example <b>2gsh.A</b> ; not all PDB entries are in the database; measurements will be displayed in a <a href="#">pocket list</a> )
<a href="#">CASTp</a> local		.poc	structure and pocket measurements (previously saved results from the <a href="#">CASTp server</a> specified by <i>name.poc</i> , which will also read <i>name.pdb</i> , <i>name.pocInfo</i> , <i>name.mouth</i> , and <i>name.mouthInfo</i> from the same location and display the measurements in a <a href="#">pocket list</a> )
CATH	cath:		protein domain structure (PDB format; file to fetch specified by <a href="#">CATH domain ID</a> )
CIF/mmCIF	cif: mmcif:	.cif	molecular structure

CIFID	cifID:		molecular structure (mmCIF format; file to fetch specified by 4-character PDB ID)
<a href="#">Gaussian formatted checkpoint</a>	fchk: gaussian:	.fchk	molecular structure
GRASP surface	graspsurf:	.srf	molecular surface generated by <a href="#">DelPhi</a> (the academic version; see also <a href="#">DelPhiController</a> ) or <a href="#">GRASP</a>
<a href="#">Gromos87</a>	gro:	.gro	molecular structure (not read as a trajectory)
<a href="#">Maestro</a> file	maestro: mae:	.mae	molecular structure
<a href="#">MD Movie metafile</a>	md: movie:		specification of <a href="#">trajectory format</a> and filenames ( <a href="#">associated tool</a> : <a href="#">MD Movie</a> )
<a href="#">MDL MOL/SDF</a>	mol: sdf:	.mol .sdf	molecular structure
<a href="#">ModBase</a>	modbase:		modeled protein structure (comparative models in PDB format to fetch from <a href="#">ModBase</a> specified by SwissProt, TrEMBL, GenPept or PIR accession code; associated information will be shown in a <a href="#">list</a> )
<a href="#">Mol2</a>	mol2:	.mol2	molecular structure
<a href="#">MS surface</a>	dms: ms:	.dms .ms	dot molecular surface
NDB	ndb:		nucleic acid structure (PDB format; file to fetch specified by <a href="#">NDB ID</a> )
<a href="#">PDB</a> (see also <a href="#">PQR</a> )	pdb:	.pdb .pdb1 .ent	molecular structure
PDBID	pdbID:		molecular structure (PDB format; file to fetch specified by 4-character PDB ID)
PDB biounit	biounitID:		known or predicted <a href="#">biological assembly</a> (file(s) to fetch specified by 4-character PDB ID, optionally with ".N" appended to specify assembly N; otherwise, if the entry has multiple assemblies, multiple files will be retrieved)
PQSID	pqsID:		predicted biological unit (file to fetch from the from the <a href="#">Protein Quaternary Structure server</a> specified by 4-character PDB ID; predictions are not available for all PDB entries; some entries have multiple predictions, and for those, multiple files will be retrieved)

<a href="#">PubChem</a> CID (input)	PubChem: pubchem:		small molecule structure (3D structure SDF fetched from the <b>Pub3D</b> database of modeled structures using a <a href="#">web service</a> provided by the <a href="#">CICC</a> at Indiana University)
<a href="#">Rich Molecular Format</a> (based on the hierarchical data format <a href="#">HDF5</a> )	rmf:	.rmf .rmf2info	hierarchical molecular structure, feature information, geometric markup (associated data and controls will be shown in an <a href="#">RMF Viewer</a> dialog)
SCOP	scop:		protein domain structure (PDB format; file to fetch specified by <a href="#">SCOP domain ID</a> )
<a href="#">SMILES</a> (input)	SMILES: smiles:		small molecule structure (SMILES string converted to 3D structure SDF using the <a href="#">smi23d web service</a> provided by the <a href="#">CICC</a> at Indiana University)
VIPERID	viperID:		icosahedral virus capsid structure (PDB format; file to fetch specified by PDB ID; capsid automatically constructed with <a href="#">Multiscale Models</a> )
<a href="#">XYZ coordinate</a>	xyz:	.xyz	molecular structure

**Electrostatic Potential**(can also be handled as [volume data](#))

**Associated tool:** [Electrostatic Surface Coloring](#)

file type	prefix	suffix	contents
APBS potential	apbs:	.dx	electrostatic potential calculated with <a href="#">Adaptive Poisson-Boltzmann Solver</a> (APBS); the <a href="#">APBS</a> tool in Chimera runs this program via web service provided by the <a href="#">NBCR</a>
DelPhi or GRASP potential	delphi:	.phi	electrostatic potential calculated with <a href="#">DelPhi</a> (the academic version; see also <a href="#">DelPhiController</a> ) or <a href="#">GRASP</a>
UHBD grid, binary	uhbd:	.grd	electrostatic potential calculated with University of Houston Brownian Dynamics (UHBD)

\*Electrostatic potential maps can also be generated by using [Coulombic Surface Coloring](#) with the [compute grid](#) option

\*\*Electrostatic potential isosurfaces are not displayed automatically, but can be shown with [Volume Viewer](#) or the [volume](#) command

**Volume Data**(see also [electrostatic potential](#))

**Associated tool:** [Volume Viewer](#)

file type	prefix	suffix	contents
<a href="#">Amira mesh</a> , scalar (not vector)	amira:	.am	scalar field data

BRIX or DSN6 density map	dsn6:	.brix .omap	crystallographic density map used by O
<a href="#">CCP4</a> density map	ccp4:	.ccp4 .map	electron density map
Chimera map (based on the hierarchical data format <a href="#">HDF5</a> )	cmap:	.cmap .cmp	electron density map
CNS or XPLOR density map	xplor:	.cns .xplor	unformatted ASCII density map
DOCK scoring grid	dock:	.bmp .cnt .nrg	<a href="#">DOCK</a> (versions 4, 5, 6) bump, contact, and energy scoring grids (suffixes not interchangeable; <i>gridname.bmp</i> required for reading <i>gridname.cnt</i> and/or <i>gridname.nrg</i> )
EDSID	edsID:		electron density map (file to fetch from the <a href="#">Electron Density Server</a> specified by 4-character PDB ID; not all PDB entries have maps available)
EDSDIFFID	edsdiffID:		electron density difference map (fo-fc) (file to fetch from the <a href="#">Electron Density Server</a> specified by 4-character PDB ID; not all PDB entries have maps available)
EMAN HDF map (based on the hierarchical data format <a href="#">HDF5</a> )	emanhdf:	.hdf .h5	electron density map
EMDBID	emdbID:		electron density map (entry to fetch from the <a href="#">Electron Microscopy Data Bank</a> ; prefix emdbfitID will additionally fetch any corresponding PDB entries, but they may or may not be in the fit positions relative to the map)
Gaussian cube grid	cube:	.cube .cub	orbitals, electron densities, other
<a href="#">gOpenMol</a> grid (see <a href="#">conversion instructions</a> )	gopenmol:	.plt	orbitals, electron densities, other
<a href="#">Image stack</a> (formats supported by <a href="#">PIL</a> )	images:	.tif .tiff .png (etc.)	series of grayscale images in multiple files or a single multipage file
<a href="#">IMOD map</a> (MRC map with signed 8-bit mode interpreted as unsigned)	imodmap:	.rec	electron density map
<a href="#">MacMolPlt</a> grid ( <a href="#">3D surfaces</a> )	macmolplt:	.mmp	3D surface data
<a href="#">MRC</a> density map	mrc:	.mrc	electron density map
NetCDF generic array	netcdf:	.nc	3D data

Priism microscope image	priism:	.xyzw	3D light or EM data
Priism time series	priism_t:	.xyzt	time series of 3D light or EM data (also starts <a href="#">Volume Series</a> )
PROFEC free energy grid	profec:	.profec	interaction free energy grid from <a href="#">PROFEC</a> (in <a href="#">Amber</a> versions 6 and 7)
Purdue image format	pif:	.pif	electron density map
<a href="#">Situs</a> map file	situs:	.situs .sit	electron density map
SPIDER volume data	spider:	.spi .vol	electron density map
<a href="#">TOM toolbox</a> EM density map	tom_em:	.em	electron density map
<a href="#">Visualization Toolkit (VTK)</a> structured points, ASCII	vtk:	.vtk	values on a grid

### Sequence

Associated tool: [Multalign Viewer](#)

file type	prefix	suffix	contents
<a href="#">Aligned FASTA</a>	afasta:	.afasta .afa .fasta .fa	sequence alignment
<a href="#">Aligned NBRF/PIR</a>	pir:	.ali .pir	sequence alignment
Clustal ALN	aln:	.aln .clustal .clustalw .clustalx	sequence alignment
GCG RSF	rsf:	.rsf	sequence alignment
<a href="#">HSSP</a> (read-only)	hssp:	.hssp	sequence alignment (other information not read); see <a href="#">HSSP database</a> and <a href="#">search interface</a>
MSF	msf:	.msf	sequence alignment
Selex	selex: pfam:	.selex .slx .pfam	sequence alignment
<a href="#">Stockholm</a>	sth: hmmer:	.sth .sto	sequence alignment
<a href="#">UniProt</a> accession or ID (input)	uniprot:		protein sequence with feature annotations

### Command Scripts

file type	prefix	suffix	contents

<a href="#">Chimera commands</a>	com: cmd:	.com .cmd	<a href="#">Chimera commands</a> (locations of files opened by a command file can be specified relative to the command file's location, or with absolute pathnames)
<a href="#">Chimera demo</a>	demo:	.src	instructions to Chimera and explanatory text for the demo viewer ( <a href="#">associated tool: Demos</a> )
<a href="#">Chimera web data</a>	chimerax:	.chimerax	instructions on data files to open, commands and code to execute (can have a <a href="#">Chimera demo</a> embedded)
Python	python: py: chimera:	.py .pyc .pyo .pyw	Python code (locations of files opened by a Python script can be specified relative to the script's location, or with absolute pathnames; for scripts with arguments, see also <a href="#">runscript</a> , startup option <a href="#">--script</a> )

### Other 3D Objects

file type	prefix	suffix	contents
<a href="#">AutoPack results</a>	apr:	.apr	collections of 3D objects from <a href="#">AutoPack</a>
<a href="#">Bild</a>	bild:	.bild .bld	graphical objects
<a href="#">Chimera markers</a>	markers:	.cmm	markers placed in 3D ( <a href="#">associated tool: Volume Tracer</a> )
<a href="#">COLLADA</a>	collada:	.dae	graphical objects (geometry nodes define Chimera <a href="#">surface pieces</a> )
<a href="#">IMOD segmentation</a>	imod:	.imod .mod	EM segmentation meshes and contours
Neuron trace ( <a href="#">SWC</a> , see also the <a href="#">NeuroMorpho FAQ</a> )	swc:	.swc	neuron reconstruction, such as from <a href="#">NeuroMorpho.Org</a>
<a href="#">Segger</a> segmentation (based on the hierarchical data format <a href="#">HDF5</a> )	segger:	.seg	segmentation surfaces and grouping hierarchy, pointer to corresponding <a href="#">volume data</a> file ( <a href="#">associated tool: Segment Map</a> )
<a href="#">Sphgen spheres</a>	sph:	.sph	spheres from the <a href="#">DOCK</a> accessory program <a href="#">sphgen</a>
STL surface	stl:	.stl	triangle-based format (binary) native to stereolithography CAD software from <a href="#">3D Systems</a> <sup>®</sup>

VIPERdb	viper:	.vdb	icosahedral virus capsid structure (PDB format, in the <a href="#">Virus Particle Explorer database</a> coordinate system; capsid automatically constructed with <a href="#">Multiscale Models</a> )
VRML	vrml:	.vrml .wrl	graphical objects (described in VRML geometry nodes)

## Fetch by ID

Files can be retrieved from various databases and opened in Chimera with **File... Fetch by ID**. Internet connectivity is required to fetch files over the Web. The command [open](#) can also fetch files by ID. Fetched files can be cached in a local [download directory](#) and reused as needed depending on the [Fetch preferences](#), [PDB preferences](#), and whether the option to **Ignore any cached data** is checked. See also: [Rapid Access](#), [Chimera input file types](#)

A database and corresponding identifier (ID code) must be indicated. Multiple identifiers for the same database can be entered, separated by spaces and/or commas.

- **NDB** - a [Nucleic Acid Database](#) identifier will be translated into a PDB ID and used to fetch a PDB-format file from the [Protein Data Bank](#)
- **PDB** - a 4-character PDB ID (Protein Data Bank Identifier) will be used to fetch a PDB-format file from the [Protein Data Bank](#):
  1. Chimera will first attempt to find the file within a local installation of the Protein Data Bank. The default places to look for a local installation are `/usr/mol/pdb/` and then `/mol/pdb/` — these can be changed by editing the Python file `share/chimera/pdbDir` within a Chimera installation.
  2. Next, Chimera will look for the file in any personal PDB directories designated in the [PDB preferences](#).
  3. Next, the PDB subdirectory of the [fetch download directory](#) will be checked (unless the use of previously fetched files has been disallowed in the [Fetch preferences](#)).
  4. Finally, if not found locally, the file will be fetched from the [Protein Data Bank web site](#) (unless fetching has been disallowed in the [PDB preferences](#)).
- **PDB (mmCIF)** - a 4-character PDB ID will be used to fetch an mmCIF-format file from the [Protein Data Bank](#)
- **PDB (biounit)** - a 4-character PDB ID will be used to fetch one or more PDB-format [biological assembly](#) files from the [Protein Data Bank](#); there may be multiple assemblies for a given entry, and a specific assembly can be designated by appending “.N” to the PDB ID. For example, `1dok` specifies all biological assemblies of entry `1dok`, `1dok.2` specifies biological assembly 2, and `1dok.1.2` specifies biological assemblies 1 and 2. Each assembly will be opened as a separate main model number, possibly containing [submodels](#).
- **CATH** - a 7-character [CATH domain identifier](#) will be used to fetch a PDB-format domain file from [CATH](#)
- **SCOP** - a 7-character [SCOP domain identifier](#) will be used to fetch a PDB-format domain file from the [ASTRAL](#) database
- **PubChem** - a [PubChem](#) compound identifier (CID) will be used to fetch a modeled 3D structure from the **Pub3D** database via a [web service](#) provided by the [CICC](#) at Indiana University. **Pub3D** is described in [Willighagen et al.](#), *BMC Bioinformatics* **8**:487 (2007). About 99% of the compounds in [PubChem](#) are available; the structure generation pipeline generally handles organic compounds, but not inorganic, metallo-, or highly unstable species such as radicals.
- **CASTp** - a 4-character PDB ID (with chain ID optionally appended, for example `2gsh.A`) will be used to fetch a structure and its precomputed pocket measurements from the [Computed Atlas of Surface Topography of proteins](#) (does not include results for all PDB entries; measurements will be

displayed in a [pocket list](#))

- **EDS (2fo-fc)** - a 4-character PDB ID will be used to fetch an electron density map from the [Electron Density Server](#) (not all PDB entries have maps available)
- **EDS (fo-fc)** - a 4-character PDB ID will be used to fetch an electron density difference map from the [Electron Density Server](#) (not all PDB entries have maps available)
- **EMDB** - a numerical identifier will be used to fetch an electron density map from the [Electron Microscopy Data Bank](#)
- **EMDB & fit PDBs** - a numerical identifier will be used to fetch an electron density map from the [Electron Microscopy Data Bank](#), along with any corresponding PDB entries, which may or may not be in the fit positions relative to the map
- **PQS** - a 4-character PDB ID will be used to fetch the predicted biological unit from the [Protein Quaternary Structure server](#) (predictions are not available for all PDB entries; some entries have multiple predictions, and for those, multiple files will be retrieved)
- **ModBase** - a SwissProt, TrEMBL, GenPept or PIR accession code will be used to fetch comparative models in PDB format from [ModBase](#) (associated information will be shown in a [model list](#))
- **VIPERdb** - a 4-character PDB ID will be used to fetch a PDB-format file from the [Virus Particle Explorer database](#) of icosahedral virus capsid structures (the capsid will be constructed automatically with [Multiscale Models](#))
- **UniProt** - a [UniProt](#) accession code or identifier will be used to fetch a protein sequence and its annotations and show them in [Multalign Viewer](#) (much as described for [PDB/UniProt Info](#), except independent of structure). This type of fetch is not saved locally even if a [download directory](#) has been specified. UniProt's [ID mapping service](#) can be used to obtain UniProt identifiers from other sequence database identifiers.

Clicking **Fetch** retrieves the indicated data and dismisses the dialog (although there is a checkbox option to **Keep dialog up after Fetch**). Clicking **Web Page** opens the home page of the chosen database if no ID code has been entered, or the specific page for the entry if a valid ID code has been entered. Only the page for the first entry is shown if multiple codes have been entered.

Clicking **Set download directory** opens the [Fetch preferences](#) for specifying a local directory in which to save fetched files.

**Close** dismisses the dialog without retrieving the data. **Help** brings up this manual page in a browser window.

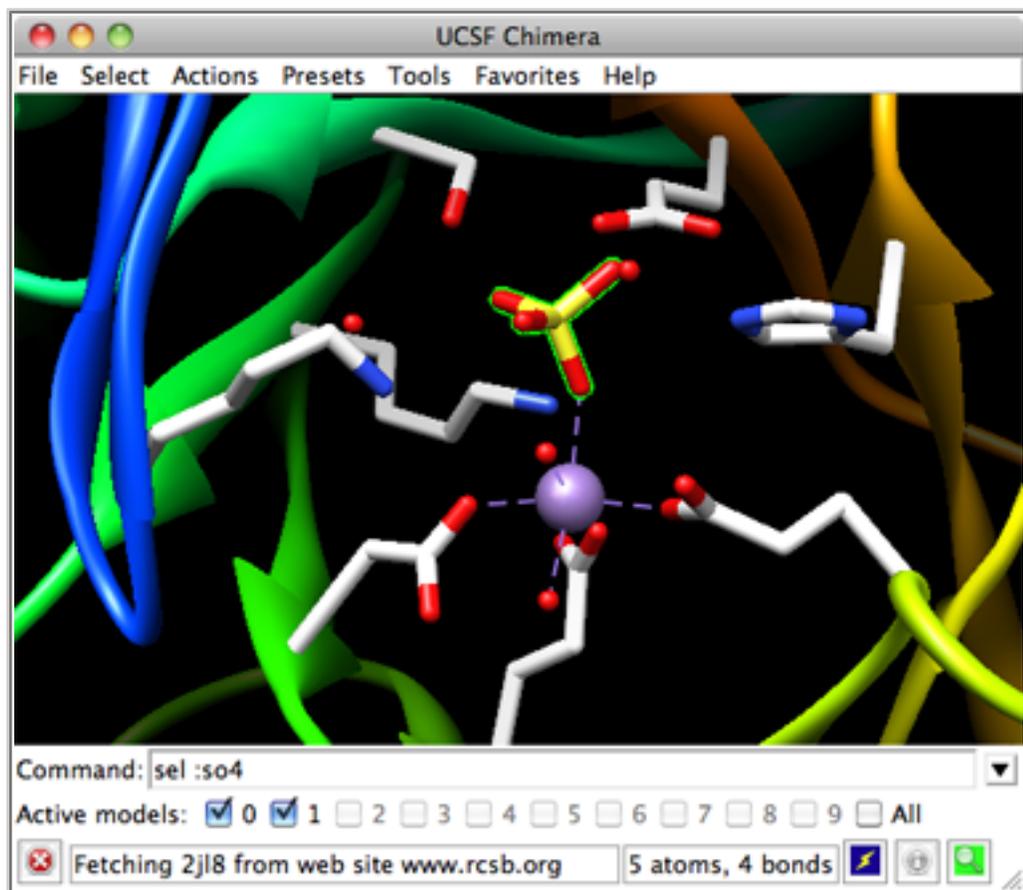
## The Chimera Window

Molecular structures and other 3D data are displayed in the *graphics window*. Initially, the [Rapid Access](#) interface may be shown instead of the graphics window, but that interface can be dismissed/reshown at any time using the lightning-bolt icon in the [status line](#) across the bottom.

The background color can be changed:

- with the command [background](#)
- in the [Background preferences](#)
- in the [Color Actions](#) dialog
- by using a [preset](#)

The first two also allow using a gradient of multiple colors or an image read from a file as the background.



Other window-related commands are [windowsize](#) and [windoworigin](#).

There is normally a *menu bar* across the top:

- [File](#)
- [Select](#)
- [Actions](#)
- [Presets](#)
- [Tools](#)
- [Favorites](#)
- [Help](#)

A pop-up version of the menu can be obtained with the **F8** key (sometimes **fn-F8**), the [Menu key](#) on a Windows keyboard, or a mouse button assigned to this function in the [Mouse preferences](#). See also: [context menus](#)

The Chimera window may also include a [toolbar](#) for icons, a [command line](#), and a [status line](#). The initial appearance of the window depends on any [startup files](#) and predefined [preferences](#).

The *toolbar* is only shown when it contains one or more icons, as specified in the [Tools preferences](#).

In *fullscreen mode* (available in **Windows**, **Linux**, and **Mac-X11** versions of Chimera), the graphics window without borders occupies an entire screen. Fullscreen mode can be toggled with the command [set/~set fullscreen](#), the **F11** key (if not used by Exposé on the Mac), and in the [General preferences](#), or specified at startup with the [--fullscreen](#) option. The menu bar, [Rapid Access](#) interface, and any other components ([tool bar](#), [command line](#), and [status line](#)) can then be opened as a separate window with the **F2** key, or just a [pop-up menu](#) can be used.



## Command Line



[Chimera commands](#) are entered at the **Command Line**. There are [several ways to start](#) the **Command Line**, a tool in the **General Controls** category. Command targets are indicated with [atom specification](#) strings, which can include names, properties, zones, and combinations of these.

**Ctrl-u** erases the command line contents; the list of past commands can be traversed with **up arrow** and **down arrow** or **Ctrl-p** and **Ctrl-n**.

The black inverted triangle to the right of the command entry field marks a pulldown menu. The menu includes:

- the most recently entered commands
- **Command History...** for showing the full [Command History](#); the number of commands to remember (default **60**) can be set in the [Command Line preferences](#)
- **Hide Command Line** (alternatively, the **Command Line** can be hidden using its [instance](#) in the bottom section of the **Tools** menu)
- **Remove duplicate consecutive commands** - where the same command was used multiple times in a row, remove all but one occurrence from the history

Below the command entry area, the number of each open model is shown in bold and a checkbox shows whether the model is [activated for motion](#). Clicking the box toggles between the activated (default) and deactivated states. Checking **All** activates all open models. Model activation status can also be controlled in the [Model Panel](#) and with the command [select](#).

## Status Line



The display of a line for status messages can be controlled in the [Messages preferences](#) and the [Keyboard Shortcuts dialog](#). Icons may be present and colored or grayed out depending on the situation:

-  the red stop icon appears on the left when a foreground task is running; clicking it aborts the task
-  a [menu selection mode](#) icon (optional): A for append, I for intersect, R for replace, or S for subtract; clicking the icon cycles through the modes
-  the lightning-on-dark-blue icon is present when [Rapid Access](#) is hidden (click to show it)
-  the lightning-on-light-blue icon is present when [Rapid Access](#) is shown (click to hide it)
-  the information icon is blue when one or more background tasks are running, a lighter blue when no tasks are running; clicking it brings up the [Task Panel](#)
-  the magnifying glass icon is colored when something is [selected](#); mousing over it reports in a text balloon what is selected, and clicking it brings up the [Selection Inspector](#)

# Chimera Menus

The major menu headings are:

- [File](#)
- [Select](#)
- [Actions](#)
- [Presets](#)
- [Volume](#) (not always present)
- [Tools](#)
- [Favorites](#)
- [Aliases](#) (not always present)
- [Help](#)

Except in the native **Mac** (non-X-Windows) version of Chimera, the menus are *tear-off*, as indicated by a dashed line above the entries when the menu contents are shown. Choosing the dashed line instead of an entry makes the menu an independent window that remains up until explicitly closed. Unavailable options are grayed out. See also: [context menus](#)

## ← File

- **Open...** bring up the [Open File dialog](#)
- **Fetch by ID...** [specify by database identifier](#) a file to be retrieved and opened
- **Restore Session...** bring up a [dialog](#) for opening a previously saved [session](#) (Python file); clicking the filename of a session shows its associated [thumbnail image](#) and [session notes](#), if any
- **Save Session** - save the current [session](#) in the working directory; available when the current session has been named with **Save Session As...** or was started by [restoring](#) an existing session; does not update the associated [thumbnail image](#), if any
- **Save Session As...** bring up a [dialog](#) for saving the current [session](#) to a specified name and location; allows including [session notes](#) and/or a thumbnail image of the current Chimera display
- **Save Image...** bring up the [Save Image dialog](#)
- **Save PDB...** bring up a dialog for [saving PDB files](#)
- **Save Mol2...** bring up a dialog for [saving Mol2 files](#)
- **Export Scene...** bring up a dialog for [exporting the scene](#) from Chimera
- **Close Session** - clear session contents, show the [Rapid Access](#) interface
- **Quit** - exit from Chimera

## ← Select

[Selection](#) designates items for subsequent [Actions](#). When nothing is selected, [Actions](#) apply to "all." (Note:

Any of the entries in **Chemistry**, **Residue...** [amino acid category](#), and **Structure** can also be used to [specify atoms](#) in the [Command Line](#).)

- **Chain**

*[only chain IDs found in currently open structures will be listed]*

- **Chemistry**

- **element**  
*[many entries]*
- **functional group**  
*[many entries]*
- [IDATM type](#)  
*[many entries]*

- **Residue**

- [amino acid category](#)  
*[many entries]*  
(the existing amino acid categories can be changed and new categories defined and listed here, using [ResProp](#))

*[only residue names found in currently open structures will be listed, sorted into nonstandard residues, standard nucleic acids, and standard amino acids]*

- **Structure**

- **backbone** (only applies to amino acids and/or nucleic acids)
  - **full** - the amide backbone in peptides, the sugar-phosphate backbone in nucleic acids
  - **minimal** - a continuous series of bonded atoms connecting the [chain trace](#) atoms (-[N-CA-C]<sub>n</sub> - in peptides and -[O5'-C5'-C4'-C3'-O3'-P]<sub>n</sub> - in nucleic acids)
- **ions** - ions, using the same [definition](#) as surface calculations
- **ligand** - ligand, using the same [definition](#) as surface calculations
- **main** - main, using the same [definition](#) as surface calculations
- **markers** - markers and links (such as from [Volume Tracer](#))
- **nucleic acid** - nucleosides, nucleotides, RNA, and DNA
- **protein** - amino acids, peptides, and proteins
- **secondary structure** (only applies to peptides/proteins; helix and strand assignments are taken from the input structure file or generated with [ksdssp](#))
  - **coil** - amino acid residues not in helices or strands
  - **helix** - amino acid residues in helices
  - **strand** - amino acid residues in strands
- **side chain/base** (only applies to amino acids and/or nucleic acids)
  - **with CA/C1'**
  - **without CA/C1'**
- **solvent** - solvent, using the same [definition](#) as surface calculations

- **Sequence...** [find and select](#) a specified amino acid and/or nucleotide sequence (atoms and bonds in any/all matching segments will be selected)

- **Atom Specifier...** use [atom specification syntax](#) for selection

- **By Attribute Value...** open the [Select by Attribute](#) tool to select by attribute values

- **Zone...** select atoms and bonds within (or beyond) a specified distance from the currently selected atoms, on a per-atom or whole-residue basis. If both the within and beyond options are checked, only the atoms that fit both criteria will be selected, for example, those within 12 Å but farther away than 8 Å.

- **Clear Selection** - clear (deselect) the current selection
- **Invert (all models)** - select all currently unselected atoms and deselect all currently selected atoms
- **Invert (selected models)** - in models containing selections only, select all currently unselected atoms and deselect all currently selected atoms
- **Select All** - select all open models and their constituent parts
- **Selection Mode** (*current\_mode*) - control whether a new selection *via* the menu will be added to, subtracted from, intersected with, or used to replace the existing selection
  - **append**
  - **intersect**
  - **replace** (default)
  - **subtract**
  - **Show (Hide) mode icon** - show, or hide if already shown, an icon in the [status line](#) for the current menu selection mode: A for append, I for intersect, R for replace, or S for subtract; clicking the icon cycles through the modes
- **Broaden** - expand the current selection up a level in the [selection cascade](#)
- **Narrow** - narrow a previously broadened selection down a level in the [selection cascade](#)
- **Undo** - undo the most recent selection operation
  
- **Name Selection...** name and save the current selection (see also the command [namesel](#) and the [Rapid Access](#) interface). [Named selections](#) can be restored using the **Select** menu (see **Named Selections** below) or the [Rapid Access](#) interface, and their names can be used for command-line [atom specification](#).
- **Named Selections** - retrieve a previously saved selection

## ← Actions

Which items are affected by **Actions** (the *targets*) depends on the current [selection](#). For actions under [Atoms/Bonds](#), [Ribbon](#), [Surface](#), [Color](#), [Label](#), [Focus](#), and [Set Pivot](#):

- When nothing is selected, the action affects all applicable items: molecule models and/or [surface models](#), depending on the action.
- Surface actions apply not only to selected [surface pieces](#), but also to the [molecular surface](#) patches of selected atoms.

[Inspection](#) and [file-writing actions](#), however, apply only to explicit selections.

- **Atoms/Bonds** (see also the commands [display](#) and [represent](#))
  - **show**
  - **show only**
  - **hide**
  - **backbone only** - only applies to residues in bonded chains of amino acids or nucleic acids
    - **chain trace** - show only one atom per residue, CA in each amino acid and C4' in each nucleic acid residue (turns on [auto-chaining](#))
    - **full** - show only the amide backbone in peptides, the sugar-phosphate backbone in nucleic acids

- **minimal** - show only a continuous series of bonded atoms connecting the [chain trace](#) atoms ( $-[N-CA-C]_n$  - in peptides and  $-[O5'-C5'-C4'-C3'-O3'-P]_n$  - in nucleic acids)
      - **side chain/base** - only applies to residues in bonded chains of amino acids or nucleic acids. Each amino acid **side chain** includes any side chain atoms from CB outward, plus the atom CA (for connectivity purposes); similarly, each nucleic acid **base** includes the base plus the sugar atom to which it is bonded, usually named C1'.
        - **show** - display side chains/bases and turn [auto-chaining](#) on
        - **show only** - undisplay other atoms in the chain and turn [auto-chaining](#) off
      - **stick** (stick [representation](#))
      - **ball & stick** (ball-and-stick [representation](#))
      - **sphere** (sphere or CPK [representation](#))
      - **wire** (wire [representation](#))
      - **wire width** (see also the command [linewidth](#)) - always affects entire molecule models even though the [targets](#) may be narrower
        - 1
        - 1.5
        - 2
        - 2.5
        - 3
        - 4
        - 5
      - **rings** (see also the commands [fillring](#) and [aromatic](#)) - aromatic rings can only be shown as filled when their aromaticity display is turned off, and the aromaticity settings always affect entire models even though the [targets](#) may be narrower
        - **fill thick**
        - **fill thin**
        - **fill off**
        - **aromatic disk**
        - **aromatic circle**
        - **aromatic off**
      - **nucleotide objects** (see also the command [nucleotides](#))
        - **off** - show atoms instead of sugar/base abstractions, remove any nucleotide ring fill
        - **settings...** open the [Nucleotides](#) tool for controlling special nucleotide representations
      - **delete** - remove atoms/bonds (**caution: irreversible**) like the command [delete](#)
  - **Ribbon** - [secondary structure ribbon](#) (see also the commands [ribbon](#) and [ribrepr](#)). Ribbons are only drawn for proteins and nucleic acids. By default, showing ribbon hides the corresponding [mainchain atoms](#) (but see [ribbackbone](#)). Protein helix and strand assignments are taken from the input structure file or generated with [ksdssp](#).
    - **show**
    - **hide**
    - **flat**
    - **edged**
    - **rounded**
    - **supersmooth**
 (note that additional styles can be defined and listed here, using the [Ribbon Style Editor](#))
  - **Surface** - [molecular surfaces](#) and [surface models](#). Even though the [targets](#) may be narrower,

representations (**solid**, **mesh**, and **dot**) always apply to entire [surface pieces](#).

- **show**
- **hide**
- **solid**
- **mesh**
- **dot**
- **transparency** (surfaces colored with [tcolor](#) will not be affected)
  - **0%**
  - **10%**
  - **20%**
  - **30%**
  - **40%**
  - **50%**
  - **60%**
  - **70%**
  - **80%**
  - **90%**
  - **100%**
  - **other...** set surface transparency percentage to a typed-in value
  - **base color** - use transparencies included in color definitions (rather than overriding them via this menu)
- **Color** (see also [color](#), [scolor](#))
  - [[top 20 colors](#)]
  - **by heteroatom** - use a built-in [color-by-element scheme](#), except leaving carbon atoms unchanged
  - **by element** - use a built-in [color-by-element scheme](#)
  - **from editor** - open the [Color Editor](#) and use its current color; apply any color adjustments made in the [Color Editor](#) until it is closed or reassigned to a different coloring operation by clicking a [color well](#). If during that time the selection is changed, any color adjustments will be applied to the new selection.
  - **none** - remove color assignments; assignments at other levels in the [coloring hierarchy](#) may become visible
  - **all options...** open the [Color Actions](#) dialog for access to [all 60 colors](#) and finer control over which types of items are colored (for example, background only or ribbons only)
- **Label** (see also the commands [label](#) and [rlabel](#) for details, including label positioning, and [labelopt](#) for custom labeling)
  - **off** - undisplay atom labels
  - **name** - label atoms by name (with any alternate location ID appended)
  - **element** - label atoms by element
  - **IDATM type** - label atoms by [atom type](#)
  - **other...** label atoms with an arbitrary string or with the values of [atom attributes](#):
    - **altLoc** - alternate location identifier, if any
    - **bfactor** - B-factor value, if any
    - (*etc.*)
 Newly generated numerical, boolean, or string-valued atom [attributes](#) will also be listed.
  - **residue**
    - **off** - undisplay residue labels
    - **name** - label residues by name
    - **1-letter code** - label standard amino acid residues by one-letter code, other

- residues by name
- **specifier** - label residues by specifier (residue number, insertion code, and chain ID); see [custom residue labeling](#) for number alone
- **name + specifier** - label residues by name and [specifier](#)
- **1-letter code + specifier** - label standard amino acid residues by 1-letter code and [specifier](#), other residues by name and [specifier](#)
- **custom...** [custom-label residues](#) with an arbitrary string and/or other residue information such as [attributes](#)
  - **options...** open the [Labels preferences](#) for setting options such as font size
- **Focus** - focus the view on the [targets](#) that are also displayed and set the [center of rotation method](#) to **center of view** (analogous to the command [focus](#)); if nothing is selected, set the [center of rotation method](#) to **front center**; an exception to the preceding is that the [center of rotation method](#) is not changed if it is **independent**
- **Set Pivot** - set the center of rotation to the center of the bounding sphere of the [targets](#) and set the [center of rotation method](#) to **fixed** (analogous to the command [cofr](#)); if nothing is selected, set the [center of rotation method](#) to **front center**. See also: [Mouse preferences](#), **Set Pivot** in the atom [context menu](#)
- **Inspect** - open the [Selection Inspector](#), which lists what items are selected, shows their attributes, and allows their attributes to be changed
- **Write List...** bring up a dialog for saving specifications as a parsable list. Specifications can be saved for the selected (or unselected) **atoms**, **bonds**, **pseudobonds**, **residues**, **molecules** (molecule models), or **models** (non-molecule models: molecular surface, surface, VRML, and volume). The [naming style](#) controls the format of the specifications. Clicking the **Log** button sends the information to the [Reply Log](#) instead of saving a file.
- **Write PDB...** bring up a dialog for [saving the selection as a PDB file](#)

## ← Presets

A *preset* is a predefined combination of display settings. Choosing an entry in the **Presets** menu applies its settings and is much easier than adjusting the many settings individually. Presets are provided for a handful of usage scenarios; of course, many more combinations of settings are possible. See also: the [preset](#) command, the [New Molecules preferences](#)

- **Custom** presets are user-defined, see [below](#)
- **Interactive** presets are meant for interactive manipulation and analysis. They may change which items (atoms, ribbons, surfaces) are displayed and how they are colored. The [background color](#) is set to black and the [ribbon path method](#) to B-spline.
  - **Interactive 1 (ribbons)** - shows most peptide and nucleic acid chains as ribbons, plus atomic detail (excluding hydrogens on carbon atoms) for residues within 3.6 Å of a [ligand](#) residue or metal ion. Atomic detail is also used for chains that are very short. Nucleic acids may be shown with special sugar and base representations (as produced by [Nucleotides](#)), with the level of abstraction dependent on size.
  - **Interactive 2 (all atoms)** - displays all atoms and bonds as wires, with heteroatoms [colored by element](#). Carbons are shown in the model colors so that models will be distinguishable from one another.

- **Interactive 3 (hydrophobicity surface)** - shows amino acid hydrophobicity in the [Kyte-Doolittle scale](#) with [colors](#) ranging from dodger blue for the most hydrophilic to white at 0.0 to to orange red for the most hydrophobic (different color-codings can be applied with [rangecolor](#) or [Render by Attribute](#)). Surfaces of nonpeptides will be colored to match the underlying atoms instead.
- **Publication** presets are intended for generating images for presentation and publication. They do not change which items are displayed or their colors, but may change the styles of the items. For example, ribbons are not hidden/shown, but any existing ribbon will be adjusted to **rounded ribbon** or **licorice** according to the chosen preset. Similarly, atoms and bonds are not hidden/shown, but any wire will be changed to sticks. The [background color](#) is set to white. Publication presets may decrease interactive performance because they increase [smoothness](#) by using finer divisions to depict curved objects (ribbons, [molecular surfaces](#), etc.). Individual display parameters are discussed in more detail in the [tips on preparing images](#).
  - **Publication 1 (silhouette, rounded ribbon)**
  - **Publication 2 (silhouette, licorice)**
  - **Publication 3 (depth-cued, rounded ribbon)**
  - **Publication 4 (depth-cued, licorice)**
- **Add Custom Presets...** open the [Presets preferences](#) for specifying directories in which to look for custom preset scripts

## ← Volume

The **Volume** menu is a replica of the **Tools... Volume Data** menu. It can be shown/hidden using **Tools... Volume Data... Volume Menu on Menubar** or the similar entry in the [Tools menu](#) of [Volume Viewer](#).

## ← Tools

The **Tools** menu contains the Chimera tools or extensions. In general, there will be several submenus of the form

- *Category*
  - *Tool1*
  - *Tool2*

where choosing an individual tool (extension) launches it. See the [Chimera Tools index](#) for descriptions of the individual tools.

If certain tools or extensions have been started and not quit during a Chimera session, their [instances](#) will appear in the bottom section of the **Tools** menu:

- *Tool\_name* (or *Tool\_name - data*)
  - **Raise** - open the tool interface (if closed) and bring it to the front
  - **Hide** - close the tool interface without exiting
  - **Quit** - exit from this instance of the tool

## ← Favorites

The **Favorites** menu contains the [Preferences](#) in addition to any subset of the entries in the [Tools menu](#) (individually specified using the [Tools preferences](#)). The default contents are

- **Model Panel** - open the [Model Panel](#), which lists the current models and enables many operations upon them
- **Side View** - open the [Side View](#) for easy adjustment of scale and clipping planes
- **Command Line** - show the [Command Line](#)
- **Reply Log** - open the [Reply Log](#) containing informational, warning, and error messages from Chimera
- **Add to Favorites/Toolbar...** open the [Tools preferences](#)
- **Preferences...** show and allow changes in [Preferences](#) settings

## ← Aliases

Start-of-line aliases defined with the command [alias](#) will be listed here. Choosing an entry from the **Aliases** menu executes the alias. The **Aliases** menu will only be present when there is at least one start-of-line alias.

## ← Help

Documentation pages requested from the **Help** menu will be shown in a browser window.

- **Search Documentation...** bring up a dialog for searching the local (bundled) Chimera documentation (which includes the [User's Guide](#), [Programmer's Guide](#), and other items listed in the [Chimera documentation index](#))

One or more terms can be entered in the search field. Combinations of terms can be indicated with **and**, **or**, **not** and parentheses. Separating terms with a space is equivalent to using **and**. Pressing return (Enter) or clicking **Search** initiates the search. Hits will be listed in the **Search Results** column as links to the corresponding documentation.

- **User's Guide** - open the [Chimera User's Guide](#)
- **Commands Index** - open the Chimera [Commands Index](#)
- **Tutorials** - open the [Tutorials](#) section of the **Chimera User's Guide**
- **Context Help** - describe the feature that is clicked next (avoid clicking the top bar of a dialog, since that will fail to bring up a help page)
- **Check for Updates** (requires internet connectivity) - see if any production releases are newer than the version in use, and if so, provide links to download them
- **Contact Us** - show [e-mail addresses](#) for asking questions or making suggestions
- **Report a Bug...** open a [form for bug submission](#)
- **Citation Info** - show [how to cite usage](#) of Chimera

- **Registration...** open a form for registration as a Chimera user
- **About UCSF Chimera** - report what version of Chimera is being used and show copyright information (like the command [version](#))

## 3D Manipulation

A mouse or touchpad (trackpad) can be used to manipulate the view of structures and other 3-dimensional data in the [Chimera graphics window](#). The [Mouse preferences](#) contain assignments for a three-button mouse, but a [one- or two-button mouse](#) can also be used. Users of the **Mac X11** version of Chimera, see [below](#). See also: [Movement Mouse Mode](#), [Constrained Move](#), [other input devices](#)

In the laboratory coordinate system, the **X axis** is horizontal in the plane of the screen, the **Y axis** is vertical in the plane of the screen, and the **Z axis** is perpendicular to the plane of the screen. By *default*, [active](#) models can be:

- **rotated** with the left mouse button in the graphics window. Rotation is about the **X** and/or **Y** axis when the cursor is in the central region of the graphics window (the cursor becomes a small circle) and about the **Z** axis when the cursor is in the periphery of the graphics window (the cursor becomes two curved arrows in yin-yang configuration). The center of rotation can be adjusted with the [Rotation tool](#) or the command [cofr](#).
- **XY-translated** with the middle mouse button (the cursor will look like a cross formed by two double-headed arrows). On **Windows**, depending on the mouse setup, [an adjustment](#) may be required.
- **scaled** (zoomed) with the right mouse button (the cursor will look like a diagonal double-headed arrow enclosing a small square); movements downward and/or to the right increase the scale, whereas movements upward and/or to the left decrease it. Interactive scaling can also be done with the [Side View](#) and possibly scrolling.
- **Z-translated** with **Ctrl**-middle mouse button (the cursor will look like a vertical double-headed arrow); movements downward and/or to the right translate structures closer, and movements upward and/or to the left translate structures farther away. Note that Z-translation is not the same as scaling.

Also by *default*,

- **Ctrl**-left mouse button performs [picking](#) (selection from the graphics window); the cursor will look like a pointing hand. Doubleclicking while picking an atom or bond elicits a [context menu](#).
- **Ctrl**-right mouse button centers a clicked item (atom, bond, [pseudobond](#), residue ribbon, or [surface piece](#)) and makes the centered point a **fixed** [center of rotation](#); clicking empty space restores the **front center** [rotation mode](#).
- Pausing the cursor over an atom or bond (without clicking any buttons) will show the corresponding label information in an **atomspec balloon**, and the PDB description of the chain, if available, in the [status line](#). Whether to show atomspec balloons can be set in the [Labels preferences](#). Similarly, many dialogs include **balloon help**, additional text that is displayed when the cursor is paused near some relevant part of the dialog. Whether to show balloon help can be set in the [Messages preferences](#).

Mouse button assignments and whether to use scrolling can be changed in the [Mouse preferences](#).

Additionally holding down the **Shift** key reduces the sensitivity to mouse manipulations in the main window and the [Side View](#) by a factor of 10.

Users of the **Mac X11** version of Chimera on Mac OS 10.5 or higher may need to turn on emulation of a 3-button mouse in the **Input** section of the **X11** preferences. This is not an issue for the native Mac (non-X-Windows) version of Chimera.

## Touchpad or One- or Two-Button Mouse

Used alone, a one-button mouse or simple touchpad click-and-drag acts as button 1, but buttons 2 and 3 can be emulated with modifier keys. On a Mac, buttons 2 and 3 can be emulated with the **option** and **command** () keys, respectively. For example, touchpad click-and-drag with the **option** key held down performs the button 2 action, XY-translation by default. See also [multitouch actions](#).

A two-button mouse is also quite workable. The two buttons can be assigned to rotation and (XY) translation, as alternative methods are available for interactive scaling ([Side View](#) and possibly scrolling and/or [touchpad pinch](#)). Button 1 is already assigned to rotation by default, but depending on whether the other mouse button is treated as button 2 or 3, it may be necessary to use the [Mouse preferences](#) to assign it to translation.

An [Apple Magic Mouse](#) allows scrolling and can be configured to act as if it had three buttons:

- “secondary click” can be enabled in the Mac System Preferences (acts as button 3 in Chimera)
- the free [MagicPrefs app](#) can be used to activate middle click (acts as button 2 in Chimera)

## Multi-Touch Actions

Whether to use multitouch gestures on Mac (default true) is specified in the [Mouse preferences](#). On a Mac touchpad, these are:

- two-finger drag - XY-rotation
- two-finger twist - Z-rotation
- pinch motion - zooming (scaling)
- three-finger drag - XY-translation

## Activation for Motion

A model must be *active* (activated for motion) to move in response to manipulations. Models are active by default. Toggling model activation status allows users to manually position one model relative to another. Models can be activated/deactivated using:

- the **A**(ctive) checkboxes or activation [function buttons](#) in the [Model Panel](#)
- the checkboxes under the [Command Line](#)
- the command [select](#)

## Context Menus

Doubleclicking while [picking](#) an atom or bond elicits a pop-up menu with entries that depend on the context:

- for an atom when 0 or >3 other atoms are [selected](#):
  - if the atom is a metal ion, **Coordination Geometry** - open the [Metal Geometry](#) tool
  - **Show Distances to Nearby Residues** - label and display residues with any atom within 3.6 Å of the current atom, show the corresponding [distance monitors](#) (ignoring atoms in the same residue as or within two bonds of the current atom), remove monitors and labels previously shown for another atom using this function; toggles to **Hide Distances...** to remove monitors and labels previously shown for the current atom
  - **Modify Atom** - open the [Modify Structure](#) dialog
  - **Set Pivot** - set fixed [center of rotation](#) at the atom
  - **Inspect** - open [Selection Inspector](#)
- for an atom when one other atom is [selected](#):
  - **Show Distance** - create a [distance monitor](#) between the two atoms
  - **Inspect** - open the [Selection Inspector](#)
- for an atom when two other atoms are [selected](#):
  - **Measure Angle** - show a measurement of the angle formed by the three atoms in the [Angles/Torsions](#) dialog
  - **Inspect** - open the [Selection Inspector](#)
- for an atom when three other atoms are [selected](#):
  - **Measure Torsion** - show a measurement of the torsion angle formed by the four atoms in the [Angles/Torsions](#) dialog
  - **Inspect** - open the [Selection Inspector](#)
- for a bond when no other bonds are [selected](#):
  - **Rotate Bond** - activate the bond for rotation, open the [Adjust Torsions](#) dialog
  - **Adjust Bond** - open the [Adjust Bonds](#) dialog to delete the bond or change its length
  - **Select Bonded** - [select](#) the flanking atoms
  - **Inspect** - open the [Selection Inspector](#)
- for a [pseudobond](#), or for a bond when other bonds are [selected](#):
  - **Select Bonded** - [select](#) the flanking atoms
  - **Inspect** - open the [Selection Inspector](#)

## Other Input Devices

- [SpaceNavigator](#)
- [Leap Motion Controller](#)

See also: [RBVI Technology Notes](#)

## SpaceNavigator

Chimera supports the [3Dconnexion](#) SpaceNavigator™ input device. Other 3Dconnexion devices may also

work. The 3Dconnexion driver must be installed, except on Linux, where [spacenvd](#) (an open-source alternative to the 3Dconnexion driver) must be installed. Thanks to Thomas Margraf, University of Hamburg, for the Linux implementation.

Besides a cap that can be tilted, rotated, and pushed/pulled in any direction, the SpaceNavigator has two buttons with the possible Chimera functions:

- **Button 1** or **Fit** centers the models and scales them to fit in the window
- **Button 2** toggles a mode allowing simultaneous rotation and translation (the default *dominant mode* only allows rotation or translation depending on which user motion has the larger amplitude)

The **Zoom Direction** controls how the cap coordinate system relates to the screen coordinate system:

- when **up/down** cap motion zooms the models, the tabletop plane maps to the plane of the screen; the systems are related by a rotation of approximately 90°
- when **closer/farther** cap motion zooms the models, the systems are approximately aligned

Zoom direction and other settings are shown in system dialogs (for example, the Windows Control Panel or Mac System Preferences), although some of the Mac controls appear to have no effect.

[Accelerators](#) (keyboard shortcuts) for changing SpaceNavigator behaviors:

- **na** - toggle between moving all models and moving only the [active](#) models
- **nd** - toggle between [dominant mode](#) and allowing simultaneous rotation and translation
- **nf** - toggle *fly-through mode*, where the device moves the viewpoint rather than the models (as if all axes were reversed); currently hard to control
- **nz** - toggle whether model in/out motion is true zooming (scaling the entire scene) or Z-translation (moving models relative to the front and back clipping planes)

Accelerators are disabled by default. One way to enable their use is with the command [ac](#).

SpaceNavigator problems on Mac:

1. Chimera responds to SpaceNavigator even when it does not have the application focus.
2. SpaceNavigator response becomes sluggish after hours of use or idle time.
3. SpaceNavigator response stops after computer wakes from sleep.

Restarting Chimera resolves problems 2 and 3.

## Leap Motion Controller

The [Leap Motion Controller](#) uses two cameras to track finger and hand movements. The [leap](#) command controls the device's mode of interaction with Chimera.

## The Viewing Tool

The **Viewing Tool** has five tabbed sections, also listed separately as Chimera [tools](#):

- [Camera](#)
- [Effects](#)
- [Rotation](#)
- [Side View](#)
- [Lighting](#)

Clicking the tab for a section brings it to the front. For the front section only, **Reset** replaces the current settings with the original “factory” defaults, **Restore** replaces the current settings with those previously saved in the [preferences file](#), and **Save** saves the current settings to the [preferences file](#). **Close** dismisses the **Viewing Tool**, and **Help** opens this manual page in a browser window.

### Camera

The **Camera** section of the [Viewing Tool](#) controls several aspects of the view, including stereo parameters. There are [several ways to start Camera](#), a tool in the **Viewing Controls** category. Default settings are indicated in **bold**. See also: [stereo](#), [set](#)

- **camera mode** - refers to any of several stereo and mono viewing options ([details...](#)):
  - **mono** (default, not in stereo)
  - stereo left eye
  - stereo right eye
  - cross-eye stereo
  - wall-eye stereo
  - red-cyan stereo - overlapping left-eye and right-eye views in different colors, to be viewed with "glasses" (often inexpensive cardboard/plastic) with colored filters
  - green-magenta stereo - similar to red-cyan, but with different colors and trade-offs
  - sequential stereo - rapid flickering between left-eye and right-eye views, to be viewed with special synchronised glasses. Sequential stereo is [not always available](#).
  - reverse sequential stereo - as above, but with the opposite convention for the two views
  - row stereo, right eye even - row-interleaved with even rows used for the right-eye view
  - row stereo, right eye odd - row-interleaved with odd rows used for the right-eye view
  - dome - angular fisheye of the hemisphere in front of the camera, with [horizontal field of view](#) locked to 90°
  - truncated dome - same as the dome mode, except with the bottom of the hemisphere cut off
  - DTI side-by-side stereo
- **projection (perspective/orthographic)** - perspective makes farther-away objects smaller, and is indicated in the [Side View](#) by red lines diverging from the eye position. An orthographic projection has no scaling-with-distance effect, and is indicated in the [Side View](#) by horizontal red lines.
- **scale factor** (**1** when a structure is first opened) - a factor reflecting the cumulative effects of scaling with the mouse, the [Side View](#), and/or the command [scale](#)
- **near plane** - Z-coordinate of the front [global clipping plane](#)

- **far plane** - Z-coordinate of the back [global clipping plane](#)
- **horizontal field of view** (allowed range 1.0-179.0°) - controls how much of the scene is visible within the graphics window; a smaller value zooms in on a smaller portion of the scene. The vertical field of view will change along with the horizontal field of view according to the aspect ratio of the graphics window.
- **Stereo parameters**
  - **units** (millimeters/centimeters/inches) - units for the following stereo parameters
  - **eye separation** (default **2.0** inches) - effective distance between the left and right eyes
  - **distance to screen** - effective distance from the viewer to the screen (or *focal plane*, even though items out of this plane do not appear “out of focus”). When viewed in stereo, items in front of the focal plane will appear to project from the screen, and items behind the plane will appear to recede behind the screen.
  - **screen width** - physical width of the entire screen (not just the graphics window). The value is initially what is reported by the windowing system, but it can be edited. Changing the value does not resize the graphics window.

The **eye separation** and **distance to screen** govern the parallax between the left- and right-eye views. The **horizontal field of view**, **distance to screen**, and **screen width** are interdependent. The *graphics window width* is determined from the **screen width** and the known pixel dimensions of the screen and graphics window. For a given graphics window width, the **horizontal field of view** and **distance to screen** are inversely related; changing one changes only the other, in the opposite direction. Changing the **screen width** value or resizing the graphics window (widthwise) changes only the **distance to screen**, in the same direction.

The **horizontal field of view**, **eye separation**, and position of the focal plane relative to the items in view can be changed interactively using the [Top View](#).

## Effects Effects

The **Effects** section of the [Viewing Tool](#) controls visual effects such as depth cueing. There are [several ways to start Effects](#), a tool in the **Viewing Controls** category. Default settings are indicated in **bold**. See also: [set](#), [background](#), [preset](#)

- **depth cueing** (**on** by default) refers to front-to-back shading with the [depth-cueing color](#). The shading increases linearly from **start** to **end**.
  - **start** (default **0.5**) - where depth-cueing starts, relative to the front (0.0) and back (1.0) [global clipping planes](#); values outside the 0-1 range are allowed
  - **end** (default **1.0**) - where depth-cueing attains full strength, relative to the front (0.0) and back (1.0) [global clipping planes](#); values outside the 0-1 range are allowed
  - **color** (a [color well](#), **No Color** by default) - the color used for front-to-back shading (depth cueing). When set to No Color, the shading color will be the same as the [background color](#).
- **silhouettes** (**off** by default) - show outlines to emphasize borders and discontinuities. This setting is global (applies to all models), but when it is on, silhouettes can be toggled for individual models with the command [setattr](#), the [Selection Inspector](#), or the [molecule model attributes panel](#).
  - **color** (a [color well](#), default **No Color**, equivalent to black)
  - **width** (default **2.0**) - linewidth

- **shadows** (off by default) - display interactive shadows
  - **quality** - a tradeoff between the smoothness of shadow borders and computational demands; simply trying the settings is recommended to determine which works best for a given system:

quality setting	graphics card memory usage (MBytes)	2D texture size
coarse	4	1024
<b>normal</b> (default)	16	2048
fine	64	4096
finer	256	8192

- **transparency:**
  - **single-layer** (on by default) - render only the topmost layer of all transparent items. Showing only the topmost layer rather than all transparent layers simplifies the display and is recommended for de-emphasizing transparent parts.
  - **flat** (off by default) - make apparent transparency independent of the viewing angle. Otherwise, transparent triangles (forming objects as well as surfaces) will appear more opaque when viewed edged-on than when viewed face-on.
- **graphics quality:**
  - **subdivision** (default 1.5, maximum 20.0) - the stick, ball-and-stick, sphere, and ribbon [representations](#) consist of curved surfaces approximated by collections of planes; increasing the subdivision level increases the number of planes and the apparent smoothness. This setting also affects curved geometric objects defined in [BILD format](#), but not [molecular surfaces](#). Display [presets](#) adjust the subdivision level to 1.5 (interactive) or 5.0 (publication).
  - **multisample** (off by default) - whether to use OpenGL multisampling, *i.e.*, multiple samples per pixel so that edges are antialiased. If not well supported by the system, rendering with this option can be very slow or reveal graphics driver bugs. Multisampling can also be enabled with the startup option [--multisample](#).

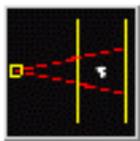
## Rotation Rotation

The **Rotation** section of the [Viewing Tool](#) controls rotation behavior. There are [several ways to start Rotation](#), a tool in the **Movement** category. See also: [cofr](#), [focus](#), [set independent](#), [Mouse preferences](#), [Set Pivot](#) and [Focus](#) in the [Actions menu](#)

- **center of rotation method:**
  - **fixed** - at a fixed point relative to the model(s), see [rotation center](#)
  - **center of models** - at the center of the bounding sphere of the displayed parts of [active](#) models
  - **independent** - a separate center for each model rather than a single collective center
  - **center of view** - continually adjusted to the current center of view (in the middle of the window, halfway between the front and back [global clipping planes](#))
  - **front center** (default) - when the view is zoomed out, behaves like **center of models**; when the view is zoomed in on parts of items, behaves like **center of view**, except the center of

rotation depth (Z-coordinate) is set to that of the frontmost displayed unclipped atom whose VDW sphere intercepts a line perpendicular to the screen in the window center. The center is not updated when only rotations are performed.

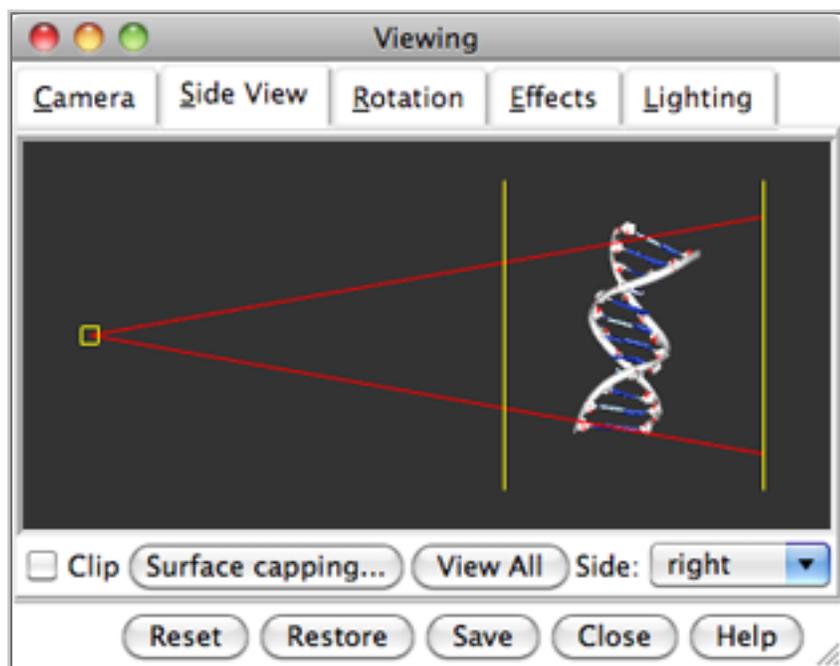
- **rotation center** (X, Y, and Z coordinates in the laboratory frame of reference) - editable only when the center of rotation method is **fixed**, meaningless when models are set to rotate independently



## Side View

The **Side View** section of the [Viewing Tool](#) provides a convenient and intuitive way to scale the view and to control [clipping](#). There are [several ways to start](#) the **Side View**, a tool in the **Viewing Controls** category.

- The yellow square on the left represents the viewer's eye position. Dragging it horizontally with the left mouse button adjusts the **scale factor**. This does *not* change the [parameter distance to screen](#). Double-clicking the eye position brings up a menu for changing the [camera mode](#).
- The vertical yellow lines represent the front and back (*hither* and *yon*) [global clipping planes](#). Each can be dragged with the left mouse button. Dragging the hither plane with the middle or right mouse button moves both clipping planes in the same direction, while dragging the yon plane with the middle or right mouse button moves the clipping planes in opposite directions.
- The red lines emanating from the eye position show the field of vision. The lines diverge when [perspective](#) is used, but are parallel in the [orthographic](#) projection.



Simultaneously holding down the Shift key reduces the speed (mouse sensitivity) of dragging operations by a factor of 10.

The **Clip** checkbox indicates [global clipping](#) status (on or off). **Surface capping...** brings up the [Surface Capping](#) dialog for controlling the appearance of planar caps where surfaces are sliced away.

Clicking **View All** turns off [global clipping](#) and adjusts the scale to include everything that is displayed. Model rotations and translations are not adjusted.

The default **Side** is **right**: it shows the relationship between the viewer and the Chimera scene from the viewer's right side. Setting **View** to **top** switches to a [top view](#) in which stereo parameters and perspective can be adjusted interactively.

## Lighting Lighting

The **Lighting** section of the [Viewing Tool](#) allows lighting parameters to be changed and [saved](#). There are [several ways to start Lighting](#), a tool in the **Viewing Controls** category. See also: [lighting](#), [lighting details](#)

**Lighting** includes two parts:

- [Intensity](#)
- [Shininess](#)

The buttons along the bottom of the dialog (**Reset**, *etc.* as described [above](#)) apply to both parts even though only one part is shown at a time.

Settings in **Lighting** (again, both parts) collectively define a scheme or *style* that can be named, saved, and later retrieved from the pulldown list marked with a black triangle in the **Lighting Settings** area. Choosing a style from the list automatically applies it. When the name **Chimera default** is shown, it is only possible to save to a different name, using **Save As...** When another name is shown, it is possible to

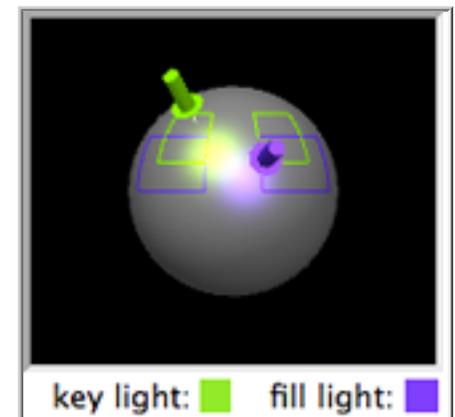
- **Save** the current settings to the name shown
- use **Save As...** to save the current settings with a new name
- **Delete** the style whose name is shown

Named lighting styles are saved in the Chimera [preferences file](#), and are only updated with any changes when **Save**, **Save As...**, or **Delete** is used. The settings in effect when a session is [saved](#) (whether or not the style has a name) are included in the [session file](#). A lighting style can also be saved, applied, or deleted with the [lighting](#) command.

The **Intensity** section includes:

- **mode** - the number and type of lights ([details...](#)):
  - **ambient** - ambient light only
  - **single** - ambient light and one directional light (the **key** light)
  - **two-point** (default) - ambient light and two directional lights: **key** and **fill**
  - **three-point** - ambient light and three directional lights: **key**, **fill**, and **back**

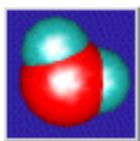
By default, only the key light gives specular highlights, and the other lights are treated as shadowless during [raytracing](#).



The directional lights can be moved interactively by dragging the color-coded solid arrows on the sphere (**green:key**, **purple:fill**, **red:back**). The back light is restricted to the rim of the sphere and is not included in the **brightness**. Outlines on the sphere in the two- and three-point modes represent commonly used light positions.

- **brightness** (default **1.16**) - the total illumination along the line of sight, disregarding any specular contributions; slider values range 0.0-2.0

- **contrast** (default **0.83**) - darkness of shading; at a given **brightness** B, decreasing the **contrast** C increases the ambient light:  $A = B(1 - C)$ . Contrast does not apply to the ambient-only mode. Values range 0.0-1.0.
- **key-fill ratio** (default **1.25**) - the ratio of (directional key + ambient) to (directional fill + ambient), disregarding any specular contributions; applies only to the two-point and three-point modes. Values range from 1.0 to an upper bound that depends on the **contrast**.



## Shininess

The **Shininess** interface controls the specular parameters of the viewed objects (rather than the light sources). Although part of the [Lighting tool](#), **Shininess** is also [handled](#) as a separate tool in the **Viewing Controls** category. See also: [lighting](#), [lighting details](#)

- **sharpness** (default **30.0**, range 1.0-128.0) - refers to the “spread” of highlights; lower values yield larger, more diffuse highlights, while higher values yield more pointlike highlights. Technically, **sharpness** is the OpenGL specular exponent. The effective highlight intensity varies as the cosine of the angle between the view direction and the direction of the reflected light, raised to the power of this exponent.
- **reflectivity** (default **1.0**, range 0.1-10.0) - scale factor for the components of the specular **color** (below). The resulting values may exceed 1.
- **color** (a [color well](#), a light gray by default) - *specular color*, that used for shiny highlights. It is added to the color that would be shown in the absence of highlights. If the color is already bright white, highlights will not be discernable; if the color is nominally green and the specular color is red, the highlights will appear to be yellow. This explanation assumes that all light sources are white or gray; however, lights have their own diffuse and specular colors, which must be [combined](#) with that of the material.

The **Shininess** interface adjusts the material properties of the Chimera default material. Stick, ball-and-stick, sphere, and ribbon [representations](#) and [molecular surfaces](#) use the default material. It is not possible to independently control the shininess of different models that use the default material. Except for surfaces generated by [Volume Viewer](#) and [Multiscale Models](#) (for which special code was written), the properties of models that do not use the default material (VRML models, for example) are not adjusted.

## Selection

In Chimera, *selection* specifies atoms, bonds, residues, molecule models, and surfaces for subsequent operations with the [Actions menu](#). When selected, these items (as well as [pseudobonds](#)) can be inspected and modified with the [Selection Inspector](#). The current selection can also be specified in [commands](#) by the word **selected**, **sel**, or **picked**.

Ways to make a selection:

- [picking](#) from the graphics window
- using the [Select menu](#)
- using the command [select](#)
- with [tools](#) such as the [Model Panel](#) or [Sequence](#)

Selecting a [ribbon](#) segment selects the atoms/bonds in the corresponding residue(s); selecting a [molecular surface](#) selects the atoms/bonds in the corresponding [surface category](#).

Ways to clear a selection:

- [picking](#) in empty space
- choosing **Select... Clear Selection** from the menu
- using the command [~select](#)
- pressing [Shift-left arrow](#)

By default, a selection is outlined in green (see the [Selection preferences](#)) and its contents are reported on a button to the right of the [status line](#). Clicking the button opens the [Selection Inspector](#).

Whether a new selection is added to, subtracted from, intersected with, or used to replace the existing selection can be set with **Select... Selection Mode** (*current\_mode*). Replacement is the default mode. Selection mode does not apply to [picking](#) from the graphics window.

Selections can be [broadened, narrowed, and inverted](#). The most recent selection operation can be undone with **Select... Undo** (or by pressing the [left arrow](#) key).

## Picking from the Graphics Window

During selection from the graphics window (*picking*), the cursor looks like a pointing hand.

- by default, **Ctrl**-button 1 (left mouse button) is assigned to picking; button assignments can be changed in the [Mouse preferences](#)
- clicking the assigned button over a selectable item selects it, whereas clicking in empty space clears the selection
- holding down the assigned button and dragging a rectangular outline selects all enclosed items
- additionally pressing the **Shift** key adds to a selection (or toggles its selection status) without

deselecting anything else

The [selection mode](#) does not apply to picking.

When a plus sign (+) has been typed into the [Command Line](#), it will be replaced by the [atom specification string](#) of the next picked atom. Each plus sign must be preceded and followed by a space (or the end of the line).

Doubleclicking while picking an atom, bond, or [pseudobond](#) elicits a [context menu](#) for [inspecting](#) the item or performing related tasks (for example, distance measurement or bond rotation).

By default, placing the mouse cursor over an object such as an atom, bond, or surface (without clicking any buttons) will show its label information in an [atomspec balloon](#). The atomspec balloon indicates what would be selected by picking at the current position, but more generally, it is useful for identifying objects without displaying their labels. Sometimes one would rather see information for the atoms under a molecular surface than for the surface itself. This can be achieved by making the surface unselectable with the mouse, for example with the command:

*Command:* [setattr](#) s piecesAreSelectable 0

Then, an atomspec balloon will report information for the atom under the cursor position, not necessarily the same as the atom giving rise to the surface patch under the cursor position. The atoms must be displayed for their atomspec balloons to appear, but they do not need to be visible; they could be behind opaque surface. However, the surface can be made transparent (with [Actions... Surface... transparency](#) or the command [surftransparency](#)) to allow viewing the underlying atoms at the same time.

## Selection with Tools

After one or more lines (models) have been chosen in the left side of the [Model Panel](#), any of several [functions](#) listed on the right side of the panel may be executed, including [select](#) and [select chain\(s\)](#).

Analogously, once one or more [pseudobond](#) groups have been chosen within the left side of the [PseudoBond Panel](#), the [select](#) button in the right side of the panel can be used to select them.

The [Sequence](#) tool shows the sequences of peptide and nucleotide chains. Selections can be made by dragging with the mouse in the sequence.

Only the most general approaches are mentioned here; many additional tools generate selections.

## Broadening, Narrowing, and Inverting Selections

A selection can be **broadened** and **narrowed** using:

- the [up arrow](#) and [down arrow](#) keys, respectively
- **Broaden** and **Narrow** in the [Select menu](#)
- the command [select](#) with the keywords **up** and **down**, respectively

Selection-level cascades:

- atom/bond ↔ residue ↔ protein secondary structure element (helix, strand, or coil) ↔ connected set of atoms (except not crossing chain ID boundaries) ↔ atoms with the same chain ID ↔ [submodel](#) ↔ molecule model ↔ all molecule models
- [pseudobond](#) (except in a [chain trace](#)) ↔ pseudobond group ↔ all pseudobond groups
- [surface piece](#) ↔ [surface model](#)

The original selection can be broadened to successively higher levels until the top level is reached, or narrowed in one step to nothing. The original selection is remembered and can be regenerated by the reverse process. Often, multiple levels collapse into one; for example, pressing the up-arrow key just once could broaden the selection from a monatomic ion to everything with the same chain ID.

If the original selection includes both atom(s) and pseudobond(s), the selection will broaden to pseudobond group in the same step as it broadens to chain, and to all pseudobond groups in the same step as it broadens to all molecule models. Although implemented as pseudobonds [chain trace](#) bonds are treated like regular bonds between residues for the purposes of broadening and narrowing selections.

See also the [keyboard shortcuts](#) **rn**, **rp**, **ri** for shifting a selection to the next residue in a chain, the previous residue, and to include intervening residues, respectively. These can be used as commands with [ac](#) (for example: **ac rn**).

A selection can be **inverted** to select the currently deselected atoms/bonds and *vice versa*:

- within models containing selections, by:
  - pressing the [right arrow](#) key
  - using **Select... Invert (selected models)**
  - using the command [select invert sel](#)
- within all models, by:
  - pressing [Shift-right arrow](#)
  - using **Select... Invert (all models)**
  - using the command [select invert](#)

## Summary of Arrow Key Shortcuts

When the keyboard focus is in the main Chimera window, arrow key shortcuts adjust the selection:

- **up arrow** - [broaden](#) the selection
- **down arrow** - [narrow](#) the selection
- **right arrow** - [invert](#) the selection within selected models
- **Shift-right arrow** - [invert](#) the selection within all models
- **left arrow** - undo the most recent selection operation
- **Shift-left arrow** - clear the selection


[Basic Functions Index](#)
[Commands Index](#)

## Atom Specification

[Overview](#)
[Hierarchical Specifiers](#)

- [Basic](#)
- [Subcategories](#)
- [Wild Cards](#)

[Zones](#)
[Built-in Classifications](#)
[Attributes](#)

- [Atom](#)
- [Residue](#)
- [Model](#)
- [Attr Names](#)

[Combinations](#)
[Surfaces, Other](#)

## Atom Specification

Atoms may be specified in [commands](#) using:

- [hierarchical specifiers](#)
- [zones](#)
- [built-in classifications](#)
- [attributes](#)

or [combinations](#) of these. Some commands accept specifications of [non-atomic models](#).

In many commands where the specification is the last argument (e.g. [color](#)), a blank specification means "all."

In commands where the specification is not necessarily the last argument (e.g. [findclash](#)), specifications with embedded spaces should be enclosed in single or double quote marks.

When a plus sign (+) has been typed into the [Command Line](#), it will be replaced by the atom specification string of the next [picked](#) atom. Each plus sign must be preceded and followed by a space (or the end of the line). In addition, the following are valid atom specifications:

- **selected**, **sel**, or **picked**, indicating atoms that are [selected](#)
- names of previously [saved selections](#), using **sel=selection\_name** or just *selection\_name*; if a saved selection has the same name as a [surface category](#), the saved selection will be used.

Those familiar with atom specification in Midas may wish to consult the [summary of differences](#).

## Hierarchical Specifiers

BASIC ATOM SPECIFICATION		
Symbol	Reference Level	Definition

#	model	number assigned to the model by default or by the user with the <a href="#">open</a> command.
:	residue	residue name <i>OR</i> residue sequence number, with any insertion code appended
::	residue	residue name
@	atom	atom name

Each file of coordinates opened in Chimera becomes a *model* with an associated model ID number. Model numbers are assigned by the user with the [open](#) command or sequentially by default. Each model contains one or more *residues*, and each residue contains one or more *atoms* with names that are unique within that residue. Thus, an atom can be specified by its model number, residue number, and atom name. The lack of a specifier is interpreted to mean all units of the associated reference level; for example, if a model number is not given, the specification refers to all models.

Residue and atom names are read from the input file. In [PDB format](#), a standard nomenclature is used for standard amino acid and nucleic acid residues. Asterisks (\*) in input atom names will be translated to prime symbols ('), but translated back if coordinates are later [saved](#) to a file. Prime symbols in input atom names are not translated on input or output.

Model and residue numbers are integers, although a residue number may have an insertion code directly appended. Multiple model numbers or residue numbers may be indicated by comma-separated lists and/or one or more ranges of the form *start-end*. The words **start** and **end** can be substituted for *start* and *end*, respectively, and **all** can be substituted for the whole range (same as no specification at that level).

Residue names and numbers cannot be specified together using a single colon, but both can be specified in a single line by using more than one colon. For example, to specify all residues named HEM and residue 52:

**:hem:52 (NOT :hem,52)**

More examples:

**#0**

- all atoms in model 0

**#3:45-83,90-98**

- residues 45-83 and 90-98 in model 3

**#0:12@CA,N**

- alpha carbon and nitrogen of residue 12 in model 0

**:12,14@CA**

- alpha carbons of residues 12 and 14

**:12:14@CA**

- all atoms of residue 12 and alpha carbon of residue 14

**:12-20@CA:14@N**

- alpha carbons of residues 12-20 and nitrogen of residue 14

**:lys**

- or -

**::lys**

- all lysine residues

In some cases, the [basic notations](#) cannot uniquely specify the model, residue, or atom of interest:

SUBCATEGORIES	
General Form	Explanation
<i>#model(s).submodel(s)</i>	when a single PDB file contains multiple MODELS, they are considered submodels 1, 2, ... of a single model in Chimera
<i>:residue(s).chain(s)</i>	when a single model contains multiple chains, a unique specification includes both residue number and chain identifier
<i>@atom(s).altloc(s)</i>	when a single residue contains alternate locations of certain atoms, an independent specification includes both atom name and alternate location identifier

*Submodel(s)* are integers and may be indicated by a single value or a range of the form *start-end*. The words **start** and **end** can be substituted for *start* and *end*, respectively, and **all** can be substituted for the whole range (same as no submodel specification).

*Chain(s)* and *altloc(s)* are alphabetical characters. Any residue in PDB HETATM records (or the mmCIF equivalent) that does not already have a chain identifier is assigned to chain **het**, unless the residue is named WAT or HOH, in which

case it is assigned to chain **water**; residue numbers are unchanged. (See also the [residue attribute isHet](#).) Residue specifications with chain IDs omitted will match residues in chains with single-character IDs but not residues in these special chains. For example, :12 includes :12.A and :12.B but not :12.water or :12.het.

Capitalization of residue and atom names, chain identifiers, insertion codes, or alternate location identifiers is not important, with one exception: when a model contains both uppercase and lowercase chain identifiers, case matters for chain specification in that model only.

Subcategorizations are appended to the basic specification. The symbol for the relevant category (**#**, **:**, or **@**) must precede the subcategory specification, although they need not be immediately adjacent. **Because commas are used only to separate values of the basic reference levels (model, residue, and atom), they cannot be used to separate values of the sublevels directly.** For example,

**#0.1-3,5**

is interpreted as submodels 1-3 of model 0 and all of model 5, while

**#0.1-3,.5**

indicates submodels 1-3 of model 0 and submodel 5 of all models.

A subcategory specification applies only to the preceding category value(s) not separated from it by any commas. Thus,

**:50.B,.D**

indicates residue 50 in chain B and all residues in chain D;

**:12-15,26-28.a,45.b@ca**

specifies CA atoms in residues 12-15 in all chains (except the [special chains het and water](#)), 26-28 in chain A, and 45 in chain B. The following specifies residues 12-15 in no-ID chains only:

**:12-15.**

More examples:

**:.b**

- chain B

**:.a@n,ca,c,o**

- peptide backbone atoms in chain A

**:195.a,221.a@n,ca,c,o**

- peptide backbone atoms in residues 195 and 221 of chain A

**:522.water**

- [water](#) residue 522 (a HETATM residue named HOH or WAT without a chain ID in the input file)

**@.a**

- all atoms with alternate location identifier A

## WILD CARDS

The **global wild card** \* matches all atoms in a residue or all residues in a model. It stands alone as a symbol, that is, it cannot be used to match *parts* of names, such as **G\*** or **\*A**. The **partial wild card** = matches parts of atom or residue names but not parts of residue sequence numbers; similarly, the **single-character wild card** ? matches single characters within residue or atom names but not single digits within residue sequence numbers. For example:

**#1:12@\***

- or -

**#1:12**

- all atoms in residue 12 of model 1

**#0,1,2:50-\*@CA**

- all alpha carbon atoms in residues 50 to the end of models 0, 1 and 2

**#2:G??**

- all residues which have three-letter names beginning with **G** in model 2

**:fmn@?1**

- atoms within residue FMN which have two-letter names ending with **1**

**@S=**

- all atoms which have names beginning with **S**; in general, this will be all sulfur atoms

```
#0@H@H?@H??
```

- all atoms with one-, two-, or three-letter names beginning with **H** in model 0

---

## Zones

Zone specifiers indicate atoms and residues that are within or beyond a given distance of the referenced atom(s). **z<** and **zr<** specify all *residues* with any atom within the given distance from the referenced atoms. **za<** specifies all *atoms* within the given distance. **z>**, **zr>**, and **za>** yield the sets complementary to their < counterparts. For example,

```
#1:gtp za<10.5
```

specifies all atoms within 10.5 Å of any atom in residue GTP, model 1.

Zone specifiers refer to atoms only, not surfaces; however, the command [zonesel](#) allows basing zones on surfaces and including surfaces in zones.

---

## Built-in Classifications

Any of the entries in the following sections of the [Select menu](#) can be used for command-line atom specification:

- **Chemistry**
- **Residue...** [amino acid category](#)
- **Structure**

For example, **Structure** includes

- type of molecule: **protein**, **nucleic acid**, [markers](#)
- protein secondary structure: **helix**, **strand**, **coil**
- [automatic categories for surface calculation](#): **main**, **ligand**, **solvent**, **ions**

The same pattern of capitalization and spaces, if any, as shown in the [Select menu](#) should be used. Where there is ambiguity, the parent menu should be

included in the specification, for example, "IDATM type.H" or "element.H" instead of "H" alone. The parent menu can be included in this manner even when not necessary.

One can also use custom amino acid categories that have been defined with [ResProp](#) and custom categories for surface calculation that have been defined with the command [msms cat \(surfcats\)](#). If a [saved selection](#) has the same name as a surface category, the saved selection will be used.

Some examples:

**side chain/base.without CA/C1'**

- or -

**without CA/C1'**

- atoms in amino acid side chains (not including CA) and atoms in nucleic acid bases (not including C1')

**#1 & Mg**

- magnesium atoms/ions in model 1

**helix & positive**

- residues in helices that are also in the **positive** [amino acid category](#)

**carboxylate**

- atoms in carboxylate groups

**solvent**

- atoms [automatically categorized](#) as **solvent**

## Attributes

[Attributes](#) are properties of atoms, residues and models. The **slash mark /** indicates specification by [attribute name](#) and value. The symbol for the relevant category (@ for [atom attributes](#), : for [residue attributes](#), # for [model attributes](#)) **must precede the slash mark**, although it need not be immediately adjacent.

Multiple attributes at the same reference level (different atom properties, for example) can follow a single slash mark and should be separated by **and** or **or**. When **and** and **or** occur in the same list, **and** has higher priority (**and-**

separated lists can be considered as grouped within parentheses).

The attribute *names* are case-sensitive; the attribute *values*, if any, are case-sensitive if specified with ==, but not if specified with =. Attribute values containing spaces (some color names, for example) must be enclosed by double quotes. The **exclamation mark !** indicates that the atoms, residues, or models must **not** match the subsequent attribute specification. For yes/no properties the syntax is *!attribute\_name*, and for multivalued properties the syntax is *attribute\_name!=value*.

Attributes with numerical values can also be used with > (greater than), < (less than), >= (greater than or equal to), and <= (less than or equal to).

Color values can be specified:

- by name, [built-in](#) or defined previously with [colordef](#)
- as a comma-separated list of component values *red,green,blue,alpha* where each value can range from zero to 1 and specification of *alpha* (1 - transparency) is optional

When placed before an attribute name, the caret ^ indicates that the atoms, residues, or models have not been assigned any value for the attribute. For example, *:/^kdHydrophobicity* designates residues (such as water or nucleic acids) that lack a [Kyte-Doolittle hydrophobicity](#) assignment.

The operators ~ and !~ can be used instead of = and !=, respectively, to indicate that the subsequent string should be treated as a [regular expression](#).

SELECTED ATOM ATTRIBUTES <sup>**</sup> —	
Atomspec Usage	Explanation
<b>altLoc</b> = <i>altloc</i>	<i>altloc</i> is the alternate location identifier of the atom
<b>areaSAS</b> = <i>sasa</i>	<i>sasa</i> is the <a href="#">solvent-accessible</a> surface area of the atom (available when a <a href="#">molecular surface</a> has been computed)
<b>areaSES</b> = <i>sesa</i>	<i>sesa</i> is the <a href="#">solvent-excluded</a> surface area of the atom (available when a <a href="#">molecular surface</a> has been computed)
<b>bfactor</b> = <i>bfactor</i>	<i>bfactor</i> is the B-factor value of the atom (see also <a href="#">Thermal Ellipsoids</a> )
<b>color</b> = <i>color</i>	<i>color</i> is the color of the atom (assigned on a per-atom basis; see <a href="#">coloring hierarchy</a> )
<b>defaultRadius</b> = <i>rad</i>	<i>rad</i> is the <a href="#">default VDW radius</a> of the atom in Å

<b>display</b>	whether display is enabled at the atom level (see <a href="#">display hierarchy</a> )
<b>drawMode=<i>mode</i></b>	<i>mode</i> can be 0 (synonyms: dot, wire, wireframe), 1 (sphere, cpk, space-filling), 2 (endcap, stick), or 3 (ball, ball and stick, ball-and-stick, ball+stick, bs, b+s); see <a href="#">draw mode</a>
<b>element=<i>atno</i></b>	<i>atno</i> is the atomic number or the element symbol
<b>idatmType=<i>type</i></b>	<i>type</i> is the <a href="#">atom type</a>
<b>label</b>	whether the atom is labeled
<b>label=<i>label</i></b>	<i>label</i> is the text of the atom label
<b>labelColor=<i>labcolor</i></b>	<i>labcolor</i> is the color of the atom label
<b>name=<i>name</i></b>	<i>name</i> is the atom name
<b>occupancy=<i>occupancy</i></b>	<i>occupancy</i> is the occupancy value of the atom
<b>radius=<i>radius</i></b>	<i>radius</i> is the current VDW radius of the atom in Å (may have been changed by the user from the <a href="#">default VDW radius</a> )
<b>serialNumber=<i>n</i></b>	<i>n</i> is the atom serial number in the input file
<b>surfaceCategory=<i>category</i></b>	<i>category</i> is the name of the surface calculation category to which the atom has been assigned <a href="#">automatically</a> or manually using <a href="#">surfcats</a>
<b>surfaceDisplay</b>	whether molecular surface display is turned on for the atom (however, this can be true even for atoms that do not contribute to the molecular surface)

Examples:

**@ca/!label and color!=green and color!=red**

- or -

**@/name=ca and !label and color!=green and color!=red**

- atoms named CA which are not labeled, and are not green or red

**@n/drawMode=1 and color=green**

- atoms named N that are green and drawn as spheres

**@n/drawMode=1 or color=green**

- atoms named N that are green and/or drawn as spheres

**@/color=yellow or color=blue and label**

- atoms that are yellow and atoms that are both blue and labeled

**@/color!=yellow or color!=blue**

- all atoms, because if an atom is yellow it fulfills the criterion of not being blue, and *vice versa*

**@/bfactor>=50**

- atoms with B-factor values greater than or equal to 50

**@/bfactor>=20 and bfactor<=40**

- atoms with B-factor values ranging from 20 to 40

SELECTED RESIDUE ATTRIBUTES <sup>**</sup> —	
Atomspec Usage	Explanation
<b>areaSAS=sasa</b>	sasa is the <a href="#">solvent-accessible</a> surface area of the residue (available when a <a href="#">molecular surface</a> has been computed)
<b>areaSES=sesa</b>	sesa is the <a href="#">solvent-excluded</a> surface area of the residue (available when a <a href="#">molecular surface</a> has been computed)
<b>isHelix</b>	whether the residue is in a helix (true is only possible for amino acids)
<b>isHet</b>	whether the residue is in PDB HETATM records (or the mmCIF equivalent)
<b>isSheet OR isStrand</b>	whether the residue is in a beta strand (true is only possible for amino acids)
<b>kdHydrophobicity=value</b>	value is the <a href="#">Kyte-Doolittle hydrophobicity</a> of the amino acid residue
<b>numAtoms=N</b>	N is the total number of atoms in the residue
<b>phi=angle</b>	angle is the protein/peptide backbone $\phi$ dihedral angle (C <sub><i>i-1</i></sub> -N-CA-C)
<b>psi=angle</b>	angle is the protein/peptide backbone $\psi$ dihedral angle (N-CA-C-N <sub><i>i+1</i></sub> )

<b>ribbonColor</b> = <i>ribcolor</i>	<i>ribcolor</i> is the color of the residue's ribbon segment (see <a href="#">coloring hierarchy</a> )
<b>ribbonDisplay</b>	whether ribbon display is turned on for the residue (however, this can be true even when the residue is a type that does not have any ribbon, such as water)
<b>ssId</b> = <i>N</i>	<i>N</i> is the secondary structure element identifier, for example, 1 for residues in the first helix and first strand (starting from the N-terminus)
<b>type</b> = <i>resname</i>	<i>resname</i> is the residue name
<b>uniprotIndex</b> = <i>N</i>	<i>N</i> is the residue number in the corresponding <a href="#">UniProt</a> sequence (discerned from HEADER and SEQRES information in the structure PDB file using a web service provided by the <a href="#">RCSB PDB</a> )

Helix and strand assignments are taken from the input structure file. When the input file lacks secondary structure information, [ksdssp](#) is called automatically to generate helix and strand assignments. The [ksdssp](#) command (or [compute SS](#) in the [Model Panel](#)) can also be used to recompute assignments.

Examples:

**:/type!=gly and type!=pro**

- all residues not named GLY or PRO

**:/isStrand :/isHelix**

- OR -

**:/isStrand or isHelix**

- all amino acid residues in beta strands or helices

**:/isStrand and isHelix**

- nothing, because the criteria are mutually exclusive

SELECTED MOLECULE MODEL ATTRIBUTES <sup>**</sup>	
Atomspec Usage	Explanation
<b>ballScale</b> = <i>factor</i>	<i>factor</i> is ball radius relative to VDW radius
<b>color</b> = <i>color</i>	<i>color</i> is the color assigned on a per-model basis (see <a href="#">coloring hierarchy</a> )

<b>display</b>	whether display is enabled at the model level (see <a href="#">display hierarchy</a> )
<b>lineWidth=<i>width</i></b>	<i>width</i> is the linewidth of bonds in the model in the wire <a href="#">representation</a>
<b>numAtoms=<i>N</i></b>	<i>N</i> is the total number of atoms in the model
<b>numResidues=<i>M</i></b>	<i>M</i> is the total number of residues in the model
<b>ribbonInsideColor=<i>color</i></b>	<i>color</i> is the color used for the insides of peptide/protein helix ribbon segments
<b>stickScale=<i>factor</i></b>	<i>factor</i> is stick radius relative to bond radius (the default bond radius is 0.2 Å)

\*\* Additional [attributes](#) can be created:

- arbitrarily with [Define Attribute](#), [defattr](#), or [setattr](#)
- by combining other attributes using the [Attribute Calculator](#)
- by various other Chimera tools such as [Add Charge](#), [Area/Volume from Web](#), [Values at Atom Positions](#), and [Multalign Viewer](#)

## Attribute Names

The preceding tables [[atoms](#)] [[residues](#)] [[molecule models](#)] include the names of some commonly used [attributes](#). Additional attributes are listed in [attribute inspector](#) dialogs, and yet others may be generated only when a particular tool is used, or created arbitrarily by the user.

Attribute name lookup in Chimera:

- attribute names are shown in the [balloon help](#) of [attribute inspector](#) dialogs
- [Select by Attribute](#) lists most numerical, string-valued, and boolean attributes of atoms, residues, and molecule models; [Render by Attribute](#) lists most numerical ones
- the command [list resattr](#) lists most residue attributes
- Data descriptors (attributes) defined in the C++ layer of Chimera can be listed by entering the following into [IDLE](#) (under **Tools... General Controls**):

```
help(chimera.Class)
```

where *Class* can be **Atom**, **Residue**, or **Molecule**. This approach also works for **Bond**, **PseudoBond**, and **PseudoBondGroup**. Attributes

defined in the Python layer will not be included, however.

## Combinations

Atom specifications can be combined with the operators:

- **&** for **intersection** (AND)
- **|** for **union** (OR)
- **~** for **negation** (NOT)

Space-delimiting these operators is optional. When **&** and **|** occur in the same list, **&** has higher priority (**&**-separated lists can be considered as grouped within parentheses).

Note that a different set of operators (**and**, **or**, **!**, *etc.*) are used to combine [attribute](#) tests; however, both types of operators can be used within the same specification.

Examples:

```
#1:/type=asp or type=glu & #0 z<10
- or -
#1:asp,glu & #0 z<10
```

- aspartate residues and glutamate residues in model 1 that are within 10 Å of model 0

```
:cys@sg & ~disulfide
- or -
:cys&S&~disulfide
```

- cysteine sulfur atoms not participating in a disulfide bond

```
ions za<4 & ~ ions
```

- atoms within 4 Å of atoms [categorized](#) as ions, excluding the ions themselves

```
ligand z<5 &~ ligand &~ solvent
```

- residues with any atoms within 5 Å of residues [categorized](#) as ligand, excluding ligand and solvent

```
Ng+|N3+
```

- guanidinium nitrogens and  $sp^3$ -hybridized, formally positive nitrogens (see [atom types](#))

---

## Surfaces and Other Model Types

Some commands can apply to things other than molecule models (atomic coordinates) and [molecular surfaces](#). To handle such cases, “atom specification” has been extended as follows:

- an entire [surface model](#) (all of its pieces) can be specified by model number preceded by #, or by [selection](#) and using the word **selected**, **sel**, or **picked**
- [surface pieces](#) can be specified individually by [selection from the screen](#) and using the word **selected**, **sel**, or **picked**
- [other types of models](#) can be specified by model number preceded by #

Which of these can be used and whether they can be [combined](#) with atom specifications depends on the command.

---

UCSF Computer Graphics Laboratory / November 2013

## Tools

See the [Tools](#) section of the [Chimera User's Guide](#) for a listing of tools (extensions) and their individual manual pages.

There are several ways to start a tool:

- by choosing its name from the [Tools menu](#)
- by specifying its name with the command [start](#)
- by choosing its name from the [Favorites menu](#) (if it has been listed there, using the [Tools preferences](#))
- by clicking its icon in the [toolbar](#) (if the icon has been placed there, using the [Tools preferences](#))
- by specifying `--start toolname` when [starting Chimera](#) from the system command line
- automatically upon [Chimera startup](#) (if previously set to do so, using the [Tools preferences](#))
- by opening a file whose [type](#) is [specific to the tool](#)

Some tools can also be started by choosing the corresponding [function](#) from the [Model Panel](#) or by using an [accelerator](#). Depending on the tool, something may happen right away, or dialog boxes requesting input files or parameter settings may appear.

[Instances](#) of certain tools that have been started or data tables that have been generated during a Chimera session are available from the **Tools** menu, allowing them to be raised, hidden, or quit. For example, if the [Command Line](#) has been started, it can be hidden by choosing **Tools... Command Line... Hide**; if a [Multalign Viewer](#) window displaying the alignment file apoex.fa becomes obscured by other windows, it can be raised with **Tools... MAV - apoex.fa... Raise**.

Regardless of whether they represent instances of tools, Chimera [active dialogs](#) can be raised using [Rapid Access](#).



## The Model Panel

The **Model Panel** lists the models in Chimera and conveniently enables [many operations](#) upon them. There are [several ways to start](#) the **Model Panel**, a tool in the **General Controls** category.

Each file of atomic coordinates opened in Chimera becomes a *model* with an associated model ID number. Surfaces and other types of models also have ID numbers. Some tools in Chimera create models that are hidden from the **Model Panel**, however.

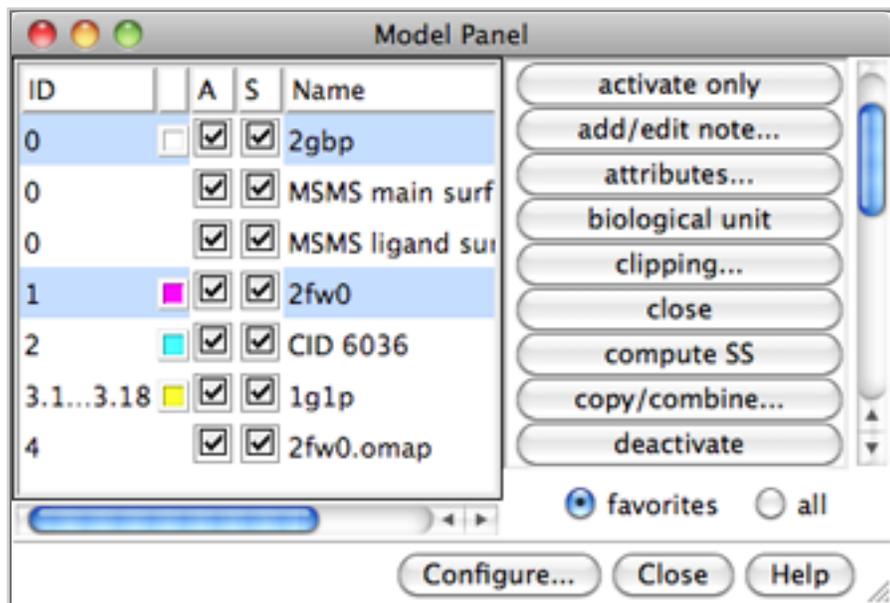
A molecular structure file may contain multiple sets of coordinates for the same set of atoms. In PDB files, these coordinate sets are delimited with `MODEL` and `ENDMDL` records. Appending a [submodel number](#) to the primary model number allows uniquely specifying atoms despite multiple occurrences of the “same” atom (same atom name, residue name, and residue number) in the file. Submodel numbers are assigned sequentially starting with 1 (`#0.1`, `#0.2`, etc.).

The term “models” will be used to indicate submodels and/or models that are not subdivided into submodels because they are operationally the same: each can be handled independently of the others and listed separately in the **Model Panel**. At first, the submodels in a model may be collapsed into a single row (for example, model #3 in the figure below), but the listing can be expanded to the individual submodels using the [group/ungroup](#) function.

The figure shows columns:

- model **ID** number
- model color (a [color well](#))
- whether **Active** ([movable](#))
- whether **Shown** (display-enabled)
- **Name** ([model name](#))

Individual models or multiple models can be chosen with the left mouse button. A block of models can be chosen by dragging, or by clicking on the first (or last) line in the desired block and then **Shift**-clicking on its last (or first) line. **Ctrl**-click toggles the state (chosen or not) of a single line.



Once one or more models have been [chosen](#) within the left side of the **Model Panel**, any of several [functions](#) represented by buttons on the right side of the panel can be executed. Just the **favorites** or **all** functions can be listed, using the setting below the list. Functions can be added to or removed from the favorites by showing **all** functions and using the **Fav** checkboxes. The [Configure...](#) button allows customizing which columns appear in the left side of the panel and which [functions](#) are applied to double-clicked models.

## Functions:

- **activate** - [activate for motion](#); the opposite of [deactivate](#)
- **activate all** - activate all models even if not chosen in the left side of the panel (the previous pattern of activities is saved and can be restored using the button's alternate function, **restore activities**)
- **activate only** - activate the chosen model(s) and deactivate all the others
- **add hydrogens...** add hydrogens to the chosen model(s); equivalent to using [AddH](#)
- **add/edit note...** add a text description for the model (the **Model Panel** can be [configured](#) to show these descriptions in a **Note** column; see also [session notes](#))
- **attributes...** list [model attributes](#) and allow changes
- **biological unit** - add copies to generate the biological unit according to [BIOMT information](#) in the input file; full atomic copies are generated for N-mers of  $N \leq 6$ , whereas for N-mers of  $N \geq 7$ , the copies are generated as [low-resolution surfaces](#) from [Multiscale Models](#). Not all structures include [biological unit information](#). See also: [fetching](#) PDB-biounit ([biological assembly](#)) and PQS files
- **clipping...** open the [Per-Model Clipping](#) tool
- **close** - close (remove) the model(s)
- **color by SS...** color peptides/proteins by secondary structure with the [Color Secondary Structure](#) tool
- **compute SS** - define the secondary structure of peptides/proteins in the chosen model(s) using [ksdssp](#) with the specified parameter values. Hitting return (**Enter**) or clicking **OK** runs the calculation and dismisses the dialog, while clicking **Apply** runs the calculation without dismissing the dialog. Clicking **Save as Defaults** saves the parameter settings in the [preferences file](#); subsequently, those settings will be used with the [ksdssp](#) algorithm when peptide/protein structures that lack secondary structure assignments are read in, when [MatchMaker](#) is used with the **Compute secondary structure assignments** option, or when **compute SS** is specifically invoked.
- **copy/combine...** combine multiple molecule models into a single model or make a copy of a single molecule model (see also the command [combine](#)). The new model will be assigned the specified **ID** number and its **name** will appear in the **Model Panel** and other model lists. A **coordinate system** (reference model) must be specified. The reference model's untransformed coordinates (if part of the combination) and current transformation will be carried over into the new model. The coordinates of any other model in the combination will be transformed relative to the reference model. To ensure that residues in the new model will be uniquely specifiable, it is necessary to allow either renaming chains (changing chain IDs) or renumbering residues when the input models contain more than one chain with the same ID. The original or **source** model(s) can be closed. Clicking **OK** generates the new model and dismisses the dialog, while clicking **Apply** generates the new model without dismissing the dialog. Changes in chain identifiers or residue numbering are reported in the [Reply Log](#).
- **deactivate** - [deactivate for motion](#); the opposite of [activate](#)
- **focus** - focus the view on the chosen model(s) and set the [center of rotation method](#) to **center of view** (as if using the command [focus](#) on those models)
- **group/ungroup** - collapse multiple chosen entries into a group with a specified name, or expand a single chosen entry that is already a group into its contents; grouping is hierarchical (groups may contain other groups)
- **hide** - disable display of the chosen model(s) (see [display hierarchy](#)); the opposite of [show](#)
- **match...** open the [MatchMaker](#) tool to superimpose structures with similar sequences and folds; at least two molecule models must be chosen for this operation to be available

- **rainbow...** color the chosen model(s) over a range with the [Rainbow](#) tool (see also the command [rainbow](#))
- **Ramachandran plot...** show the distribution of peptide  $\phi, \psi$  angles (a [Ramachandran plot](#)) for each chosen model that contains protein; also implemented as the command [ramachandran](#)
- **rename...** change the name of each chosen model and/or [group](#). The name is initially based on input filename or assigned by Chimera when creating the model (for multi-Mol2 files, see also the [New Molecules preferences](#)).
- **render attribute...** open the [Render/Select by Attribute](#) tool to depict (with color, *etc.*) or [select](#) atoms, residues, and molecule models by their [attribute](#) values
- **save sphgen** (available when a [sphere file](#) has been opened) - save spheres or atoms as a [sphere file](#) in the [sphgen](#) format used by [DOCK](#)
- **select** - [select](#) the chosen model(s) for subsequent operations or commands
- **select chain(s)...** [select](#) subsequently specified chains in the chosen model(s)
- **sequence...** show chain sequences (start the [Sequence](#) tool)
- **show** - enable display of the chosen model(s) (see [display hierarchy](#)); the opposite of [hide](#)
- **show all atoms** - show the chosen model(s) *and* display all atoms within them
- **show only** - show the chosen model(s) and hide all of the others
- **surface main** - for the chosen model(s), display the molecular surface of any atoms [categorized](#) as **main**
- **tile...** open [Tile Structures](#) to arrange the (two or more) chosen models in a plane
- **toggle active** - toggle the status of each chosen model between [activated and deactivated for motion](#) (make the status of each its current opposite)
- **trace backbones** - in each chosen model, display only a simplified trace for bonded chains containing amino acid or nucleic acid residues, consisting of real backbone bonds ( $-\text{[N-CA-C]}_n$  - in peptides and  $-\text{[O5'-C5'-C4'-C3'-O3'-P]}_n$  - in nucleic acids)
- **trace chains** - in each chosen model, display only a simplified trace for bonded chains containing amino acid or nucleic acid residues, consisting of one atom per residue (connect the CA atoms in peptides, C4' atoms in nucleic acids)
- **transform as...** transform each (initially) chosen model using the rotation/translation matrix of the single model chosen when the dialog's **OK** button is clicked
- **UniProt info...** start [PDB/UniProt Info](#) to retrieve [UniProt](#) annotations and map them to sequence
- **write PDB** - save models [as PDB files](#)

**Close** dismisses the **Model Panel**. **Help** opens this manual page in a browser window.

## Model Panel Configuration

The **Configure...** button brings up another panel with three sections for customizing the **Model Panel**:

- The **Buttons** section was used in older versions of Chimera to classify [functions](#) as frequently or infrequently used. Currently Chimera uses a favorites list instead, where just **favorites** or **all** functions can be shown in the **Model Panel**. To configure which functions are favorites, change from showing just **favorites** to showing **all** in the **Model Panel**, then use the **Fav** checkboxes.
- The **Columns** section determines which model attributes will be shown in the left side of the

**Model Panel.** The descriptions below are in the order in which the columns would be shown, from left to right.

- **ID** - model (and submodel, if applicable) number
- **[color]** - for each molecule model, a [color well](#) showing the [model-level color](#); blank for molecular surface models and [other model types](#). Clicking a color well opens the [Color Editor](#) and allows the color to be changed.
- **Active** - whether the model is [activated for motion](#); see [activate](#) and [deactivate](#)
- **Shown** - whether the model is shown; see [hide](#) and [show](#)
- **Name** - model name; see [rename](#)
- **Note** - user-added descriptive text for the model; see [add/edit note](#)
- **Input file** - for local files, a pathname (relative if the file is in a subdirectory of the working directory, otherwise full); for files [fetched](#) from databases, an identifier or filename for that database entry

**Show model color behind model name** shows the [model-level color](#) of each molecule model in the **Name** column, with the text of the name when the model is [chosen](#), otherwise with a swatch of color behind the name.

- The **Double Click** section controls which [functions](#) will be executed upon a model when the model is double-clicked in the left side of the **Model Panel**. The default is [attributes](#); however, there can be zero or many functions in the list for execution. Clicking an entry in the **Function menu** on the right causes it to appear in the **Execution list** on the left. Clicking an entry in the **Execution list** removes it from that list. The functions will be executed in the order listed.

**Model Panel** customization settings are stored in a user's [preferences file](#).

## The PseudoBond Panel

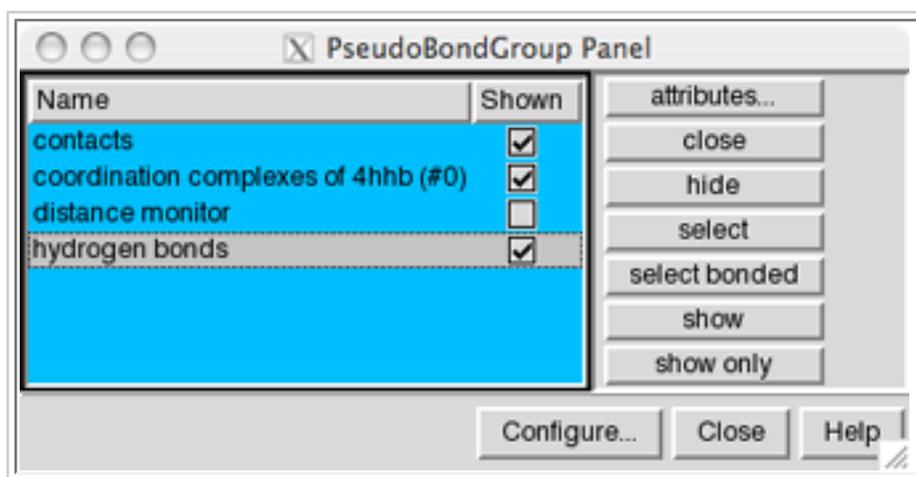
*Pseudobonds* are drawn to signify connections other than standard covalent bonds, such as [distance monitors](#), [hydrogen bonds](#), [contacts](#), and [metal coordination bonds](#). They are also used to indicate missing segments or loops in crystal structures. A *pseudobond group* is a named set of zero to many pseudobonds that can be treated collectively.

Pseudobonds can be created arbitrarily (drawn between any pairs of atoms) using [PseudoBond Reader](#).

The **PseudoBond Panel** lists the current pseudobond groups and conveniently enables many operations upon them. There are [several ways to start](#) the **PseudoBond Panel**, a tool in the **General Controls** category.

Once one or more groups of pseudobonds have been chosen within the left side of the **PseudoBond Panel**, any of several [functions](#) represented by buttons on the right side of the panel can be executed.

Individual pseudobond groups or blocks of groups can be chosen with the left mouse button. **Ctrl**-click toggles the state (chosen or not) of single line. A block of pseudobond groups can be chosen by dragging, or by clicking on the first (or last) line in the desired block and then **Shift**-clicking on its last (or first) line.



**Configure...** allows customization of which settings are shown as checkboxes on the left side of the panel, which buttons are included in the right side of the panel, and what [functions](#) are applied to double-clicked pseudobond groups. **Close** dismisses the **PseudoBond Panel**. **Help** brings up the **PseudoBond Panel** manual page in a browser window.

The functions are:

- **attributes...** list [pseudobond group attributes](#) and allow changes
- **close** - remove the chosen group(s)
- **hide** - disable display of the chosen group(s) (see [display hierarchy](#)); the opposite of **show**
- **select** - [select](#) the chosen group(s) for subsequent operations or commands
- **select bonded** - [select](#) the atoms flanking pseudobonds in the chosen group(s)
- **show** - enable display of the chosen group(s) (see [display hierarchy](#)); the opposite of **hide**
- **show only** - show the chosen group(s) and hide all of the others

## PseudoBond Panel Configuration

The **Configure...** button brings up another panel with three sections for customizing the **PseudoBond Panel**. The three sections are organized like index cards with their names on tabs across the top: **Buttons**, **Columns**, and **Double Click**. Clicking on a tab brings the corresponding card to the front.

1. The **Buttons** section shows all the functions that can be listed on the right side of the **PseudoBond Panel**. If a checkbox is activated, the adjacent function will be included in the list.
2. The **Columns** section determines which pseudobond group attributes will be shown in the left side of the **PseudoBond Panel**:
  - o **Name** - name of the group of pseudobonds
  - o **Shown** - whether the group of pseudobonds is shown; see [hide](#) and [show](#)
3. The **Double Click** section controls which [functions](#) will be executed upon a pseudobond group when the group is double-clicked in the left side of the **PseudoBond Panel**. The default is [attributes...](#); however, there can be zero or many functions in the list for execution. Clicking an entry in the **Function menu** on the right causes it to appear in the **Execution list** on the left. Clicking a command in the **Execution list** removes it from that list. The functions will be executed in the order listed.

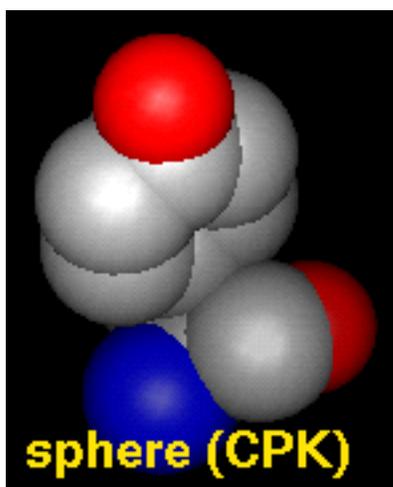
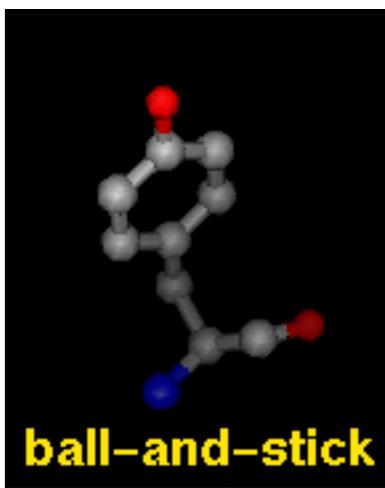
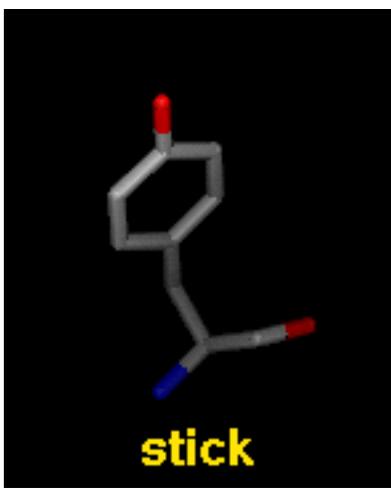
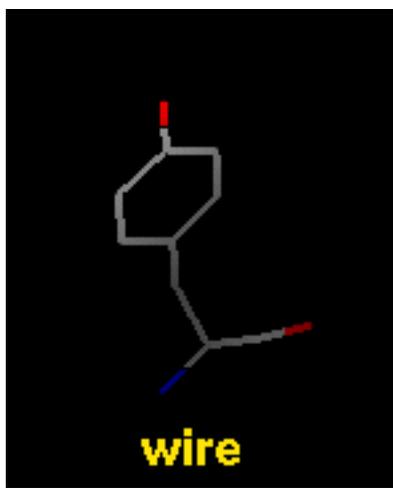
## Molecule Display

See also: [colors](#), [tips on preparing images](#), [volume display](#), [geometric objects](#)

### Atoms/Bonds

Several molecular display styles (*representations*) are available in Chimera. Atoms/bonds can be shown as:

- **wire** - a simple line drawing with dot atoms and wire bonds
- **stick** - “endcap” atoms and stick bonds
- **ball-and-stick** - ball atoms (atomic [VDW radius](#) × [ball scale](#)) and stick bonds
- **sphere** - sphere atoms (full [VDW radius](#)) and wire bonds; also called Corey-Pauling-Koltun (CPK)



A model can only have one wire linewidth, but individual atoms and bonds can be shown in different representations. Atom and bond displays can be combined with [ribbons](#) and [surfaces](#). See also: [presets](#), [New Molecules preferences](#)

The command [represent](#) sets atom/bond styles, [bondrepr](#) sets bond styles, and [linewidth](#) changes wire linewidth. Atom/bond styles and wire linewidth can also be changed using the [Actions menu](#), the

[molecule model attributes panel](#), and the [Selection Inspector](#). Ball display radius is the product of the atomic [VDW radius](#) and the [ball scale](#) attribute of a molecule model (default 0.25). Stick radius is the product of the individual [bond radius](#) (default 0.2 for all bonds) and the [stick scale](#) attribute of a molecule model (default 1.0). Ways to change [attribute](#) values include the command [setattrr](#) and the [Selection Inspector](#). Different linewidth, ball-scale, and stick-scale values can also be specified in the [New Molecules preferences](#).

[Pseudobonds](#) are drawn to show connections other than covalent bonds. For example, they can represent clashes, hydrogen bonds, or distance measurements. Pseudobond styles can be changed using the [pseudobond attributes panel](#) and the [Selection Inspector](#).

In addition:

- rings can be filled using the command [fillring](#)
- circles/disks indicating aromaticity can be shown with the command [aromatic](#)
- special representations of nucleic acid sugars and bases can be generated with the [Nucleotides](#) tool or [nucleotides](#) command
- anisotropic B-factors can be shown as ellipsoids with the [Thermal Ellipsoids](#) tool or [aniso](#) command

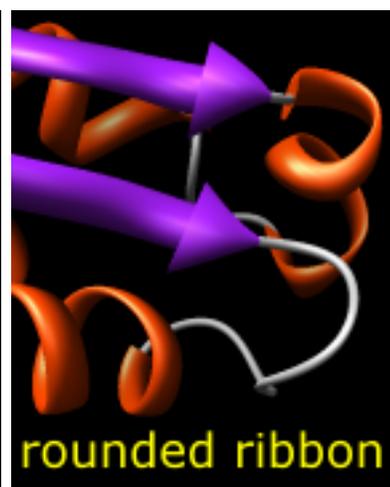
## Ribbons

Protein and nucleic acid chains can be shown with ribbons. Protein helix and strand assignments are taken from the input structure file or generated with [ksdssp](#). For nucleic acids, the ribbon simply follows the backbone. See also: [PipesAndPlanks](#), [Nucleotides](#)

Built-in ribbon styles include:

- flat
- edged
- rounded

Ribbon segments for individual residues can be shown and hidden independently and displayed in different styles. New ribbon [styles](#),



[scalings](#) (secondary-structure-dependent width and height), and [residue classes](#) (which atoms define the ribbon path) can be created using [Ribbon Style Editor](#). The command [ribbon](#) displays ribbons; style can be adjusted with [ribrepr](#), scaling with [ribscale](#), and residue class with [ribclass](#). Ribbon display and style can also be controlled with the [Actions menu](#); ribbon display, style, and scaling can be controlled in the residue section of the [molecule model attributes panel](#) or [Selection Inspector](#).

The ribbon path is interpolated and may deviate from the exact positions of the backbone atoms ([more...](#)).

The method of path calculation can be set with [ribspline](#) or in the molecule section of the [molecule model attributes panel](#) or [Selection Inspector](#).

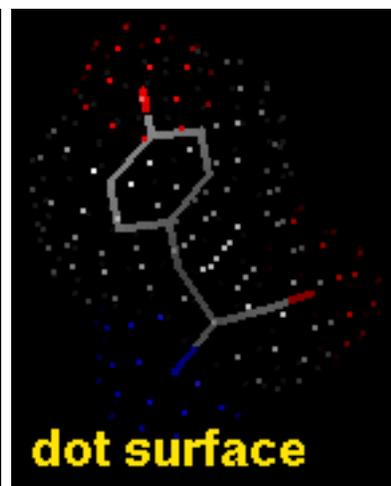
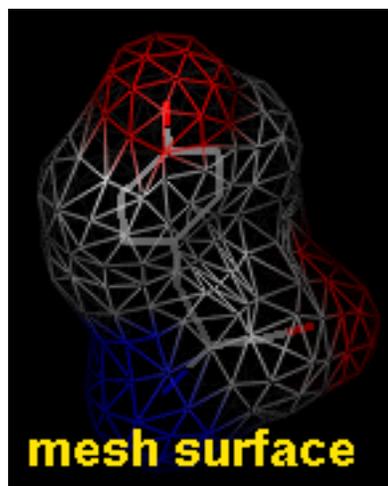
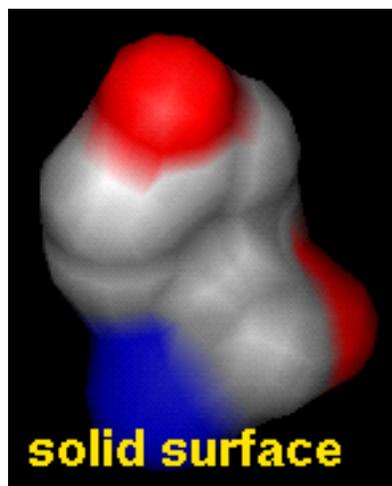
The insides of ribbons in protein helices can be colored separately using the [Color Actions](#) dialog, the [molecule model attributes panel](#), or the command [setattr](#).

## Molecular Surfaces

Chimera shows *solvent-excluded* molecular surfaces, composed of probe contact, toroidal, and reentrant surface. These differ from *solvent-accessible* surfaces, which are traced out by the probe center.

Molecular surface styles are:

- **solid**
- **mesh**
- **dot**



A molecular surface model can only have one style, but molecular surface patches for individual atoms can be shown/hidden independently and can vary in [color](#) and transparency.

The atoms in a molecular model are automatically partitioned into [categories for surfacing](#): **ligand**, **ions**, **solvent**, or **main**. By default, protein and nucleic acid chains are classified as **main**, and if multiple chains are in contact, they will be enclosed in a single surface. However, if separate surfaces for the separate chains are preferred, either of the following can be used:

- [split](#) command to make each chain a separate model
- [surfcut](#) command to create custom surface categories followed by [surface](#) to display the corresponding surfaces

When a molecular surface is computed, total solvent-excluded and solvent-accessible surface areas are reported in the [Reply Log](#), along with the values for each disconnected part, or *component*. The values are calculated analytically. Multi-component surfaces are common; for example, a **main** surface could include one or more completely enclosed interior pockets, and a **ligand** surface could separately enclose each of multiple ligand molecules.

Analytical solvent-excluded and solvent-accessible surface areas per atom and residue are assigned as [attributes](#) named **areaSES** and **areaSAS**, respectively. These may include contributions from more than one [component](#). For example, one side of an atom could form part of an interior cavity while the other side could form part of the exterior surface. Calculating [relative exposure](#) (a normalized surface area) may be helpful for classifying amino acid residues as buried or exposed.

The command [surface](#) displays molecular surface, [surfrepr](#) sets which style is shown, and [surftransparency](#) adjusts surface transparency. Molecular surface display, style, and transparency can also be controlled with the [Actions menu](#), the [molecular surface attributes panel](#), the [Selection Inspector](#), and the command [setattr](#). Except for the [Actions menu](#), these also allow changes in probe radius, vertex density, mesh line width, and dot size. Parameters for subsequently generated molecular surfaces can be set in the [New Surfaces preferences](#).

In Chimera, molecular surfaces are created with embedded software from the [MSMS package](#), described in:

[Reduced surface: an efficient way to compute molecular surfaces](#). Sanner MF, Olson AJ, Spohner JC. *Biopolymers*. 1996 Mar;38(3):305-20.

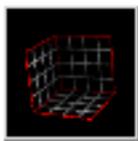
MSMS surface calculations may fail numerically, especially on large structures; see [surface calculation failures and workarounds](#).

Dot molecular surfaces in [MS/DMS format](#) can also be displayed in Chimera.

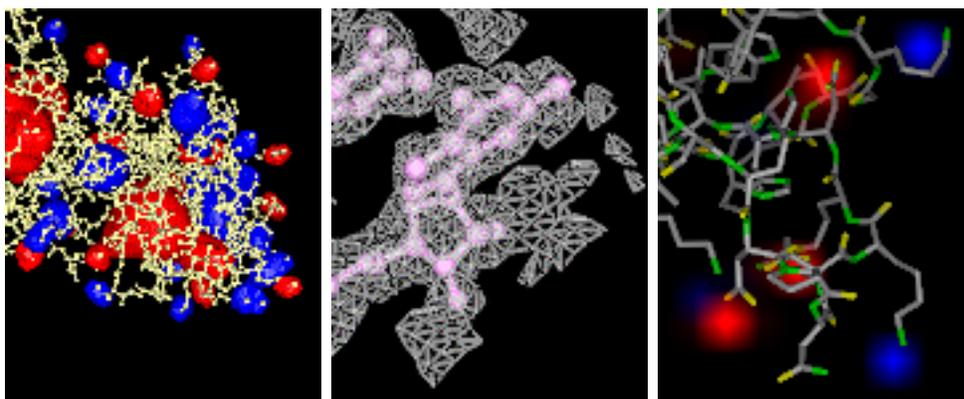
## VDW Surfaces

A van der Waals (VDW) surface differs from a molecular surface in that fine crevices are not smoothed. A VDW dot surface can be displayed with the command [vdw](#) and its dot density adjusted with [vdwdensity](#). The VDW dot size and dot density can also be adjusted in the [molecule model attributes panel](#). Note that the [sphere](#) representation also shows the VDW surface.

Chimera also displays [other surfaces](#) that are not necessarily molecular.


[Tools Index](#)
**Volume Viewer**
[Introduction](#)
[Startup and Input](#)
[Menus and Buttons](#)
[Features](#)
[Display](#)
[Surface/Mesh](#)
[Solid](#)
[Regions and Subsamples](#)
[Planes](#)
[Adding a File Type](#)
[Limitations](#)
[Technical Notes](#)

**Volume Viewer**

**Volume Viewer** is a tool for visualizing [volume data](#), 3D numerical data sets such as electron density maps. The data can be shown as solid or mesh isosurfaces (contour surfaces) or as partially transparent solids:



electrostatic  
potential  
(surface)

electron density map  
(mesh)

electrostatic  
potential  
(solid)

The state of **Volume Viewer** is included in saved [sessions](#). Many of its features are also implemented as the command [volume](#), and several related tools can be accessed from the **Volume Viewer** [Tools menu](#).

See also: [vop](#), [mask](#), [measure](#), [molmap](#), [Keyboard Shortcuts](#), [IMOD files](#), the [Density Display](#) image tutorial, and on the Chimera web site, the [Image Gallery](#) and [Guide to Volume Display](#)

**Volume Viewer** and some associated tools are described in:

[Visualizing density maps with UCSF Chimera](#). Goddard TD, Huang CC, Ferrin TE. *J Struct Biol*. 2007 Jan;157(1):281-7.

**Volume Viewer** is under development, and some [limitations](#) may be addressed in future versions.

## STARTUP AND INPUT

There are [several ways to start Volume Viewer](#), a tool in the **Volume Data** category. Starting **Volume Viewer** without a data file opens an "empty" interface. Choosing [File... Open map](#) from the interface will then bring up a [dialog](#) for opening any of the registered [volume data](#) types, including [electrostatic potential](#). Alternatively, simply [opening](#) a file of [volume data](#) (other than [electrostatic potential](#)) will automatically start **Volume Viewer**. Additional file types can be [defined](#). See also: [fetching](#) EDS files

Data are read when needed for display or a calculation, usually right when a file is opened. However, if only a [subsample](#) is shown initially, only those values are read, and if there is no display, only the file header is read. During loading of a large data set, the Chimera [status line](#) reports progress and allows the read to be canceled.

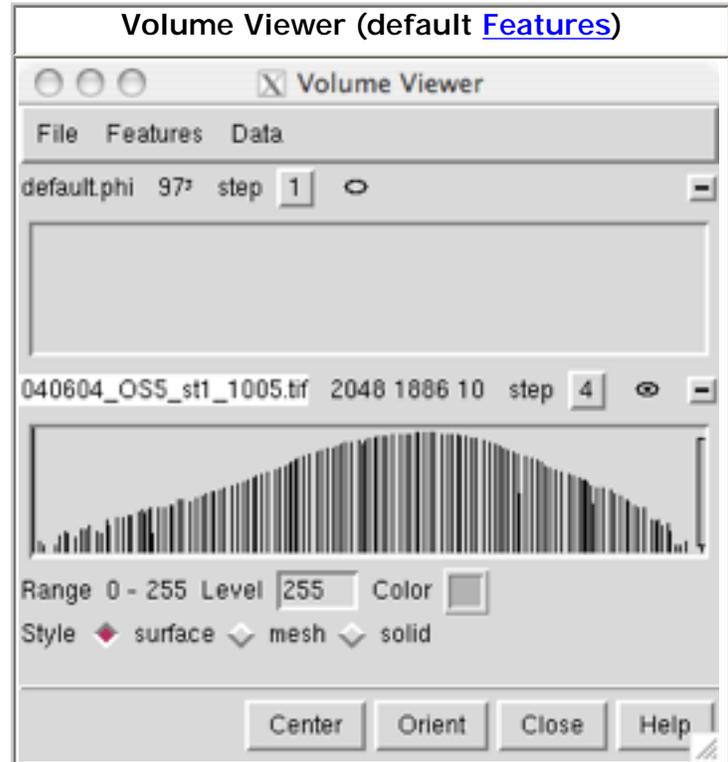
By default, [volume data](#) other than [electrostatic potential](#) will be displayed automatically if it does not exceed a certain size (see [Show data when opened...](#) in the [Data display options](#)). Otherwise, a data set can be displayed by clicking its

name, histogram area, or the eye icon to the right of the **step** menu. In the figure, no histogram bars are shown for **default.phi** because the data has not been read. Clicking to display the data will trigger reading it. A data set that is already displayed can be hidden by clicking its eye icon.

Data sets can be opened, closed (deleted), duplicated, and saved using the **Volume Viewer File menu**. Multiple data sets can be open at the same time and more than one [subregion](#) or [subsample](#) from the same or different sets can be displayed.

Settings in **Volume Viewer** refer to the currently active data set, or *current set*. The name of the current set is highlighted; clicking a data set's name, dimensions, or histogram makes it the current set, as does choosing its name from the [Data menu](#) or [Data set list](#). The *current display region* may correspond to part (a [subregion](#)) or all of the current set.

The value type of a data set can be 8-, 16-, or 32-bit signed integer 8-, 16-, or



32-bit unsigned integer, or 32- or 64-bit floating-point. The value type for the current set is shown in a popup balloon when the cursor is placed on the word **Range** below the histogram(s) in **Volume Viewer**.

A Priism or NetCDF file may contain more than one 3D array of values (more than one *component*), for example corresponding to the different wavelengths used to image a specimen. When multicomponent data is read, each component is treated as if it were a separate data set.

## VOLUME VIEWER MENUS AND BUTTONS

**Volume Viewer** includes many [features](#), and it is not practical to show them all at once. The basic dialog includes menus across the top ([File](#), [Features](#), [Data](#), and [Tools](#)) and [buttons](#) across the bottom.

The **File** menu includes operations associated with data files:

- **Open map...** brings up a [dialog](#) for opening [volume data](#) files.
- **Save map as...** brings up a [dialog](#) for saving the data in the [current display region](#) as a file in [MRC](#), NetCDF, Chimera map, or BRIX format. The file header will include information (as shown in the [Coordinates section](#)) for converting between grid indices and Cartesian coordinates. When [zoning](#) is in effect, data will be written out for a region enclosing the zone, with values outside the zone set to zero.
- **Duplicate** makes a copy of the [current set](#) so that opening the same file multiple times is not required for simultaneously displaying different [subregions](#) or different representations of the data.
- **Remove surface** deletes any **surface** or **mesh display** of the [current set](#).
- **Close map** removes the [current set](#) and all of its displays; this can also be done by clicking a data set's [minus-sign button](#) or closing the model with the same name as the [current set](#) using the [Model Panel](#).

The [Features menu](#) lists potential sections of the **Volume Viewer** dialog.

Choosing a data set from the **Data** menu makes it the [current set](#).

The **Tools** menu provides access to several related tools:

- [Fit in Map](#) - fit atoms into a map (volume data) or one map into another
- [Surface Color](#) - color surfaces by volume data values
- [Color Zone](#) - color surfaces to match atoms, split volume data by the resulting color zones
- [Morph Map](#) - morph between two related sets of volume data
- [Volume Tracer](#) - place markers within volume data and connect them with smoothed paths
- [Volume Filter](#) - smooth or transform volume data
- [Hide Dust](#) - hide small disconnected bits of a surface

- [Segment Map](#) - partition density maps
- [Fit to Segments](#) - fit structures into map segmentation regions
- [MultiFit](#) - fit multiple structures into density using a web service hosted by the [UCSF RBVI](#)
- [Volume Eraser](#) - interactively zero out parts of volume data
- [Measure Volume and Area](#) - measure surface areas and surface-enclosed volumes
- [Measure and Color Blobs](#) - color and measure disconnected parts of a contour surface
- [Volume Mean, SD, RMS](#) - calculate statistics for volume data
- [Values at Atom Positions](#) - map volume data to atom positions and assign the values as an atom [attribute](#)
- [Volume Series](#) - display a series of volume data
- **Volume Menu on Menubar** - replicate the **Tools... Volume Data** menu as a top-level **Volume** menu in Chimera and store the behavior in the [preferences file](#) (choosing this item again removes the top-level **Volume** menu)

The buttons across the bottom are always present:

- **Center** scales and centers the view and adjusts the clipping planes to contain the [current display region](#).
- **Orient** reorients all open models so that the axes of the [current set](#) are pointing in the standard directions: X horizontal in the plane of the screen increasing rightward, Y vertical in the plane of the screen increasing upward, and Z perpendicular to the screen increasing outward from the screen.
- **Close** dismisses **Volume Viewer**. If the panel is closed accidentally, it can be reopened in the same state by restarting **Volume Viewer**.
- **Help** opens this manual page in a browser window.

## FEATURES

The **Features** menu lists potential sections in the **Volume Viewer** dialog, with checkboxes controlling which sections are shown. Clicking the adjacent checkbox expands the **Volume Viewer** interface to show the corresponding section. A section can be closed by unchecking its entry in the **Features** menu, unchecking its [feature button](#), or clicking its close button () if present.

- [Atom box](#)
- [Brightness and Transparency](#)
- [Coordinates](#)
- [Data display options](#)
- [Data set list](#)
- [Display style](#)
- [Feature buttons](#)
- [Named regions](#)
- [Orthogonal planes](#)

- [Planes](#)
  - [Precomputed subsamples](#)
  - [Region bounds](#)
  - [Solid rendering options](#)
  - [Subregion selection](#)
  - [Surface and Mesh options](#)
  - [Threshold and Color](#)
  - [Zone](#)
- 
- **Show only default panels** restores the previously [saved](#) state of the **Features** menu (and thus which sections are present in the **Volume Viewer** dialog)
  - **Save default panels** saves the state of the **Features** menu (and thus which sections are present in the **Volume Viewer** dialog) in the Chimera [preferences file](#)
  - **Save default dialog settings** saves many current **Volume Viewer** settings other than the state of the **Features** menu in the [preferences file](#). Settings that are specific to a particular volume data set are not saved in this way, but can be saved along with an entire [Chimera session](#) in which **Volume Viewer** is being used.
  - **Use factory default settings** resets the **Features** menu and other **Volume Viewer** settings to the factory defaults without changing the [preferences file](#)

### ← Feature buttons

This section contains a row of small, unlabeled buttons representing the *other* possible sections of the **Volume Viewer** dialog. Checking a feature button has the same effect as checking a box in the [Features menu](#). Although arranged horizontally, the buttons are in the same order as the menu entries. Placing the cursor over a feature button shows its section name.

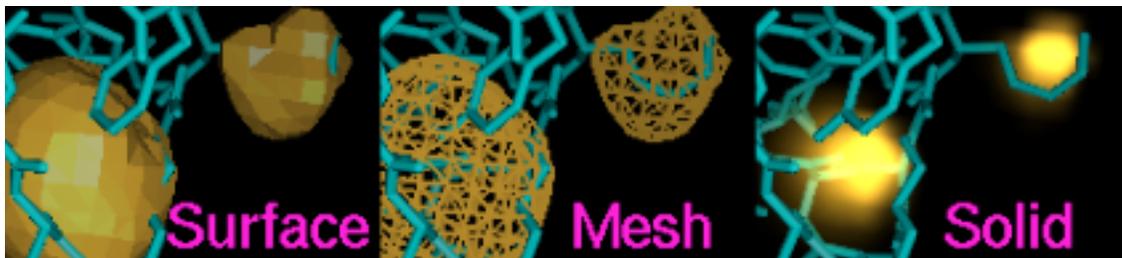
### ← Coordinates

Values in the **Coordinates** section are used to convert between the grid indices of the [current set](#) and Cartesian coordinates. They are included in the header when a data set is [saved](#) to a file. The values can be edited; changes take effect when the Enter (return) key is pressed. **Origin index** refers to the grid indices of the XYZ origin; **center** moves the XYZ origin to the center of the grid. Fractional and negative values are allowed, as the origin is not required to coincide with a grid point or even to fall within the grid. **Voxel size** refers to the data point spacing (resolution). Although only one number will be shown if the size is the same along the three axes, three different numbers separated by spaces can be entered. After any changes, the origin index and voxel size can be **reset** to their initial values. **Cell angles** are the angles in degrees between the crystallographic axes. The **Rotation axis** and **angle** describe a transformation. When a data set is created by [resampling](#) (see [Subregion selection](#)), these fields show its transformation relative to the original data set.

## ← Data set list

Clicking the name of a data set in the **Data Sets** list makes it the [current set](#).

## DISPLAY



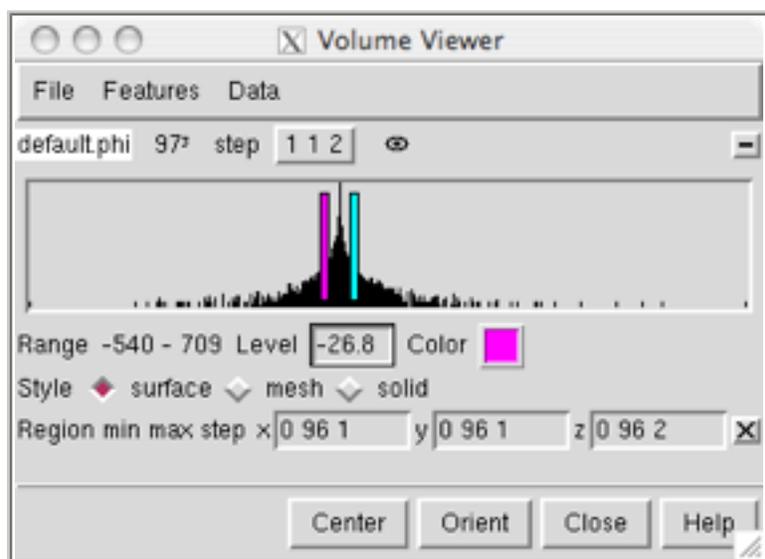
A data set can be displayed by clicking its name, histogram area, or the eye icon (small oval) to the right of the **step** menu. The oval has a dot in the middle when the data set is displayed. A data set that is already displayed can be hidden by clicking its eye icon.

## ← Display style

The **Style** or *rendering mode* to use for the [current set](#) can be **surface**, **mesh**, or **solid**. The **surface** and **mesh** modes depict isosurfaces (contour surfaces) consisting of a large collection of triangles. In the **surface** mode, each triangle is drawn as a plane of color. In the **mesh** mode, just the triangle edges are displayed. In the **solid** mode, the data is shown as a semitransparent solid.

## ← Threshold and Color

A histogram of values is shown for each data set (up to the [maximum number](#) specified in the [Data display options](#)). The name of the data set, dimensions in grid units of the display region, and **step** size(s) are shown above each histogram. A data set can be displayed/undisplayed by clicking its **eye icon** and closed (deleted) by clicking the **minus-sign button** on the far right.



The name of the [current set](#) is highlighted. Clicking the name, dimensions, or histogram of a data set makes it the [current set](#), as does choosing a different **step** setting or turning on its display.

The **step** setting controls sampling density; a value of **1** means all the data points are used to generate the display, while **2** means every other data point is taken along each axis, reducing the number of points used by a factor of eight. Step sizes can be different in the X, Y, and Z directions, in which case three numbers are shown. Different step sizes can be entered in the [Region bounds section](#), also shown in the figure. Changing a step value will change the data size limit for automatic step adjustment (see [Adjust step...](#) in the [Data display options](#)).

Histogram bar heights are on a logarithmic scale. The **Range** of values in the [current set](#) and its threshold controls are shown below all of the histograms. The *thresholds* control how data values are mapped to the [display](#):

- [surface or mesh](#)
- [solid](#)

## ← Data display options

**Show outline box using color** [[color well](#)] **linewidth** (**1** by default) shows the bounding box of the [current display region](#) using the specified color and linewidth. The color can be changed by clicking the [color well](#). Outline box display is **off** by default.

**Maximum number of histograms shown** (**3** by default) sets how many data set histograms can be shown in the [Threshold and Color section](#). The [current set](#) is always included, and specifying a data set for which the histogram is not shown as the current set will “bump” one of the others out of the section.

**Initial colors** [a series of [color wells](#)] are the colors to be used for successively opened sets of data. Clicking a [color well](#) allows the color to be changed.

**Update display automatically** (**on** by default) causes automatic display updates upon changes in [rendering mode](#), [Threshold and Color](#) settings, or rendering options ([Surface and Mesh options](#), [Solid rendering options](#)). Otherwise, it is necessary to click the [eye icon](#) to update the display after such changes. This option does not apply to display after:

- a file is first opened
- changes in **min**, **max**, or **step** values ([Region bounds](#))
- [subregion selection with the mouse](#) (but see [auto subregion display](#))

**Show data when opened if smaller than [size] Mvoxels** (**on** by default, with *size=256*) enables automatic initial display of data sets smaller than a specified size. (Regardless of this setting, however, [electrostatic potential](#) is not displayed initially.) The default limit of 256 Mvoxels is equivalent to a 512 x 512 x 1024 data set. *Size* does not have to be an integer; for example, 0.25 could be used. When the option is on, a newly opened data set that exceeds the size limit will not

be displayed until its [eye icon](#) is clicked. When the option is off, the icon must always be clicked to generate an initial display. Turning the option on or off or changing the *size* will not affect the display of any data that has already been opened. If a [precomputed subsample](#) is being read, the *size* is compared to that of the subsampled data.

**Show plane when data larger than [*size*] Mvoxels** (on by default, with *size*=256) specifies that a single plane normal to the Z axis should be displayed when a newly opened data set exceeds the specified size. (See also [Planes](#).)

**Adjust step to show at most [*size*] Mvoxels** (on by default, with *size*=1) allows limiting the display to a reasonable amount of data. Different values of *size* can be used for different data sets. The default limit of 1 Mvoxel ( $2^{20}$  or 1,048,576 voxels) is equivalent to a 128 x 128 x 64 data set. *Size* does not have to be an integer; for example, 0.25 could be used. When this option is on, [step](#) is automatically adjusted by powers of 2 to comply with the Mvoxel limit (increased when the extents are increased, if necessary, and decreased when the overall region size is small enough for finer sampling). Changing a step value in the [Threshold and Color section](#) or the [Region bounds section](#) will change the Mvoxel size limit to what is then shown.

**Data cache size (Mb)** (512 by default) controls whether volume data is kept in memory when it is not being displayed. A cache can improve performance, since accessing data in the cache is faster than reading it from disk. It is necessary to press Enter (return) after changing the cache size. Undisplayed data is purged from the cache to maintain the size, but if the displayed data alone requires more space, it will not be purged. The least recently displayed data is purged. The data cache only accounts for part of the memory used by **Volume Viewer**. Because a significant amount of additional memory is occupied by surfaces and color arrays used in rendering, the cache size should be set to about 1/3 or 1/2 of the desired total memory usage by **Volume Viewer**. Clicking **Current use** opens a dialog showing how much memory is occupied by volume data (not including the surfaces, *etc.* mentioned above).

**Zoom and center camera when region changes** (off by default) automatically readjusts the view when the bounds of the [current display region](#) change.

## ← Brightness and Transparency

**Brightness** scales the intensity of the color of the [volume display](#). Values range from 0.01 to 10, where **1** (the default) produces no change relative to the color defined in the [Threshold and Color section](#).

For **surface** and **mesh displays**: the **Transparency** is the fraction of light transmitted from behind the surface or a line of mesh. Values range from 0 to 1, where **0** (the default) corresponds to no transparency. (See also [Dim transparent surface/mesh](#) in the [Surface and Mesh options](#).)

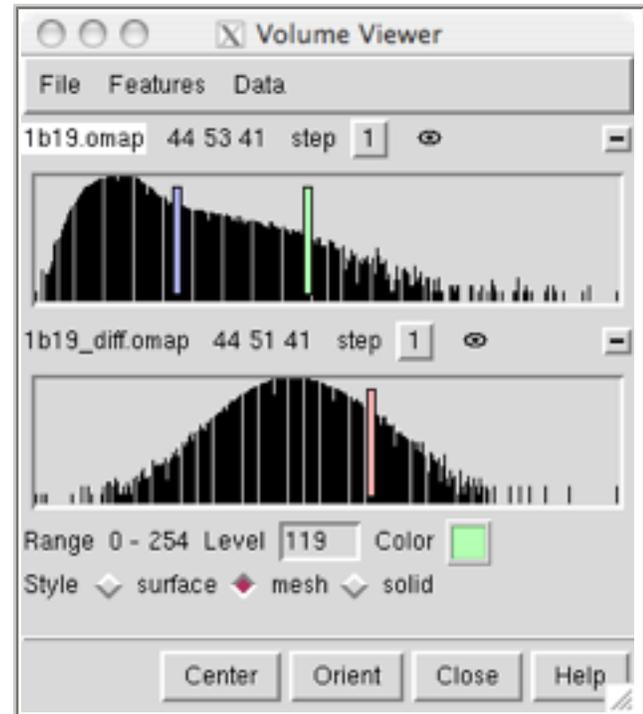
For **solid displays**: the **Transparency** setting modulates transparency values [from](#)

the [histogram](#) in a way that compensates for the thickness of the display ([details](#)). Values range from 0 to 1, default **0.5**. By default, more transparent voxels are made dimmer ([Dim transparent voxels](#) in the [Solid rendering options](#)).

## SURFACE OR MESH DISPLAY

For **surface** and **mesh displays**, each threshold is shown as a vertical bar. Initial thresholds are set automatically. For unsigned data types, an initial threshold is set so that 1% of voxels (1% of the volume encompassed by the data region) lie above it; for signed data types, positive and negative thresholds are placed symmetrically about zero.

In the figure, the histogram of the [current set](#) is on top. The **Range** of values in the [current set](#) and its threshold controls are shown below both of the histograms.



Any number of thresholds (contour levels) can be added by Ctrl-clicking with the left mouse button on the histogram. Ctrl-clicking on an existing threshold deletes it. Of course, different thresholds can have different settings for **Level** and **Color**; the settings shown apply to the threshold most recently moved or clicked, the *current threshold*. A threshold can be moved by changing the **Level** and then pressing Enter (return) or by dragging it horizontally with the left mouse button. Holding the Shift key down reduces the speed (mouse sensitivity) of threshold dragging tenfold, allowing finer control. Setting a contour level very low can generate a surface with many triangles that takes a long time to display.

Each threshold is shown in the same color as its display. The **Color** can be changed by clicking the [color well](#). [Brightness and Transparency](#) can be adjusted.

## ← Surface and Mesh options

**Surface smoothing iterations** [*i*] **factor** [*f*] (off by default, with  $i=2$  and  $f=0.3$ ) smooths **surface** and **mesh displays** by moving each vertex toward the average position of its neighboring vertices (those connected by triangle edges). A vertex is moved a fraction  $f$  (ranging from 0 to 1) of the way toward the average position of its neighbors, and each vertex is moved once per iteration.  $i$  iterations are performed, where  $i$  is a positive integer. When [Subdivide surface...](#) is on, it may be necessary to use more smoothing iterations and/or a higher factor to achieve a smoothness similar to that of the unsubdivided surface. See also: [sop smooth](#)

**Subdivide surface [*j*] times** (off by default, with  $j=1$ ) increases the number of triangles used in **surface** and **mesh displays**. Subdivision can help to produce smoother surfaces when combined with the **Surface smoothing...** option. During a single round of subdivision, each triangle is divided into four smaller triangles by connecting the midpoints of the triangle edges. Thus, the number of triangles is increased by a factor of  $4^j$ , where  $j$  is a positive integer. See also: [sop finerMesh](#)

**Smooth mesh lines** (off by default) turns on anti-aliasing to smooth lines in **mesh displays**. Mesh lines with **transparency** > 0 can only be smoothed when **Dim transparent surface/mesh** is on. OpenGL can render smooth lines using anti-aliasing, but the method has a side effect: dense meshes look brighter from some viewpoints and darker from others, depending on the order in which the lines were drawn.

**Square mesh** (on by default) displays a subset of the lines in the triangular mesh. Lines in the square mesh show the intersection of the XY, YZ, and XZ grid planes with the contour surface. Regardless of this setting, triangles will still be drawn in the planar surfaces at box faces shown with the **Cap high values...** option.

**Mesh line thickness [*k*] pixels** ( $k=1$  by default) is the pixel line width used in **mesh displays**; it must be a positive integer.

**Dim transparent surface/mesh** (on by default) decreases the brightness of **surface** and **mesh displays** as their **Transparency** is increased. Specular highlights are also dimmed. Otherwise, increasing the transparency of a color (without changing its brightness) allows more light to shine through, resulting in an increase in the overall brightness. When dimming is on, OpenGL (alpha,1-alpha) blending is used instead of (1,1-alpha) blending.

**Mesh lighting** (on by default) makes the inside of a contour volume shown in the **mesh rendering mode** dimmer than the outside. With mesh lighting on, the brightness varies according to the angle between each surface point normal and the line of sight; brightness is maximal when the outward-facing normal is parallel to the line of sight and pointing at the user (see more on the definition of "outward" under **Light flip side...**). With mesh lighting off, the brightness is uniform regardless of the angle between the normal and the line of sight.

**Two-sided surface lighting** (on by default) makes both the inside and outside of a contour volume shown in the **surface rendering mode** bright. Otherwise, only the outside is lit (see more on the definition of "outside" under **Light flip side...**). The brightness of each lit side varies according to the angle between a surface point normal and the line of sight; brightness is maximal when the normal is parallel to the line of sight.

**Light flip side for thresholds < 0** affects **surface** and **mesh displays**. The **surface** mode is affected only when lit from a single side (when **Two-sided surface lighting** is off). The **mesh** mode is affected only when **Mesh lighting** is on. When **Light flip side...** is off, the side toward smaller or more negative data

values is always treated as the outside of a contour volume. When **Light flip side...** is on, the side toward larger or more positive values is treated as the outside for any negative thresholds (contour levels), while the side toward smaller or more negative values is treated as the outside for any positive thresholds. Flip side lighting is appropriate when displaying data for which the sign is meaningful, such as electrostatic potential. In other cases, the data set is better described as a monotonic range of values, and where it lies relative to zero is not meaningful.

**Cap high values at box faces** makes isosurfaces (**surface** and **mesh displays**) cover faces of the volume data box where high values would be exposed. This option should be turned off when the objects of interest are represented by lower or more negative data values. The default setting depends on **volume data type**: **on** for most formats, **off** for formats that are commonly inverted (Purdue image format) or used for signed data ([electrostatic potential](#), molecular orbitals).

## SOLID DISPLAY

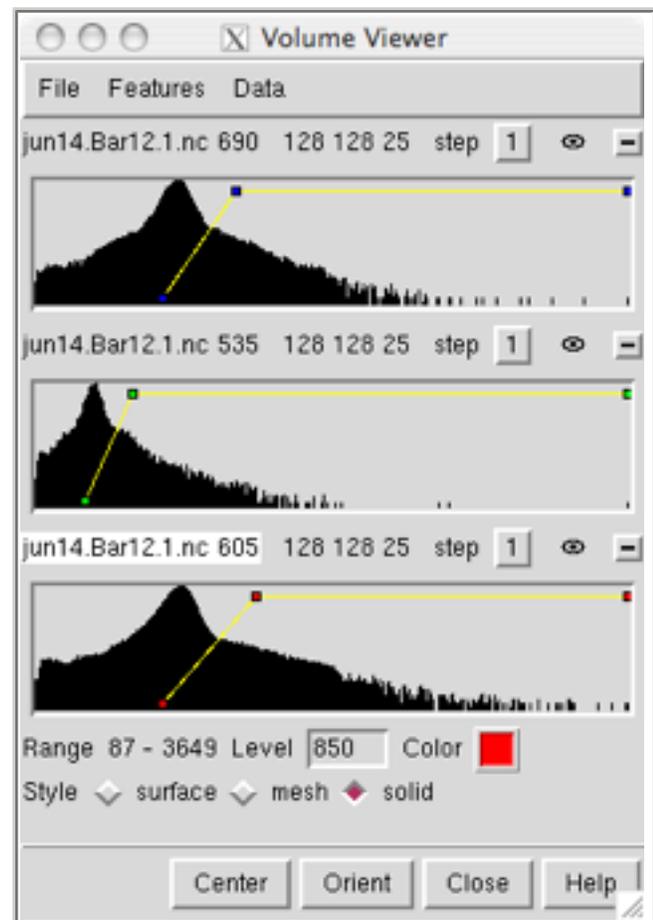
For **solid displays**, each threshold is shown as a small square. Initial thresholds are set automatically. For unsigned data types, initial thresholds are set to give zero intensity for the 10% of voxels (in the data region) with the lowest values, scaling up to 0.99 intensity for the 1% of voxels with the highest values; for signed data types, positive and negative thresholds are placed symmetrically about zero.

The figure shows three **components** of light microscope data. The **Range** of values in the **current set** and its threshold controls are shown below all of the histograms.

Any number of thresholds can be added by Ctrl-clicking with the left mouse button on the histogram. Ctrl-clicking on an existing threshold deletes it. Of course,

different thresholds can have different settings for **Level** and **Color**; the settings shown apply to the threshold most recently moved or clicked, the *current threshold*. A threshold can be moved by changing the **Level** and then pressing Enter (return) or by dragging it horizontally *and/or vertically* with the left mouse button. Holding the Shift key down reduces the speed (mouse sensitivity) of threshold dragging tenfold, allowing finer control. The rightmost threshold does not have to be at the far right of the histogram, and a threshold at a lower data value can be higher (vertically) than a threshold at a higher data value.

Each threshold is shown in the same color as its display. Different thresholds for



the same data set can be colored alike (as in the [figure](#)) or differently. The **Color** can be changed by clicking the [color well](#). **Brightness and Transparency** can be adjusted.

The thresholds and connecting lines on each histogram define a *transfer function* that maps data values to colors and intensities. For each voxel, the data value is compared to the thresholds on the histogram. The colors and intensities of the closest threshold at a lower data value (to the left) and the closest threshold at a higher data value (to the right) are linearly interpolated. Voxels with data values greater than the rightmost threshold or less than the leftmost threshold are colorless and completely transparent. Color is defined by red, green, blue and opacity components. The intensity at a threshold is further scaled by its vertical position, where 0 is the bottom of the histogram and 1 is the top. Rendering time does not depend on the positions of the thresholds, but increases with greater numbers of thresholds.

### ← Solid rendering options

**Color mode** refers to color pixel format (OpenGL texture format). Memory usage can be tailored, potentially to allow faster updates and display of larger maps.

- **auto** (default) - set the mode according to whether multiple colors and transparency are being used
  - **opaque** - ignore transparency and use the single- or multiple-color opaque mode as appropriate
  - **multi-color transparent** - RGBA
  - **multi-color opaque** - RGB
  - **single-color transparent** - luminance + alpha
  - **single-color opaque** - luminance
- 
- 4 bits
  - **8 bits** (default)
  - 12 bits
  - 16 bits

**Projection mode** allows memory-efficient display of large data sets such as tomograms. Displaying just the planes perpendicular to one data axis uses one-third the memory of the **x, y or z planes** option, which displays the planes along the data axis most perpendicular to the screen at a given data orientation. The **perpendicular to view** option uses planes that may not be aligned with any data axis (3D texture mapping). If this option is chosen but not supported by the computer hardware, an empty red outline box will be shown; if it is supported but the texture is too large, an empty yellow outline box will be shown.

- **auto** (default) - use **z planes** for volumes with X or Y dimensions at least 4 times greater than Z, otherwise **x, y or z planes**
- **x, y or z planes**
- **x planes**
- **y planes**
- **z planes**
- **perpendicular to view**

**Maximum intensity projection** (off by default) shows, at each pixel, the the most intense color value underlying the pixel along the line of sight. The maximum intensities of the red, green, and blue color components are determined separately. Transparency is not used; that is, the [Transparency](#) and the vertical positions of the thresholds are ignored when this setting is on. Maximum intensity projection can be useful for enhancing detail. Unphysical effects can result, but are usually not very noticeable; examples include the disappearance of a dim spot when it passes in front of a brighter spot and the simulation of a single spot when the maximal values of different color components under the same pixel actually come from different spots.

**Dim transparent voxels** (on by default) scales voxel [brightness](#) in **solid displays** by a factor of  $(1 - \text{transparency})$ . Otherwise, increasing the transparency also makes a volume appear brighter, because less light is blocked.

**Solid brightness correction** (off by default) Without this correction, the apparent [brightness and transparency](#) of **solid displays** (in [projection modes](#) other than **perpendicular to view**) depend on the viewing angle relative to the XYZ data axes. For a cube-shaped volume with equal resolution in the X, Y, and Z dimensions, the brightness drops and the transparency increases by a factor of  $3^{1/2}$  (approximately 1.7) as the viewing angle is changed from along any axis to along the cube diagonal. The brightness correction remedies this, but doubles rendering time.

**Minimize texture memory use** (off by default) causes **solid displays** (in [projection modes](#) other than **perpendicular to view**) to reuse a single 2D texture instead of allocating separate textures for every plane of the data volume. This is useful for viewing large data sets that would otherwise fail to display, but it can degrade performance (increase the lag time) when the display is rotated and during remote display.

**Solid linear interpolation** (on by default) refers to linear interpolation of the [brightness and transparency](#) between voxels in **solid displays**. Turning interpolation off may yield a pixelated appearance, but may decrease the rendering time, depending on the graphics hardware.

## REGIONS AND SUBSAMPLES

The default settings will often suffice for displaying the data without performance problems. If there are problems, however, the first issue to address is the size of the data set: whether the information can be handled and displayed in a reasonable amount of time. Typically, an isosurface for a 64 x 64 x 64 data set can be displayed in a few seconds on a late-1990's machine. The Chimera web site includes a [benchmark table](#) listing the volume data sizes that can be handled at 10 frames per second on various computer systems. With **solid displays**, performance is optimal when region dimensions are powers of two. A status line near the bottom of the panel reports on the progress of a calculation. Some experimentation may be required to balance appearance and performance for a given data set. The size of a 3D data set is essentially the total number of values,

X x Y x Z. It is related to both the physical dimensions (for example, Å along each axis) and the resolution (spacing) between data points.

To decrease size, one can focus on a smaller physical region (*subregion*) and/or use a sparser grid of points (*subsample*) from the original data set. [Step](#) values > 1 indicate subsampling. Subsamples can be [precomputed](#).

[File... Duplicate](#) makes a copy of the [current set](#) so that it is not necessary to open the same file multiple times to simultaneously display different [subregions](#) or representations of the same data.

### ← Region bounds

The physical extents of the displayed data can be decreased by increasing the **min** grid indices and/or decreasing the **max** grid indices. The **step** values control the level of sampling; for example, a step of 2 means that every other grid point is used to calculate the display. Because they refer to grid indices, the **min**, **max**, and **step** values must be integers. Pressing Enter (return) after entering new values or clicking the [eye icon](#) will update the display. Changing a step value will change the data size limit for automatic step adjustment ([Adjust step...](#) in the [Data display options](#)). The original full data set can be restored by changing the **min**, **max**, and **step** values back to their original settings or by clicking **Full** in the [Subregion selection section](#).

### ← Subregion selection

**Select subregions using [button] mouse button** allows [subregions](#) of the [current set](#) to be defined with the mouse. The indicated mouse button is reassigned from its [existing function](#) to volume selection in the graphics window.

In the following description of selection boxes, “clicking” and “dragging” refer to operations with whichever mouse button has been assigned to subregion selection. When there is no pre-existing selection box:

- Dragging sweeps out a box aligned with the volume data axes, the smallest possible box containing all voxels under the initial and final cursor positions. Depending on the viewing angle, however, the box may be surprisingly large. It may be helpful to [Orient](#) the volume beforehand.
- Clicking outside the [current display region](#) creates a selection box the same size as the [current display region](#).

After a selection box has been created, it can be erased or further adjusted. Clicking erases the selection box and starts over. Dragging from a point over the box moves the frontmost face; also holding down Shift moves the the rearmost face instead of the frontmost. Dragging from a point not over the box translates the whole box in X and Y; Shift-dragging translates the box in the Z dimension.

The **auto** display option (**off** by default) automatically updates the [current display region](#) after the subregion selection box is changed. Otherwise, it is necessary to click **Crop** to apply the new boundaries. Clicking **Full** restores the original extents of the data set.

**Rotate selection box** deactivates other models so that only the selection box is rotated by the mouse. After box reorientation, clicking **Resample** interpolates from the data to generate a new set of data bounded by the box, with points spaced according to the **Resampling voxel size**. Resampling (interpolation) is required to generate values along axes that differ from the original data axes; a voxel size no larger than that of the original set (see [Coordinates](#)) is recommended to limit loss of resolution.

After various [subregions](#) of the [current set](#) have been used as the [current display region](#) during a session, it is possible to go **Back** and **Forward** among them. The subregion farthest **Back** is the earliest created within the session. Up to 32 subregions can be stored and traversed. Changes in **min** and **max** values but not **step** sizes (in [Region bounds](#)) are retained in this history. Note that [resampling](#) creates a separate data set with its own history; the resampled data set is not included in the history of the set from which it was derived.

### ← Named regions

[Subregions](#) can be named and then restored by name. **Add** adds the name entered in the name field to the list of named regions, **Delete** deletes the region with the name shown in the name field from the list of named regions, and **Show** allows restoration (from a pulldown menu) of any of the named regions on the list. When **Add** is clicked, the **min** and **max** values (see [Region bounds](#)) of the [current display region](#) are saved. A named region can be restored by typing its name in the **Named region** field and pressing Enter (return).

### ← Zone

Clicking **Zone** limits the display to areas within a specified distance (**Radius**) of the currently [selected](#) atoms, if any (works for **surface** and **mesh** but not **solid displays**; see [limitations](#)). The [current display region](#) is actually larger than the zone, but areas outside the zone are masked. By default, a zone is based only on distances to any [selected](#) atoms (and [Volume Tracer](#) markers); the [bondzone](#) command indicates that points along any [selected](#) bonds (and [Volume Tracer](#) links) should also be used.

The **Radius** can be adjusted by moving the slider or by typing in a new value. Values larger than what is attainable by moving the slider all the way to the right can be typed. When zoning is in effect, recontouring affects only the block of data enclosing the zone (the [current display region](#)) rather than the entire [current set](#) of data. If the set of selected atoms is changed, **Zone** must be clicked again to move the zone accordingly. Otherwise, the zone will remain centered on the formerly selected atoms. Clicking **Zone** will not update the display when no atoms are

selected. Multiple isosurfaces of a given data set will be zoned simultaneously. However, the data set can be [duplicated](#) to allow differential zoning of different isosurfaces. **No Zone** turns off zoning and restores the original extents of the data. Zoning should be turned off during the use of [Subregion selection](#) buttons. (See also [Atom box](#).)

Clicking **Mask** creates a new data set of the same dimensions, but with the values outside the zone set to zero. See also: [vop zone](#)

### ← Atom box

Clicking the **Box around selected atoms...** button limits the [current display region](#) to a box with X,Y,Z dimensions enclosing the currently [selected](#) atoms, if any. The **padding** value increases the box size in every direction from the minimum needed to enclose the atoms. (See also [Zone](#).)

### ← Precomputed subsamples

Precomputed [subsamples](#) are especially useful when the data set is large; they facilitate switching between larger regions at low resolution and smaller regions at high resolution. Precomputed subsample files decrease memory usage and time spent reading data. After a data file is opened, its contents are not read into memory until the data is displayed (by clicking the [eye icon](#), or automatically as specified with [Show data when opened...](#) in the [Data display options](#)). If [step](#) sizes are set greater than 1 before the data is displayed, only a smaller precomputed subsample (if the appropriate one is available) will be read into memory. Subsamples of the original data can be created with the program [downsize.py](#). Besides the more efficient use of memory, another reason to use precomputed subsamplings instead of generating them on the fly is that [downsize.py](#) offers a variety of methods for determining the subsample values. The subsample files must be associated with the correct full data file when they are opened. This is done by opening the full data file, making sure it is the [current set](#), and then using the **Open...** button in the **Precomputed subsamples** section to open the subsample files. Along with the full data, the subsamples are listed in the pulldown **Precomputed subsamplings** menu according to the [step](#) sizes [inferred](#) to have been used to generate them from the full data file. When the [eye icon](#) is clicked, the [appropriate subsample](#) (based on **step** values) is used.

It is also possible to open and work with precomputed subsample files in the “normal” way (without associating them with the full data file). However, this does not allow for easy switching among different subsamples of the same full data set.

### ← Planes

The **Planes** section allows displaying a slab of the data, **Depth** planes thick and perpendicular to the specified **Axis**. The **Plane** value is the grid index of the first displayed plane along the axis. (See also [Show plane...](#) in the [Data display](#)

[options](#).)

Clicking **One** displays a single plane of data and switches the [display style](#) to **solid**. Slab thickness can be changed by editing the **Depth** value, and the slab can be repositioned along the axis by editing the **Plane** value, moving the slider, or [using the mouse](#). Only planes that are multiples of the [step](#) size will be shown, and **Depth** refers to the number of planes shown rather than a grid index range. Clicking **All** shows the full thickness of the data along the axis. **Preload** loads data for the current [step](#) size into memory beforehand to allow faster switching between the display of different planes.

**Move planes using [button] mouse button** allows repositioning the slab with the mouse. The indicated mouse button is reassigned from its [existing function](#) to dragging a slab along its axis in the graphics window. This mouse mode also works with [orthogonal planes](#).

Single planes of data can be shown as *contour lines* or spots. Using the **surface** or **mesh style** and turning off [Cap high values at box faces](#) (in [Surface and Mesh options](#)) shows contour lines of the specified [Mesh line thickness](#). Using the **surface style** with capping turned on shows filled spots.

Values in a single plane of data can be plotted as surface heights with the command [topography](#). See also: [vop tile](#)

## ← Orthogonal planes

The **Orthogonal planes** section facilitates viewing planes perpendicular to the **x**, **y**, and/or **z** data axes simultaneously. The **Box** option allows viewing the orthogonal planes as the six box faces of the [current display region](#). Choosing any of the **Orthoplanes** options turns on the [outline box](#), which includes lines where the planes intersect.

Each plane or box face can be moved along its axis with the mouse if the [Move planes...](#) option (in the [Planes](#) section) has been activated.

The planes are shown in the **solid style** using the **opaque color mode**. Opacity prevents confusion that could arise from seeing one plane through another. Viewing can also be enhanced by making the [background color](#) gray instead of black to increase contrast at data boundaries.

## ADDING A FILE TYPE

A new file format can be added using Python code. The code must read the desired file type into a NumPy Python array and specify how to map data indices into Chimera XYZ positions by giving an XYZ origin (position for index 0,0,0) and XYZ step size (spacing between grid points). In the chimera/share/VolumeData directory, look at griddata.py to see the base class for volume data; this file

contains useful comments. The base class defines the interface to the volume data used by **Volume Viewer**. For an example file reader, look at `xplor_format.py` and `xplor_grid.py` in the `xplor` directory. For [multiple-component data](#) (*i.e.*, data with more than one value at each grid point), look at `priism_format.py` and `priism_grid.py` in the `priism` directory. The `*_format.py` file is the file reader, and `*_grid.py` wraps the data in a grid object suitable for use by **Volume Viewer**. The last step is to add an entry to the list of known file types in `fileformats.py`.

## LIMITATIONS

**Zoning does not work for solid displays.** Restricting the display with [Zone](#) works in the **surface** and **mesh** modes, but not in the **solid** mode.

**Superimposing transparent models.** A superposition of two models with transparency will not be shown correctly. The models will be rendered separately, and the one rendered second will not take account of the opacity of the first model. A superposition of one transparent model with one or more opaque models will be displayed correctly, however.

**Data size limit for surface calculations.** Contour surfaces cannot be calculated for volume data sets with more than  $2^{32}$  voxels (4 Gvoxel) because 32-bit integer variables are used to index the volume data values. This limit applies even to 64-bit machines.

**Transfer function limitations.** The conversion of data values to intensities (which occurs in the **solid** mode) only allows linear interpolation.

**Transparent solid edge voxels.** **Solid** displays are missing a half-voxel on all sides. This is due to problems in OpenGL with linear interpolation near the boundary of a texture.

**Slow remote display.** Remote display is particularly slow with [transparent surface](#) and **mesh** displays, since the whole surface must be sent to the displaying machine every time the orientation is changed. The **solid** mode does not require every change in orientation to be transmitted, but remote display is still significantly slower than direct display.

## TECHNICAL NOTES

**Solid transparency details.** For [solid](#) displays, the [transfer function](#) maps a voxel's data value to a vertical position (ranging from 0 to 1) on the histogram:

$$T_{\text{histo}} = (1 - \text{vertical position})$$

If the [Transparency setting](#) is zero ( $T_{\text{set}} = 0$ ), the voxel will be completely opaque:

$$T_{\text{final}} = 0$$

Otherwise, the transparency value from the histogram is modulated by  $T_{\text{set}}$  in a way that depends on the number of data planes displayed:

$$T_{\text{final}} = T_{\text{histo}}^{(T_{\text{histo}} * T_{\text{set}} * \text{thickness})^{-1}}$$

$$\text{thickness} = \min(N_x, N_y, N_z)$$

where  $N_x$ ,  $N_y$ , and  $N_z$  are the number of X, Y, and Z data planes displayed, respectively. Increasing the [Transparency setting](#) and/or the thickness of the displayed data will make the exponent smaller and (because  $T_{\text{histo}} \leq 1$ ) individual voxels more transparent.

**Electrostatic potential file types.** DelPhi and GRASP \*.phi files are both accepted as input; the file reader automatically recognizes the phi file type and reads the data accordingly.

**Output MRC files.** MRC files written out with [File... Save map as](#) are in the MRC 2000 format. Older MRC-reading programs or programs expecting CCP4 format will read such files successfully but may not align the data correctly with other data sets.

**Byte order.** Different computers store the bytes of a single numeric value in different orders. PowerPC processors use one order (called big-endian), while Intel/AMD x86 family processors use the opposite order (called little-endian). If a binary data file is created on one machine and read on another having the opposite byte order, byte-swapping is required to get the correct numeric values. The DSN6 file reader assumes that the file was written in big-endian byte order. DSN6 files can be read on big- or little-endian machines. All of the other file readers detect file byte order and machine byte order, then perform byte-swapping if necessary.

**Rendering code.** **Surface** and **mesh** displays are of class [SurfaceModel](#), implemented in the Chimera `_surface` module; **solid** displays are of class `Volume_Model`, implemented in the Chimera `_volume` module.

**Memory usage varies by file type.** For the DSN6, Delphi, and XPLOR formats, the whole file is read in order to extract a [subregion](#); thus, specifying a subregion will speed up rendering and display but will not conserve memory. By contrast, only the specified subregions are read from MRC, NetCDF, and Priism files.

## Attribute Inspectors

Items in Chimera such as atoms, bonds, residues, and models have *attributes*: properties with [names](#) and values ([details...](#)).

The [Selection Inspector](#) provides access to attributes of items that are [selected](#).

The [Model Panel](#) lists models of various types. Once a model has been chosen in the left side of the [Model Panel](#), clicking the **attributes...** button in the right side brings up an attribute inspector:

- [Molecule](#)
- [MSMSModel](#)
- [SurfaceModel and Volume](#)
- [VRMLModel and Volume\\_Model](#)

Similarly, the [PseudoBond Panel](#) lists [pseudobond groups](#). Once a group has been chosen in the left side of the [PseudoBond Panel](#), clicking the **attributes...** button on the right side brings up an attribute inspector:

- [PseudoBondGroup](#)

Descriptions in the inspector dialogs are not necessarily identical to the [attribute names](#) used in commands such as [setattr](#), but for most attributes, placing the mouse cursor over the description shows the attribute name and other information in balloon help.

Attributes of [segmentation regions](#) can be inspected and new attributes created using the [Region Attributes dialog](#).

See also: [setattr](#), [defattr](#), [vdwdefine](#)

## Coloring

Chimera includes [many colors](#), and more can be created with the [Color Editor](#) or the command [colordef](#). There is a coloring [hierarchy](#): per-atom colors and residue-level ribbon colors override the model-level color.

- the [Actions... Color menu](#) can color atoms/bonds, ribbons, surfaces, and labels; choosing **all options...** opens the [Color Actions](#) dialog for more colors and finer control over which types of items are colored (background only, ribbons only, *etc.*)
- background color can also be set with the command [background](#) or the [Background preferences](#)
- the [color](#) command can color atoms/bonds, ribbons, surfaces, and labels
- the command [ribinsidecolor](#) specifies a separate color for the insides of protein helix ribbons
- the command [modelcolor](#) sets model-level color
- [volume data](#) display, including colors, can be controlled with [Volume Viewer](#) or the command [volume](#)

Many additional tools or commands assign colors based on data, including:

- [Color Secondary Structure](#) - set colors for helix, strand, and other segments of peptides/proteins
- [Render by Attribute](#) - show [attributes](#) such as B-factor, charge, conservation, and hydrophobicity with colors (see also the command [rangecolor](#)), radii, or worm thickness
- [Rainbow](#) (see also the command [rainbow](#)) - color progressively from one end of a chain to the other, or make each model a different color
- [Color Zone](#) - color surfaces to match nearby atoms or [markers](#)
- [Surface Color](#) (see also the command [scolor](#)) - color surfaces to show [volume data](#) values or distance from a point, axis, or plane
- [Coulombic Surface Coloring](#) (see also the command [coulombic](#)) - color molecular surfaces by Coulombic electrostatic potential

See also: [tips on preparing images](#), the [Palette Editor](#)

## Color Wells

Many dialogs contain *color wells*. Clicking on a color well opens the [Color Editor](#) and *activates* the well so that it reflects any color changes within the [Color Editor](#). The border of the color well turns white to signify activation. Clicking another color well activates the newly clicked well and deactivates any previously activated wells, whereas Shift-clicking a color well toggles the activation status of that well without changing the activation status of any other color wells.

The color defined in the [Color Editor](#) can be dragged and dropped into any color well (whether activated or not). A color can also be dragged from one color well and dropped in another in the same or a different Chimera dialog.



Examples of dialogs with color wells:

- the [Selection Inspector](#)
- the [Model Panel](#)
- [FindHBond](#)

## Surface Color Source

Atoms and models have surface color assignments that can differ from each other and from their own color assignments. The default behavior is for the visible surface color(s) to match the visible atom color (s), determined by the [coloring hierarchy](#) mentioned above. The level in the hierarchy used as the source for visible surface colors can be changed with:

- the command [surfcOLOR](#)
- the [molecular surface attributes panel](#)

## Clipping

Clipping planes cut away parts of structures, surfaces, and objects. Chimera includes [global](#) and [per-model](#) clipping planes. Clipping plane status, locations, and orientations are included in [saved positions](#) and [sessions](#). By default, clipped surfaces appear solid rather than hollow because planar caps are drawn where the surfaces are cut away. Cap appearance and whether caps are shown can be controlled using [Surface Capping](#).

### Global Clipping Planes

The *global* clipping planes affect all models and demarcate the front and back of the visible scene. The front (*hither*) and back (*yon*) global clipping planes are always perpendicular to the line of sight. They can be moved interactively in the [Side View](#) or adjusted with the commands [clip](#), [section](#), and [thickness](#).

Global clipping is **off** by default, meaning that the planes are automatically adjusted to about the bounding sphere of the displayed items. Turning on global clipping discontinues the automatic adjustment and allows items to become clipped.

Global clipping can be turned on by:

- checking the **Clip** option in the [Side View](#)
- using the command [clip on](#)

It is also turned on as a side effect of:

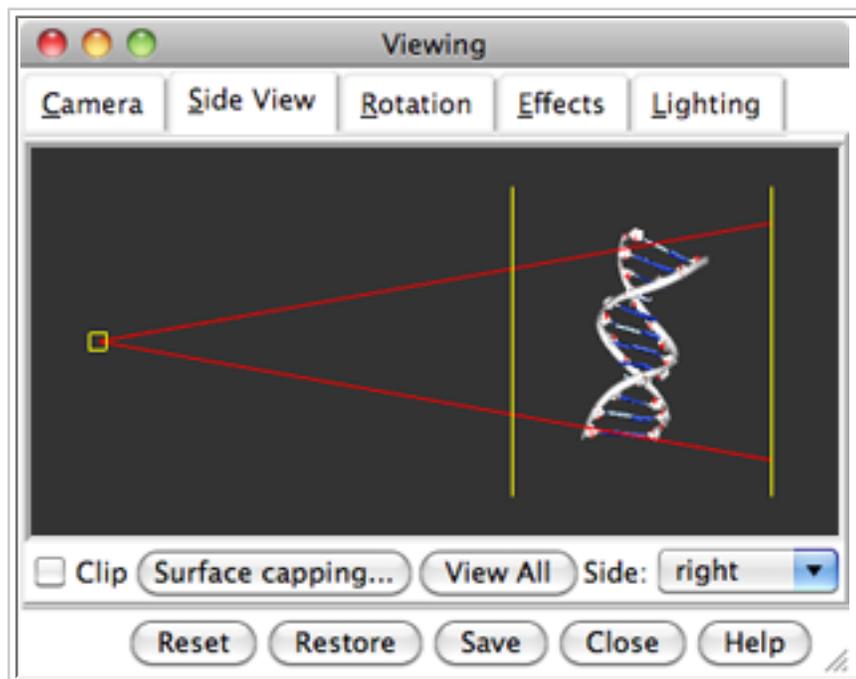
- moving the planes, for example, in the [Side View](#) or with commands ([clip](#), [section](#), [thickness](#))
- adjusting the view to include specific items, with [Actions... Focus](#) on a [selection](#) or the command [focus](#) or [window](#) on [specified](#) items

Global clipping can be turned off by:

- unchecking the **Clip** option in the [Side View](#)
- using the command [clip off](#)

It is also turned off as a side effect of adjusting the view to include all displayed items, by:

- clicking **View All** in the [Side View](#)



## Clipping

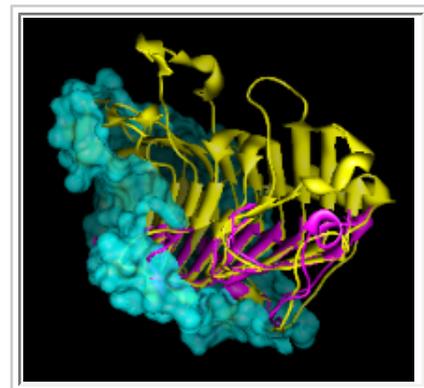
- using [focus](#) or [window](#) without arguments (or using [~focus](#))
- using [Actions... Focus](#) with nothing [selected](#)

In addition, [resetting](#) a [saved position](#) or [restoring a session](#) may turn clipping on or off.

[Depth cueing](#) is modulated by the positions of the global clipping planes.

## Per-Model Clipping Planes

A *per-model* clipping plane affects only a single model and can face in any direction. Since a [molecular surface](#) and its underlying structure are separate models, they can be clipped separately, as shown in the figure. Up to one plane per model can be activated and positioned with the [Per-Model Clipping](#) tool or the command [mclip](#). All parts of a model to one side of its clipping plane can be shown, or just a slab of specified thickness.



---

UCSF Computer Graphics Laboratory / September 2011

## Saving Images

Images can be rendered directly with Chimera or [raytraced](#) with POV-Ray using **File... Save Image** or the **copy** command. See also: [tips on preparing images](#), [making movies](#), [exporting a scene](#)

The top section of the **Save Image** dialog contains a file browser for specifying output location, **File name**, and **File type**. Formats include:

- **EPS** - Encapsulated PostScript
- **JPEG** - see [JPEG quality](#)
- **PS** - PostScript
- **PNG** (default) - [raytracing](#) always produces this format
- **PPM**
- **TIFF** - LZW-compressed
- **TIFF-fast** - uncompressed (larger files)

When the [Image type](#) is **stereo pair**, additional format options are stereo JPEG (\*.jps) and stereo PNG (\*.pns). Viewing such files as standard JPEG and PNG shows side-by-side images, but special viewers are available to show them as stereo. Free viewers include [StereoPhoto Maker](#) and [JPSViewer](#).

Several dialog settings such as the [supersampling level](#) and [print resolution](#) are saved in the [preferences file](#).

### Image Size:

- **Use print units** (default off) - whether image width and height are expressed in physical units of length rather than pixels
- **Units** (pixels by default, but centimeters/**inches**/millimeters/points if [using print units](#)) - units in which image width and height are specified; all except pixels are units of length (72 points = 1 inch)
- **Image width** - image width in [units](#) (default is the current pixel width of the [graphics window](#))
- **Image height** - image height in [units](#) (default is the current pixel height of the [graphics window](#))
- **Adjust field of view** (no/yes/**stereo cameras**) - when [using print units](#), whether to compute the image field of view from the specified image width instead of the graphics window width. This may provide a better sense of depth. The **stereo cameras** setting (default) indicates making the adjustment for stereo modes only (not when the [Image type](#) is **same as screen** and the graphics window [camera mode](#) is **mono**).
- **Maintain current aspect ratio** (default on) - whether to constrain the image width/height ratio to match the width/height ratio of the [graphics window](#). If so, changing the image width automatically adjusts the height and *vice versa*. If not, image width and height can be changed independently: the **Grow to Fit** button resizes the graphics window to match the image aspect ratio by increasing one window dimension (*i.e.*, width or height), while **Shrink to Fit** resizes the

graphics window to match the image aspect ratio by decreasing one window dimension. If the window has not been grown or shrunk to match the image aspect ratio, a saved image will include more than what is displayed.

- **Print resolution (dpi)** (default **100.0**) - number of pixels per inch in the final image; image dimensions in [print units](#) are converted to inches internally and multiplied by the dpi to give the output dimensions in pixels

### Image Options:

- **JPEG quality** (integer value in the range 5-95, default **90**) - quality setting for [JPEG output](#); higher values give larger files
- **Rendering:**
  - **Chimera** (default) - Chimera rendering, normally offscreen ([details...](#)). Images can be [supersampled](#), that is, initially generated at a higher resolution and then sampled down to the final size.
  - **POV-Ray** - [raytrace with POV-Ray](#) rather than rendering directly with Chimera. Raytracing can be very slow but includes fancier effects such as high-quality shadows. First, POV-Ray input files containing the scene (\*.pov) and raytracing options (\*.ini) are generated, and then the raytracing calculation is run as a background job that can be monitored or canceled using the [Task Panel](#). The **POV-Ray Options** button opens the [POV-Ray Options preferences](#). Raytraced images are saved in [PNG format](#).
- **Supersample** (1x1/2x2/3x3/4x4) - for [Chimera rendering](#), how many pixels to sample in the X and Y dimensions for each pixel in the final saved image. 1x1 corresponds to no supersampling. Higher values increase the smoothness of edges in saved images and increase calculation time with little effect on file size. 3x3 is generally recommended when supersampling is done. A potential disadvantage of supersampling is that lines such as [silhouettes](#) may become unexpectedly thin in the final image because there is a system-dependent limit on how wide they can be drawn in the initial image. The effective maximum linewidth is reported at the bottom of the dialog. It may be possible to achieve the desired linewidth by reducing the supersampling level and/or the pixel dimensions of the image.
- **Image type** - stereo/mono options for [Chimera rendering](#)
  - **same as screen** (default) - use the current [camera mode](#) of the graphics window for the image. If the mode is *sequential stereo*, two separate image files (left-eye and right-eye views) will be saved.
  - **lenticular** - regardless of the graphics window camera mode, save multiple separate images of views from slightly different angles for combination into a [lenticular image](#)
    - **Lenticular images** (default **12**) - how many images to save in the **lenticular** mode. The files will be named by appending a number to the specified file name, before any file type suffix.
  - **stereo pair** - regardless of the graphics window camera mode, save a *cross-eye stereo* pair as a single image twice as wide as the specified [size](#), optionally using one of the special [stereo formats](#). (To save a *wall-eye stereo* image, set the graphics window to that [camera mode](#) and use the **same as screen** image mode instead.)
- **Transparent background** - whether to include opacity values (alpha) in the output file ([PNG or TIFF](#) only), to facilitate combining the image with different backgrounds in image-editing applications. With background transparency, if the Chimera [background](#) is a single solid color, it

will be completely invisible in the saved images. (**Note:** TIFF images with background transparency may not be interpreted correctly by Adobe Photoshop.) Background transparency can also be enabled with the command [set bgTransparency](#) or the startup option `--bgopacity`. For background transparency in [raytraced](#) images, see the [POV-Ray Options preferences](#).

**Image Description** - text annotation to be placed in the saved image file (does not affect image appearance). Clicking **Image Credits** opens the [Image Credits preferences](#).

Clicking **Save** dismisses the dialog and initiates saving the image, whereas **Close** simply dismisses the dialog. **Citing Chimera** shows [how to credit Chimera](#), **Tips** shows the [tips on preparing images](#), and **Help** opens this manual page in a browser window.

Chimera performs offscreen rendering as permitted by the system. Offscreen rendering is not supported by certain older machines. On those systems, the image will be redrawn in the graphics window, piece by piece depending on the specified [image size](#) and degree of [supersampling](#); during this process, the graphics window should not be obscured by other windows or moved offscreen, even partially.

## Tips on Preparing Images

The following tips mainly apply to high-quality rendering directly in Chimera. Images rendered in Chimera are often clearer and more illustrative than those made with the (noninteractive) [raytracing](#) option, and can even include [shadows](#).

The [tutorials](#) include step-by-step examples of preparing images in Chimera. Many [display styles](#) and [colors](#) are available.

**Presets** are predefined combinations of display settings. A preset can be applied by choosing it from the [Presets menu](#) or by using the [preset](#) command. Further changes can be made after a preset has been applied. Types of presets:

**interactive** - for interactive viewing in Chimera

**publication** - for making publication/presentation images

**custom** - defined by the user in a script, see the [Presets preferences](#)

Publication presets make the background white, increase [smoothness](#), and adjust display styles, without changing which items are displayed or their colors. See also: [alias](#)

**Background color** is set to white by the [publication presets](#) mentioned above, but can be set to [any color](#) with the [Color Actions](#) dialog, the [background](#) command, or the [Background preferences](#). The latter two can also set the background to a gradient of multiple colors or to an image read from a file. If system hardware permits, images can be saved with a [transparent background](#).

**Silhouettes** are outlines that emphasize borders and discontinuities. Although shown in the interactive display, these are mainly intended for output images, as [supersampling](#) makes them look much smoother in the image than on the screen. Whether silhouettes are shown and their linewidth and color can be controlled with the [Effects tool](#) or the [set](#) command. [Publication presets](#) #1 and #2 turn on silhouettes and turn off [depth cueing](#). This silhouettes setting is global (applies to all models), but when it is on, the silhouettes for an individual model can be toggled with the command [setattr](#), the [Selection Inspector](#), or

the [molecule model attributes panel](#).

**Depth cueing** is progressive shading from front to back, also known as fog. It can be controlled with the [Effects tool](#) or the [set](#) command. The depth cueing color tracks the background color by default, but it can be changed independently with the [Color Actions](#) dialog, the [Effects tool](#), or the [set](#) command. [Publication presets](#) #3 and #4 turn on depth cueing and turn off [silhouettes](#).

**Lighting.** Chimera lighting modes may include ambient (nondirectional) light and up to three directional lights:

- ambient** - ambient-only, giving an unshaded, flat appearance
- single** - single directional light + ambient
- two-point** - two directional lights + ambient
- three-point** - three directional lights + ambient

A simple, line-drawing-like appearance can be achieved by combining the ambient-only mode with [publication preset](#) #1 or #2 (white background, [silhouettes](#)). Lighting mode, brightness, contrast, light directions, and [shininess](#) can be controlled with the [Lighting tool](#) and [lighting](#) command.

**Smoothness** can be increased by increasing the pixel dimensions of an image (its resolution). Additionally, independent of resolution:

- **Atoms, bonds, ribbons.** [Display styles](#) other than wire (which is not recommended for publication images) can be smoothed by raising the **subdivision** level with the [Effects tool](#) or the [set](#) command. This setting also affects curved **geometric objects** defined in [BUILD format](#). The [publication presets](#) increase **subdivision** to 5.0 if the current value is lower.
- **Surfaces.** The smoothness of a [molecular surface](#) can be increased by increasing its **vertex density** with the command [setattr](#), the [Selection Inspector](#), or the [molecular surface attributes panel](#) (see also the [New Surfaces preferences](#)). This parameter can also be specified when a surface is displayed with the [surface](#) command. The [publication presets](#) raise **vertex density** to 10.0 for any existing molecular surfaces with lower values. Separate smoothing options are available in [Multiscale Models](#) and [Volume Viewer](#) for the surfaces they create (see also the command [sop smooth](#)).
- **Supersampling.** The smoothness of edges in saved images can be increased by supersampling, where Chimera automatically saves an image at higher resolution (larger pixel dimensions) than requested and then samples it back down to the requested final size. The default level of supersampling, 3x3, is usually recommended for publication images and gives much smoother edges than no supersampling (1x1). For individual images, the supersampling level is specified in the [Save Image dialog](#) or with the [copy](#) command's **supersample** keyword. The command [movie record](#) and dialogs for [recording movies](#) also include supersampling options. Supersampling increases calculation time with little effect on file size. A potential disadvantage of supersampling is that lines such as [silhouettes](#) or hydrogen-bond representations may become unexpectedly thin in the final image because there is a system-dependent limit to how wide they can be drawn in the initial image. The **File... Save Image** dialog reports the effective maximum linewidth given the current supersampling level and image size. It may be possible to achieve the desired linewidth by reducing the supersampling level and/or the pixel dimensions of the image.

**Ribbon path.** By default, the ribbon path is a smoothed spline that may deviate from the true positions of

the backbone atoms, making bonds to sidechains appear unnaturally short or long ([details...](#)). The path calculation can be adjusted with the command [ribspline](#), the [molecule model attributes panel](#), and the [Selection Inspector](#). For proteins, a cardinal spline without smoothing is constrained to pass through  $\alpha$ -carbon coordinates, but gives a very rumped ribbon; a cardinal spline with strand-only smoothing may be a useful compromise (e.g., [ribspline cardinal smooth strand](#)).

**Shadows.** Shadows can make images more dramatic and enhance the sense of depth. However, they can also make images more complex and darken areas of interest; surfaces may benefit more from shadows than would “busy” images with ribbons and sticks. If system hardware permits, *interactive* shadows can be enabled with the [Effects tool](#) or the command [set shadows](#). Shadow locations and darkness are controlled by the [lighting](#) directionality and contrast, respectively. Interactive shadows are often good enough for presentation images, and their appearance can be improved by increasing the shadow **quality** setting in the [Effects tool](#) (albeit at the cost of increased graphics memory usage). They can also be used to preview shadow locations for noninteractive rendering. High-quality, *noninteractive* shadowed images can be produced by [raytracing with POV-Ray](#). Raytracing can be quite slow, among other [limitations](#). Noninteractive shadowed images can also be generated with the commands [conic](#) and [neon](#).

**Clipping planes** cut away portions of structures, surfaces and objects. The [global clipping planes](#) shown in the [Side View](#) affect all models and can only be perpendicular to the line of sight. In addition, each model can have a [per-model clipping plane](#) oriented at any angle. [Surface Capping](#) controls whether clipped surfaces appear solid or hollow.

**Transparency.** By default, only the topmost layer of all transparent items is shown. This is recommended because it simplifies the display and effectively de-emphasizes those parts. Whether a single transparent layer or multiple layers is shown can be controlled with the [Effects tool](#) and the command [set singleLayer](#). Transparency is normally angle-dependent, such that transparent triangles (forming objects as well as surfaces) appear more opaque when viewed edge-on than when viewed face-on. However, the angle dependence can be turned off with the [Effects tool](#) or the command [set flatTransparency](#).

- **Surfaces** ([surface models](#)) can be made transparent without otherwise changing their colors using the [Actions... Surface menu](#) or the command [transparency](#).
- **Atoms, bonds, and ribbons** can be made transparent without otherwise changing their colors using the command [transparency](#), or they can be assigned transparent colors directly. These more general approaches also work for **surfaces**. A transparent color can be defined in the [Color Editor](#) and applied using the [Actions... Color menu](#) or various [coloring](#) commands. Transparent colors can also be specified in commands by name (previously assigned with [colordef](#)) or as four comma-separated values (red, green, blue, and opacity components, each in the range 0-1).

**Labels and Arrows.** 2D text, [symbols](#), and arrows of multiple colors and sizes can be added to the display with the [2D Labels](#) tool (or command [2dlabels](#)). Such 2D annotations are drawn in front of any displayed objects and do not move when the objects are moved. By contrast, the standard Chimera “3D” labels (shown with the [Actions... Label](#) menu or commands [label](#) and [rlabel](#)) are associated with atoms and move along with them. The font and size of all 3D labels collectively and whether they should always be drawn in front are set in the [Labels preferences](#).

**Geometric Objects.** Spacefilling objects including 3D arrows can be created arbitrarily with the command

[shape](#) or in [BILD format](#). Axes (cylinders), planes (discs), and centroids (spheres) can be defined from sets of atoms using [Axes/Planes/Centroids](#) or the command [define](#), and ellipsoids using [measure inertia](#). The [PipesAndPlanks](#) tool shows protein helices and strands as cylindrical “pipes” and rectangular-box “planks,” respectively, with thin connectors. Axis, plane, and centroid objects as well as those from [shape](#) and [measure inertia](#) are [surface models](#), which have the following advantages over the VRML models from [BILD format](#) and [PipesAndPlanks](#):

- they better handle [transparency](#) and/or [interactive shadows](#)
- they can be recolored with the [Actions menu](#), [color](#), or [scolor](#)

Markers (spheres) can be placed and paths between them drawn with [Volume Tracer](#). Markers are implemented as atoms and thus also respond to the [Actions menu](#) and various commands. See also: [Cage Builder](#), [Thermal Ellipsoids](#), [hk cage](#), [Nucleotides](#), [Scale Bar](#)

**Color Keys.** A color key shows how a coloring scheme relates to quantities. Such coloring schemes are applied by various tools and commands, including [Render by Attribute](#), [Surface Color](#), and [rangecolor](#). Color keys suitable for publication images can be created with the [Color Key](#) tool or [colorkey](#) command.

**Stereo.** Wall-eye, cross-eye, [red-cyan](#), and [green-magenta](#) stereo images can be saved by changing the graphics window to the corresponding [camera mode](#) with the [Camera](#) tool (or the command [stereo](#)) and using the **same as screen** [Image type](#) in the [Save Image dialog](#). Another way to save cross-eye stereo images is with the **stereo pair** [Image type](#); in that case, it does not matter what [camera mode](#) is being used in the graphics window, but the resulting image will be twice as wide as the specified [size](#).

**Color space.** Some publications require images to be in the CMYK color space. Chimera currently saves images in only the RGB color space, so a separate application such as Adobe Photoshop® must be used to switch between the two.

**Choosing colors.** Several factors should be considered in choosing colors, including what the colors are meant to indicate, their distinguishability from each other and from the background, and whether viewers may have color vision deficiencies. Useful web sites include:

- [ColorBrewer](#) - multipurpose color-scheme picker; 3- to 12-color schemes are displayed on a map and rated by suitability for red-green colorblind viewers, printing, and/or photocopying. The HEX color specifications at this site are [Tk codes](#), which can be used as color names in many Chimera commands and in the [Color Editor](#).
- [Vischeck](#) - simulates the effects of color-blindness on your own uploaded image or a specified web page
- [IDEA vision demo](#) - simulates several types of color-blindness using images already at the site

See also: [Color Editor](#), [Palette Editor](#)

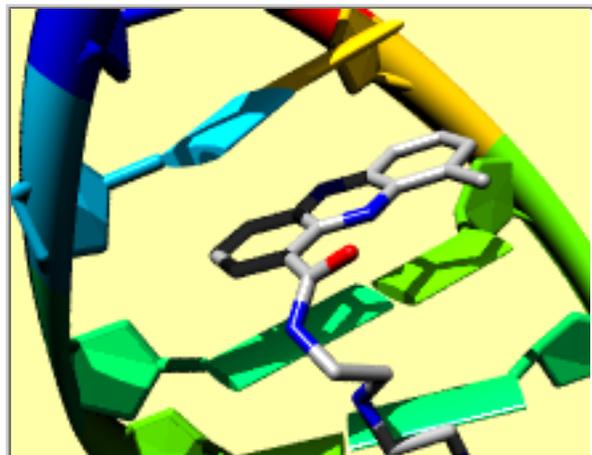
## Raytracing with POV-Ray

The [Persistence of Vision Raytracer](#) (POV-Ray) is included with Chimera. POV-Ray images can be saved:

- with [Save Image](#)
- with the [copy](#) command
- when [making movies](#)

Only the PNG image format can be saved (see [limitations](#)). For each raytraced image, Chimera first generates two POV-Ray input files:

- scene description (\*.pov file)
- raytracing options (\*.ini file)



The raytracing calculation is then run as a background job that can be monitored or canceled using the [Task Panel](#). Several raytracing parameters can be set in the [POV-Ray Options preferences](#).

See also: [exporting a scene](#), [tips on preparing images](#)

### Lights and Shadows

Chimera includes ambient light and up to three directional lights: key, fill, and back ([details...](#)). Lighting parameters can be adjusted with the [Lighting tool](#) or [lighting](#) command and are propagated to POV-Ray for raytracing. By default, only the key light produces shiny highlights and shadows. Shadows from raytracing can be made less severe by decreasing the **contrast**. Also, it may be useful to move the lights (change their directions) and/or adjust the **key-fill ratio**.

Interactive shadows can be used to preview shadow locations. If system hardware permits, these can be enabled with the [Effects tool](#) or the command [set shadows](#). In some cases, interactive shadows may suffice as an alternative to raytracing; their appearance can be improved by increasing the shadow **quality** setting in the [Effects tool](#) (albeit at the cost of increased graphics memory usage).

### Balancing Time Requirements and Results

Sometimes a considerable speedup in raytracing can be obtained with little change in output appearance by adjusting settings in the [POV-Ray Options preferences](#). This is particularly true for [movies](#), which are compressed to some extent during encoding. Changes to consider include:

- decreasing the **quality** to 4 or 5 [[quality details](#) at the POV-Ray site]
- increasing the **antialias threshold** to 1.5 or even higher (maximum 3.0) [[antialias details](#) at the POV-Ray site]
- if objects are not intentionally sliced, making sure that [global clipping](#) is turned off; to adjust

depth cueing, use the **start** and **end** parameters in [Effects](#) rather than moving the global clipping planes, which would turn clipping on

There is no single answer as to which values will optimally balance time requirements and results, as it will depend on the contents of the scene, the available computational power, and the patience of the user. Running a few tests is recommended before committing to raytracing a large set of images with a particular set of parameters.

Raytracing jobs can be monitored and canceled using the [Task Panel](#).

## Raytracing Limitations

Raytracing can be quite slow; see [balancing time requirements and results](#).

Using transparency or [clipping](#) may increase rendering time significantly. If objects are not actually clipped, clipping should be turned off, as mentioned [above](#).

The `max_trace_level` parameter is set to 10 to improve multi-level transparency [[details](#) at the POV-Ray site], but this also increases time and memory requirements relative to lower levels.

Only the PNG format can be saved.

If image dimensions are specified in [units of length](#) rather than pixels, they are multiplied by the [resolution](#) to give the correct number of output pixels, but the dimensions in units of length are “forgotten” (not included in the image file).

A raytraced image from POV-Ray may differ from the view in Chimera in several ways. Some differences are desirable, such as the presence of shadows. Others reflect current limitations of the POV-Ray format and/or its implementation in Chimera:

- colors may differ slightly
- shading along [ribbons](#) may be stairstepped; using the **supersmooth** style instead of **rounded** reduces this effect
- [light source](#) directions but not colors are translated from Chimera
- although object [shininess and brightness](#) levels (but not specular color) are translated from Chimera, apparent shininess may differ
- if [depth cueing](#) is used, the background of the raytraced image:
  - will not be transparent even if the [transparent background](#) option is turned on
  - will match the [depth-cueing](#) color even if the background is a different color in Chimera
- the following are omitted or otherwise not handled: [silhouettes](#), [selection highlighting](#), [stereo](#), [Volume Viewer](#) transparent solid displays, and normals on dot and mesh surfaces

## Making Movies

Chimera includes the following approaches for capturing image frames and assembling them into a movie file:

- [Movie Recorder](#) is a graphical interface with buttons to start recording, stop recording, and encode the saved images as a movie file.
- The command [movie](#) provides access to the same features, but with keywords and arguments. This allows integration of movie recording and [content](#) within a single [script](#).
- [MD Movie](#) provides an interface for [recording trajectories](#) such as from molecular dynamics or [Morph Conformations](#)
- [Morph Map](#) creates an interpolated progression between two sets of [volume data](#) and provides an interface for [recording](#) the result
- [Animation](#) (under development) allows saving Chimera [scenes](#), arranging them into a [timeline](#), and [recording](#) the resulting animation
- Additional features step through different displays over time and can be used together with [Movie Recorder](#) or the [movie](#) command:
  - [Volume Series](#) (or command [vseries](#)) displays an ordered sequence of [volume data](#) sets
  - the [planes](#) feature in [Volume Viewer](#) (or [volume](#) command with the [planes](#) keyword) displays sequential planes or slabs of data
  - volume morphing can be scripted with [vop morph](#)
  - molecular trajectory playback can be scripted with [coordset](#)

The image frames can be rendered directly by Chimera or created by [raytracing with POV-Ray](#). See below for information on developing [movie content](#) and troubleshooting [problems in PowerPoint](#). See also: [tips on preparing images](#)

### ← Movie Content

During recording, structures can be manipulated interactively and [commands](#) entered. However, for reproducibility and smoother results, often a Chimera [command file](#) (script) will be used instead. The commands listed below are especially useful for scripting continuous motion and/or specifying movie content. See also: [movie script examples](#), [video mini-examples](#) for individual commands

[2dlabels](#) - display 2D labels, which can be faded in or out over a specified number of frames

[background](#) - set background color(s) or image

[clip](#) - move [global clipping planes](#)

[cofr](#) - change the center of rotation

[coordset](#) - play through frames of a trajectory

[fly](#) - smoothly traverse a series of positions previously saved with [savepos](#)

[freeze](#) - stop all motion

[mclip](#) - control [per-model clipping](#)

[move](#) - translate models

[movie](#) - capture image frames and assemble them into a movie file

[perframe](#) - specify commands to be executed at each display frame  
[play](#) - script various complex motions  
[reset](#) - interpolate from the current position to one previously saved with [savepos](#)  
[rock](#) - rotate models back and forth (oscillate)  
[roll](#) - rotate models  
[rotation](#) - rotate bonds (adjust torsions)  
[savepos](#) - save and name a “position” (model transformations, scale, *etc.*)  
[scale](#) - scale the view  
[scene](#) - save and restore “scenes” (positions plus styles, colors, labels, *etc.*)  
[section](#) - move the clipping planes in parallel  
[select](#) - [activate/deactivate](#) models for motion  
[set](#) - set visual effects, make models rotate about individual centers, echo command-file lines as they are executed  
[thickness](#) - move the clipping planes in opposite directions  
[transparency](#) - make atoms/bonds, ribbons, and surfaces transparent  
[turn](#) - rotate models  
[vop morph](#) - morph (interpolate) between two or more volume data sets  
[vseries](#) - display an ordered sequence of volume data sets  
[wait](#) - suspend command processing for a specified number of frames or until motion has stopped  
[windowsize](#) - set the dimensions of the graphics window

In addition, the commands [sleep](#) and [pause](#) are useful for introducing pauses during live script execution, but not in movies, as no new frames are drawn during such pauses.

The graphics window should be adjusted to the desired dimensions before recording, for example with the command [windowsize](#). Using a small and standard size (320 x 240, 640 x 480, or 800 x 600 pixels) may solve some [playback problems](#). On certain platforms, other windows should not overlap the Chimera graphics window during recording ([details...](#)).

## ← Frame Rate and Movie Speed

In Chimera, the target or intended graphics frame rate is 60 fps (default). The actual frame rate may be lower depending on computer performance and what is being displayed. Recording generally slows the process further; however, the frame rate of the resulting movie is controlled independently with an encoding parameter (default 25 fps). Thus, movie playback may be faster or slower than the original process in Chimera.

The command [set maxFrameRate](#) can be used to set the Chimera target rate to match the movie frame rate, so that “live” execution of a script will show the movie contents at the same speed as the resulting movie (or at least, as similar a speed as possible given performance limitations). The [Animation](#) tool sets the target rate to 25 fps temporarily during timeline playback. The [accelerator](#) `rt` (command [ac rt](#)) allows monitoring the actual frame rate.

## ← Movie Command File Examples

Simply [opening](#) a [command file](#) in Chimera will execute its contents. Command script execution can be

aborted by pressing the **Esc** (escape) key, or paused/resumed (with no new frames being drawn during the pause) by pressing **Shift-Esc**. Opening any of the following example files in Chimera will show the content but not create a movie, as the movie-recording commands have been commented out with **##**.

- [1gfl.com](#) - fairly simple; fetches green fluorescent protein **1gfl** from the [Protein Data Bank](#), rotates it, changes display styles
- [tumble.com](#) - also simple; shows a small molecule rotating and casting shadows on a planar object (see [Interactive shadows](#) in the Chimera Feature Highlights)
- Several entries in the [Chimera Animation Gallery](#) include associated command files and (if needed) other data, for example:
  - [Ball-and-socket motion](#) - morphing
  - [Myosin thick filament analysis](#) - EM data, fitting
  - [Binding footprint](#)

The online movie-making tutorials ([2009](#), [2012\\*](#)) and the [Chimera Animation Gallery](#) contain additional examples. See also: [video mini-examples](#) for individual commands

\*contains videos in H.264 format, which Firefox does not play

## ← Problems in PowerPoint

On **Windows**, some Chimera-generated movies cannot be inserted into PowerPoint presentations, or can be inserted but will not play.

Using a small and standard frame size (320 x 240, 640 x 480, or 800 x 600 pixels) may solve some playback problems. Before recording, the Chimera window can be set to specific dimensions using the command [window size](#).

In PowerPoint 2000, only the Chimera **MPEG-1** format works, but the quality is very low. Using PowerPoint 2003 or newer is recommended. With PowerPoint 2003 on Windows, the Chimera **AVI MSMPEG-4v2** format is recommended. Although a movie in this format can be inserted into a presentation, it may play as a black rectangle. To remedy this, try reducing the video acceleration in Windows Media Player (instructions are for Media Player 10.0, but should be similar for other versions):

1. start Windows Media Player (for example, by right-clicking a movie icon and choosing **Play with Media Player**)
2. right-click the top bar of the Windows Media Player window and choose menu entry **Tools / Options...**
3. click the **Performance** tab in the **Options** dialog
4. click the **Advanced...** button in the **Video Acceleration** panel
5. turn off **Use video mixing renderer**

After PowerPoint 2003 has been restarted, the movie should play correctly within the presentation. We have not tested movie playing in PowerPoint 2007. A more drastic alternative to the last two steps above is to move the slider in the **Video Acceleration** panel in the **Performance** tab to **None**.

Video acceleration in Media Player is not the only potential source of trouble. If the problem persists, see

the following information from Microsoft:

- [Troubleshooting movies in PowerPoint](#)
- [PowerPoint 2007 movie checklist](#)

---

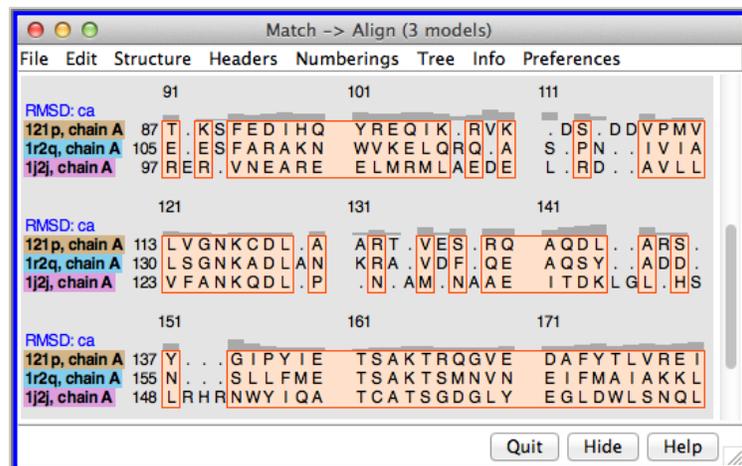
UCSF Computer Graphics Laboratory / June 2014


[Tools Index](#)
[Multalign Viewer](#)
[Introduction](#)
[Startup and Input](#)
[Sequence Window](#)
[Sequence Names](#)
[Numbering](#)
[Headers](#)
[Regions](#)
[Trees](#)
[Editing and Saving](#)
[Percent Identity](#)
[Annotations](#)
[Sequence-Structure Association](#)
[Basic Crosstalk](#)
[Secondary Structure](#)
[Residue Attributes](#)
[Superposition](#)
[RMSD](#)
[Interface to Modeller](#)
[Preferences](#)
[Appearance](#)
[Structure](#)
[Headers](#)
[Regions](#)
[Menu Listing](#)
[File](#)
[Edit](#)
[Structure](#)
[Headers](#)
[Numberings](#)
[Tree](#)
[Info](#)
[Preferences](#)


## Multalign Viewer (Sequence [Sequence](#))

Multalign Viewer shows amino acid and nucleotide sequences:

- multiple sequence alignments
  - read from files ([several formats](#))
  - created by other tools in Chimera ([MatchMaker](#), [Match -> Align](#), [Align Chain Sequences](#))
  - pseudo-multiple alignments from [Blast Protein](#)
- individual sequences
  - of structures in Chimera (when started as **Sequence**)
  - read from files ([FASTA](#), [NBRF/PIR](#) formats)
  - [fetched](#) from [UniProt](#)



The sequences are automatically [associated](#) with structures in Chimera. Associated structures are not required, but association enables useful sequence-structure [crosstalk](#) such as bidirectional selection. Measures of sequence conservation and spatial variability can be shown as [headers](#) above the sequences, and sequence feature [annotations](#) from [UniProt](#) and the [Conserved Domain Database \(CDD\)](#) can be shown as colored boxes. An [interface to Modeller](#) allows comparative (homology) modeling using a template structure, as well as building missing segments without a template. **Multalign Viewer** status and alignment data, but not [preference settings](#), are included in saved [sessions](#).

See also: [Chimera tutorials](#) (in particular, [Sequences and Structures](#), [Superpositions and Alignments](#), [Comparative Modeling](#), and on the Chimera web site, [Mapping Sequence Conservation](#)) and the following reference:

[Tools for integrated sequence-structure analysis with UCSF Chimera](#). Meng EC, Pettersen EF, Couch GS, Huang CC, Ferrin TE. *BMC Bioinformatics*. 2006 Jul 12;7:339.

### STARTUP AND INPUT

There are [several ways to start Multalign Viewer](#), a tool in the **Sequence** category. Explicitly starting **Multalign Viewer** brings up a [dialog](#) for opening a [sequence alignment](#). In fact, simply [opening](#) a file in any of the registered [sequence alignment formats](#) (without starting **Multalign Viewer** first) will automatically use it to show the alignment. Several alignments can be open at the same time, each in a separate [sequence window](#).

Individual sequences of structures in Chimera can be shown:

- by [starting](#) the **Sequence** tool (**Sequence** category)
- by using the command [sequence](#)
- by choosing one or more molecule models in the left side of the [Model Panel](#) and clicking the **sequence...** button in the right side of the panel

If there is more than one bonded multiresidue chain in the model(s), the chains will be listed in a dialog for designating which sequences to **Show**. The sequence of each chosen chain will be shown in a separate [window](#), including any residues not in the structure itself but in SEQRES records in the input structure file. Residues with standard names are assigned the corresponding standard one-letter codes, while certain nonstandard residues are assigned the same one-letter codes as closely related standard residues:

- ASH (protonated ASP) = D
- CYX (disulfide-bonded CYS) = C
- GLH (protonated GLU) = E
- HID/HIE/HIP (different protonation states of HIS) = H
- HYP (hydroxyproline) = P
- MSE (selenomethionine) = M
- +N (methylated nucleic acid) = N, where N = A/C/G/T/U
- additional relationships described in MODRES records in PDB input files

In addition, the two unusual genetically encoded amino acids are equivalenced as follows for alignment scoring with amino acid similarity matrices:

- U (selenocysteine) is scored as C
- O (pyrrolysine) is scored as X

## THE SEQUENCE WINDOW

The figure shows part of the sequence window contents for the input file [apoex.fa](#). In addition to the sequences and their [names](#), [numbering](#) and [header lines](#) may be displayed. Font size, sequence wrapping behavior, and the coloring of residue one-letter codes can be controlled in the [Appearance preferences](#). Colored boxes enclosing one or more residues are called [alignment regions](#); double-clicking a region opens the [Region Browser](#). The [Edit menu](#) allows [searching](#) for a string or pattern of residues in the sequences, or by full or partial sequence name.

When the mouse focus is in the sequence window, the **Page Down** key (or **space**) moves the view down to start with the block below the topmost block whose beginning is currently visible;

**Page Up** (or **Shift-space**) moves the view up to start with the block above the topmost block whose beginning is currently visible. Vertical alignment scrolling can also be done with the mouse wheel.

	121	131	141
Consensus	A R L g A D M E D v	r n R L v Q Y R s E	v q a M L G Q S t E
Conservation			
APE_BOVIN	A R L G S D M E D L	R N R L A Q Y R S E	V Q A M L G Q S T E
APE_PIG	A R V G A D M E D V	R N R L V L Y R S E	V H N M L G Q T T E
APE_MOUSE	A R L G A D M E D L	R N R L G Q Y R N E	V H T M L G Q S T E
APE_RAT	A R L G A D M E D L	R N R L G Q Y R N E	V N T M L G Q S T E
APE_PAPAN	A R L G A D M E D V	R S R L V Q Y R S E	V Q A M L G Q S T E
APE_MACFA	A R L G A D M E D V	R S R L V Q Y R S E	V Q A M L G Q S T E
<b>APE_HUMAN</b>	A R L G A D M E D V	<b>G</b> R L V Q Y R G E	V Q A M L G Q S T E
APE_RABIT	A . L E A D M E D V	C N R L A Q Y R G E	A Q A M L G Q S T E
APE_CAVPO	A R L G A D M E E V	R N R L S Q Y R S E	V Q A M L G Q S S E
C60940	A R L R A D M E D V	R N R L T Q Y R G E	L Q A M L G Q S S E
Y13652	G K L Q T D M T D A	K E R S T Q Y L Q E	L K T M M E Q N A D

	151	161	171
Consensus	E L R a R l a s H l	R K l r K R L l R D	a d D L Q K R l A V
Conservation			
APE_BOVIN	E L R A R M A S H L	R K L P K R L L R D	A D D L K K R L A V
APE_PIG	E L R S R L A S H L	R N V R K R L V R D	T E D L Q K R L A V
APE_MOUSE	E I R A R L S T H L	R K M R K R L M R D	A E D L Q K R L A V
APE_RAT	E L R S R L S T H L	R K M R K R L M R D	A D D L Q K R L A V
APE_PAPAN	E L R A R L A S H L	R K L R K R L L R D	A D D L Q K R L A V
APE_MACFA	E L R A R L A S H L	R K L R K R L L R D	A D D L Q K R L A V
<b>APE_HUMAN</b>	E L R V R L A S H L	<b>R K L R K R L L R D</b>	A D D L Q K R L A V
APE_RABIT	E L A R A F S S H L	R K L R K R L L R D	A E D L Q K R M A V
APE_CAVPO	E L R A R L T S H P	R K M K R R L Q R D	I D E L Q K R M A V
C60940	E L R A R F A S H M	R K L R K R V L R D	A E D L Q R R L A V
Y13652	D V K N R V G T Y T	R K L K K R L N K D	T E E I R N T V A T

Mousing over sequence names and residues shows information near the bottom of the window. Mousing over a sequence name reports any [associated structure\(s\)](#), the vertical position in the alignment (sequence is Nth of M sequences), and sequence length. Mousing over a residue symbol reports the corresponding residue in associated structure(s), if any, otherwise

the sequence position. Reporting the position in non-structure-associated sequences is turned off for larger alignments (>250,000 characters). The system “copy” shortcut (for example, command-C on Mac) will copy the contents of the [active region](#), if any, otherwise the entire sequence(s), into the text buffer.

The **Hide** button closes the sequence window without changing the state of **Multalign Viewer**. The sequence window can be reinvoked using the **Raise** option for the **Multalign Viewer instance in the Tools menu** (abbreviated **MAV**). This is also useful when the window has become obscured by other windows. **Help** opens this manual page in a browser window, and **Quit** exits from **Multalign Viewer**.

## SEQUENCE NAMES

Initial sequence names are taken from the input file or based on the corresponding structure chain. A sequence name can be changed by choosing [Edit... Edit Sequence Name](#) from the **Multalign Viewer** menu, and in the resulting dialog, indicating the desired sequence and entering a new name. Double-clicking a name in the [sequence window](#) brings up the same dialog, already set to that sequence. [Edit... Find Sequence Name](#) allows searching by full or partial sequence name. In the [sequence window](#), ellipses (...) are used to shorten names longer than the maximum (default 30 characters) specified in the [Appearance](#) section of the [Multalign Viewer preferences](#).

## NUMBERING

Numbering can be displayed over the alignment and to the left and/or right of the sequences. Numbering displays can be controlled with the [Numberings menu](#).

*Alignment numbering* (for the alignment as a whole) includes gap positions and always starts at 1. Whether it should be shown initially can be set in the the [Headers preferences](#). When present, alignment numbering is shown above any [headers](#).

*Sequence numbering* (for individual sequences) includes only non-gap positions and can start with numbers other than 1. Sequence start numbers are set to 1 when an alignment is opened, except:

- when a sequence is generated from a structure chain in Chimera (for example, with [Match -> Align](#)), its start number is set to the number of the first residue in the structure chain
- when all sequence names in the input file are of the form *string/N-M*, the trailing */N-M* is stripped to generate the [names](#) shown in the [sequence window](#), and the sequence start numbers are set to the respective values of *N*

Sequence start numbers can be changed arbitrarily with [Numberings... Adjust Sequence Numberings](#). Even with a correct start number, however, a sequence's numbering will be incorrect following any missing internal residues (for example, in a sequence derived from a structure missing a loop). When an alignment is [saved](#), */N-M* can be appended to each sequence name, where *N* is the sequence's start number and *M* is calculated from *N* and the total number of residues in the sequence.

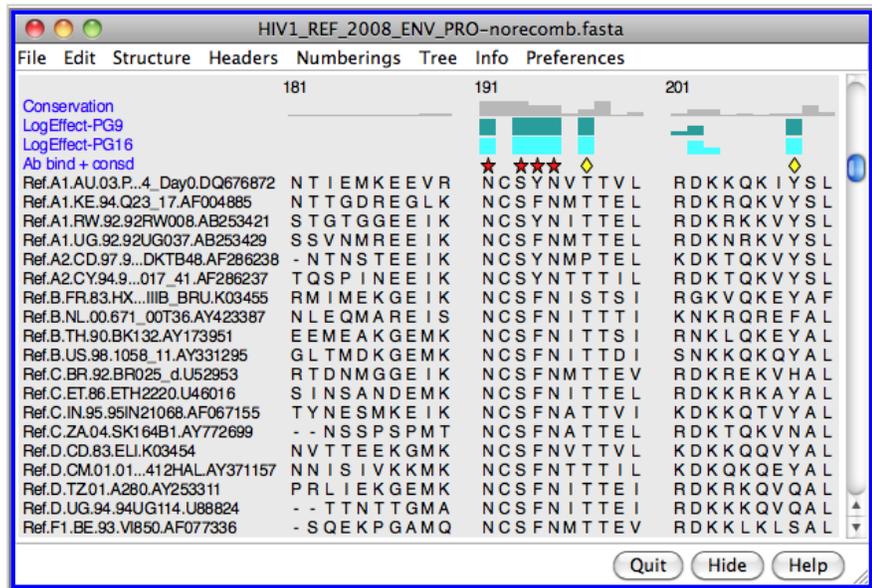
## ALIGNMENT HEADERS

Headers are lines of information above the sequences in the [sequence window](#). They can be hidden/shown with the checkboxes in the **Multalign Viewer Headers menu**. Some headers are available automatically, and custom headers can be [defined](#) using a [simple format](#). The

[Headers preferences](#) control how **Consensus** and **Conservation** header values are calculated.

A header may contain numerical quantities (shown as a histogram), characters, or symbols. If the values in a numeric header fall within the range of 0 to 1, they will be used directly as histogram bar heights. If the range extends below 0 or above 1, the values will be converted into histogram bar heights ranging from 0 to 1 according to:

$$\text{for values } \geq 0, \quad 1 - \frac{1}{2} e^{-\text{value}}$$

$$\text{for values } < 0, \quad \frac{1}{2}(e^{\text{value}})$$


However, the original values (not those used for histogram display) are assigned as [residue attributes](#) and written out when header contents are [saved](#).

Headers may be *dynamic*, with values that update automatically when independent variables such as the sequence alignment or structure [associations](#) are changed, or *static*, with values that stay the same.

The following dynamic headers are available by default:

- **Consensus** - [consensus sequence](#) as defined in the [Headers preferences](#)
- **Conservation** - [conservation](#) as specified in the [Headers preferences](#)
- **Charge variation** - range of residue formal charges, assuming -1 for D/E, +1 for H/K/R, and 0 for all other types
- **RMSD** (root-mean-square distance) per-column spatial variation among the [associated](#) structures, only calculated for columns with at least two associated structure residues:
  - **RMSD: ca** - using [one point per residue](#), CA in amino acids, usually C4' in nucleic acids
  - **RMSD: backbone** - based on peptide backbone atoms (N, CA, C, O) in amino acids, minimal backbone (O5', C5', C4', C3', O3', P) in nucleic acids
  - **RMSD: full** - for residues of the same name, based on all distances between atoms of the same name (without correcting for symmetries between equivalent atoms with different names, *e.g.*, OD1 and OD2 in aspartic acid); if not all structure residues associated with the column have the same name, falls back to the **RMSD: backbone** calculation

See also: [superposition assessment](#), [residue attributes](#)

Arbitrary, user-defined headers can also be shown:

- Static headers can be added by reading in a [header definition file](#) with [Headers... Load](#) in the **Multalign Viewer** menu. Loading values for a header with the same name as an existing header will create an additional header line, not replace the values in the existing header line. GC (Generic per-Column) [annotations](#) in alignments in [Stockholm format](#) are also treated as static headers.
- Dynamic headers can be defined with Python code. For examples, see the file `share/MAVHeader/ChimeraExtension.py` within the Chimera installation (it defines the **Charge variation** and **RMSD** headers described [above](#)). Rather than changing this file in the `MAVHeader` module, which might be overwritten whenever a new version of Chimera is installed, users should create an analogous file named `ChimeraExtension.py` in a different directory. The [Tools preferences](#) can be used to add that directory's location (the directory above it) to the places Chimera should look for extensions.

The following apply to headers other than [alignment numbering](#):

- Headers can be reordered (a header line is placed below the others when turned on with the [Headers menu](#)).
- Header line contents can be saved to a file ([Headers... Save](#) in the **Multalign Viewer** menu).
- Clicking within a header [selects](#) any structure residues [associated](#) with that column of the alignment.
- Values of headers are available as [residue attributes](#) named `mavHeaderName` in any [associated](#) structures. Numerical attributes will appear in the attribute lists of [Render/Select by Attribute](#); character attributes will be listed only in the [Select by Attribute](#) portion. It may be necessary to [Refresh](#) the attribute menus or values in these tools to update them with any changes in **Multalign Viewer** header information.

## ALIGNMENT REGIONS

Alignment *regions* are colored boxes or outlines that enclose one or more residue symbols in the [sequence window](#). A single region can contain any number of disjoint and/or abutting rectangular blocks. Pausing the cursor over a region (but not directly over a residue symbol) shows the region name in a pop-up balloon (see the [Regions preferences](#)), and double-clicking a region opens the [Region Browser](#).

Several actions in **Multalign Viewer** and other tools generate regions, and they can also be created manually. [UniProt](#) sequence feature annotations are loaded automatically as regions for sequences [fetched](#) from UniProt, and annotations from UniProt and/or the [Conserved Domain Database](#) can be added to other sequences (those already open in Chimera) using [Info... UniProt/CDD Annotations](#).

The initial display of manually created regions and those created by [selection](#) or [structure association](#) can be set in the [Regions preferences](#). The display and colors of existing regions can be controlled with the [Region Browser](#). Be aware that [reordering the sequences](#) will delete existing multisequence regions.

A region can be created manually by dragging with the left mouse button within the [sequence window](#). Dragging downward into the following block highlights to the end of the preceding block. **Shift**-dragging with the left mouse button *adds to the active region*. **Ctrl**-dragging *creates* a new region and makes it the [active region](#).

The *active region* is the region most recently created manually, clicked on in the [sequence window](#), or designated as **Active** in the [Region Browser](#). The corresponding parts of any [associated](#) structures are highlighted in the main Chimera window by becoming [selected](#). Only one region can be active at a time. In the [sequence window](#), the active region is indicated with a dashed outline; clicking the active region deactivates it, and clicking a different region deactivates the former active region and makes the new region active. A region with no interior color is only responsive to clicks on its borders. Where regions [overlap](#), only the highest is responsive to clicks.

The **Region Browser** can be opened by double-clicking a region in the [sequence window](#) or by choosing [Info... Region Browser](#) from the **Multalign Viewer** menu.

Region Browser for apoex.fa

Regions applicable to: APE\_HUMAN

A	S	Color Well	Name	Start	End
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	mismatches (1) of 1le4 (#0) chain A	131	131
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	missing structure (2 gaps) of 1le4 (#0) chain A	1	321
chain:					
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Apolipoprotein E id=PRO_0000001987	19	321
glycosylation site:					
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	N-linked (Glc) (glycation) evidence=14	94	94
glycosylation sites:					
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	O-linked (GalNAc...) evidence=1	26	318
glycosylation sites:					
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	O-linked (GalNAc...) evidence=15	213	309
modified residue:					
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Methionine sulfoxide status=by similarity	144	144
region of interest:					
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	8 X 22 AA approximate tandem repeats	81	256
region of interest:					
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Heparin-binding	163	237
region of interest:					
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	LDL receptor binding status=potential	159	169

Raise Lower Copy Rename Delete Info

Data source  
Show these kinds of regions...  
 built-in  UniProt

activates

Choosing in table:  shows  hides others except missing structure, Chimera selection

Include gaps: false

Close Help

Regions applicable to the **entire alignment** or to any of the individual sequences in the alignment can be listed using the pulldown menu near the top of the dialog. Regions applicable to the entire alignment include the **Chimera selection** and any hand-drawn regions, even if they only enclose parts of a single sequence. Regions applicable to an individual sequence include [UniProt and CDD annotations](#) and regions created by [structure association](#). If [UniProt and/or CDD annotations](#) have been added, checkboxes under **Data source** allow including/excluding specific sets of regions from the listing.

**Region Browser** columns:

- **Active** - whether the region is the [active region](#)
- **Shown** - whether the region is shown in the [sequence window](#) (however, it could still be obscured by [overlapping](#) regions)
- region interior color (a [color well](#))
- region border color (a [color well](#))
- **Name** - region name
- **RMSD** - region [overall RMSD](#), if applicable (entire-alignment regions only)
- **Start** - position in [alignment numbering](#) of the first residue in the region (individual-sequence regions only)
- **End** - position in [alignment numbering](#) of the last residue in the region (individual-sequence regions only)

Clicking a row in the **Region Browser** highlights the line and *chooses* the region. More than one region can be chosen at a time, and checkbox settings specify whether **Choosing in table**

shows all chosen regions, hides other regions, and/or makes the last chosen region the [active region](#).

If **Include gaps** is false, the [chosen region\(s\)](#) will be drawn to enclose only residues, not gap positions. Although a given region may then appear as disjoint blocks, it will still be a single region.

Since regions may overlap, a region can be considered higher or lower than another. When the mouse focus is in the [sequence window](#), the **up arrow** and **down arrow** keys can be used to raise and lower the [active region](#), respectively, and pressing the **Delete** key will delete the [active region](#). Alternatively, **Region Browser** buttons can be used to act on the [chosen region\(s\)](#):

- **Raise** - place region on top (in front) of any overlapping regions in the [sequence window](#) and move it to the top of the list in the **Region Browser**
- **Lower** - place region below (behind) any overlapping regions in the [sequence window](#) and move it to the bottom of the list in the **Region Browser**
- **Copy** - copy region to user-specified name
- **Rename** - change region name
- **Delete** - remove region; the **Chimera selection** region cannot be deleted, but can contain zero residues
- **Info** - report region contents in the [Reply Log](#), in terms of [alignment numbering](#), and positions within each sequence ([sequence numbering](#)), and residue numbers within any associated structures

Many other operations create or modify regions, including:

- sequence-structure [association](#)
- [selection](#) within sequence-associated structures
- [searching](#) for occurrences of a particular string or pattern of residues
- showing [secondary structure](#), *actual* in structure-associated sequences, *predicted* in sequences not associated with structures
- reading a sequence coloring format (SCF) file or seqsel file ([File... Load SCF/Seqsel File](#)) generated by the program [JEvTrace](#). The two [formats](#) are slightly different, but both contain information for coloring regions in the sequence and in any [associated](#) structures. The file reader automatically determines which format is being used. The formats are simple enough that users wishing to color residues alike in sequence and structure (for any reason) may wish to create a file manually and read it in by this route.

## TREES

A [Newick-format](#) tree corresponding to the alignment can be read and displayed to the left of the sequences with [Tree... Load](#) in the **Multialign Viewer** menu. A newly loaded tree will replace any previously shown tree. The tree can be hidden/shown with [Tree... Show Tree](#).

The sequence names in the tree must be the same as the [sequence names](#) in the alignment, and neither set of names can be a subset of the other. If the order of sequences in the tree differs from that in the alignment, a dialog will appear, requesting permission to reorder the alignment sequences for clearer display (to avoid tree branch crossings). **Reordering will delete multisequence regions**, although the **Chimera selection** region (if any) will subsequently reappear.

When the tree includes branch lengths, both solid and dotted lines are used for display, with horizontal solid line lengths proportional to branch lengths. When the tree does not include branch lengths, only solid lines are used.

Clicking a node in the tree makes it red and chooses it for subsequent operations. [Tree... Extract Subalignment](#) copies the sequences defined by the chosen node into a separate [sequence window](#). The alignment of the sequences is kept exactly the same as in the full alignment, including any all-gap columns, so that the subalignment is initially the same length as the full alignment. If desired, all-gap columns can be deleted with the [Edit menu](#).

## EDITING AND SAVING THE ALIGNMENT

Sequences can be [added](#), [deleted](#), [reordered](#), [realigned](#), and [renamed](#) using the **Multalign Viewer Edit menu**. A sequence to [add](#) can be pasted as plain text, read from a file, inferred from a structure, or retrieved from [UniProt](#) along with its feature annotations. Several parameters can be adjusted to control how the new sequence is aligned with the others. Sequence [reordering](#) will delete multisequence [regions](#), although the **Chimera selection** region (if any) will subsequently reappear. Gap columns can be [inserted](#) and [removed](#) using the **Edit menu**, and a sequence can be [copied as plain text](#) for pasting into another application window. The [active region](#) contents (if single-block) can be opened in a new [sequence window](#) with **Edit... Region → New Window**.

**Edit... Realign Sequences** uses a Clustal Omega or MUSCLE web service to generate a new alignment from all of the sequences in the current alignment.

Manual editing allows rearranging residues and gaps without changing the sequences. In other words, residues *within* a sequence cannot be created, deleted, mutated, or reordered, but gap positions can be created and deleted.

**Edit... Show Editing Keys** in the **Multalign Viewer** menu brings up a dialog summarizing **Ctrl**-key editing functions. The [active region](#) can be moved one column at a time with **Ctrl**-left arrow and **Ctrl**-right arrow, or as far as possible in one step with **Shift-Ctrl**-arrow (until the active region abuts either some other residues or the end of the alignment). If the active region already abuts the alignment start or end, new columns will be created as needed to accommodate the movement. Such movements can be undone step-by-step with **Ctrl**-down arrow or collectively with **Ctrl-Esc**. Only **Ctrl**-key editing can be undone, not the larger-scale operations described below (sequence addition, deletion, and reordering). Such larger-scale operations clear the editing history, but otherwise all **Ctrl**-key editing operations are retained. A previously undone movement can be redone with **Ctrl**-up arrow. Undo/redo repositions residues but not the [active region](#) box, since the active region could have been changed during editing.

Control Key + ...	Function	Notes
← / →	move current region one column left/right	1
Shift + ← / →	move current region max left/right	1,2
↓	undo	3
↑	redo	
Escape	undo all	3
[1] Must have all gaps in region's adjacent left/right column [2] Motion stops when next column is not all gaps [3] Can only undo edits listed above (e.g. not add/delete sequences)		

Choosing **File... Save As** from the **Multalign Viewer** menu brings up a [dialog](#) for saving the sequence alignment to a file. It is possible to save just the [active region](#) to a file; the region may consist of disjoint sections, but each sequence (row) included must contain the same set of columns. All-gap columns (for example, arising when a region to be saved includes only a subset of the sequences) can be omitted from the output, and [numbering can be appended](#) to the [sequence names](#). The [sequence alignment formats](#) available for saving are the same as those that can be read. An alignment saved in [Stockholm format](#) will automatically include [annotations describing the secondary structure](#) of any [associated](#) structures.

The [sequence window](#) contents can also be saved as an EPS file (**File... Save EPS** in the **Multalign Viewer** menu).

## PERCENT IDENTITY

Pairwise percent identity can be calculated all-by-all or for specified pairs of sequences in the alignment. **Info... Percent Identity** in the **Multalign Viewer** menu opens a dialog for choosing which sequences to compare (from functionally interchangeable pulldown menus labeled **Compare** and **with**):

- all sequences (default)
- individual sequences listed by name

There is also a choice of denominator, what to **divide by**:

- **shorter sequence length** (default)
- **longer sequence length**
- **non-gap columns in common**

**Apply** calculates the percent identity without dismissing the dialog, while **OK** calculates the percent identity and dismisses the dialog. The percent identity is reported in the status area near the bottom of the [sequence window](#) and the Chimera [Reply Log](#). **Close** simply dismisses the dialog, and **Help** opens this manual page in a browser window.

## ANNOTATIONS

Sequence feature annotations from [UniProt](#) and the [Conserved Domain Database \(CDD\)](#) are treated as [regions](#) that can be displayed as colored boxes in the [sequence window](#). Sequence annotations from UniProt are present automatically for sequences [fetched](#) from UniProt, and UniProt or CDD annotations can be added to other sequences (those already open in Chimera) using [Info... UniProt/CDD Annotations](#).

The remainder of this section refers to annotations not handled as [regions](#).

Annotations and comments associated with an alignment as a whole (as opposed to individual columns or sequences) can be viewed and edited using [Edit... Alignment Annotations](#) in the **Multalign Viewer** menu. Changes are not assessed for correctness.

Alignment *annotations* are shown in the upper part of the dialog. Each has two parts, a name and a value (generally text). For example, **MSF length** is a possible name and **366** is a possible associated value. A new annotation can be added by clicking **New**, specifying a name, and then entering a value in the adjacent field. Values can have multiple lines even though only one is shown; pressing return in a value field starts a new line, but the previous information is retained. Clicking **Delete** and then choosing an annotation name removes the corresponding annotation.

Alignment *comments*, shown in the lower part of the dialog, may consist of multiple lines of free-form text.

Not all [sequence alignment formats](#) can accommodate annotations. Thus, [saved](#) files may not include this information or reflect any changes that have been made. Only the [Stockholm format](#) accommodates arbitrary [annotations](#), whereas both Stockholm and RSF formats allow for [comments](#).

Markups in [Stockholm format](#) are handled as follows:

- GF (Generic per-File) markups are interpreted as alignment [annotations](#) or [comments](#) (GF CC indicates the latter).
- GC (Generic per-Column) markups are treated as [headers](#); they can be shown above the alignment and are available as [residue attributes](#) in any [associated](#) structures.
- GS (Generic per-Sequence) markups are not displayed by **Multalign Viewer**.
- GR (Generic per-Sequence and per-Column) markups are not displayed by **Multalign Viewer**, but when an alignment is [saved](#) in Stockholm format, the file will automatically include GR markup lines describing the secondary structure of any [associated](#) structures.

The GR markup line for a sequence and its [associated](#) structure starts with

```
#=GR seq-name Chimera_actual_SS_struct-name
```

The structure name is included because there can be more than one structure associated with a given sequence. Chimera-generated GR secondary structure markups may include the symbols:

symbol	meaning
.	gap in sequence
H	helix
E	strand
C	other structure
X	sequence residue not associated with structure residue

If a sequence did not have a GR SS markup in the original input file and is associated with only one structure, another markup line with the same contents but named **SS** instead of **Chimera\_actual\_SS\_struct-name** will also be included.

## SEQUENCE-STRUCTURE ASSOCIATION

When a sequence alignment and one or more structures are opened in Chimera (in any order), a structure chain will be associated automatically with a sequence in the alignment if their sequences can be aligned without too many mismatches ([details...](#)). Associations can be [changed or added](#) if the automated procedure does not give the desired result. However, when the sequence of a structure in Chimera is shown [individually](#) via the **Sequence** tool, the sequence and the corresponding structure chain are associated and cannot be dissociated.

The [names](#) of structure-associated sequences are shown in bold over a box indicating the [model-level color](#) of the structure. For example, **APE\_HUMAN** in the [sequence window figure](#) is associated with a structure whose model-level color is tan. If multiple models are associated with the same sequence, the box is shown as a dark green dashed outline.

Association enables several types of sequence-structure crosstalk:

- **Associated structure reported on mouseover.** When the cursor is placed over the *name* of a sequence, information on the sequence and any associated structure(s) is shown in the status area near the bottom of the [sequence window](#). When the cursor is placed over a sequence *residue* that is associated with a residue in one or more structures, information on the structure residue(s) is given. Association information can be [saved to a file](#).
- **Clicking/dragging on sequence → structure selection.**
  - When a [region](#) is made the [active region](#) (for example, when newly created by dragging), any associated structure residues will be [selected](#).
  - Clicking within a [header](#) line will [select](#) any structure residues associated with that column of the alignment.
- **Structure selection → sequence region.** By default, making a [selection](#) in one or more sequence-associated structures will create a [region](#) named **Chimera selection** in the associated sequences. Every residue with at least one atom or bond selected will be included in the region. The color of the region and whether it should be created can be controlled in the [Regions preferences](#). The region will not be created when the selection merely reflects the current [active region](#).
- **Focusing the view.**
  - Clicking with the right mouse button on a structure-associated residue in the [sequence window](#) will focus the graphics window view on the corresponding structure residue(s).
  - **Shift**-right-clicking within a [region](#) (or right-clicking within the region but not on a residue) will focus the view on all of the structure residues associated with the region.

When the current [selection](#) includes one or more sequence-associated residues, [Structure... Expand Selection to Columns](#) will expand the selection to include all residues associated with the same column(s) of the sequence alignment. This can be reversed with **Select... Undo** in the main Chimera menu.

The allowable number of mismatches for automatic association is user-specified as a proportion of the total number of residues in the structure chain (1/10 by default; see the [Structure preferences](#)). For automatic association, gaps in the structure sequence relative to the sequence in the alignment file can only occur where residues are missing from the structure (for example, a flexible loop with insufficient density for coordinates to be determined). The order in which

the sequence and structure files are opened does not matter. Associations are reported in the status area near the bottom of the [sequence window](#) and the Chimera [Reply Log](#). A structure (even if it has multiple chains) cannot be associated with more than one sequence, but a single sequence can be associated with more than one structure. If more than one sequence matches a given structure chain, the single best-matching sequence is associated.

Sequence-structure association may create new [regions](#). If all residues of a sequence are matched, no region is created. Otherwise, regions are created containing the matched segment(s), any mismatches, and any gaps in the structure relative to the sequence. The region names report the associated structure chain and the number of gaps or mismatches. The initial colors and display status (shown or hidden) of these regions can be controlled in the [Regions preferences](#).

### Changing Associations

Associations can be changed or added using [Structure... Associations](#) in the **Multalign Viewer** menu. The resulting dialog lists the molecule models open in Chimera. An association is specified by choosing a chain (if the model has more than one chain) and the name of the sequence to associate with that chain. The choices for association include the sequences in the alignment displayed by **Multalign Viewer** and **none**. When the setting is **none**, there is an option to **associate with best match**, *i.e.* to compare the structure chain with all of the sequences in the alignment and associate it with the one that yields the fewest mismatches.

For the first submodel in an ensemble of structures (multiple submodels with the same primary model number, such as from NMR), there will be an additional button to **Propagate** its association setting to all of the other submodels in the ensemble. This facilitates associating the whole ensemble with the same sequence for purposes of [superposition](#) and/or calculating [RMSD headers](#).

Associations will be made as specified, no matter how inappropriate. **Apply** performs the associations without dismissing the panel, while **OK** performs the associations and dismisses the panel. **Close** dismisses the panel without changing the associations. **Help** opens this manual page in a browser window.

### Automatic Structure Loading

[Structure... Load Structures](#) in the **Multalign Viewer** menu can be used to open structure files corresponding to sequences that are not already structure-associated. The corresponding structure files are determined from the [sequence names](#) using the rules given in the [Structure preferences](#), then retrieved and opened (as described for [Fetch by ID](#)). If **Automatically load Structures** is turned on (also in the [Structure preferences](#)), this will occur as soon as an alignment is read.

## SECONDARY STRUCTURE

In sequences associated with structures, [regions](#) named **structure helices** (pale yellow with gold outline) and **structure strands** (pale green with darker outline) can be shown/hidden with the **Multalign Viewer** menu setting [Structure... Secondary Structure... show actual](#). This setting is initially turned on for individual sequences displayed with the **Sequence** tool.

Protein helix and strand assignments are taken from the input structure file or generated with [ksdssp](#). Helix and strand regions can overlap when more than one structure is associated with a sequence. (Regardless of whether the regions exist, however, an alignment saved in [Stockholm format](#) will automatically include [annotations describing the secondary structure](#) of any associated structures.)

In sequences not associated with structures, [regions](#) named **predicted helices** (gold outline) and **predicted strands** (green outline) can be shown/hidden with the **Multalign Viewer** menu setting [Structure... Secondary Structure... show predicted](#). The prediction is done with [GOR](#).

## RESIDUE ATTRIBUTES

Structure residues [associated](#) with residues in a sequence alignment shown by **Multalign Viewer** are assigned [attributes](#):

- **mavPercentConserved** - the value for a residue is the percent conservation of the most prevalent residue type at the corresponding position in the alignment.
- **mavHeader**, where *Header* is the name of a [header](#) line - the value for a residue is the character or number (shown with bar height in a histogram) in the header line at the corresponding position of the alignment. For example:
  - if [Conservation](#) is shown as a histogram, there will be a **mavConservation** attribute with numerical values; if it is shown with characters, the attribute values will be characters
  - if the [RMSD: ca](#) header is shown, there will be a corresponding residue attribute named **mavRMSDca**

Structure residues not associated with residues in the sequence alignment are not assigned attribute values.

[Structure... Select \(or Render\) by Conservation](#) in the **Multalign Viewer** menu opens the corresponding portion of [Render/Select by Attribute](#), set to the residue attribute **mavConservation** if numerical, otherwise **mavPercentConserved**.

Numerical attributes will appear in the attribute lists of [Render/Select by Attribute](#); character attributes will be listed only [Select by Attribute](#) portion. It may be necessary to [Refresh](#) the attribute menus or values in these tools to update them with any changes in **Multalign Viewer** header information.

**Note:** if a structure is associated with more than one alignment, the attribute values are based on the alignment for which association occurred or changed most recently. Generally this is the alignment most recently created or opened, but a different alignment can be forced as the “winner” by choosing [Structure... Associations](#) from its menu and toggling an existing association to **none** and then back (each time clicking **Apply**).

## STRUCTURAL SUPERPOSITION

[Structure... Match](#) in the **Multalign Viewer** menu allows structures to be superimposed based on the alignment of their [associated](#) sequences. The dialog contains two subpanels, each listing the structures [associated](#) with sequences in the alignment. One reference structure should be chosen in the left side, but any number of structures to be matched (superimposed) with it can be chosen in the right side. Residues in each **Structure to match** are paired with the aligned (in the sequence alignment) residues of the **Reference structure**. If both structures are [associated](#) with the same sequence, the correspondence is even more obvious. Fitting uses one point per residue: CA in amino acid residues and C4' in nucleic acid residues. If a nucleic acid residue lacks a C4' atom (some lower-resolution structures are P traces), its P atom will be paired with the P atom of the aligned residue. The number of atom pairs fitted and the resulting RMSD are reported in the Chimera [Reply Log](#) and the status area near the bottom of the [sequence window](#).

**Match highly conserved residues only** causes only the well-conserved (at least 80%) positions in the alignment to be used for the least-squares fit. These are the positions shown as capital letters in the [consensus sequence](#).

**Match active region only** causes only the positions in the current [active region](#) of the alignment to be used for matching.

**Use pseudobonds to show matched atoms** indicates that lines ([pseudobonds](#)) should be drawn between the matched atoms. For each matched pair of structures, a [pseudobond](#) group is created and colored uniquely (in order, the [named colors](#) dark green, dodger blue, sienna, yellow, spring green, purple, gray, and coral are used). Each group is named **matches of...**, where the rest of the name indicates the structures and chains that were matched. The [PseudoBond Panel](#) can be used to change the appearance of or delete the [pseudobonds](#).

**Iterate by pruning long atom pairs until no pair exceeds [x] angstroms** refers to an iterative fitting procedure: in each cycle, atom pairs are removed from the match list and the remaining pairs are fitted, until no matched pair is more than  $x$  apart (default 2.0 Å). The atom pairs removed are either the 10% farthest apart of all pairs or the 50% farthest apart of all pairs exceeding the cutoff, whichever is the lesser number of pairs. The result is that the best-matching "core" regions are maximally superimposed; conformationally dissimilar regions such as flexible loops are not included in the final fit, even though they may be aligned in the sequence alignment.

**Create region showing matched residues** indicates that the residues paired to generate the final fit should be shown as a [region](#) named **matched residues**.

**Apply** performs the matching (superposition) without dismissing the dialog. **OK** performs the matching and dismisses the dialog, **Cancel** dismisses the dialog without performing a match, and **Help** opens this manual page in a browser window. See also: [Chimera superposition methods](#)

## SUPERPOSITION ASSESSMENT

The spatial variation among multiple superimposed structures can be shown and analyzed using one or more of the [RMSD headers](#) and corresponding [residue attributes](#) (**mavRMSDca**, **mavRMSDbackbone**, and **mavRMSDfull**). All of the structure residues associated with a column in the alignment are assigned the same value of a given attribute.

An *overall RMSD* value is calculated for each [region](#) in which a block encloses at least one column [associated](#) with at least two structure residues. All pairwise within-column distances are included in the calculation of the overall RMSD, and all distances are weighted equally (columns may not be weighted equally, as they may be associated with differing numbers of structure residues). The current overall RMSD for a region is reported at the bottom of the [sequence window](#) and in the [Reply Log](#) as the region is created, and values that are updated as needed are shown in the **RMSD** column of the [Region Browser](#).

Pairwise assessments of spatial variation can be performed by choosing [Structure... Assess Match](#) from the **Multalign Viewer** menu. Each pairwise comparison creates an attribute of the residues of one structure containing the distances from the sequence-aligned residues of a reference structure.

A **Reference structure** should be chosen from the pulldown menu of structures [associated](#) with sequences in the alignment. One or more **Structures to evaluate** should be chosen from the list of remaining structures. The structures should already be superimposed, but the superposition can have been generated in any way, including manually (not necessarily using [Structure... Match](#)).

[One point per residue](#) is used for the comparison. The distance between each reference-evaluation pair of residues aligned in the sequence alignment (or corresponding to the same residue, if the reference and evaluation structures are [associated](#) with the same sequence) is measured. The distances are assigned as a [residue attribute](#) of the evaluation structure(s), named **matchDist** by default. After attribute values have been assigned, [Select by Attribute](#) will appear, set to the new residue attribute.

**OK** performs the comparison and dismisses the dialog, while **Close** just dismisses the dialog. **Help** opens this manual page in a browser window.

## INTERFACE TO MODELLER

The program [Modeller](#) performs protein comparative modeling and related calculations. Chimera provides a graphical interface to running Modeller, either locally or via a web service hosted by the [UCSF RBVI](#). Modeller use requires a license key, available free of charge to academics upon [registration](#). Two types of calculations are available:

1. [Comparative \(homology\) modeling](#), interface shown by choosing [Structure... Modeller \(homology\)](#)

from the **Multalign Viewer** menu. In comparative modeling, theoretical models of a protein are generated using at least one known related structure and a sequence alignment of the known and unknown structures. The protein to be modeled is the *target*, and a related known structure used for modeling is a *template*. The inputs for comparative modeling can be generated in several ways:

- Using a pre-existing target-template sequence alignment:
  - [open](#) the [sequence alignment file](#) and template structure(s)
  - make sure the template structure(s) are [associated](#) with the alignment
- Starting with a structure already known to be a suitable template:
  - [open](#) the structure(s)
  - if a single template structure, start [Sequence](#) to show its sequence
  - if multiple template structures, use the [Align Chain Sequences](#) tool to generate a multiple alignment from their sequences
  - [add](#) the target sequence to align it to the template sequence(s) ([Edit... Add Sequence](#) in the **Multalign Viewer** menu)
  - if needed, [realign](#) the sequences to further improve the multiple alignment ([Edit... Realign Sequences](#) in the **Multalign Viewer** menu)
- Searching the PDB for structures related to the target (for multidomain targets, see [mda](#)):
  - start [Blast Protein](#) (under **Tools... Sequence** in the main Chimera menu), paste in the target sequence as plain text, search the **pdb** database for similar sequences, *i.e.*, structures that could be used as templates; from the list of results, choose one or more hits, show them in MAV (**Multalign Viewer**) as a sequence alignment with the query, and load the corresponding structure(s)
    - or -
  - [fetch](#) the target sequence from [UniProt](#), use the **Multalign Viewer Info menu** to start [Blast Protein](#), proceed as above
    - or -
  - [open](#) a file containing just the target sequence ([FASTA](#) or [NBRF/PIR](#) format), use the **Multalign Viewer Info menu** to start [Blast Protein](#), proceed as above

\* **Note:** an alignment from Blast is only a pseudo-multiple alignment, a consolidation of the pairwise alignments of individual hits to the query. Since alignment quality is important for comparative modeling, it may be helpful to [realign](#) the sequences with true multiple-alignment software ([Edit... Realign Sequences](#) in the **Multalign Viewer** menu). Further, Blast may omit parts of the sequences of hit structures. To generate a multiple alignment containing the entire sequences, load the structures of the Blast hits of interest, then proceed as described [above](#) for starting with a structure.
- 2. [Building](#) parts of a protein without using a template. The interface is shown by choosing [Structure... Modeller \(loops/refinement\)](#) from the **Multalign Viewer** menu. Missing segments can be built *de novo*, or existing segments *refined* by generating additional possible conformations. Parts that need building or refinement are often (but not always) loops between secondary structure elements.

## PREFERENCES

The **Preferences** section of the [Multalign Viewer menu](#) controls preferences specific to **Multalign Viewer**. Settings are saved to the Chimera [preferences file](#) as soon as they are changed.

**Multalign Viewer** preferences are grouped together in sections shown as index cards:

- [Appearance](#) - sequence display parameters
- [Structure](#) - structure-loading rules
- [Headers](#) - calculation of sequence alignment [headers](#)
- [Regions](#) - creation and display of [regions](#) (colored boxes)

**Close** dismisses the **Multalign Viewer** preferences tool, and **Help** opens this manual page in a browser window.

Note that a typed-in value will not be applied until **Apply**, **OK** (which also dismisses the preferences tool), or the Enter (return) key has been pressed. Changes in other types of settings

take effect and are saved immediately.

The **Appearance** section of the [Multalign Viewer preferences](#) controls sequence text arrangement and coloring. Most of the preferences in this section can be set separately for **Multiple alignments** and **Single sequences**.

- **Line wrapping** - whether sequences should be shown as a single horizontal bar or wrapped to form successive blocks
  - **Wrap to new line every [n]0 residues** (default for single sequences) - wrap at the specified interval (default  $n=5$ , or 50 positions) regardless of the number of sequences
  - **Wrap if [k] or fewer sequences** (default for multiple alignments, not relevant to single sequences) - wrap at the interval specified above if the alignment contains no more than  $k$  sequences (default 8)
  - **Never wrap**
  - **Space between wrapped blocks** - whether to pad blocks vertically (default true for alignments, false for single sequences)
- **Font**
  - **Use [m] point** (Courier/Helvetica/Times), optionally **Bold** ( $m=12$  by default)
- **Spacing**
  - **Column separation (pixels)** (default 0 for alignments, -2 for single sequences)
  - **Line separation (pixels)** (default 1 for alignments, 8 for single sequences)
  - **Space after every 10 residues** (default true for alignments, false for single sequences)
- **Residue letter coloring**
  - **Color scheme**
    - **black** (default for single sequences) - color all letters black
    - **Clustal X** (default for alignments) - use [Clustal X coloring](#), which depends on both residue type and the pattern of conservation within a column
    - **ribbon** - color one-letter codes to match the [associated](#) structure residues, the average color where multiple structures are associated, and black where no structures are associated; [ribbons](#) will match the lettering, but need not be displayed for this option to work. Possible uses include making the letters match a structure that has been colored [by attribute](#), by domain, or over a [rainbow](#) range of hues.
    - **Kyte-Doolittle hydrophobicity** - color one-letter codes by [Kyte-Doolittle hydrophobicity](#), ranging from red for the most hydrophobic amino acids to blue for the most hydrophilic, with black in the middle
    - custom coloring schemes can be listed here by describing them in the Clustal X [parameter file format](#) and reading them in with [File... Load Color Scheme](#)
- **Sequence names**
  - **Use ellipsis for names longer than [j]** ( $j=30$  characters by default)

The **Structure** section of the [Multalign Viewer preferences](#) sets rules for automatic [sequence-structure association](#) and structure loading.

- **Auto-association** - criteria for automatic [sequence-structure association](#)
    - **Allow one mismatch per [n] structure residues** ( $n=10$  by default)
  - **Structure loading** - whether/how [sequence names](#) relate to the names of the corresponding structure files; whether to retrieve and open such files (as described for [Fetch by ID](#)) as soon as a sequence alignment is read. No more than one structure per sequence will be loaded. If a sequence name corresponds to both a SCOP file and a PDB file, the SCOP file will be loaded (unless the **Load SCOP...** option is turned off).
    - **Load SCOP file for each unassociated sequence whose name appears to be a [SCOP ID \(sid\)](#)** (on by default)
    - **Load PDB file for each unassociated sequence whose name...**
      - **is < pdb code >**  
(e.g. "1gcn")
      - **starts with < pdb code >**  
(e.g. "1k6w\_A")
      - **contains "pdb|" + < pdb code >**  
(NCBI-style; e.g. "gi|1421399|pdb|1FTZ")
      - **starts with [string] + < pdb code >**  
where *string* is user-specified
- These correspondence rules are not mutually exclusive; all but the last are on by default.

- **Automatically load structures** (off by default) - whether to retrieve and open structures as soon as a sequence alignment is read; if not, the procedure can still be invoked with [Structure... Load Structures](#) in the **Multalign Viewer** menu
- **Do not autoload if more than [M] structures would load** (on and  $N=10$  by default)

The **Headers** section of the [Multalign Viewer preferences](#) controls how alignment [headers](#) are calculated and whether alignment [numbering](#) is shown initially.

- **Consensus style (majority in column including gaps/majority in column ignoring gaps)** - what to show in the [header](#) line named **Consensus**. The consensus sequence shows the residue type most prevalent (not necessarily >50%) at each position in the alignment; highly conserved residues (80% or greater) are capitalized and shown in purple, except that the completely conserved residues are shown in red. If several residue types are equally the most prevalent, one is chosen at random to appear in the consensus sequence. When gaps are included, however, the consensus will show a gap at positions where a gap is the most prevalent or equally the most prevalent as one or more residue types.
- **Conservation style (AL2CO/Clustal histogram/Clustal characters/identity histogram)** - what to show in the [header](#) line named **Conservation** and assign as the [mavConservation attribute](#) of residues in [associated](#) structures. The values can be [saved to a file](#).

In a **Clustal histogram**, full bar height indicates complete identity, 2/3 bar height indicates Clustal [strong group](#) conservation, and 1/3 bar height indicates Clustal [weak group](#) conservation.

**Clustal characters** are the characters used in Clustal format to indicate conservation: "\*" for complete identity, ":" for [strong group](#) conservation, and "." for [weak group](#) conservation.

An **identity histogram** shows what proportion of the sequences have the most prevalent nongap character per alignment column. Histogram bar heights are proportional to  $(M-1)/(N-1)$ , where M is the number of occurrences of the most prevalent residue type at a position in the alignment and N is the number of sequences in the alignment. Thus, if every sequence has a different residue at a given position, the bar height is zero, not  $1/N$ . However,  $M/N$  values (not the values used for display purposes) are used for the [mavConservation attribute](#) and written out when conservation values are [saved](#).

**AL2CO (ALignment 2 COnservation)** provides more sophisticated options for calculating conservation, including choice of equation (entropy, *etc.*), sequence weighting, and smoothing over windows. Users should cite:

[AL2CO: calculation of positional conservation in a protein sequence alignment](#). Pei J, Grishin NV. *Bioinformatics*. 2001 Aug;17(8):700-12.

#### AL2CO parameters:

- **Frequency estimation method** (unweighted/modified Henikoff & Henikoff/**independent counts**) - whether/how to weight the sequences
- **Conservation measure** (**entropy-based**/variance-based/sum of pairs) - which of the AL2CO equations to use for conservation
- **Averaging window** (default 1) - window width, or how many alignment positions to average in a sliding window. The result is assigned to the central column of the window when the width is odd, the column left of center when the width is even (*e.g.*, the fourth position in a window of 8), with various adjustments at the termini. The authors of AL2CO recommend a width of 3 for motif analyses.
- **Gap fraction** (default 0.5) - maximum fraction of sequences with gaps in a column for conservation to still be calculated; for example, if the **gap fraction** is 0.25, conservation values will only be calculated for columns in which at least three-fourths of the sequences have residues
- **Sum-of-pairs matrix** [choices: BLOSUM-30, BLOSUM-35, BLOSUM-40, BLOSUM-45, BLOSUM-50, BLOSUM-55, BLOSUM-60, **BLOSUM-62** (default),

BLOSUM-65, BLOSUM-70, BLOSUM-75, BLOSUM-80, BLOSUM-85, BLOSUM-90, BLOSUM-100, BLOSUM-N, HSDM, identity, PAM-40, PAM-120, PAM-150, PAM-250, SDM] - which amino acid similarity matrix to use with the AL2CO sum of pairs [Conservation measure](#)

- o **Matrix transformation (none/normalization/adjustment)** - whether/how to modify the matrix used with the AL2CO sum of pairs [Conservation measure](#)

$$\text{normalization: } S'_{ab} = S_{ab} / (S_{aa}S_{bb})^{1/2}$$

$$\text{adjustment: } S''_{ab} = 2S_{ab} - 1/2(S_{aa} + S_{bb})$$

Conservation values from AL2CO are in standard deviations from the mean (sometimes called Z-scores) and can range from  $-\infty$  (least conserved) to  $+\infty$  (most conserved).

- **Show numbering initially (true/false)** - whether to show [alignment numbering](#) when an alignment is opened (does not apply to single sequences)

The **Regions** section of the [Multalign Viewer preferences](#) controls whether a [selection](#) will be shown as a [region](#) and sets initial region colors.

- **Show Chimera selection as region (true/false)** - whether to show a [selection](#) within structures as a [region](#) named **Chimera selection** within [associated](#) sequences (if any); a residue will be included in the region if any of its atoms are selected
- **Chimera selection region border color** (a [color well](#), **No color** by default) - initial border color for the region showing the Chimera [selection](#)
- **Chimera selection region interior color** (a [color well](#), **green** by default) - initial interior color for the region showing the Chimera [selection](#)
- **New region border color** (a [color well](#), **No color** by default) - initial border color for [manually created regions](#)
- **New region interior color** (a [color well](#), **white** by default) - initial interior color for [manually created regions](#)
- **Show balloon help for regions (true/false)** - whether to show the region name in a balloon when the cursor is paused over the region (but not over a residue symbol, as that would instead report the [associated](#) structure residue)
- **Structure association** - whether and how to show regions created during [association](#)
  - o **Draw region depicting successful matches** (on/off; [color wells](#) for the border and interior, default **black** and **No color**, respectively)
  - o **Draw region depicting mismatches** (on/off; [color wells](#) for the border and interior, default **No color** and **red**, respectively)
  - o **Draw region depicting missing structure** (on/off; [color wells](#) for the border and interior, default **red** and **No color**, respectively)

## MENU LISTING

The **Multalign Viewer** menu includes the following sections:

- [File](#)
- [Edit](#)
- [Structure](#)
- [Headers](#)
- [Numberings](#)
- [Tree](#)
- [Info](#)
- [Preferences](#)

## File

- **Save As...** bring up a [dialog](#) for [saving the alignment](#) to a file. The [sequence alignment formats](#) available for saving are the same as those that can be read. Options:
  - **Restrict save to [active region](#)** (off by default)
  - **Omit all-gap columns** (default on)
  - **[Append sequence numberings](#) to sequence names** (default on if [sequence numbering](#) is displayed, otherwise off)
- **Save EPS...** bring up a [dialog](#) for saving an Encapsulated PostScript file of the [sequence window](#) contents. Options:
  - **color mode** (color/grayscale/black & white)
  - **rotate 90** (true/false)
  - **extent** ([visible region](#)/entire alignment)
  - **hide tree control nodes** (true/false) - whether to omit the small squares from a [tree](#) display

If the alignment is [wrapped](#) (or would wrap if it were longer), one EPS file will be saved with the specified name. Otherwise, the sequence [names](#) and alignment will be saved in two separate EPS files with "-names" and "-alignment" appended to the specified name (before the ".eps" suffix, if present).
- **Save Association Info...** save a text file that states which structure is associated with which sequence and lists the correspondences between alignment positions and structure residues. The **Residue naming style** indicates how residues will be specified in the file:
  - **simple** (for example, **PHE 16**)
  - **[command-line specifier](#)** (for example, **:16**)
- **Load SCF/Seqsel File...** read a [JEVTrace](#) file; color sequence [regions](#) and the corresponding parts of any associated structures accordingly
- **Load Color Scheme...** read a [parameter file](#) to set the coloring of residue codes in the [sequence window](#). When a scheme is loaded, the residue codes are colored accordingly and the name of the custom coloring file is added to the list of choices for [Residue letter coloring](#) in the [Appearance preferences](#). The [dialog](#) for loading the file includes an option to **Make this scheme the default**. The location of each custom coloring file is saved in the [preferences file](#). If a custom coloring file is later renamed, moved, or deleted, trying to load the corresponding scheme will generate an error.
- **Hide** - close the sequence window without changing the state of **Multalign Viewer** (the window can be reopened using the **Raise** option for the **Multalign Viewer [instance in the Chimera Tools menu](#)**, abbreviated MAV)
- **Quit** - exit from **Multalign Viewer**

## Edit

- **Copy Sequence...** copy a sequence as text that can be pasted into another application window. Options:
  - **Remove gaps from copy** (default on) - exclude gap characters (typically present when the sequence is part of a multiple alignment)
  - **Restrict copy to [active region](#)** (default off) - only include residues that are within the [active region](#), even if they form disjoint segments

Note: when the mouse focus is in the [sequence window](#), the system "copy" shortcut (for example, command-C on Mac) will copy the contents of the [active region](#), if any, otherwise the entire sequence(s), into the text buffer. Multiple sequences or their parts will be separated by newlines.
- **Reorder Sequences** - open a dialog for reordering sequences. The dialog lists the names of the sequences in their current order. The order can be changed by clicking on a name and moving it to the **top**, **bottom**, **one higher**, or **one lower**. Any order can be achieved through combinations of these operations. **OK** and **Apply** perform the reordering in the [sequence window](#) with and without dismissing the dialog, respectively. Sequence reordering **will delete multisequence [regions](#)**, although the **Chimera selection** region (if any) will subsequently reappear.
- **Insert All-Gap Columns** - open a dialog for inserting all-gap columns into the alignment (presumably to facilitate subsequent [editing](#) that moves residues into the new space). The dialog allows specification of how many gap columns to add, where to insert them, and the character to use for gap positions. **OK** and **Apply** insert the gap columns with and without dismissing the dialog, respectively.
- **Delete Sequences/Gaps...** open a dialog for deleting sequences and/or all-gap columns. One or more sequences can be slated for deletion by choosing their names from the list. **Ctrl**-click toggles the status of a sequence. To choose a block of sequences without dragging, click on the first (or last)

and then **Shift**-click on the last (or first) in the desired block. **Delete all-gap columns** indicates that columns containing only gaps (evaluated after sequence deletion, if any) should be removed. Clicking **Apply** (or **OK**, which also closes the dialog) performs the indicated deletions.

- **Add Sequence** - open a dialog for adding a new sequence to the alignment. **The existing alignment will not be changed, except that gaps may be inserted between its columns.** Unless simply appended as described below, a new sequence will be added to the alignment using the Needleman-Wunsch algorithm and the indicated [scoring parameters](#) as applied to **All sequences** or only the **Specified sequences**. Either way, the result will include all sequences, but sometimes a better alignment can be obtained by omitting hard-to-align sequences (those not similar to the new sequence) from the scoring calculations. Clicking **Apply** (or **OK**, which also closes the dialog) adds the sequence to the alignment.

Sequence input options:

- **Plain Text** - add a sequence that has been entered or pasted in the **Sequence** field as plain text. It will be converted to upper case and any spaces removed automatically. The sequence should not contain gap characters, unless (with gaps) it is exactly the same length as the existing alignment and the option to **Simply append...** is used.
- **From Structure** - add the sequence of a structure chain open in Chimera
- **From File** - add one or more sequences from a file:
  - A single sequence or multiple unaligned sequences (no gap characters) can be read from a file in [FASTA](#) or [NBRF/PIR](#) format. The sequences do not need to be of equal length; they will be added one by one in their order of occurrence in the file, each addition leaving the preceding alignment unchanged except by the possible insertion of gap columns.
  - An alignment (possibly containing gap characters) in any of the [known formats](#) can be appended if it is exactly the same length as the existing alignment and the option to **Simply append...** is used.
- **From UniProt** - add a sequence from [UniProt](#) and assign its feature annotations as [regions](#) (much as described for [PDB/UniProt Info](#), except independent of structure). UniProt's [ID mapping service](#) can be used to obtain UniProt identifiers from other sequence database identifiers.

Alignment parameters:

- **Matrix** (default **BLOSUM-62**) - [substitution matrix](#) for scoring the alignment. The score for aligning a residue in the new sequence with a column in the existing alignment is the sum of the matrix values for the residue versus each of the residues in the column (and zero versus a gap position) divided by the number of sequences (rows) in the existing alignment. This contribution over the entire alignment is the *residue similarity score*. In some cases there will also be a [secondary structure score](#).
- **Opening penalty** (default **12**) - penalty for opening a gap in the existing alignment, regardless of whether it is already flanked by gaps in any of the sequences. For opening a gap in the new sequence, this penalty is scaled by the fraction of sequences in the existing alignment not gapped at that position. When [secondary structure scoring](#) is used, secondary-structure-specific gap opening penalties apply instead.
- **Extension penalty** (default **1**) - penalty for extending a gap between columns of the existing alignment. For extending a gap in the new sequence, this penalty is scaled by the fraction of sequences in the existing alignment not gapped at that position.
- **Character** (default is the most prevalent gap character in the existing alignment, or the character **.** if the alignment lacks gaps) - character for displaying gap positions created during alignment of the new sequence
- **Include secondary structure score (N%)** - include a secondary structure term in the alignment score and use secondary-structure-specific gap opening penalties. This option only applies when one is adding a sequence [from a structure](#) and at least one other sequence in the alignment is already [associated](#) with a structure. Detailed settings can be shown by clicking **Show parameters**, hidden again by clicking **Hide parameters**. *N* reflects the relative weights of the terms, which can be adjusted by moving the slider. If *N* is 30, for example,

$$\text{total score} = 0.70(\text{residue similarity score}) + 0.30(\text{secondary structure score}) - \text{gap penalties}$$

The values in the secondary structure **Scoring matrix**(for all pairwise combinations of **H**helix, **S**strand, and **O**other) and the secondary-structure-specific **Gap opening penalties** can be adjusted. The secondary structure score is computed in the same way as described for the [residue similarity score](#), except that sequences without associated structures contribute

neither to the score nor to the sequence count used for normalization. When secondary structure scoring is used (even if the slider is set to zero),

- the secondary-structure-specific **Gap opening penalties** are used instead of the single [Opening penalty](#)
- the **Extension penalty** remains the same
- the total penalty is computed as described [above](#), except that sequences without associated structures contribute neither to the penalty nor to the sequence count used for normalization

Contributions from multiple structures associated with the same sequence in the alignment are averaged so that the sequence only counts “once.”

The alignment parameter values last used for adding a sequence are retained. However, **Reset...** buttons are provided for restoring the general alignment parameters (those other than secondary structure scoring) and secondary structure scoring parameters to their factory defaults.

- **Realign Sequences...** use a Clustal Omega or MUSCLE web service to [realign](#) all of the sequences in the current alignment, either replacing the current alignment or opening the result in a new [sequence window](#)
- **Alignment Annotations...** open a dialog for viewing and editing [alignment annotations](#)
- **Edit Sequence Name...** allow entering a new [name](#) for the sequence chosen from the pulldown menu
- **Show Editing Keys...** bring up a dialog summarizing [Ctrl-key editing functions](#)
- **Region → New Window** - open [active region](#) contents in a new [sequence window](#) (single-block regions only)

The next three options allow searching for occurrences of a specified string or pattern of residues in one or all sequences; matches are made into a [region](#) named **search result**:

- **Find Subsequence...** find matches to a string, potentially with **Ambiguity codes**:
  - **none** - only allow exact matches of one-letter codes
  - **protein** - in addition to exact one-letter code matching, allow B to match aspartic acid (D) and asparagine (N) and Z to match glutamic acid (E) and glutamine (Q)
  - **nucleic acid** - in addition to exact one-letter code matching, allow R to match the standard purine nucleotides (A, G), Y to match the standard pyrimidine nucleotides (C, T, U), and N to match any of the five (A, C, G, T, U)
- **Find Regular Expression...** find matches to a pattern specified as a [regular expression](#)
- **Find PROSITE Pattern...** find matches to a pattern specified with [PROSITE pattern syntax](#) (residue codes must be capitalized; extended syntax involving an asterisk, *e.g.* <{C}\*>, is not supported)
- **Find Sequence Name...** find sequences whose names contain or exactly match an entered string, where capitalization is important. Clicking **Find** shows a matching sequence name in red and scrolls the alignment vertically to center on it. The sequence's position in the alignment (#N of M) and number of residues is reported near the bottom of the sequence window. If there are multiple matches, clicking **Find** again locates the next one.

## Structure

- **Load Structures** - for sequences not already [associated](#) with a structure, retrieve and open the corresponding structure files (as described for [Fetch by ID](#)); correspondence is determined from [sequence names](#) according to the rules given in the [Structure preferences](#)
- **Match...** open a dialog for [superimposing structures](#) based on the sequence alignment
- **Assess Match...** open a dialog for [evaluating](#) how well the sequence-aligned residues of structures are superimposed in space
- **Modeller (homology)...** show interface to [comparative modeling with Modeller](#)
- **Modeller (loops/refinement)...** show interface to [building or refinement with Modeller](#)
- **Associations...** open a dialog for [adding or changing associations](#) between structures and sequences
- **Secondary Structure**
  - **show actual** - whether to show [regions](#) named **structure helices** (pale yellow with gold outline) and **structure strands** (pale green with darker outline) in sequences with [associated](#) structures. Protein helix and strand assignments are taken from the input structure file or generated with [ksdssp](#).

- **show predicted** - whether to show [regions](#) named **predicted helices** (gold outline) and **predicted strands** (green outline) in sequences without [associated](#) structures. Helices and strands in these sequences are [predicted using GOR](#).
- **Select by Conservation...** open the [Select by Attribute](#) tool to [select](#) residues in sequence-[associated](#) structures by [conservation](#)
- **Render by Conservation...** open the [Render by Attribute](#) tool to color or otherwise change the appearance of residues in sequence-[associated](#) structures by [conservation](#)
- **Expand Selection to Columns** - if any sequence-[associated](#) residues are [selected](#), expand the selection to include all residues associated with the same column(s) of the sequence alignment

## Headers

- **Save...** save a [header definition file](#) containing alignment positions and header values. Header values are the characters or numbers (shown with bar heights in a histogram) in a [header](#) line such as [Conservation](#) in the [sequence window](#). No-value positions (for example, when the AL2CO [Conservation style](#) is used, positions with a gap in more than [gap fraction](#) of the sequences) can be omitted from the file.
- **Load...** read a [header definition file](#) to show a new header line and have its values available as [residue attributes](#) in any [associated](#) structures

The available [headers](#) are listed in the bottom part of the menu; checkboxes control which are shown in the [sequence window](#):

- [Consensus](#)
- [Conservation](#)
- *another\_header\_name*
- *(etc.)*

## Numberings

- **Overall Alignment** - whether to show [alignment numbering](#)
- **Left Sequence** - whether to show [sequence numbering](#) to the left of each row of sequences
- **Right Sequence** - whether to show [sequence numbering](#) to the right of each row of sequences
- **Adjust Sequence Numberings...** open a dialog for specifying sequence [start numbers](#). One or more sequences can be chosen with the mouse. **Ctrl**-click toggles the status of a sequence. To choose a block of sequences without dragging, click on the first (or last) and then **Shift**-click on the last (or first) in the desired block. Clicking **Apply** (or **OK**, which also closes the dialog) applies the new **Start** number to the chosen sequence(s).

## Tree

- **Load...** read a [tree](#) in [Newick format](#) and display it to the left of the sequences in the [sequence window](#)
- **Show Tree** - hide/show the previously loaded tree
- **Extract Subalignment** - copy the subalignment defined by the [chosen node](#) of the [tree](#) into a separate [sequence window](#)

## Info

- **Percent Identity...** calculate the [percent identity](#) between pairs of sequences in the alignment
- [Region Browser](#) - manage the existing [regions](#)
- **Blast Protein...** perform a [protein BLAST](#) search with a specified sequence
- **UniProt/CDD Annotations...** map feature annotations from [UniProt](#) or the [Conserved Domain Database \(CDD\)](#) onto a sequence as [regions](#); if the matching database entry lacks feature annotations, or no match is found, no regions will be generated.

- [UniProt](#) annotations are fetched as an XML file that can be cached and reused as needed depending on the [Fetch preferences](#) and whether the option to **Ignore any cached data** is checked. The relevant entry can be identified:
  - **from UniProt ID** - directly by the specified ID, for example, **Q92731** or **esr2\_human**
  - **from PDB code and chain ID** - using the same lookup mechanism as the [PDB/UniProt Info](#) tool. Namely, structure annotations and any PDB-UniProt mapping information (UniProt ID and residue number correspondences) are fetched from the [RCSB PDB](#). The UniProt ID is reported in the [Reply Log](#).
  - **by Blast search of UniProt (may take minutes)** - [UniRef100](#) is BLAST-searched for sequences similar to the query; this may take several minutes and is handled as a [background task](#). Only the top hit is used, regardless of the quality or presence of feature annotations. The UniProt ID and E-value of the top hit are given in the [Reply Log](#). *The top hit may not be identical to the query sequence, and scientific judgement must be exercised as to the transferability of any annotations.*

Possible actions are to annotate the sequence with the UniProt sequence features and/or show the corresponding UniProt web page(s).

- [CDD](#) annotations are obtained by scanning the query sequence against the position-specific score matrices (PSSMs) in the CDD default database, which includes NCBI-curated domains and data imported from Pfam, SMART, COG, PRK, and TIGRFAM (as in [CD-search](#)). This may take several minutes and is handled as a [background task](#). Only the top hit per sequence portion is used as a source of annotations, providing it meets the E-value criterion of 0.01. The [ID](#) of the hit PSSM (representing a conserved domain) is reported in the [Reply Log](#), and the **Also show...** option indicates whether to open the corresponding CDD web page. *Scientific judgement must be exercised as to the transferability of any annotations.* The **CDD info** button opens the general help page at the CDD website.

In the [Region Browser](#), the initial ordering of annotations from either database is alphabetical, right after any missing structure region. If annotations are obtained from both sources, all annotations from the later-used source will be listed above all annotations from the source used earlier.

## [Preferences](#)

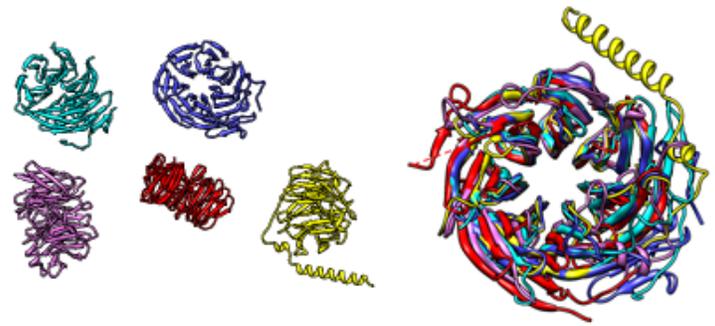
---

UCSF Computer Graphics Laboratory / October 2014

## Superimposing Structures

There are several ways to superimpose structures in Chimera:

- [MatchMaker](#) (or command [matchmaker](#)) performs a fit after automatically identifying which residues should be paired. Pairing uses both sequence and secondary structure, allowing similar structures to be superimposed even when their sequence similarity is low.



The figure shows five distantly related proteins (pairwise sequence identities < 25%) from the [SCOP WD40 repeat-like superfamily](#) before and after [MatchMaker](#) superposition with default parameters.

- Structures can be matched [using a pre-existing sequence alignment](#) shown in [Multalign Viewer](#).
- The exact atoms to pair can be specified with the [match](#) command. This works on any type of structure, while the preceding methods apply only to peptide and nucleotide chains.
- Structures can be superimposed manually by [activating/deactivating](#) them for motion and [using the mouse](#).

Except for manual matching, the methods allow iterative exclusion of poorly superimposed atoms from the fit.

For the special case of an *ensemble*, which contains multiple sets of coordinates for exactly the same atoms, see also: [Ensemble Match](#), [MD Movie \(RMSD analysis\)](#)

A multiple sequence alignment can be generated from a set of superimposed structures using [Match -> Align](#).

See also: [matrixcopy](#), [measure rotation](#), [rmsd](#), [measures of structural similarity](#)

### MatchMaker vs. Match

Usually [MatchMaker](#) (or command [matchmaker](#)) provides the easier route to superimposing related proteins or nucleic acids. Unlike [match](#), it does not require the user to specify which atoms should be used. On the other hand, it:

- takes more time due to the sequence alignment step (determining residue pairing)
- offers less control over which atoms are used for fitting: always uses [one point per residue](#)
- recomputes protein secondary structure assignments; this promotes consistency in assignments between structures and can improve how they are superimposed, but the [option to reassign secondary structure](#) should be turned off if one would rather use the assignments in the input files

Of course, [match](#) would be the preferred method for superimposing sets of atoms representing known local similarities within globally dissimilar structures such as proteins of different folds.

---

UCSF Computer Graphics Laboratory / March 2010

## Building Structures, Modifying and Saving Data

The state of Chimera can be saved to a [session](#) file and later restored.

- Atomic coordinates can be modified or generated in Chimera in many ways, including
  - [bond rotation](#)
  - amino acid [mutation and rotamer swapping](#)
  - [adding hydrogens](#)
  - [deleting atoms](#)
  - [structure building](#)
  - [Modeller calculations](#): comparative (homology) modeling, building missing segments, refinement
  - [minimization](#)
  - generating multimers ([sym](#), [Unit Cell](#), [Multiscale Models](#))
  - repositioning by [superimposing structures](#) or [fitting](#) into maps
  - [renumbering residues](#) and [changing chain IDs](#)

and saved using the command [write](#) or the dialogs described below for [PDB](#) and [Mol2](#).

- [Sequence alignments](#) can be generated in Chimera
  - by sequence search and retrieval of hits with [Blast Protein](#) (pseudo-multiple)
  - by [Match -> Align](#) from the spatial overlap of superimposed structures (multiple)
  - as a by-product of superposition with [MatchMaker](#) (pairwise)
 and [edited and saved](#) using [Multalign Viewer](#).

- [Volume data](#) (maps) can be modified or generated in Chimera by
  - [Volume Viewer](#) (see [Coordinates](#), [Region bounds](#), [Subregion selection](#), and [Zone](#))
  - [Volume Eraser](#)
  - [Volume Filter](#)
  - [molmap](#) to simulate a map from atomic coordinates
  - commands [volume](#), [vop](#), [mask](#)
  - [keyboard shortcuts](#) for [volume editing](#)
  - [fitting](#) (but see note on [saving after fitting](#))

and saved using [Volume Viewer](#) (the [File menu](#)) or the command [volume](#). Related data:

- markers/paths can be created and saved with [Volume Tracer](#)
- segmentations can be created and saved with [Segment Map](#)

See also: [exporting a scene](#), [Write DMS](#), [Write Prmtp](#)

### Saving PDB Files

The dialog for saving [PDB](#) files can be invoked with **File... Save PDB** or **Actions... Write PDB** in the [Chimera menu](#), the **Write PDB...** button on the [Selection Inspector](#), or [write PDB](#) in the [Model Panel](#). The dialog resembles other [open/save dialogs](#), but has additional specific options. See also: [write](#)

Individual models or blocks of models can be chosen from the **Save models** list with the left mouse

button. Ctrl-click toggles the status of an individual model. To choose a block of models without dragging, click on the first (or last) and then Shift-click on the last (or first) in the desired block. Only molecule models are listed.

Options:

- **Save displayed atoms only** - only write coordinates of atoms that are displayed
- **Save selected atoms only** - only write coordinates of atoms that are [selected](#)

Option when only one model is present:

- **Use untransformed coordinates** - write coordinates without applying any rotations and translations that have been performed in Chimera

Option when more than one model is present:

- **Save relative to model [*model*]** - save coordinates relative to the untransformed coordinates of a specified reference *model* (otherwise, transformed coordinates will be saved). This is useful for preserving the spatial relationship between models. For example, if model 1 has been matched or docked to model 0, saving 1 relative to 0 results in the models being matched or docked in the same way when the model 0 file (original) and the model 1 file (saved relative to 0) are reopened.

Option when more than one model is [chosen](#):

- **Save multiple models in**
  - **a single file** - one PDB file containing all chosen models, with MODEL/ENDMDL records delimiting the coordinates of each. Technically, this treatment is only correct for models with identical sets of atoms, but may be convenient in the more general case of disparate models. In the case of multiple models containing different sets of atoms, the CONECT records should be removed before the file is read back into Chimera, as there is no mechanism for handling the different CONECT records for the different models.
  - **multiple files [file name must contain \$name or \$number]** - a different PDB file for each model, named by substituting the model name for **\$name** and/or the model ID number (*model* or *model.submodel*) for **\$number** in the specified file name; if this substitution does not produce unique file names, each file will contain only the last appropriate model

If one or more of the chosen models contains a trajectory, there will also be an option to **Save the current frame** or **all frames**. In this context, "all frames" means all frames whose coordinate sets have been [read in](#) with [MD Movie](#). A model will not be recognized as a trajectory until more than one set of coordinates has been read.

Besides atomic coordinates, [HELIX and SHEET records](#) reflecting the current protein [secondary structure assignments](#) are written, along with any other header lines read from PDB input. Even if protein helix/strand assignments have not been changed in Chimera, the output HELIX and SHEET records may differ from the input because helices are written assuming the right-handed  $\alpha$  type, and strands are written as if each were a separate sheet.

## Saving Mol2 Files

[Mol2 format](#) includes atom types and (optionally) partial charges. When a structure is read from a Mol2 file, the [Sybyl atom types](#) and any partial charges are stored as the atom [attributes mol2type](#) and [charge](#). The input bond orders are also stored. Unless the user specifies [writing Amber/GAFF types instead](#), output Mol2 files include Sybyl atom types. The Sybyl types read from input will be used, if available; otherwise, [Chimera atom types](#) will be translated into Sybyl atom types. As there is not a simple one-to-one relationship between Chimera and Sybyl atom types, *users may wish to check the Sybyl type assignments*, particularly for nonstandard residues. Similarly, if bond orders were not read from Mol2 input, they will be guessed upon output. Values for the [charge attribute](#) can be assigned within Chimera, for example using [Add Charge](#). If no [charge](#) values have been read or assigned, the partial charges will be written as zero.

The dialog for saving [Mol2](#) files can be invoked with **File... Save Mol2** in the [Chimera menu](#). The dialog resembles other [open/save dialogs](#), but has additional specific options. See also: [write](#)

Individual models or blocks of models can be chosen from the **Save models** list with the left mouse button. Ctrl-click toggles the status of an individual model. To choose a block of models without dragging, click on the first (or last) and then Shift-click on the last (or first) in the desired block. Only molecule models are listed.

Option when only one model is present:

- **Use untransformed coordinates** - write coordinates without applying any rotations and translations that have been performed in Chimera

Option when more than one model is present:

- **Save relative to model [*model*]** - save coordinates relative to the untransformed coordinates of a specified reference *model* (otherwise, transformed coordinates will be saved)

Option when more than one model is [chosen](#):

- **Save multiple models in**
  - **a single file [individual @MOLECULE sections]**
  - **multiple files [file name must contain \$name or \$number]** - a different Mol2 file for each model, named by substituting the model name for **\$name** and/or the model ID number (*model* or *model.submodel*) for **\$number** in the specified file name; if this substitution does not produce unique file names, each file will contain only the last appropriate model
  - **a single file [combined @MOLECULE section]** - if the models are all single-residue and the model names are unique but the residue types are not, the model names will be used as substructure names, and the option to include residue numbers in substructure names (below) will be ignored

Further options:

- **Use Sybyl-style hydrogen naming (e.g. HE12 rather than 2HE1)** - move leading numerals to the ends of hydrogen names
- **Include residue sequence numbers in substructure names** - generate substructure names with

embedded residue numbers, for example, ALA85 rather than ALA

- **Write Amber/GAFF atom types instead of Sybyl atom types** - include the Amber/GAFF atom types assigned by [Add Charge](#) (or command [addcharge](#)) instead of [Sybyl atom types](#); if any atom in the [chosen](#) models lacks an Amber/GAFF type assignment, file-writing will halt and an error message will be displayed. *Mol2 files with Amber/GAFF types are not suitable for reading back into Chimera, as the types are likely to be misinterpreted.*
- **Write current selection to @SET section of file** - make the currently [selected](#) atoms a SET (as used to specify the rigid portion of a ligand in [DOCK](#))

## Saving Maps after Fitting

Simply saving a map to a file after [fitting](#) will not save its new orientation, because the map file formats do not include rotation/translation information. One way to save the new orientation is to save a Chimera [session](#), which preserves the orientations of all models. However, if data files are needed for use in another program:

- When a map has been fit to an atomic structure, [save the PDB file](#) relative to the map rather than *vice versa*.
- When a map has been fit to another map, a new map can be created in the new orientation by resampling the map that was fit, on the grid of the map to which it was fit. Resampling can be performed with the command [vop resample](#). Resampling uses interpolation and may cause some loss of resolution. The new, resampled map can be saved as described [above](#).

## Chimera Sessions

A Chimera *session* (the state of Chimera during use) can be [saved](#) and [restored](#). A session file consists of Python code that reconstitutes [most aspects](#) of Chimera by displaying data and performing other operations.

Molecular coordinate and sequence alignment data are included in the session file; however, [volume data](#) files must still be present to restart a session in which they were open. When a session with volume data is restarted, the user may be queried about file location(s) if the data or the session file have been moved since the session was saved.

Ways to save a session:

- [File... Save Session](#)
- [File... Save Session As...](#) (allows saving an associated thumbnail image and [session notes](#))
- the command [save](#)

Ways to restart a session:

- [File... Restore Session...](#) (clicking a session name in the file browser shows its associated thumbnail image and [session notes](#), if any)
- [File... Open...](#) ([file type](#) Python)
- the command [open](#) ([file type](#) Python)

A session may not restore correctly, or at all, in a version of Chimera older than that used to create the session. If a session includes a structure that has duplicate atom names within the same residue, it will not restore correctly.

Restoring a session creates a *compiled version of the file* (binary) with the same name except **\*.pyc** instead of **\*.py**. The binary speeds up session restoration and will be used (if present) even when the \*.py file has been specified. Further, opening the \*.pyc file directly will start the session even if the \*.py file has been deleted. However, keeping the \*.py file is recommended:

- the \*.py file will work with subsequent versions of python, whereas the \*.pyc file will only work with the same version of python that generated it
- the \*.py file can be viewed and edited with a text editor, whereas the binary \*.pyc file cannot
- while \*.py files are transferable among different types of computers, \*.pyc files generated on one system may not be read correctly on another

## Merging Sessions

If models are already open when a session is being restored, the user will be asked whether the pre-existing models should be closed. If not, the sessions will be merged; data from the incoming session will be opened, but its environmental settings such as background color and [effects](#) will not be applied.

When sessions are merged, models in the incoming session will be assigned new ID numbers and

transformed so that the model with the lowest ID in the incoming session will have the same transform as the pre-existing lowest-ID model. If these transformations are not as desired, they can be adjusted:

- [matrixcopy](#) can be used to apply the transformation of one model to another
- [reset](#) can be used to restore default or previously saved positions

Regardless of whether pre-existing models are closed when a session is restored, any pre-existing [2D labels](#) and [color key](#) will be removed, while pre-existing [sequence alignments](#) will be retained.

## What Sessions Include

The following are saved in session files:

- display status, colors, and [styles](#) of atoms, bonds, [pseudobonds](#), [ribbons](#), and [molecular surfaces](#)
- atomic coordinates, including alternate locations and changes such as from [bond rotation](#)
- PDB headers and residue secondary structure assignments
- atomic B-factor (isotropic and anisotropic), occupancy, [radius](#), and [charge](#) values
- [attributes](#) created with [Define Attribute](#)
- [Nucleotides](#) and [Thermal Ellipsoids](#) representations
- [Structure Measurements](#) including axis, plane, and centroid representations
- trajectory frames that had been [read in](#) prior to saving, and if all frames, the [MD Movie controller](#) dialog
- other model types: [\(nonmolecular\) surface](#), [volume and VRML](#) (except a VRML model opened from a file that is then moved before the session is saved)
- sequence data in [Multalign Viewer](#), including regions, associations, and trees
- standard labels, status of [2D Labels](#), [Color Key](#), and [Scale Bar](#)
- window size, model translation/rotation, and global scale
- [clipping plane](#) status, locations, and orientations
- saved [positions](#)
- saved [scenes](#) and [Animation](#) timeline contents
- color definitions (see [colordef](#)) and aliases (see [alias](#))
- [selections](#), current and saved (except of [surface models](#))
- viewing parameters: [depth cueing](#), [subdivision](#), [silhouettes](#), [projection mode](#), [center of rotation method](#), [lighting](#), [shininess/brightness](#)
- [camera mode](#) and [stereo parameters](#) (however, the camera mode will not be restored if it is [sequential stereo](#), or if Chimera was already in [sequential stereo](#) prior to session restoration)
- [Blast Protein](#) results, [ModBase Model List](#), [CASTp Pocket List](#)
- status, information, and/or models from [Cage Builder](#), [Color Zone](#), [Icosahedron Surface](#), [Metal Geometry](#), [Multiscale Models](#), [Rotamers](#), [Surface Capping](#), [Surface Color](#), [Surface Zone](#), [ViewDock](#), [Volume Tracer](#), [Volume Viewer](#)
- session thumbnail image and notes, if any, from using [Save Session As](#) and/or [Notepad](#)

Not included:

- the status of many other tools
- [preferences](#) settings (the [preferences file](#) can be changed between saving and restarting a session)
- which [preferences file](#) is being used (an issue when there are different preferences files in

different locations and a session file has been moved from its original location)

---

UCSF Computer Graphics Laboratory / February 2013

## Preferences File

A Chimera preferences file contains settings from the [Preferences dialog](#) and additional information such as recently accessed files and directories.

A particular preferences file can be specified with [--preferences](#) at [startup](#). Otherwise, Chimera will look for a file named **preferences** first in the current directory, then in a **.chimera** subdirectory of the current directory, then in the **.chimera** subdirectory of the user's home directory. On Windows, what is considered the home directory in this context depends on a user's profile, but is often **C:\Documents and Settings \username\Application Data\**. The file location is reported in the [Preferences preferences](#). Preference changes and other information from the current session will be saved to this file unless it has been designated [read-only](#).

This system allows a user to maintain a single set of preferences for all Chimera sessions, or different preferences specific to data in different directories, or a combination of these.

If no preferences file is found, Chimera will look for a writable location for a new preferences file in the following order: the user's home directory (a subdirectory **.chimera** with a **preferences** file within it will be created), a subdirectory **.chimera** of the current directory, the current directory. If none of these directories are writable, an error message will appear.

## Preferences Dialog

Choosing **Favorites...** Preferences from the menu opens the **Preferences** dialog. Options within are grouped by **Category**:

- [Animation](#)
- [Background](#)
- [Command Line](#)
- [Fetch](#)
- [General](#)
- [Image Credits](#)
- [Labels](#)
- [Messages](#)
- [Mouse](#)
- [New Molecules](#)
- [New Surfaces](#)
- [PDB](#)
- [POV-Ray Options](#)
- [Preferences](#)
- [Presets](#)
- [Selection](#)
- [Tools](#)
- [Web Access](#)

For just the [category](#) being shown: **Reset** replaces the current settings with the original defaults, **Restore** replaces the current settings with those previously saved in the [preferences file](#), and **Save** saves the current settings to the [preferences file](#). Equivalent operations can be performed for [all categories](#) at once in the [Preferences section](#) of the [Preferences dialog](#).

**Help** opens this manual page in a browser window, and **Close** dismisses the [Preferences dialog](#).

Default settings are indicated below in **bold**.

## ← Animation

The **Animation** section of the [Preferences dialog](#) is only available after a [scene](#) has been saved or the [Animation](#) tool started. See also: [scene](#)

- **Scene name prefix** (default nothing) - text to prepend to the integer names of scenes saved with the [Animation](#) dialog
- **Scene thumbnail size** (default **48** pixels) - width and height of scene thumbnails in the [Animation](#) and [Rapid Access](#) interfaces
- **Transition name prefix** (default tr)
- **Warn about model closure** (**true**/**false**) - whether to show a warning and allow cancellation when the user attempts to close a model that is included in a saved scene

## ← Background

The **Background** section of the [Preferences dialog](#) includes settings for the [graphics window](#) background. See also: [background](#)

- **Background method** (**solid**/gradient/image) - whether the background should be a single solid color (default), a gradient of multiple colors, or an image
- **Background color** (a [color well](#), default **No Color**, equivalent to black) - color to use when the [background method](#) is **solid**
- **Background gradient** (a [palette well](#), default **HLS**-interpolated **white** to **blue**) - [palette](#) to use when the [background method](#) is **gradient**; left-to-right order in the palette well corresponds to bottom-to-top order in the graphics window
  - **opacity** (default **1.0**) - gradient opacity, ranging from 0.0 (completely transparent) to 1.0 (completely opaque); transparency reveals the current [solid background color](#). If a transparent gradient is used together with a [transparent background](#), the background of an image saved from Chimera as PNG or TIFF will not only show the blending of the gradient over the solid color, but also have the same overall opacity as was specified for the gradient.
- **Background image** (an image well, default **No image**) - image to use when the [background method](#) is **image**; clicking the well brings up a file browser for locating and opening an image file, where accepted formats include PNG, TIFF, and JPEG. The image can be fit into the graphics window in any of the following ways:
  - **zoomed** (default) - automatically scaled to fill the window, top/bottom or sides cropped as needed (no image distortion)

- **stretched** - automatically scaled to fill the window, stretched or squashed as needed
- **tiled** - tiled to fill the window
- **centered** - centered with any surrounding border shown in the current [solid background color](#)

Additional image settings:

- **scale** (default **1.0**) - image scale factor, applies only when the image is **tiled** or **centered**
- **opacity** (default **1.0**) - image opacity, ranging from 0.0 (completely transparent) to 1.0 (completely opaque); transparency reveals the current [solid background color](#). If a transparent background image is used together with a [transparent background](#), the background of an image saved from Chimera as PNG or TIFF will not only show the blending of the background image over the solid color, but also have the same overall opacity as was specified for the background image.

## ← Command Line

The **Command Line** section of the [Preferences dialog](#) is only available after the [Command Line](#) has been shown.

- **Files to read at startup** - what [command files](#), if any, should be [opened](#) when the [Command Line](#) is started. There are two defaults:
  - **\$HOME/.chimera/midasrc** (in a subdirectory named .chimera of the user's home directory; on Windows, what is considered the home directory in this context depends on a user's profile, but is often **C:\Documents and Settings\username\Application Data\**)
  - **.chmidasrc** in the current directory
 and either or both are executed (in the order listed), if found. Files can be added to or deleted from the list. Whether files are listed by name followed by path or with the path preceding the name is controlled in the [General preferences](#).

Startup files are especially useful for defining [aliases](#) and [color names](#).

- **Number of commands to remember between sessions** (default **60**) - length of command history to save to the [preferences file](#) when exiting from Chimera; the saved commands will be shown in the [Command History](#) in the next session that uses the same [preferences file](#).

## ← Fetch

The **Fetch** section of the [Preferences dialog](#) controls handling of data retrieved over the Web from various databases. See also: [open](#), [PDB preferences](#), [Fetch by ID](#)

- **Save fetched files (true/false)** - whether to save fetched files locally; even if this option is **true**, no files will be saved unless a download directory has also been specified
- **Use local files (true/false)** - whether to look in the download directory and use a previously fetched copy (if any) when [Fetch by ID](#) or the [open](#) command is used to fetch a file from a database; when true, can be overruled by using the [Fetch by ID](#) option to **Ignore any cached data**
- **Download directory** (default **~/Downloads/Chimera**) - where to store fetched files locally; subdirectories for the different databases will be created as needed. On Windows XP, the default is **~/My Documents/Downloads**. If the specified location does not exist, fetched files will not be cached.

## ← General

The **General** section of the [Preferences dialog](#) includes:

- **Confirm quit** (always/conditional/never) - when to ask for confirmation before exiting Chimera; if **conditional**, only when certain tools detect unsaved work. If **always** or **conditional**, a conditional check will also apply to closing a session. Regardless of this setting, confirmation will not be requested when the command [stop](#) is encountered within a [command file](#).
- **File lists path style** (**file - leading path/full path**) - how to list files and their locations in certain dialogs (such as **Files to read at startup** within the [Command Line preferences](#)); a change will not be evident until after Chimera is restarted
- **Fullscreen graphics** (true/false) - whether the graphics window is in [fullscreen mode](#) (available in Windows, Linux, and Mac-X11 versions of Chimera)
- **Try to avoid placing dialogs over main window** (true/false) - whether to try to place additional Chimera dialogs alongside the graphics window instead of on top of it
- **File browser confirms overwrite** (true/false) - whether to ask for confirmation before overwriting a file with a [save dialog](#)
- **Use short open/save browser** (true/false) - whether [open/save dialogs](#) should show only the two lowest directory levels ([more details](#))
- **Open dialog starts in directory from last session** (true/false) - whether the first [open/save dialog](#) invoked during a session should start in the directory where a file was last opened or saved in the previous session (**true**, default on Windows and Mac). Otherwise (**false**, default on Linux), it will start in the current working directory.
- **Debug OpenGL on startup** (true/false) - whether to open [Debug Graphics Driver](#) at startup to selectively enable/disable OpenGL features before starting Chimera. This should be used only to find workarounds for [graphics driver bugs](#), because in the absence of such problems, changing the settings from the system defaults is expected to degrade Chimera performance and/or appearance. [Debug Graphics Driver](#) can also be called with the startup option `--debug-opengl`.
- **Update check interval** (never/daily/weekly/biweekly/monthly/quarterly) - how often to check whether a newer version of Chimera (snapshot or production release) is available
- **Initial window size** (**remember last/fixed**) - how to determine the size of the window at startup
- **Fixed size** - pixel width and height for fixed [initial window size](#)

In the **Mac** version of Chimera only:

- **Menus in windows on Mac** (true/false) - whether to show Chimera menus in-window as well as across the top of the screen

## ← Image Credits

The **Image Credits** section of the [Preferences dialog](#) specifies information to be included in saved image files (TIFF, PNG). This information does not affect the image itself.

- **Artist** (user's login name by default) - optional, name of the image creator
- **Copyright** (blank by default) - optional, a copyright statement

## ← Labels

The **Labels** section of the [Preferences dialog](#) contains settings for 3D labels (those which move along with the structure) and pop-up atomspec balloons. The method of [residue label positioning](#) can be set in the [molecule model attributes panel](#) or with the command [setattr](#). See also: [2D Labels](#)

- **Label font (Sans Serif/Serif/Fixed) (Normal/Bold/Italic/Bold Italic)** (6/8/9/10/11/12/16/18/24/30/36) - font typeface, style, and size to use for 3D labels
- **Labels on top (true/false)** - whether to draw 3D labels in front regardless of their Z-offsets
- **Show atomspec balloon (true/false)** - when the cursor is paused over an object (atom, bond, surface, *etc.*), whether to show the corresponding label information in a balloon in the graphics window, and PDB chain information, if available, in the [status line](#). The number of decimal places shown for bond distances can be set with the [Distances](#) tool.
- **Atomspec display style (simple/command-line specifier/serial number)** - how to list atoms in [atomspec balloons](#), [Structure Measurements](#), and [Metal Geometry](#). Atom naming styles:
  - **simple** - residue name, residue specifier, and atom name (for example, **HIS 16.A ND1**)
  - **command-line specifier** - a [specification string](#) that could be used in the [Command Line](#) (for example, **:16.A@ND1**)
  - **serial number** - atom serial number (for example, **126**)
 Model number (for example, **#0**) will also be included when multiple models are present.

## ← Messages

The **Messages** section of the [Preferences dialog](#) controls the disposition of various messages and whether balloon help is shown.

- **Show status line (true/false)** - whether to show a [line for status messages](#) under the main Chimera window (otherwise, status messages are not visible); can also be set from the [Accelerators dialog](#)
- **Clear status line after** (5 seconds/10 seconds/20 seconds/**30 seconds**/1 minute/never) - how long a status message will be shown in the status line (when not overwritten by subsequent status messages); applies to status messages appearing in certain dialogs as well as the main Chimera status line
- **Show balloon help (true/false)** - whether to show balloon help, explanatory text that pops up when the cursor is left in certain positions
- **Command (reply log only/dialog)** - whether command messages should appear in the [Reply Log](#) only, or also generate a dialog
- **Warning (reply log only/dialog)** - whether warnings should appear in the [Reply Log](#) only, or also generate a dialog
- **Error (reply log only/dialog)** - whether error messages should appear in the [Reply Log](#) only, or also generate a dialog

## ← Mouse

Mouse button functions including the [manipulation of models](#) can be reassigned using the **Mouse** section of the [Preferences dialog](#). Functions may be assigned to the left, middle, and right mouse buttons alone and in combination with the **Ctrl** key. (Modifier keys allow accessing these functions with a [touchpad or](#)

[simpler mouse](#).) There can be no more than one function checked per row (per button or Ctrl-button).

icon	meaning	default assignment	icon	meaning	default assignment
	rotation	1 (left button)		<a href="#">picking</a> (selection from screen)	Ctrl-1
	XY-translation	2 (middle button)		Z-translation	Ctrl-2
	scaling	3 (right button)	center	click item to center, set as <b>fixed</b> <a href="#">center of rotation</a> ; click empty space to restore <b>front center mode</b>	Ctrl-3
	pop-up <a href="#">Chimera menu</a>	(none) see <a href="#">alternatives</a>	Label	label-dragging (XY, + <b>Shift</b> for Z)	(none)

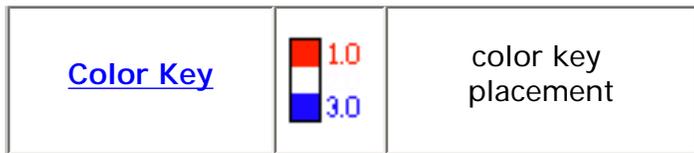
- **Continue rotation after mousing** (true/false) - whether rotation should continue when the corresponding mouse button is released while the cursor is in motion
- **Use scrolling** (true/false) - whether mouse or touchpad scrolling should also scale the view. On a Mac, it may be necessary to set the [multitouch preference](#) to **false** to make touchpad scrolling available.

In the **Mac** version of Chimera only:

- **Use multitouch gestures on Mac** (true/false) - whether to respond to multitouch gestures on a Mac touchpad (see [multitouch actions](#))

Some tools create additional mouse modes, for example:

tool	icon	meaning
<a href="#">Adjust Torsions</a>	rotate bond	bond rotation
<a href="#">2D Labels</a>		2D label placement



## ← New Molecules

The **New Molecules** section of the [Preferences dialog](#) controls how subsequently opened molecule models will initially appear. **New Molecules** settings can be circumvented by opening structures with the command [open noprefs](#); this prevents inconsistent behavior of [command files](#) and [demos](#) potentially caused by different preference settings of different users. By default, these preferences are not applied to structures opened via [Chimera web data \(chimerax\)](#) files, but `noprefs="false"` can be set in the chimerax file to indicate that the preferences should be applied. Regardless of `noprefs` options, however, the preferences for metal complex depiction and mol2 naming are always applied. See also: [open](#), [molecule model attributes](#), [presets](#)

- **smart initial display** (true/false) - show subsequently opened molecule models with settings similar to the **ribbons preset**, but without rainbow-coloring chains or altering global parameters such as [background color](#). Several other **New Molecules** preferences are overridden and grayed out when **smart initial display** is set to true:
  - **ribbon display** (off/on) - whether to show ribbon; ribbons are only drawn for proteins and nucleic acids. Protein helix and strand assignments are taken from the input structure file or generated with [ksdssp](#).
  - **ribbon cross section** (edged/flat/rounded/any additional [styles](#) made with the [Ribbon Style Editor](#)) (see also [ribrepr](#))
  - **ribbon scaling** (Chimera default/licorice/any additional [scalings](#) made with the [Ribbon Style Editor](#)) (see also [ribscale](#))
  - **ribbon hides backbone atoms** (true/false) - whether ribbon display hides backbone atoms for the corresponding residues (see the command [ribbackbone](#) for details)
  - **show atoms** (true/false) - whether to display all atoms/bonds (except backbone when hidden by ribbon)
  - **per-atom coloring** (none/by element/by heteroatom) - whether to leave atom colors unassigned (**none**; atoms will inherit [model colors](#)) or to assign [element colors](#) to all atoms (**by element**) or only the non-carbon atoms (**by heteroatom**)
  - **atom style** (dot/ball/endcap/sphere) - atom [draw mode](#) (excluding [monatomic ions](#))
  - **bond style** (wire/stick) - bond [draw mode](#)
  - **auto-chaining** (off/on) - whether to connect atoms that precede and follow undisplayed segments (whether to draw [pseudobonds](#) between them)
- **use new color for each model** (true/false) - set the model-level color (see [coloring hierarchy](#)) differently for each new molecule model; [submodels](#) with the same main model number will be assigned the same model-level color. This setting can also be controlled with [set/~set autoColor](#). If the solid [background color](#) is **black** or **white**, automatic model colors are **0:tan**, **1:sky blue**, **2:plum**, **3:light green**, **4:salmon**, **5:light gray**, **6:deep pink**, **7:gold**, **8:dodger blue** and **9:purple**

(see [named colors](#)):



Thereafter, or for all models if the solid [background color](#) is not **black** or **white**, an algorithm is used to produce colors distinguishable from the background and from other models.

- **otherwise, use color** (a [color well](#), by default a light gray) - model-level color if not using a new color for each model
- **stick scale** (default **1.0**) - scale factor for bonds in the **stick** [draw mode](#). The **stick scale** is multiplied by individual bond radii (default **0.2** angstroms) to generate stick radii in angstroms. Changing **stick scale** is preferable to changing all of the bond radii in a model, because the former will also scale singleton atoms in the **endcap** [draw mode](#) appropriately. Either way, the other **endcap** atoms (those participating in bonds) will be scaled to match the thickest of the attached bonds.
- **ball scale** (default **0.25**) - scale factor for atoms in the **ball** [draw mode](#). The **ball scale** is multiplied by individual atom [VDW radii](#) to generate ball radii in angstroms.
- **line width** (default **1.0**) - pixel width of lines depicting bonds (when in the **wire** draw mode)
- **monatomic ion style** (dot/ball/endcap/sphere) - [draw mode](#) to use for monatomic [ions](#)
- **metal complex representation** (dashed lines/solid lines/springs) - how to show metal coordination [pseudobonds](#)
- **metal complex color** (a [color well](#), **medium purple** by default) - color to use for metal coordination [pseudobonds](#)
- **metal complex line width** (default **2.0**) - pixel width of lines representing metal coordination [pseudobonds](#)
- **Mol2 model naming** (file name/Mol2 molecule name/Mol2 molecule comment) - how to name models read from Mol2 format: by file name, by molecule name given within the file (on the line below @<TRIPOS>MOLECULE), or by comment text (the sixth line below @<TRIPOS>MOLECULE)

## ← New Surfaces

The **New Surfaces** section of the [Preferences dialog](#) sets parameters for subsequently created [molecular surfaces](#). See also: [surface](#), [molecular surface attributes](#), [presets](#)

- **representation** (solid/mesh/dots) - display type
- **probe radius** (default **1.4**) - radius in Å of the probe sphere used to compute the surface. A larger probe decreases surface bumpiness because it fits into fewer crevices. A radius of 1.4 Å is commonly used to approximate a water molecule.
- **vertex density** (default **2.0**) - vertices per Å<sup>2</sup>. Greater density results in a smoother surface but increases computational demands for calculating and moving the surface.
- **line width** (default **1.0**) - pixel width of lines used in the mesh [surface representation](#)
- **dot size** (default **1.0**) - pixel size of dots used in the dot [surface representation](#)
- **show disjoint surfaces** (**true/false**) - whether to check for multiple disconnected parts of a surface ([components](#)) rather than assuming there is only one; increases calculation time

## ← PDB

The **PDB** section of the [Preferences dialog](#) includes:

- **Fetch from web as necessary (true/false)** - whether to retrieve a file specified by PDB ID from the [Protein Data Bank web site](#) when it is not found locally (see the [Web Access preferences](#) for proxy settings, if needed). Chimera first looks for a local installation of the Protein Data Bank, then in any [personal PDB directories](#), and then in the [fetch download directory](#) (unless disallowed in the [Fetch preferences](#)) before resorting to a web fetch. The default places to look for a local installation are `/usr/mol/pdb/` and then `/mol/pdb/` — these can be changed by editing the Python file `share/chimera/pdbDir` within a Chimera installation.
- **Personal PDB directories** - additional directories in which to look for PDB files. Directories can be added to or deleted from the list. The directory structure can be flat, where a single directory contains files named `xxxx.pdb` (and `xxxx` is the PDB ID), or hierarchical, with pathnames of the form `xy/pdb1xyz.ent`, where `1xyz` is the PDB ID and subdirectory names are the two middle characters of the IDs. PDB filenames of the form `pdb1xyz.ent` can also be used in the flat directory structure.

## ← POV-Ray Options

The **POV-Ray Options** section of the [Preferences dialog](#) sets specifications for [raytracing with POV-Ray](#).

- **POV-Ray executable [ location ]** - the location of the POV-Ray executable. This is filled in automatically for the executable included with Chimera, but another location could be specified.
- **Show preview (true/false)** - whether to show a preview; regardless of this setting, however, a preview will not be shown during [movie recording](#) or when Chimera is in [nogui mode](#).
- **Quality** (default **9**, range 0-11) - output image quality, adjusted by omitting calculations that are normally performed. For example, values  $< 4$  yield images without shadows [[quality details](#) at the POV-Ray site].
- **Antialias (true/false)** - whether multiple samples should be averaged to produce a single output pixel; otherwise, POV-Ray will simply trace one ray per pixel. Analogous to (but independent of) [Chimera image supersampling](#), this supersampling in POV-Ray smooths edges in output images [[antialias details](#) at the POV-Ray site].
- **Antialias method (fixed/recursive)** - how pixels should be supersampled when [antialiasing](#) is used. In the **fixed** method, POV-Ray initially traces one ray per pixel. Each pixel is compared to its neighbors to the left and above. If the colors of two pixels differ by more than the [antialias threshold](#), each of the pixels will be subsampled according to the [antialias depth](#) (if the depth is 3, a grid of  $3 \times 3 = 9$  samples will be taken per output pixel). In the **recursive** method, POV-Ray initially traces 4 rays at the corners of each pixel. Where the colors of adjacent samples differ by more than the [threshold](#), subsamples will be taken and compared. This process is repeated until adjacent colors agree within the threshold, or the maximum number of subdivision cycles (the specified [antialias depth](#)) has been performed.
- **Antialias depth** (default **3**, range 1-9) - meaning depends on which [antialias method](#) is used
- **Antialias threshold** (default **1.0**, range 0.0-3.0) - two colors are compared by summing the absolute differences in their red, green, and blue components; differences greater than the threshold trigger further sampling when [antialiasing](#) is used. Lower threshold values increase the number of samples and the calculation time, whereas a threshold of 3.0 will suppress further sampling.
- **Jitter (true/false)** - whether to introduce noise by randomly wiggling the supersample locations when [antialiasing](#) is used; not recommended for [movie recording](#), as the random noise will make pixels vary and flicker from frame to frame [[jitter details](#) at the POV-Ray site]
- **Jitter amount** (default **0.5**) - values are specified relative to 1.0, which is the most [jitter](#) that can be performed while retaining supersamples within the original pixel

- **Transparent background** (true/false/same as chimera) - whether to make the background transparent by including transparency (alpha) values for output pixels [[output alpha details](#) at the POV-Ray site]. The **same as chimera** option means to follow the current [transparent background](#) setting in the [Save Image dialog](#). It is also necessary to turn off [depth-cueing](#) to obtain background transparency in raytraced images.
- **Wait for POV-Ray to finish** (true/false) - whether to prevent further commands from executing until POV-Ray has finished generating the image. During [movie recording](#), the behavior will always be as if this were set to **true**.
- **Keep POV-Ray input files** (true/false) - whether the POV-Ray input files containing the scene (\*.pov) and raytracing options (\*.ini) should be kept or deleted after the image has been rendered

## ← Preferences

The **Preferences** section of the [Preferences dialog](#) includes:

- **Preferences file** - the [location](#) of the [preferences file](#) for the current session
- **Read-only** - whether the [preferences file](#) for the current session should be treated as read-only. The initial setting reflects the file permissions at Chimera startup. When **Read-only** is **false**, Chimera may update the file with changes in preference settings and other information; switching to **true** prevents subsequent changes in the file. When **Read-only** is **true**, Chimera will not update the file; switching to **false** (if the file is already writable, or is owned by the user) will enable its modification by Chimera.
- **All categories:**
  - **Restore to saved settings** - replace the current settings in [all categories](#) with those previously saved in the [preferences file](#)
  - **Reset to factory defaults** - replace the current settings in [all categories](#) with the original defaults
  - **Save** - save the current settings in [all categories](#) to the [preferences file](#)

## ← Presets

The **Presets** section of the [Preferences dialog](#) allows specifying **Personal preset directories** in which Chimera should look for custom preset scripts. Directories can be added to or deleted from the list. Preset script file types:

- [Chimera commands](#) (\*.com, \*.cmd)
- Python code (\*.py)

Custom presets will appear in the [Presets menu](#) in alphabetical order by name, which is determined from script filename: *name.preset.suffix* where *suffix* indicates the file type, as listed above. Custom presets can also be accessed with the [preset](#) command.

## ← Selection

The **Selection** section of the [Preferences dialog](#) controls how [selections](#) are highlighted.

- **Selection highlight method (outline/fill)** - how a [selection](#) is highlighted in the graphics window
- **Selection highlight color** (a [color well](#), green by default) - what color to use for highlighting a [selection](#) in the graphics window

## ← Tools

The **Tools** section of the [Preferences dialog](#) allows control over the following **Settings** for each entry in the [Tools menu](#):

- **Auto Start** - whether to launch the tool automatically upon [Chimera startup](#)
- **On Toolbar** - whether to include the tool icon in the [toolbar](#)
- **In Favorites** - whether to include the tool in the [Favorites menu](#)
- **Confirm on Start** - whether to ask for confirmation before starting the tool

The [toolbar](#) is shown when it includes at least one icon.

- **Toolbar placement (left/right/top/bottom/Rapid Access)** - whether to put the toolbar along one side of the Chimera graphics window, or only in the [Rapid Access](#) interface

The **Locations** section allows users to specify additional directories in which Chimera should look for tools. Directories can be added to or deleted from the list.

## ← Web Access

The **Web Access** section of the [Preferences dialog](#) includes:

- **Confirm open of commands or code (once per session/each time/never)** - whether/how often to [ask for confirmation](#) before opening certain types of [data linked to web pages](#).
- **Accept web data (true/false)** - whether the running instance of Chimera can be used as a helper application for [data linked to web pages](#) (and can receive files specified with `chimera --send`). On **Mac**, this preference is not shown, but is effectively **true** unless Chimera was started from the terminal command line. A browser must be [configured](#) to send the files to Chimera. If there is no running instance of Chimera enabled to accept web data, another instance of Chimera will be started and used to open the data. If there are multiple running instances of Chimera set to accept web data, the file will be sent to the instance that most recently had focus (was most recently clicked into).
- **Use HTTP proxy (true/false)** - whether to use a proxy to connect to the internet through a firewall (for example, to fetch a PDB file from the [Protein Data Bank web site](#)); the server and port number settings (below) should generally be the same as those used for web browsers
- **Proxy server** - the hostname or IP address of the proxy server
- **Proxy port** - proxy port number to use

# Help

Extensive help is available for Chimera; the [User's Guide](#) and [Programmer's Guide](#) are written in HTML and can be displayed using the web browser of choice. This HTML documentation is bundled with a Chimera download and can be accessed from the Chimera **Help** menu. Use of the bundled documentation is recommended because it is synchronized with the locally installed version of Chimera. (The development version of the **User's Guide** is [available online](#) for viewing, but may describe features that have not yet been released.)

**Help** menu contents:

- **Search Documentation...** [search](#) the local (bundled) [Chimera documentation](#)
- **User's Guide** - open the [Chimera User's Guide](#)
- **Commands Index** - open the Chimera [Commands Index](#)
- **Tutorials** - open the [Tutorials](#) section of the **Chimera User's Guide**
- **Context Help** - describe the feature that is clicked next (avoid clicking the top bar of a dialog, since that will fail to bring up a help page)
- **Check for Updates** (requires internet connectivity) - see if any production releases are newer than the version in use, and if so, provide links to download them
- **Contact Us** - show [e-mail addresses](#) for asking questions or making suggestions
- **Report a Bug...** open a [form for bug submission](#)
- **Citation Info** - show [how to cite usage](#) of Chimera
- **Registration...** open a form for registration as a Chimera user
- **About UCSF Chimera** - report what version of Chimera is being used and show copyright information

To call up the manual page for a specific [command](#), type into the [Command Line](#):

[help](#) *command*

whereas the following will show the entire [Commands Index](#):

[help](#) commands

The [help](#) command without any arguments calls up the [documentation search](#) dialog.

Many of the tools and interfaces have **Help** buttons that open their respective manual pages. Some tools and interfaces also supply *balloon help*, explanatory text that pops up when the cursor is paused over some relevant part of a dialog. Whether to show balloon help can be set in the [Messages preferences](#).

When the graphics window has mouse focus, pausing the cursor over an atom or bond (without clicking any buttons) shows an *atomspec balloon* with information such as atom name, residue number, and bond length. The style of atomspec balloons and whether they should be shown can be set in the [Labels preferences](#).

## Stereo

The *camera mode* refers to any of several stereo and mono viewing options. The camera mode of the Chimera graphics window can be controlled with the command [stereo](#), the [Camera](#) tool, or the menu raised by double-clicking the eye position in the [Side View](#). It can also be specified at Chimera startup with [--stereo](#). The [Image type](#) (camera mode for output image files) can differ from the mode shown in the graphics window.

Available camera modes:

- **mono** - standard single view, not in stereo
- **stereo left eye** - single view from the left-eye position
- **stereo right eye** - single view from the right-eye position
- **cross-eye stereo** - side-by-side views, with the left-eye view on the right and the right-eye view on the left
- **wall-eye stereo** - side-by-side views, with the left-eye view on the left and the right-eye view on the right
- **red-cyan stereo** (also known as **anaglyph**) - overlapping left-eye and right-eye views in different colors, to be viewed with “glasses” (often inexpensive cardboard/plastic) with colored filters. The red channel is used for the left-eye view, green and blue for the right-eye view. This setting works best with red-cyan glasses, but reasonably well with red-blue or red-green glasses. Some ghosting will always be present unless the glasses are perfectly matched to the frequencies of light emitted by the display. Stereo cues will be greater when colors with a balance of red and blue/green components are used, for example, grayscale, magenta, and yellow rather than red, green, blue, or cyan. The red, green, and blue component values of a color can be viewed by entering its [name](#) in the [Color Editor](#).
- **green-magenta stereo** (also known as **trioscopic**) - similar to red-cyan stereo, but with green and magenta instead of red and cyan, and different trade-offs. From [Wikipedia](#): “better reds, oranges and wider range of blues than red/cyan” and “less chromatic aberration.”
- **sequential stereo** - rapid flickering between left-eye and right-eye views, to be viewed with special synchronised glasses. Sequential stereo needs [special hardware](#).
- **reverse sequential stereo** - as above, but swapping the two views to accommodate devices that use the opposite convention
- **row stereo, right eye even** - row-interleaved stereo, with even rows used for the right-eye view and odd rows used for the left-eye view. Row-interleaved stereo is used by 3D displays from [Miracube](#), [Zalman](#), and others. Users of these devices can try both row stereo options in Chimera to see which gives the best result.
- **row stereo, right eye odd** - as above, except swapping the two views to accommodate devices that use the opposite convention
- **dome** - angular fisheye of the hemisphere in front of the camera, with [horizontal field of view](#)

locked to 90°

- **truncated dome** - same as the dome mode, except with the bottom of the hemisphere cut off
- **DTI side-by-side stereo** - for stereo viewing with [DTI](#) technology

# Keyboard Shortcuts

## Main Window

The [Keyboard Shortcuts](#) extension defines many simple shortcuts (*accelerators*) for operations in Chimera. When the cursor is in the main Chimera window, character keystrokes may be interpreted as [accelerators](#), [commands](#), or information intended for another dialog (although focus can be restored to the main window by mouseclick). An accelerator takes effect as soon as it is typed, whereas a command takes effect after return (**Enter**) is pressed. See the [Keyboard Shortcuts](#) page for more on [keystroke interpretation](#).

Several [arrow key shortcuts](#) adjust the Chimera [selection](#) contents.

**Tab** (and often **Shift-Tab**) can be used to navigate among user interface elements, including icons in the [toolbar](#). Pressing the **space bar** is equivalent to clicking the highlighted element.

## Menu Shortcuts

Menu shortcuts are not available in the native (non-X11) Mac version of Chimera.

A top-level menu with an underlined character in its name can be opened by pressing the **Alt** key and the character. (Exception: this does not work in the Mac/X11 version of Chimera, but the other shortcuts described below will work in menus that have been [torn off](#).) Entries in the open menu with an underlined character can be chosen by pressing that character. For example, **Alt-f** opens the [File menu](#), and subsequently, **c** invokes **Close Session**. The **arrow** keys can be used to navigate within the menus. Pressing the **space bar** is equivalent to a mouseclick (chooses the highlighted menu item or, if no item is highlighted, closes the menu).

The entire Chimera menu can also be [popped up](#).

## Dialogs

When the mouse focus is on a particular dialog: **Tab** (and often **Shift-Tab**) can be used to navigate among user interface elements, **arrow** keys can be used to navigate within lists and menus, and the **space bar** acts like a mouseclick. For example, when the cursor is over the [Preferences dialog](#), the category can be changed by pressing **Tab** until the button next to **Category** is highlighted, pressing **space bar** to open the menu of categories, using the **arrow** keys to navigate to the desired category, and then pressing **Enter** (or the **space bar**, except on Windows) to choose the category. The **Esc** (escape) key can be used to close the menu of categories without choosing another category.

**F1** invokes the help page for a dialog, if available. Most dialogs can be closed with the **Esc** key. Where possible, **Esc** is associated with the **Cancel** action. Some dialogs have a default action, indicated by highlighting of the corresponding button, that can be invoked by hitting return (**Enter**).

On a Mac, the mouse focus can be cycled among the Chimera dialogs (including the main window) using **⌘-`**, and the non-X11 version also allows using **Shift-⌘-`** to cycle in the opposite direction.

## Text-Editing Shortcuts

Chimera uses the Tk [default bindings](#). Some shortcuts for editing text depend on the windowing system:

X11		
Action	Key	Alternate
Cut	Ctrl-x	F20
Copy	Ctrl-c	F16
Paste	Ctrl-v	F18
PasteSelection	Button 2 Release	
Undo (if implemented)	Ctrl-z	
Redo (if implemented)	Ctrl-Shift-Z	

win32		
Action	Key	Alternate
Cut	Ctrl-x	Shift-Delete
Copy	Ctrl-c	Ctrl-Insert
Paste	Ctrl-v	Shift-Insert
PasteSelection	Button 2 Release	
Undo (if implemented)	Ctrl-z	
Redo (if implemented)	Ctrl-y	

aqua		
Action	Key	Alternate
Cut	⌘-x	F2
Copy	⌘-c	F3
Paste	⌘-v	F4
PasteSelection	Button 2 Release	
Clear	Clear	
Undo (if implemented)	⌘-z	
Redo (if implemented)	⌘-y	

## Keyboard Shortcuts

The **Keyboard Shortcuts** extension allows simple [keyboard shortcuts](#) (*accelerators*) to be used for many operations in Chimera. See also: [general keyboard shortcuts](#)

Use of these shortcuts is disabled by default, but can be turned on:

- with the command [ac](#)
- by starting **Keyboard Shortcuts** (for example, with **Tools... General Controls... Keyboard Shortcuts**) and turning on **Enable keyboard shortcuts** in the [dialog](#)

The shortcut mode can be disabled:

- with the [accelerator](#) `af`
- with the [accelerator](#) `cl` (also shows the [Command Line](#))
- by turning off **Enable keyboard shortcuts** in the [dialog](#)

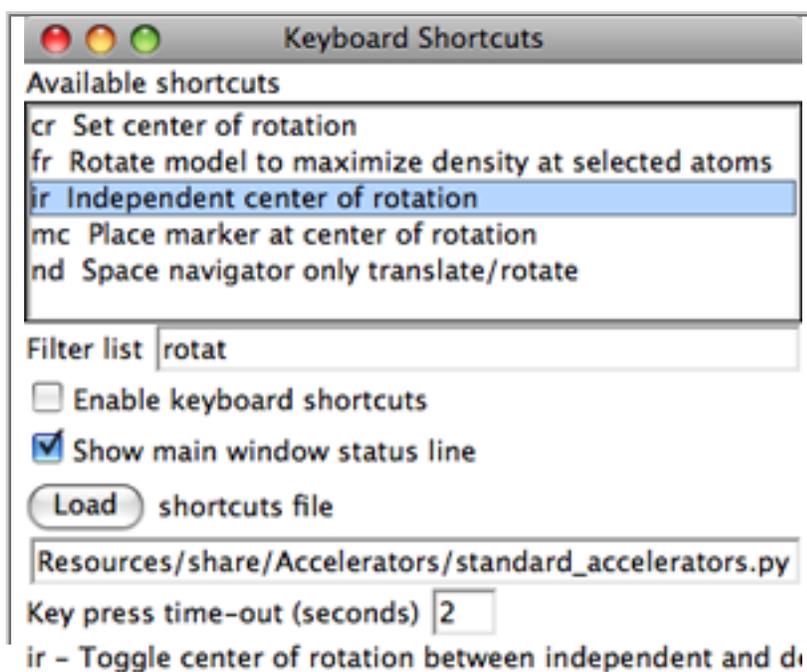
Checking/unchecking **Enable keyboard shortcuts** changes the [preferences file](#) to automatically enable/disable the shortcut mode when Chimera is next started. Other methods of turning accelerators on or off do not alter the preferences.

[Many accelerators](#) are included with Chimera, and more can be [added](#). They usually consist of two characters, but any number of characters is possible. Uppercase letters are sometimes used and are distinct from lowercase. Several provide capabilities not accessible through the graphical interface. Keyboard shortcuts are not entered in the [Command Line](#) except via the command [ac](#).

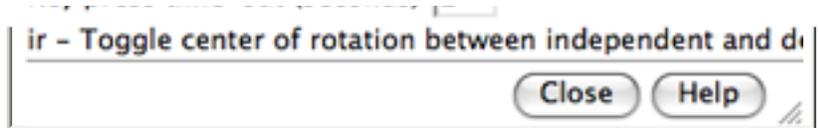
## Keyboard Shortcuts Dialog

There are [several ways to start Keyboard Shortcuts](#), a tool in the **General Controls** category. The resulting dialog lists the known accelerators along with brief descriptions. Clicking on an entry in the list shows a more detailed description (if available) at the bottom of the dialog.

- **Filter list** limits the entries to those containing the entered text.
- **Enable keyboard shortcuts** controls whether accelerators are enabled.
- **Show main window status line** controls whether a [status line](#) is shown under the Chimera window. Keystrokes interpreted as accelerators and messages reporting their effects appear in the status line.



- **Load shortcuts file** reads accelerator definitions from the Python file whose pathname is shown in the field below. New keyboard shortcuts can be [added](#) by putting their definitions in a Python file and then using **Load**.
- **Key press time-out (seconds)** (default 2) controls how long keystrokes that form part of an accelerator are remembered.



Settings in the dialog are saved in the [preferences file](#), except that the status of **Enable keyboard shortcuts** is only saved when the option is manually checked/unchecked by the user.

## Accelerator or Command?

Keystrokes may be interpreted as accelerators, as commands, or as information intended for other dialogs. Sometimes it is necessary to click in the graphics window to take back focus from some other dialog so that accelerator or command keystrokes will register.

An accelerator takes effect as soon as it is typed; there is no need to press return (Enter). A command takes effect after return is pressed. How these forms of input are distinguished:

- If accelerators are on and the **Command Line** has not been shown, keystrokes are interpreted as accelerators when the mouse focus is in the graphics window.
- To allow both types of input, accelerators should be turned on *after* the **Command Line** has been shown. If the **Command Line** is started after accelerators, it will receive all of the keystrokes, even if subsequently [hidden or quit](#). In this situation, to enable both types of input:
  - turn or toggle accelerators on again
  - use the *command* [ac](#)
- When both types of input are enabled as described above, one must click in the graphics window to indicate that keystrokes should be interpreted as accelerators and click in the command-line area to indicate that keystrokes should be interpreted as commands.

The accelerator **hc** hides the **Command Line** while the accelerator **cl** opens it and turns off accelerators.

## Time-outs and Entry Errors

Keystrokes being interpreted as accelerators are shown in the [status line](#); the [time-out](#) setting controls how long a partial accelerator entry is retained. If the wrong key has been pressed, simply waiting for the [time-out](#) period to elapse allows a fresh start. Alternatively, pressing the escape key clears the partial accelerator from memory. Finally, if no accelerator starts with the key that has been pressed, an "unknown accelerator" status message results and the keystroke is not retained. Completed accelerator operations are reported in the [status line](#) and [Reply Log](#).

## Adding Keyboard Shortcuts

The standard Chimera keyboard shortcuts are defined in

share/Accelerators/standard\_accelerators.py

within the Chimera installation directory. New accelerators can be added by writing their definitions in a Python file and reading the file with **Load** on the [Keyboard Shortcuts dialog](#).

Example file:

```
def register_accelerators():

    from Accelerators import standard_accelerators
    standard_accelerators.register_accelerators()

    from Accelerators import add_accelerator
    add_accelerator('ry', 'Rotate models about y axis', rotate_y)
    add_accelerator('rs', 'Stop model rotation', stop_rotation)

def rotate_y():
    'Spins models about the y axis'
    import Midas
    Midas.roll(axis = 'y')

def stop_rotation():
    'Stops spinning models'
    import Midas
    Midas.freeze()
```

The `register_accelerators()` function is called by the Accelerators extension. The first two lines load the standard accelerators. The next three lines define accelerators **ry** and **rs** to spin the models about the Y axis and to stop the spinning. The routines that spin and stop the spinning use the Chimera Midas module. The comment strings in the `add_accelerator` lines are the short descriptions that appear in the top part of the [Keyboard Shortcuts dialog](#) and, when the accelerator operation is performed, in the status line; the comment string within each function is the explanation shown at the bottom of the [Keyboard Shortcuts dialog](#) when the accelerator list entry is clicked.

Since accelerators take effect immediately (without return), a new accelerator cannot start with an existing accelerator. For example, if **ab** is a known accelerator, accelerators of the form **abx**, where *x* is any character, cannot be used.

## Limitations

### **A click in the graphics window may be required.**

Sometimes it is necessary to click in the graphics window to take back focus from some other dialog so that accelerator or command keystrokes will register.

### **"Off by one" situations can occur.**

Suppose **sv** (open the [Side View](#)) is intended, but "xv" is typed accidentally. If there is no accelerator starting with "x," an "unknown accelerator" status message results and only the "v" is retained in memory. If **sv** is then typed before the [time-out](#) period has elapsed, the keystrokes will be interpreted as the accelerator **vs** (show volume) and another "v" is retained in memory until the [time-out](#) period has again elapsed (or the escape key is pressed).

**Accelerators could be shown on menu entries.**

The keyboard shortcuts could be shown on the corresponding menu entries to make them easier to learn. More accelerators could be defined so that they are available for all suitable menu entries.



## Task Panel

The **Task Panel** is an interface to jobs started by Chimera. It provides information on previous and currently running jobs and allows canceling background tasks.

There are [several ways to start](#) the **Task Panel**, a tool in the **General Controls** category, including clicking the information icon  in the [status line](#). The icon is blue when one or more background tasks are running, a lighter blue when no tasks are running; clicking it shows the **Task Panel**.

**Clear Finished** removes listings of any canceled or finished jobs. **Help** opens this manual page in a browser window, and **Close** dismisses the dialog.

---

UCSF Computer Graphics Laboratory / September 2010



IDLE

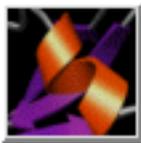
**IDLE** (for **I**nteractive **D**eve**L**opment **E**nvironment) is effectively a Python shell for Chimera. It can be used simply to enter individual Python commands, but is especially useful in the development of more complex scripts and programs.

There are [several ways to start IDLE](#), a tool in the **General Controls** category.

See also: [Reply Log](#)

---

UCSF Computer Graphics Laboratory / August 2009



## Color Secondary Structure

**Color Secondary Structure** colors peptides/proteins by secondary structure: helix, strand, or coil (non-helix, non-strand). Helix and strand assignments are taken from the input structure file or generated with [ksdssp](#). Molecules other than peptides and proteins are not affected. See also: [rainbow](#)

There are [several ways to start Color Secondary Structure](#), a tool in the **Depiction** category.

Individual models or blocks of models can be chosen from the **Models** list with the left mouse button. Ctrl-click toggles the status of an individual model. To choose a block of models without dragging, click on the first (or last) and then Shift-click on the last (or first) in the desired block.

Checkboxes control which types of secondary structure will be colored, and the colors can be changed using the adjacent [color wells](#).

- **Helix** (a [color well](#), orange red by default)
- **Strand** (a [color well](#), purple by default)
- **Coil** (a [color well](#), gray by default)

Coloring can be applied to ribbons, atoms, and/or surfaces.

**OK** performs the coloring and dismisses the dialog, whereas **Apply** performs the coloring without dismissing the dialog. **Defaults** restores the [color wells](#) to the default colors. **Close** dismisses the dialog, and **Help** opens this manual page in a browser window.



## Rainbow

**Rainbow** uses a range of colors, changing the color per residue, chain, or molecule model. There are [several ways to start Rainbow](#), a tool in the **Depiction** category. It is also implemented (with more options than the graphical interface) as the command [rainbow](#). See also: [Color Secondary Structure](#)

Individual models or blocks of models can be chosen from the **Models** list with the left mouse button. Ctrl-click toggles the status of an individual model. To choose a block of models without dragging, click on the first (or last) and then Shift-click on the last (or first) in the desired block.

Options for coloring the chosen molecule models are to **Change color every**:

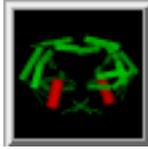
- **residue** (starting over for each chain)
- **chain** (starting over for each model)
- **model**

[WATER and HET chains](#) are not affected when the color is changed per residue or chain. The coloring cannot be limited to subset of the residues in a chain or a subset of the polymer chains in a model. To assign a unique color to each helix and/or strand in a protein chain, use the [rainbow](#) command instead.

The **Color range** is defined by five [color wells](#), corresponding to starting color, three intermediate interpolation colors, and ending color. The default colors are **blue, cyan, green, yellow, and red**, in that order. To use a different number of colors (two to four, or more than five) to define the range, use the [rainbow](#) command instead.

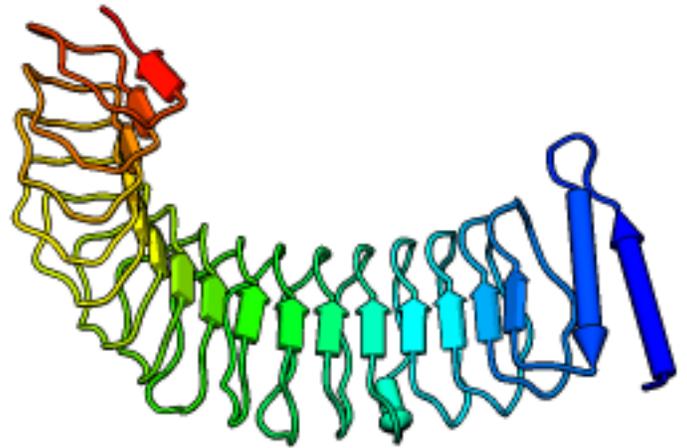
Apparent color is determined by a [hierarchy](#); briefly, atom colors assigned on a per-atom basis and ribbon colors assigned on a per-residue basis override the model-level color. When coloring per residue or chain, **Rainbow** sets individual atom colors and per-residue ribbon colors; when coloring per model, it sets model-level colors and removes any atom-level and ribbon color assignments within the models.

**OK** performs the coloring and dismisses the dialog, whereas **Apply** performs the coloring without dismissing the dialog. **Defaults** restores the [color wells](#) to the default colors. **Close** dismisses the dialog, and **Help** opens this manual page in a browser window.



## PipesAndPlanks

**PipesAndPlanks** creates a VRML representation of a protein in which helices are shown as “pipes” (cylinders), strands as “planks” (rectangular boxes), and the remaining coil as thin connectors. Arrowheads can be included to indicate chain directionality. Helix and strand assignments are taken from the input structure file or generated with [ksdssp](#). **PipesAndPlanks** does not work for nonprotein molecules such as nucleic acids. See also: [Axes/Planes/Centroids](#), [Nucleotides](#), [Rainbow](#), [ribbons](#), [geometric objects](#), the [Pipes and Planks tutorial](#)



There are [several ways to start PipesAndPlanks](#), a tool in the **Depiction** category. The model of interest should be chosen from the pulldown list of open molecule models (indicated by the black inverted triangle near the top of the dialog). Hiding [ribbons](#) is recommended; otherwise, they will be shown at the same time as the pipes-and-planks representation.

**Helix color**, **Strand color**, and **Coil color** assignments (default **No Color**) can be changed by clicking the respective [color well](#) and using the [Color Editor](#). A **No Color** setting indicates that each pipe, plank, or stretch of coil should be colored using the [residue color](#) (ribbon color) of its first residue. If a protein chain had been [rainbow-colored](#) from N-terminus to C-terminus, for example, each secondary structure element would be shown with a different color in the rainbow range.

**Helix edge color**, **Strand edge color**, and **Coil edge color** are set to **No Color** by default; if set to a color, outlines in that color will be drawn for the corresponding part. These outlines may be useful in addition to standard [silhouettes](#).

**Helix radius**, **Strand width**, **Strand thickness**, **Coil width**, and **Coil thickness** default to fixed values of 1.25, 2.5, 1.0, 0.25, and 0.25 Å, respectively. If **Fixed [helix radius, strand width, or strand thickness]** is instead set to **false**, the specified value is ignored and the parameter instead depends on the distances from the constituent  $\alpha$ -carbons to the axis of the helix or strand. If the ratio of maximum to minimum distance from  $\alpha$ -carbons to the axis exceeds the **[Helix or Strand] split threshold** (and the corresponding **Split curved...** option is **true**), the pipe or plank will be split into two parts, each with its own axis.

**Show arrow on helix** and **Show arrow on strand** (both default **true**) place arrowheads on the C-terminal ends of secondary structure elements to indicate N→C directionality.

Depiction of coils can be turned off by setting **Display coil** to **false**. **Coil subdivisions** (default **10**) refers to the number of segments per residue used to represent coils.

Clicking **OK** creates the display and dismisses the dialog; clicking **Apply** creates the display without dismissing the dialog. **Close** dismisses the dialog, and **Help** opens this manual page in a browser window.

The pipes-and-planks VRML representation is assigned the same model number as the corresponding molecule. The VRML model can be hidden/shown or closed with the [Model Panel](#), or hidden/shown with the command [objdisplay](#).



## Nucleotides



**Nucleotides** creates special nucleotide-specific displays, including VRML representations of the base and sugar moieties. Such displays are generally combined with various [atomic representations](#) and/or [ribbons](#). **Nucleotides** representations are included in saved [sessions](#). See also: [fillring](#) and the following reference:

[Nucleic acid visualization with UCSF Chimera](#). Couch GS, Hendrix DK, Ferrin TE. *Nucleic Acids Res.* 2006 Feb 14;34(4):e29.

There are [several ways to start Nucleotides](#), a tool in the **Depiction** category. It can be called from the [Atoms/Bonds](#) section of the [Actions menu](#). It is also implemented as the command [nucleotides](#). Options:

- **Show backbone as:**
  - **atoms & bonds** - include all currently displayed nucleic acid backbone atoms
  - **ribbon** (default) - use the **nucleic acid rotated** [residue class](#)
  - **classic ribbon** - uses the **nucleic acid** [residue class](#)
- **Show sidechain (sugar / base) as:**
  - **atoms & bonds** - include all currently displayed nucleic acid sidechain atoms
  - **fill / fill** - show the sugar (ribose) and base moieties as filled rings
  - **fill / slab** (default) - fill in sugar rings, show bases as slabs (see [Slab Options](#) and [Slab Style](#))
  - **tube / slab** - show sugars as tubes connecting the bases to the backbone, show bases as slabs (see [Slab Options](#) and [Slab Style](#))
  - **ladder** - show H-bonded residue pairs as rods or ladder rungs (see [Ladder Options](#))

Ring fill thickness tracks the thickness of the surrounding bonds: a thin layer for wires, thicker for sticks. The thickness of sticks and of tube sugars can be adjusted by changing the [stick scale](#) (a molecule model [attribute](#)).

- **Show base orientation (true/false)** - show the positive faces of bases with bumps. The positive faces are those which point towards the 3' end of a strand in right-handed A- and B-DNA (the positive Z direction in the [standard reference frame](#)). The bumps will be displayed for bases shown as slabs or with thick fill, but not where the fill is a thin layer.

Clicking **NDB Colors** sets atom and ribbon colors according to the convention used in the [Nucleic Acid Database \(NDB\) Atlas](#): A red, T blue, C yellow, G green, and U cyan.

Special nucleotide representations will be created only where the corresponding atoms are displayed, and will be colored to match the atoms. **OK** updates the display and dismisses the dialog; **Apply** updates the

display without dismissing the dialog. The option **Restrict OK/Apply...** limits any changes (including setting **NDB Colors**) to nucleotide residues in the current [selection](#). **Close** dismisses the dialog, and **Help** opens this manual page in a browser window.

Nucleotide displays other than atoms, bonds, and ribbons are opened as a VRML model with the same model number as the corresponding molecule model. These models can be closed or hidden with the [Model Panel](#), or undisplayed/displayed (hidden/shown) with the command [objdisplay](#).

## Slab Options

- **Thickness** (default 0.5 Å) - slab thickness; the other dimensions depend on the [slab style](#)
- **Slab object** - slab shape
  - **box** (default) - a rectangular box: rectangular in the plane of the base with rectangular cross-section
  - **tube** - rectangular in the plane of the base with elliptical cross-section, large enough to enclose the corresponding rectangular box
  - **ellipsoid** - elliptical in the plane of the base with elliptical cross-section, large enough to enclose the corresponding rectangular box
- **Hide base atoms** (true/false) - whether to hide the atoms and bonds in bases when slabs are shown
- **Separate glycosidic bond** (true/false) - whether tube sugar representations should each consist of a single straight segment (default) or two segments, one along the glycosidic (sugar-base) bond; applies only to [tube / slab](#) display with base-anchored [slab styles](#) (**long**, **skinny**).

## Slab Style

A *slab style* refers to the slab dimensions and location relative to the atoms in purine and pyrimidine bases. The settings define a virtual rectangle in the plane of the base, as shown in the schematic diagrams in the dialog. Ellipsoid [slab objects](#) will be drawn to enclose the the virtual rectangle. Changes in **Slab Style** settings will be shown in the schematics, but will not affect the Chimera display until either **Apply** or **OK** is clicked.

The built-in slab styles are **big**, **fat**, **long** (default), and **skinny**. Slab styles can be named, saved, and later retrieved from the pulldown list indicated by the solid black triangle. When the name of a built-in style is shown, it is only possible to save to a different name, using **Save As...** When another name is shown, it is possible to:

- **Save** the slab style to the name shown
- use **Save As...** to save the slab style to a new name
- **Delete** the slab style whose name is shown (if previously saved)

Custom slab styles are saved in the Chimera [preferences file](#), and are only updated with any changes when **Save**, **Save As...** or **Delete** is used.

The **Anchor** is the point of reference for the virtual rectangle:

- **sugar** - at the sugar end of the glycosidic bond (atom C1')

- **base** - at the base end of the glycosidic bond

**Purine and Pyrimidine** settings each include:

- **Lower left** - position of the rectangle's lower left corner (as shown in the schematic) in Å from the anchor; the first number is distance along the X-axis in the [standard reference frame](#) (positive means to the right) and the second is distance along the Y-axis in the [standard reference frame](#) (negative means downward)
- **Upper right** - position of the rectangle's upper right corner (as shown in the schematic) in Å from the anchor; the first number is distance along the X-axis in the [standard reference frame](#) (positive means to the right) and the second is distance along the the Y-axis in the [standard reference frame](#) (negative means downward)

The *standard reference frame* is described in:

[A standard reference frame for the description of nucleic acid base-pair geometry.](#) Olson WK, Bansal M, Burley SK, Dickerson RE, Gerstein M, Harvey SC, Heinemann U, Lu XJ, Neidle S, Shakked Z, Sklenar H, Suzuki M, Tung CS, Westhof E, Wolberger C, Berman HM. *J Mol Biol.* 2001 Oct 12;313(1):229-37.

## Ladder Options

- **Skip non-base H bonds** (true/false) - whether to consider only base atoms when identifying H-bonds; otherwise, all nucleic acid atoms will be considered (default)
- **Show stubs** (true/false) - whether to show bases without H-bonds to other bases as half-rungs (default) or not at all
- **Rung radius** (default 0.45 Å) - radius to use for ladder rungs showing H-bonds between bases, and for stubs representing bases that lack H-bonds to other bases. When the **Skip...** option is false, any H-bonds involving non-base atoms (on either or both ends) will be shown with a smaller radius corresponding to the smallest ribbon dimension, or if the backbone is not shown as a ribbon, the model stick size.
- **Use existing H-bonds** (true/false) - whether to use H-bonds previously calculated with [FindHBond](#) (or the command [findhbond](#)) instead of running a new calculation. If this option is true and a [pseudobond group](#) named **hydrogen bonds** exists, another H-bond calculation will not be performed (even if the H-bonds are for some other structure).
- **Relax H-bond constraints** (on by default) - whether to relax the strict [geometric criteria](#) for identifying H-bonds. The default relaxation amounts are 0.4 Å and 20.0°.

## Limitations

**Some nonstandard residues not handled.** Residues handled are A, DA, T, DT, C, DC, G, DG, U, PSU (pseudouridine base), and I (inosine base), as well as certain modified versions:

- described in MODRES records
- in which the base ring atoms are unchanged (for example, methylated or brominated derivatives)

After showing these modified bases as slabs, it may be necessary to undisplay extra base atoms that were not hidden automatically.



## Ribbon Style Editor

**Ribbon Style Editor** allows ribbon [scalings](#), [styles](#), and [residue classes](#) to be edited directly and/or named and saved for subsequent use.

Ribbons for peptides, proteins, and nucleic acids can be displayed using the command [ribbon](#) or [Actions... Ribbon... show](#). Ribbon [style](#) can be switched with [ribrepr](#) or [Actions... Ribbon... style-name](#). Ribbon [scaling](#) can be switched with [ribscale](#). Ribbon [residue class](#) can be switched with [ribclass](#), and spline method with [ribspline](#).

There are [several ways to start Ribbon Style Editor](#), a tool in the **Depiction** category. It has three panels shown as index cards:

- [Scaling](#)
- [Cross Section](#)
- [Residue Class](#)

**Close** dismisses the dialog, and **Help** opens this manual page in a browser window.

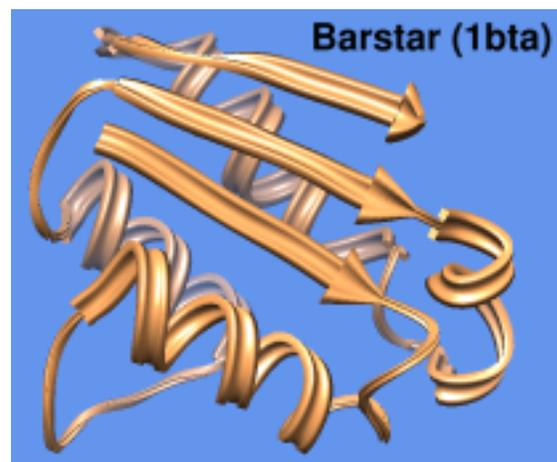
### SCALING

The *scaling* of a ribbon is its secondary-structure-specific dimensions. Named scalings can be accessed with the command [ribscale](#).

**Known problem:** reassigning secondary structure (for example, by using [ksdssp](#)) sets the scaling of some residues to **Chimera default**; any other scalings previously in effect need to be reapplied.

In the **Scaling** panel of the [Ribbon Style Editor](#), changes can be entered directly, or the arrows on either side of a field can be clicked to increase or decrease the value by 0.05-Å increments. Only values greater than zero are allowed. Changes will not take effect until either **Apply** or **OK** is clicked; **OK** also dismisses the panel. **Restrict OK/Apply...** allows the scaling to be applied to only the current [selection](#). If a scaling has been named, it can also be applied using the command [ribscale](#).

If a ribbon were laid out on a flat surface, width would be the dimension parallel to the surface but perpendicular to the long axis of the ribbon, and height would be the dimension normal to the surface. The **Width** and **Height** values are doubled to give the maximum ribbon dimensions (the dimensions mapped to the [cross-section grid](#)). For example, with any of the [built-in styles](#), a **Width** of 0.9 yields a ribbon 1.8 Å wide.



[ribbon scaling](#): "Chimera default"  
custom [ribbon style](#): "[aitch](#)"

Nucleic acid structures are shown with the **Nucleic** ribbon. For proteins, **Helix** and **Sheet** assignments are taken from the input structure file or generated with [ksdssp](#); the remainder is shown as **Coil**. Strands (the **Sheet** segments) are shown by default as arrows pointing in the C-terminal direction. To also show helices as arrows, see [below](#). Setting the **Arrow (tip)** and **Arrow (base)** widths the same as the **Sheet** width turns off the arrows. The scaling named **licorice** is effectively independent of helix/strand assignments because all widths are the same and all heights are the same.

The settings in the **Scaling** panel collectively define a scaling, which can be named, saved, and later retrieved from the pulldown list indicated by the solid black triangle next to the **Ribbon Scaling** field. When the name of a built-in scaling (**Chimera default** or **licorice**) is shown, it is only possible to save to a different name, using **Save As...** When another name is shown, it is possible to:

- **Save** the scaling to the name shown
- use **Save As...** to save the scaling to a new name
- **Delete** the scaling whose name is shown (if previously saved)

Named scalings are saved in the Chimera [preferences file](#), and are only updated with any changes when **Save**, **Save As...**, or **Delete** is used. The settings in effect when a session is [saved](#) (whether or not the scaling has a name) are included in the [session file](#).

The **Arrow** parameters apply only to strands, not helices. To show arrowheads at the ends of helices, the workaround is to reassign the helices as strands. This can be done with commands something like:

Command: [select](#) helix

Command: [setattr](#) r isStrand true sel

Command: [setattr](#) r isHelix false sel

Command: [~select](#)

Alternatively, the residue secondary structure attributes could be changed in the [Selection Inspector](#) instead of with [setattr](#).

## CROSS SECTION

## Custom style “aitch”

The *style* of a ribbon is defined by its *cross section* (general shape perpendicular to the long axis) and how the cross section is [smoothed](#). Named styles can be accessed with the command [ribrepr](#) and from the [Actions... Ribbon menu](#).

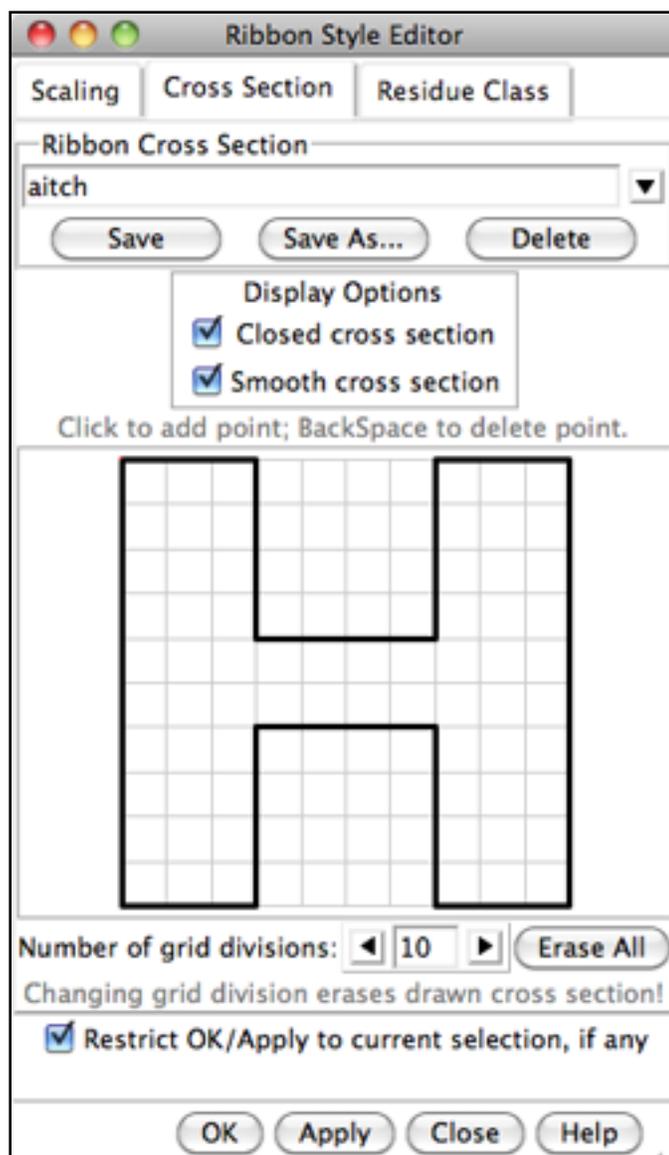
In the **Cross Section** panel of the [Ribbon Style Editor](#), a cross-section can be defined by placing points on a grid. The **Number of grid divisions** can be adjusted as needed for the complexity of the desired cross-section. A ribbon with a particular cross-section is subject to whichever [scaling](#) is in effect. The extent of the grid X-axis equals the width and the extent of the grid Y-axis equals the height. Using the built-in styles as examples:

- the **edged (sharp)** style corresponds to a square outlining the grid; when width differs from height, the cross-section is “squashed” to a rectangle
- the **flat (ribbon)** style corresponds to a horizontal line bisecting the grid; only the width matters, as this style has no height
- the **rounded (round, smooth)** style corresponds to the largest circle that fits in the grid; when width differs from height, the cross-section is “squashed” to an oval
- the **supersmooth** style is the same as **rounded**, except that its cross section is defined with a greater number of points (useful when generating [raytraced](#) images)

A style is further defined by **Display Options**:

- **Closed cross section** - close the cross section automatically as it is being drawn on the grid (a line will connect the first and last points drawn); unnecessary if the closing line is created manually
- **Smooth cross section** - round the corners of the cross section when drawing the ribbon

Changes will not take effect until either **Apply** or **OK** is clicked; **OK** also dismisses the panel. **Restrict OK/Apply...** allows the style to be applied to only the current [selection](#). If a style has been named, it can also be applied using the command [ribrepr](#) or by choosing its name from the [Actions... Ribbon menu](#).



The settings in the **Cross Section** panel collectively define a style, which can be named, saved, and later retrieved from the pulldown list indicated by the solid black triangle next to the **Ribbon Cross Section** field. When the name of a [built-in](#) style is shown, it is only possible to save to a different name, using **Save As...** When another name is shown, it is possible to:

- **Save** the style to the name shown
- use **Save As...** to save the style to a new name
- **Delete** the style whose name is shown (if previously saved)

Named styles are saved in the Chimera [preferences file](#), and are only updated with any changes when **Save**, **Save As...**, or **Delete** is used. The settings in effect when a session is [saved](#) (whether or not the style has a name) are included in the [session file](#).

## RESIDUE CLASS

A ribbon *residue class* is defined by which atoms control the path of the ribbon and which atoms are

considered part of the backbone (*mainchain atoms*). The path of the ribbon is controlled by two atoms per residue: the *guide atom* sets ribbon location, and the *orientation atom* sets the plane of the ribbon.

Named classes can be accessed with the command [ribclass](#).

Built-in residue classes include:

class	guide atom	orientation atom	mainchain atoms
amino acid	CA	O	N,CA,C,O
nucleic acid	C5'	C1'	P,OP1,OP2,OP3,O5',C5',O3' (and synonyms)
nucleic acid p trace	P	C1'	P

There is also a **nucleic acid rotated** class, which differs from **nucleic acid** only in that the option to [rotate](#) the ribbon is on.

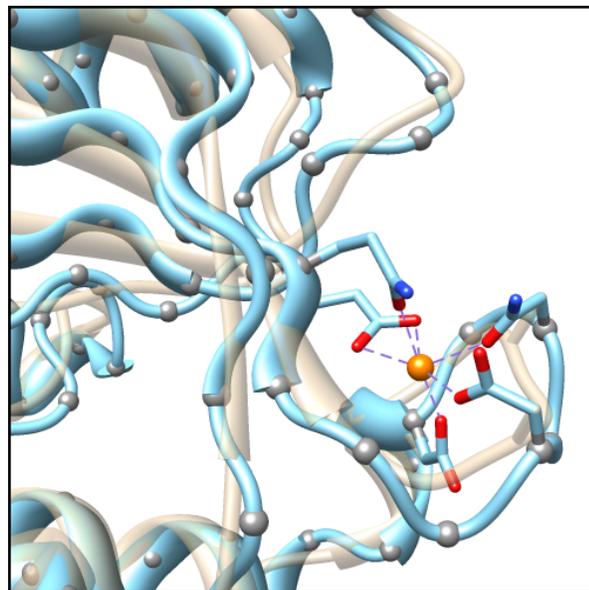
The ribbon path is smoothed over successive residues. By default, when ribbon is shown for a residue, the [mainchain atoms](#) of that residue are hidden, except where bonded to other displayed atoms (such as the sidechain). For drawing these bonds, the apparent locations of the mainchain atoms are adjusted to fall on the ribbon. Thus, bonds from ribboned mainchain to sidechain may appear as unnaturally short or long, but measurements involving mainchain atoms (bond length, *etc.*) will still reflect their real, unadjusted positions. The command [ribbackbone](#) enables simultaneous display of ribbon and the mainchain atoms in their true positions. However, the atoms may be “floating” away from the ribbon in some places.

There is a trade-off between the smoothness of the ribbon path and how closely it follows the true positions of the atoms. Two methods of determining the path are available:

- *B-spline* (default) - uses five successive residues and may not coincide exactly with any [mainchain atoms](#)
- *cardinal spline* - uses four successive residues and coincides exactly with the [guide atoms](#), unless additional smoothing is applied; to minimize the appearance of kinks, a tubular ribbon (for example, [style: rounded](#) and [scaling: licorice](#)) may be preferred

These and related parameters can be controlled with the command [ribspline](#), the [molecule model attributes panel](#), and the [Selection Inspector](#).

The **Residue Class** panel of the [Ribbon Style Editor](#) allows new ribbon residue classes to be defined. Names for the **Guide atom**, **Orientation atom**, and **Mainchain Atoms** must be supplied; letters in these names are automatically entered in uppercase. A mainchain atom's **Position** is how far along a residue's ribbon segment the atom is placed when adjusted to fall on the ribbon (possible values range from 0 to 1). Clicking **Set** adds the contents of the **Name** and **Position** fields to the **Mainchain Atoms** list. Choosing (clicking) an entry in the list and then clicking **Remove** deletes that entry.



**bspline: tan;**  
**cardinal (unsmoothed): light blue;**  
gray balls:  $\alpha$ -carbon positions

**Further Options:**

- **Non-helix/sheet uses "Nucleic" scaling** - whether the ribbon for residues without secondary structure assignments should have the **Nucleic** dimensions in the current [scaling](#) scheme (otherwise the **Coil** dimensions will be used)
- **Rotate orientation 90 degrees** - whether the plane of the ribbon should be perpendicular to a line drawn from the middle (widthwise) of the ribbon to the orientation atom (otherwise parallel)

Changes will not take effect until either **Apply** or **OK** is clicked; **OK** also dismisses the panel. **Restrict OK/Apply...** allows the changes to be applied to only the current [selection](#). If a class has been named, it can also be applied using the command [ribclass](#).

The settings in the **Residue Class** panel collectively define a ribbon residue class, which can be named, saved, and later retrieved from the pulldown list indicated by the solid black triangle next to the **Ribbon Residue Class** field. When the name of one of the [built-in classes](#) is shown, it is only possible to save to a different name, using **Save As...** When another name is shown, it is possible to:

- **Save** the class to the name shown
- use **Save As...** to save the class to a new name
- **Delete** the class whose name is shown (if previously saved)

Named classes are saved in the Chimera [preferences file](#), and are only updated with any changes when **Save**, **Save As...**, or **Delete** is used. The settings in effect when a session is [saved](#) (whether or not the class has a name) are included in the [session file](#).

The **Render/Select by Attribute** interface performs several functions related to [attributes](#):

- [visual rendering](#) of attribute values
- [selection](#) by attribute values
- [saving](#) attribute values to an [assignment file](#)

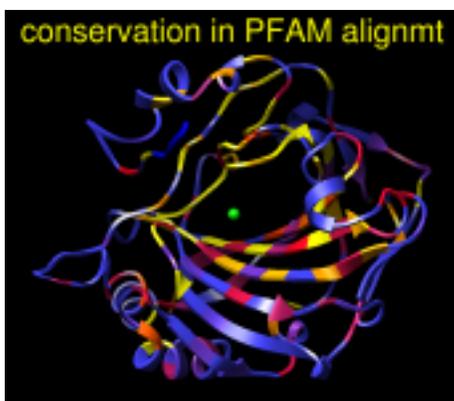
Attributes of atoms, residues, molecule models, and [segmentation regions](#) are handled.

## Render by Attribute

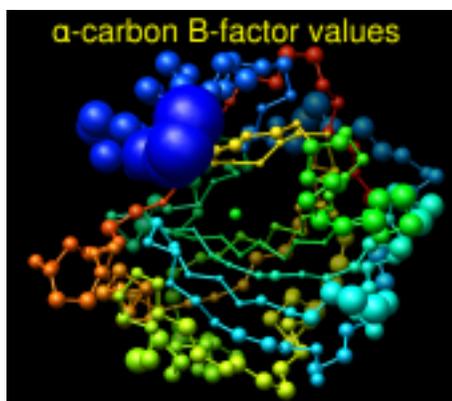
Render by  
Attribute

Render by Attribute can show [attribute](#) values with colors, atomic radii, or *worms* (modified ribbons):

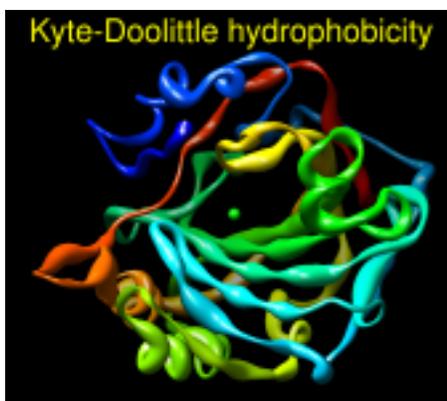
### [Colors](#)



### [Radii](#)



### [Worms](#)



There are [several ways to start](#) **Render by Attribute**, a tool in the **Depiction and Structure Analysis** categories. See also: [Select by Attribute](#), [Define Attribute](#)

The first step is to indicate whether attributes of **atoms**, **residues**, **molecules** (molecule models), or map [segmentation regions](#) will be used. Next, the **Models** and **Attribute** of interest should be chosen.

The **Models** list includes all open models of the appropriate type. Individual models or blocks of models can be chosen with the left mouse button. Ctrl-click toggles the status of an individual model. To choose a block of models without dragging, click on the first (or last) and then Shift-click on the last (or first) in the desired block.

The available [attributes](#) (those of the [chosen models](#)) are listed in a pulldown menu. Only *numerical attributes* are available for rendering. These may include:

- **atoms**
  - **bfactor** - B-factor (temperature factor; generally read from [PDB input](#))
  - **occupancy** - occupancy (generally read from [PDB input](#))
  - other atom attributes added by Chimera, such as **areaSAS** and **areaSES** when a [molecular surface](#) has been computed, **charge** from [Add Charge](#), *etc.*
  - [custom](#) atom attributes

- **residues**
  - **chi1, chi2**, ... amino acid sidechain angles
  - **kdHydrophobicity** - [Kyte-Doolittle hydrophobicity](#) (available only for standard amino acids)
  - **phi, psi** - peptide backbone angles
  - other residue attributes added by Chimera, such as **areaSAS** and **areaSES** when a [molecular surface](#) has been computed, **mavConservation** and other [sequence-derived attributes](#) from [Multalign Viewer](#), *etc.*
  - [custom](#) residue attributes
  - residue averages of constituent atom attributes, considering only atoms with values for that attribute
  - residue totals of constituent atom attributes
- **molecules** (molecule models)
  - model attributes added by Chimera, such as [model scores](#) from comparative (homology) modeling
  - [custom](#) model attributes
  - model averages of constituent residue attributes, considering only residues with values for that attribute
  - model totals of constituent residue attributes
- **segmentation regions**
  - **npoints** - number of density map grid points enclosed in the region (roughly proportional to volume; shown as **grid points** in the [Region Attributes dialog](#))
  - [custom](#) segmentation region attributes

Custom (user-defined) attributes of atoms, residues, and molecule models can be created:

- arbitrarily with [Define Attribute](#), [defattr](#), or [setattr](#)
- by combining other attributes using [Attribute Calculator](#)

Custom attributes of [segmentation regions](#) can be created arbitrarily with the [Region Attributes dialog](#).

The name of an (atom, residue) attribute controls whether it is automatically available as an average or total at the next higher level (residue, model). If **area**, **charge**, or **volume** is a component of the name, the attribute will be totaled; if not, it will be averaged. Name components are indicated with underscores or camel-caps. For example, the components of **my\_charge** are **my** and **charge**; the components of **voronoiVolume** are **voronoi** and **volume**.

If **Render/Select by Attribute** was opened before values were assigned for some attribute, the new attribute can be listed using **Refresh... Menus**. When attribute values have been changed or parts of the [chosen models](#) have been deleted, the display can be updated with **Refresh... Values**. The data will refresh automatically, however, when the set of [chosen models](#) is changed.

Once an attribute has been chosen, a histogram of the corresponding values is shown. Histogram bar heights are logarithmic by default, but can be changed to linear (and back) with the **Scaling** menu.

*Thresholds* for defining the interpolation function (for [Colors](#), [Radii](#), or [Worms](#)) are shown as vertical bars on the histogram. Clicking on a threshold shows its **Value** in black. Clicking elsewhere within the histogram shows the **Value** (X-coordinate) of the mouseclick in gray. A threshold can be moved by

changing its **Value** and then pressing Enter (return) or by dragging it horizontally with the left mouse button. Holding the Shift key down reduces the speed (mouse sensitivity) of threshold dragging tenfold, allowing finer control. Thresholds can be added by Ctrl-clicking with the left mouse button on the histogram. Ctrl-clicking on an existing threshold deletes it.

The rendering can be restricted to the current [selection](#) (**Restrict OK/Apply...**). **OK** performs the rendering and dismisses the dialog, whereas **Apply** performs the rendering without dismissing the dialog. **Close** simply dismisses the dialog. **Help** opens this manual page in a browser window.

## Colors

Atoms and [surfaces](#) (molecular, VDW) can be colored by atom, residue, or molecule model attribute values; ribbons can be colored by residue or molecule model attribute values; [segmentation regions](#) can be colored by segmentation region attribute values. Each [threshold](#) or vertical bar on the histogram has a **Value** (position on the histogram) and a **Color**. Together, the thresholds specify how colors map to values. Thresholds can be moved, added, or deleted as described [above](#), and their colors can be changed:

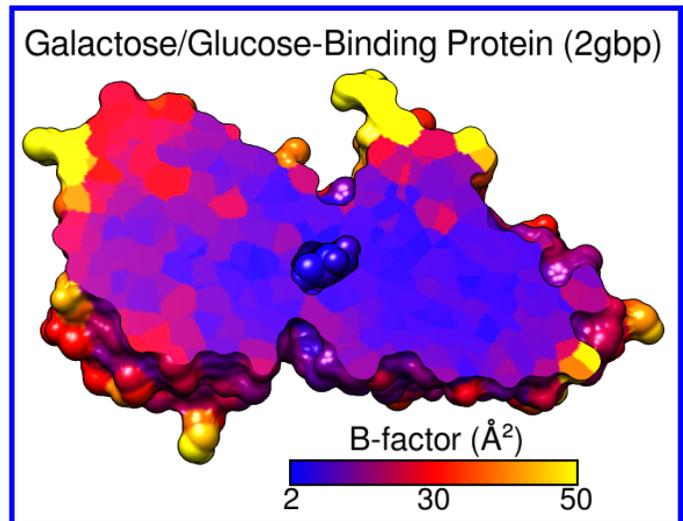
- The [color well](#) marked **Color** applies to the threshold most recently moved or clicked. Clicking it opens the [Color Editor](#), allowing the color to be changed interactively.
- Choosing a **Palette** changes the colors without changing the number of thresholds or their positions:
  - **Blue-Red** - from blue to white to red
  - **Cyan-Maroon** - from a medium cyan to white to maroon
  - **Gray** - grayscale, starting with black and ending with white
  - **Rainbow** - standard rainbow colors, starting with red and ending with blue (to achieve a full rainbow spectrum, the number of thresholds should be increased to at least five beforehand)
  - **Red-Blue** - from red to white to blue

The desired number of thresholds should be present before the palette is chosen. The order of the colors can be reversed and/or individual colors can be changed interactively after a palette has been chosen.

- Clicking **Reverse threshold colors** reverses the current colors (whether set individually or using a palette) without changing the number of thresholds or their positions.

An attribute value is compared to the thresholds on the histogram. Color is defined by red, green, blue and opacity/transparency components. The color of the closest threshold at a lower value (to the left) and the color of the closest threshold at a higher value (to the right) are linearly interpolated. Items with attribute values less than the leftmost threshold are colored according to the leftmost threshold, while items with attribute values greater than the rightmost threshold are colored according to the rightmost threshold. The **Keep opaque** options for atom and ribbon coloring indicate that any transparency in the color definitions should be ignored.

There may be cases where the atoms (residues, molecule models) have no value for the chosen attribute.



For example, nucleic acid residues do not have [kdHydrophobicity](#) values, and residues not associated with a position in a sequence alignment open in [Multalign Viewer](#) do not have `mavPercentConserved` values. In these cases, the **No-value color** (also shown in a [color well](#)) applies.

**Create corresponding color key** opens the [Color Key](#) dialog and populates it with the current colors and values; a color key can then be created interactively with the mouse.

See also: [rangecolor](#), [Values at Atom Positions](#), [Surface Color](#), [Coulombic Surface Coloring](#)

## Radii

Atom radii can be assigned according to atom, residue, or molecule model attribute values. The **Atom style** controls which [draw mode](#) will be used for the affected atoms (**ball** or **sphere**). Ball radius equals atom radius multiplied by a molecule model's [ball scale](#) factor, whereas sphere radius is the same as atom radius.

Besides a **Value** (position on the histogram), each [threshold](#) has an associated **Atom radius**, expressed in Å. The **Atom radius** value corresponds to the threshold most recently moved or clicked and can be adjusted by entering a different number.

The [thresholds](#) define a function that maps attribute values to atom radii. For each atom (residue, molecule model) the attribute value is compared to the thresholds on the histogram. The atom radius of the closest threshold at a lower value (to the left) and the atom radius of the closest threshold at a higher value (to the right) are linearly interpolated. Atoms (or atoms within residues or models) with attribute values less than the leftmost threshold are assigned the atom radius of the leftmost threshold, while those with attribute values greater than the rightmost threshold are assigned the atom radius of the rightmost threshold.

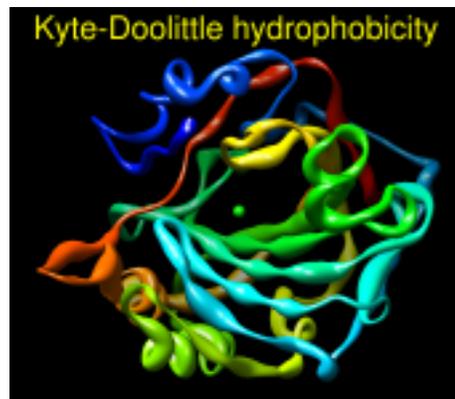
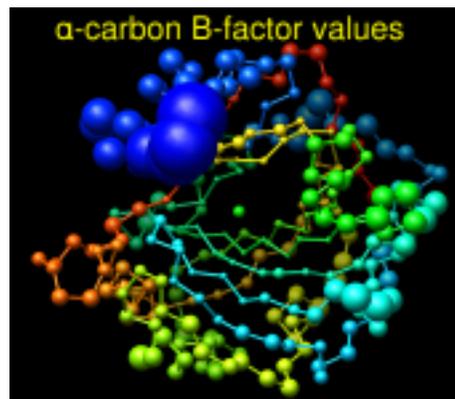
When there are atoms (residues, molecule models) with [no value](#) for the chosen attribute: if **Affect no-value atoms** is **true**, atoms (or atoms within residues or models) with no value will be assigned the **No-value radius**. Otherwise, the radii of these atoms will not be changed.

Radii can be returned to their [default values](#) with the command `~vdwdefine`.

## Worms

Worms are similar to the “sausage” or “putty” representations in other graphics programs, where fatness (radius) shows the attribute values of residues or molecule models. Worms are actually ribbons with a special [scaling](#) based on attribute values instead of secondary structure.

Rendering as worms displays a ribbon for the affected residues in the [style](#) named [rounded](#) with the special [scaling](#). Like other ribbons, worms can be shown only for amino acid and nucleic acid residues bonded to at least one other residue. The **Worm style** controls whether the radius is smoothed continuously along the ribbon (**smooth**) or changes abruptly for



each residue (**segmented**). To obtain standard, secondary-structure-dependent ribbons after rendering as worms, it is necessary to render again with the **Worm style** set to **non-worm**.

Besides a **Value** (position on the histogram), each [threshold](#) has an associated **Worm radius**, expressed in Å. The **Worm radius** value corresponds to the threshold most recently moved or clicked and can be adjusted by entering a different number.

The [thresholds](#) define a function that maps attribute values to worm radii. For each residue (or molecule model) the attribute value is compared to the thresholds on the histogram. The worm radius of the closest threshold at a lower value (to the left) and the worm radius of the closest threshold at a higher value (to the right) are linearly interpolated. Residues (or residues within models) with attribute values less than the leftmost threshold are assigned the worm radius of the leftmost threshold, while those with attribute values greater than the rightmost threshold are assigned the worm radius of the rightmost threshold.

When there are residues (or molecule models) with [no value](#) for the chosen attribute: if **Affect no-value residues** is **true**, residues (or residues within models) with no value will be shown as worms with the **No-value worm radius**, unless the **Worm style** is **non-worm**. Rendering with the **non-worm** style restores the standard ribbon for all affected residues.

## Select by Attribute

**Select by Attribute** allows [selection](#) of atoms, residues, molecule models, and map [segmentation regions](#) by their [attribute](#) values. It can be opened by choosing **By Attribute Value...** from the Chimera [Select menu](#) or [Structure... Select by Conservation](#) from the menu in in [Multalign Viewer](#). See also: [Render by Attribute](#)

One must [choose one or more models](#) and then one of the [attributes](#) available for the chosen models. These may include the [numerical attributes](#) available for [rendering](#) plus various string-valued and boolean attributes assigned by Chimera or [defined by the user](#). Some examples:

- **atoms**
  - **display** - whether display is enabled at the atom level (see [display hierarchy](#))
  - **idatmType** - [atom type](#)
  - **surfaceCategory** - the name of the category to which the atom has been assigned [automatically](#) or manually using [surfcat](#)
- **residues**
  - **isHelix** - whether the residue is in a helix (true is only possible for amino acids)
  - **isHet** - whether the residue is in HETATM records in the input PDB file
  - **isStrand** - whether the residue is in a beta strand (true is only possible for amino acids)
  - **type** - residue name
- **segmentation regions**
  - **placed**

Once an attribute has been chosen, a [histogram](#) (if numerical), [list](#) (if character strings), or [set of checkbox options](#) (if boolean) of the corresponding values is shown. The data can be [refreshed](#) after new attributes have been added, attribute values have been changed, or parts of the [chosen models](#) have been deleted.

- Within a histogram, the bar [scaling](#) can be logarithmic or linear. Two *markers* are shown as vertical bars. Clicking on a marker shows its **Value** in black. Clicking elsewhere within the histogram shows the **Value** (X-coordinate) of the mouseclick in gray. A marker can be moved by changing its **Value** and then pressing Enter (return) or by dragging it horizontally with the left mouse button. Holding the Shift key down reduces the speed (mouse sensitivity) of marker dragging tenfold, allowing finer control. The mutually exclusive options are to **Select**:
  - **between markers (inclusive)**
  - **outside markers**
  - **no value** - applies to atoms (residues, molecule models) with [no value](#) for the chosen attribute (only listed when present)
- Within a list of character strings, simply clicking an entry toggles its status between will/will not be selected.
- For boolean attributes, the mutually exclusive options are:
  - **False**
  - **True**
  - **No value** - applies to atoms (residues, molecule models) with [no value](#) for the chosen attribute (only listed when present)

**OK** performs the selection and dismisses the dialog, whereas **Apply** performs the selection without dismissing the dialog. **Close** simply dismisses the dialog. **Help** opens this manual page in a browser window.

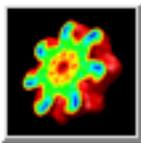
## Saving Attributes

An attribute and its associated values can be saved to an [attribute assignment file](#) by choosing **File... Save Attributes** from the menu of [Render/Select by Attribute](#).

The attribute to save, its level (**atoms**, **residues**, **molecules**, or [segmentation regions](#)\*) and the model(s) to include should be chosen from the menus in the resulting [save dialog](#). Additional options:

- **Restrict save to current selection, if any** - when a [selection](#) exists, whether to write out only the attribute values of the [selected](#) items (atoms, residues, models) within the set of chosen models
- **Include model numbers in output** - whether [atom specification strings](#) in the output [assignment file](#) should include model numbers. Omitting the model number allows later use of the file to assign attribute values when the model has been opened with a different number.

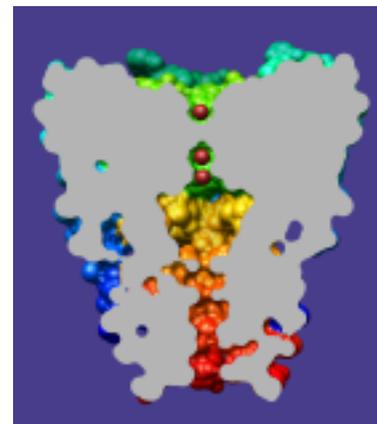
\* Files for [segmentation regions](#) are mainly informational, as they cannot be used to assign attributes ([Define Attribute](#) only handles atoms, residues, and molecule models).



## Surface Capping

**Surface Capping** draws planar caps that hide the interior of a [surface model](#) sliced by a [clipping plane](#). **Surface Capping** information is included in saved [sessions](#). See also: [Surface Color](#), [Color Zone](#), [clip](#), [mclip](#)

There are [several ways to start Surface Capping](#), a tool in the **Depiction** and **Surface/Binding Analysis** categories. It can also be called from the [Side View](#) and [Per-Model Clipping](#) dialogs, and is implemented as the command [sop cap](#).



Options:

- **Cap surfaces at clip planes** - whether to show caps
- **Use cap color** [\[color well\]](#) - whether to use the [color well](#) color for caps. Otherwise, cap color will be:
  - for molecular surfaces, the model-level color of the corresponding molecule model (see [coloring hierarchy](#))
  - for other surface models, the surface color in single-color mode
 This option is ignored when caps are colored with [Surface Color](#).
- **Cap style**
  - **solid** (default)
  - **mesh**
- **Mesh subdivision factor** [\[f\]](#) - how finely to subdivide cap surfaces. By default,  $f=1.0$ , which makes cap triangles approximately the same size as triangles in the surface being capped. Larger values of  $f$  yield smaller triangles and finer color gradations (when the cap is multicolored with [Surface Color](#)), but increase computational demands. There is no reason to increase the subdivision factor when caps are a single solid color. The number of cap triangles scales as  $f^2$ ; a very high subdivision factor ( $f > 10$ ) can make cap calculation extremely slow and cause Chimera to run out of memory and crash.
- **Cap to clip plane distance** [\[d\]](#) - the cap must be offset at least slightly from the clipping plane ( $d=0.01$  by default). Without an offset, the cap itself can be clipped (invisible) depending on floating-point rounding errors and the specific graphics hardware being used. If  $d$  is large, however, the mismatch between the cap and the cut edge of the surface will be evident.

Changes in numerical parameters take effect after return (**Enter**) is pressed. Caps are recalculated automatically when the clipping plane is moved, the surface is moved relative to the clipping plane, or the shape of the surface changes. Exceptions: a change in the resolution of a [Multiscale Models](#) surface or the shape of a [molecular surface](#) (MSMS model) may not trigger a cap update. An update can be triggered by moving the clipping plane slightly or toggling the [Cap surfaces at clip planes](#) option.

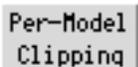
## LIMITATIONS

**Lack of per-model settings.** The settings apply to all surfaces. It is not possible to cap one surface but not another, or to use different colors, styles, or other capping parameters for different surfaces. This is a limitation of the user interface, not the underlying implementation.

**Slow interactive rotation.** Rotating the model causes the cap calculation to check if the near clipping plane intersects the surface at each frame of the rotation. This can slow down interactive rotation.

**Artifacts in cap.** Artifacts such as streaks and dots in a cap can occur when points in the border are very close together. Slight changes in the surface shape or the position of clipping plane relative to the surface will generally solve the problem.

**Surfaces with a boundary.** The cap calculation assumes that the intersection of the clipping plane and the surface forms closed loops. If the surface has a boundary (for example, a isosurface might end abruptly at the edge of the corresponding volume data), then its intersection with the clipping plane may form a non-closed curve, which will not be capped. The basic difficulty in this case is that the surface does not separate its interior from its exterior.



## Per-Model Clipping

**Per-Model Clipping** allows different models to be clipped in different ways. Unlike the [global clipping planes](#), a per-model clipping plane:

- affects only one model
- can be placed at any angle relative to the line of sight

Except in [slab mode](#), a per-model clipping plane is a single plane that separates the visible and invisible portions of a model; whether it behaves like a "front" or "back" depends on the direction it is facing. In [slab mode](#), a per-model clipping plane behaves like a front-back pair of planes locked in parallel.

Each model can have one per-model clipping plane. A molecule model and its associated [molecular surface](#) are treated as separate models. Per-model clipping planes and slabs are saved in [sessions](#).



There are [several ways to start Per-Model Clipping](#), a tool in the **Depiction** category. Per-model clipping can also be controlled with the command [mclip](#).

The **Model** pulldown menu shows the *current model* and allows switching between models.

Activating **Enable clipping** turns on a clipping plane for the [current model](#). The plane's starting position is perpendicular to the line of sight with its center of rotation at the current center of view; the visible portion of the model is behind the plane.

Activating **Use slab mode with thickness [d]** (with  $d > 0$ ) limits the visible portion of the [current model](#) to a slab  $d$  units thick (default 5.0). The units depend on the data but are usually Å.

The clipping plane can be moved [with the mouse](#) and/or by using:

- **Orient plane** - place the plane perpendicular to the line of sight with its center of rotation at the current center of view, with the visible portion of the model behind the plane
- **Flip plane** - flip the plane so that the previously visible portion of the model is invisible and vice versa
- **Align plane** - align the clipping plane with that of another model (chosen from the pulldown menu); both planes must be [enabled](#)

**Surface capping...** brings up the [Surface Capping](#) dialog for controlling the appearance of planar caps where surfaces are sliced away.

**Close** closes the **Per-Model Clipping** interface and turns off mouse control of all per-model clipping planes. Any per-model planes that have been enabled, however, will remain enabled until explicitly disabled (by unchecking **Enable clipping**). **Help** opens this manual page in a browser window.



## Mouse Modes

Activating **Adjust clipping with mouse as below** allows the clipping plane to be moved independently with the mouse. The **Clip Motion Assignments** determine which buttons are used. **Translation** moves the plane along its normal without changing the angle; the direction of translation caused by a particular cursor motion depends on the angle of the plane. **Rotation** changes the angle of the plane, with cursor control similar to that of standard rotation. During such manipulations, the plane's position is illustrated with a semitransparent disk. The plane-controlling mouse modes are shown in the [Mouse preferences](#) after **Per-Model Clipping** has been started.

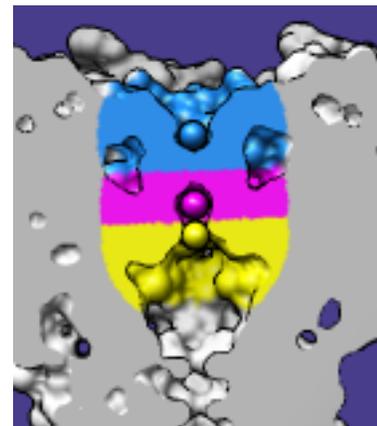
Changing the [current model](#) turns off mouse control of the previous model's clipping plane. When mouse control is turned off for all per-model clipping planes, the mouse buttons return to their previous functions (as shown in the [Mouse preferences](#)). During standard rotation and translation operations, each per-model clipping plane will move along with its associated model. If a model is not [active](#), neither it nor its clipping plane will move.


 A rectangular button with a light gray background and a thin border. The text "Color" is on the top line and "Zone" is on the bottom line, both in a dark gray, sans-serif font.

## Color Zone

**Color Zone** colors [surface models](#) and their [caps](#) to match nearby [selected](#) atoms or [markers](#). When a [volume data](#) isosurface is colored by zone, the data can be [split](#) into multiple sets accordingly. **Color Zone** information is included in saved [sessions](#). See also: [Segment Map](#), [Surface Color](#), [Surface Zone](#), [Surface Capping](#), [bondzone](#), [msc](#)

There are [several ways to start Color Zone](#), a tool in the **Depiction** and **Volume Data** categories (including from the [Volume Viewer Tools](#) menu). It is also implemented as the **zone** option of the command [scolor](#).



The **Surface** of interest should be chosen from the list of available surface models. Surfaces from [Volume Viewer](#) will be named according to the volume data set. Surfaces from [Multiscale Models](#) will be named for the PDB file from which the surfaces were generated.

Clicking **Color** colors vertices in the chosen surface that are within the specified distance (**Coloring radius**) of any of the currently [selected](#) atoms. When multiple selected atoms are within the cutoff distance, a vertex will be colored to match the closest one. Vertices not within the cutoff of any selected atom are assigned the default surface color (single-color mode). Each surface triangle is colored by linearly interpolating its vertex colors. Colors are defined by red, green, blue and opacity/transparency components.

By default, a zone is based only on distances to any [selected](#) atoms (and [Volume Tracer](#) markers); the [bondzone](#) command indicates that points along any [selected](#) bonds (and [Volume Tracer](#) links) should also be used. Vertices closer to a generated bond point than to an atom in the [selection](#) will then be assigned the [bond color](#).

The **Coloring radius** can be adjusted by moving the slider or by typing in a new value. If the set of selected atoms is changed, or if a selected atom is recolored or repositioned relative to the surface, **Color** must be clicked again to update the surface coloring. Clicking **Color** will not update the surface colors when no atoms are selected.

Clicking **Uncolor** returns the surface to its default color (single-color mode).

When the zone-colored surface is a contour surface from [Volume Viewer](#), clicking **Split Map** splits the corresponding [volume data](#) into separate data sets for the zones. A data set is created for each distinct color and for the "uncolored" data (in the zone remaining when the color zones are subtracted from the original volume). Values outside the respective zones are set to zero. The new data sets are named by appending numbers (0, 1, 2, ...) to the name of the original volume data. The number 0 is associated with the uncolored region, whereas the other numbers correspond to the distinct colors in an unpredictable order. The original volume data is hidden but not modified. Each data set can be [saved to a file](#) with [Volume Viewer](#).

**Close** dismisses the dialog, and **Help** opens this manual page in a browser window.

## LIMITATIONS

**Only one radius used.** There is no way to use different radii for different atoms.

**Surfaces cannot be made transparent independent of the atoms.** Surfaces colored with **Color Zone** to match nearby atoms can only be transparent if the corresponding atom colors are transparent. Transparent colors can be created with the [Color Editor](#) or the command [colordef](#).

---

UCSF Computer Graphics Laboratory / March 2011



## PseudoBond Reader

[Pseudobonds](#) are lines drawn between atoms to signify connections other than standard bonds.

**PseudoBond Reader** creates arbitrary pseudobonds by reading in a text file that lists the pairs of atoms to be connected. Alternatively, if only a few pseudobonds need to be created, they can be added interactively as [distance monitors](#) (the distance labels can be turned off). See also: [Metal Geometry](#)

There are [several ways to start PseudoBond Reader](#), a tool in the **Depiction** category. Starting **PseudoBond Reader** brings up a dialog for opening a user-created file of pseudobond information. Each line in the file describes one pseudobond and is of the form:

```
atom1 atom2 [color [label_text]]
```

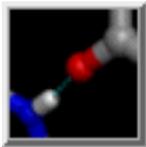
where *atom1* and *atom2* are [atom specifications](#) narrow enough to identify single atoms. Square brackets enclose optional fields. Fields are separated by spaces, except that *label\_text* can contain any number of spaces. If a label is desired, a color must also be specified. Colors can be specified by [name](#) or by [Tk color code](#). Examples of suitable lines of input:

```
#0:1@ca #0:2@ca dodger blue this is "dodger blue"
:3@n #2:4@o #55ff88 a Tk code
#0:279.a@n #1:133.b@n red
:7@ca :8@ca
```

The label for the first pseudobond in the example is **this is "dodger blue"** and the label for the second is **a Tk code** (the other two do not have labels). The label will be the same color as the corresponding pseudobond; label font and size can be set in the [Labels preferences](#).

Pseudobonds generated with the same input file are placed into a single [group](#). Pseudobond **group name**, **line width**, and default model numbers for the specified atoms can be supplied. If no group name is given, the pathname of the input file is used. A default model number is used only for lines of input in which it is not specified. Since groups can be treated collectively in terms of attributes, it may be convenient to generate different sets of pseudobonds with different input files (to yield distinct groups). If pseudobonds with the same group name already exist, the new pseudobonds can either be added to the group or used to replace its previous members (if **clear group...** is set to **true**).

The display, line style, color, and other attributes of pseudobonds and their groups can be controlled using the [pseudobond attributes panel](#), the [Selection Inspector](#), and the command [setattr](#).



## FindHBond

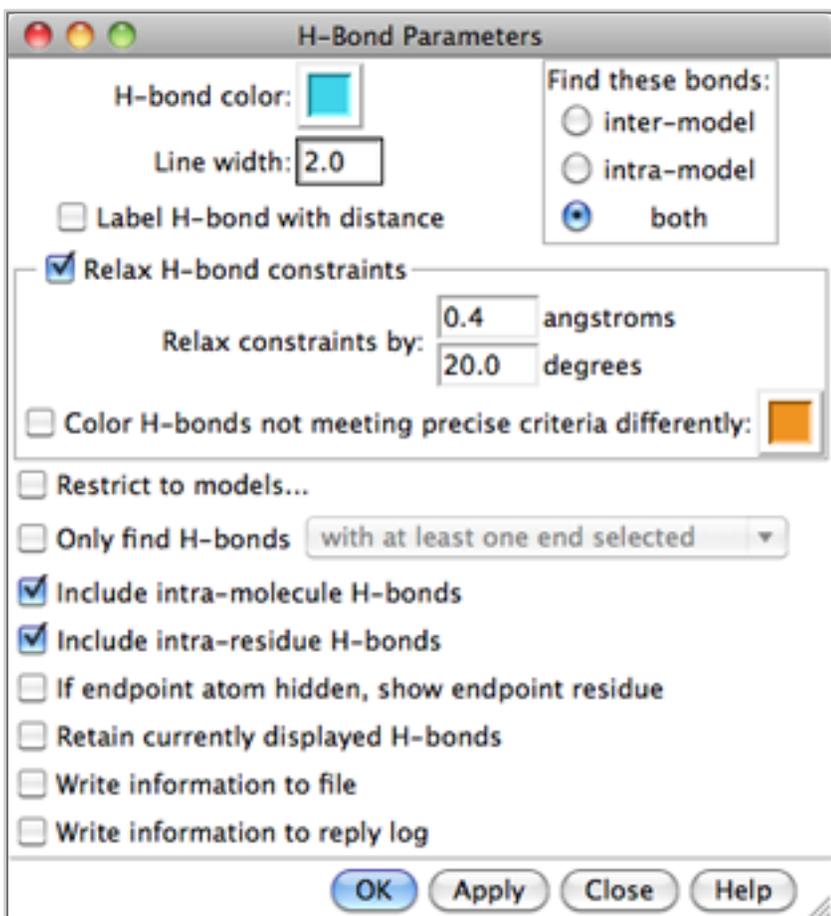
**FindHBond** uses [atom types](#) and [geometric criteria](#) to identify possible hydrogen bonds (H-bonds). It is not necessary for hydrogen atoms to be present. All potential H-bonding interactions fulfilling the criteria are shown. For example, even if it is not possible for a particular hydroxyl group to donate a hydrogen bond to two particular acceptors *simultaneously*, both possibilities will be displayed if the hydroxyl lacks an explicit hydrogen atom. If the hydroxyl group has an explicit hydrogen atom, however, only H-bonds compatible with the position of the hydrogen will be found. See also: [AddH](#), [Distances](#), [Find Clashes/Contacts](#), [Intersurf](#), [Rotamers](#), [Metal Geometry](#)

There are [several ways to start FindHBond](#), a tool in the **Structure Analysis** and **Surface/Binding Analysis** categories. **FindHBond** is also implemented as the command [hbonds](#) (or [findhbond](#)).

The H-bonds will be shown as lines (a group of [pseudobonds](#) named **hydrogen bonds**) between the donor and acceptor atoms, or if hydrogens are present, between the hydrogens and acceptor atoms. Appearance options:

- **H-bond color** (a [color well](#))
- **Line width**
- **Label H-bond with distance** - if active, there will be a button to **Show distance formatting options**, namely the [Distances](#) panel for setting the number of decimal points and whether the Å symbol is displayed (other settings in that tool will not affect **hydrogen bonds** pseudobonds)

\*\*If the endpoint atoms are subsequently moved relative to each other, the lines will “stretch” rather than disappearing, even if the atoms no longer meet the criteria for hydrogen bonding.\*\* After such changes, the H-bonds should be updated by re-running the calculation or removed with the command [~hbonds](#).



**Relax H-bond constraints** indicates that tolerances to **Relax constraints by** should be applied to the precise [geometric criteria](#). These criteria come from a survey of high-resolution structures; empirically, tolerances of **0.4 angstroms** and **20.0 degrees** work well for most macromolecular structures. H-bonds within the tolerances but that do not meet the precise criteria can be colored differently than the H-bonds that meet the precise criteria.

The scope of the calculation is controlled under **Find these bonds:**

- **inter-model** - between models only
- **intra-model** - within models only
- **both** (default)

Regardless of the scope, the calculation will not include interactions between different [submodels](#) of the same model (although this can be done with the [hbonds](#) command). The calculation can be further restricted with the options:

- **Restrict to models...** show a list of molecule models from which one or more can be chosen. Individual models or blocks of models can be chosen with the left mouse button. Ctrl-click toggles the status of an individual model. To choose a block of models without dragging, click on the first (or last) and then Shift-click on the last (or first) in the desired block.
- **Only find H-bonds**
  - with at least one end selected
  - with exactly one end selected
  - with both ends selected
  - between selection and atom spec... [ [atom-spec](#) ]

Whether to limit H-bond detection based on the current [selection](#). In the last option, the selection and the [atom-spec](#) may overlap, but the specification will not overrule other restrictions (for example, if the specified atoms are in a different model than the selected atoms but **inter-model** detection is turned off, no H-bonds will be found).

- **Include intra-molecule H-bonds** - whether to include intramolecular H-bonds, where a molecule is defined as a covalently bonded set of atoms
- **Include intra-residue H-bonds** - whether to include H-bonds between atoms in the same residue

Additional options:

- **If endpoint atom hidden, show endpoint residue** - turn on the display of residues containing an H-bonding atom if that atom is not displayed initially (By default, if the atom on either end of a [pseudobond](#) representing an H-bond is not shown, the pseudobond itself is not shown, although it still exists; displaying the atom allows the H-bond to be shown.)
- **Retain currently displayed H-bonds** - retain a previously determined set of H-bonds through a subsequent round of H-bond detection
- **Write information to file** - write H-bond information (atom specifications and distances) to a file. The [dialog](#) for saving the file includes a **Naming style** option to control [how atoms will be listed](#):
  - **simple** (for example, **HIS 16.A ND1**)
  - [command-line specifier](#) (for example, **:16.A@ND1**)
  - **serial number** (for example, **126**)
- **Write information to reply log** - write H-bond information (atom specifications and distances) to the [Reply Log](#). The **Atomspec display style** setting in the [Labels preferences](#) controls [how atoms will be listed](#).

**OK** starts the calculation and dismisses the panel, while **Apply** starts the calculation without dismissing the panel. The calculation may take several seconds; progress is reported in the [status line](#). **Close** dismisses the panel without performing any calculation. **Help** brings up this manual page in a browser window.

## METHOD AND CRITERIA

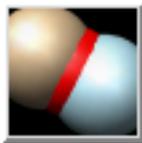
The geometric criteria are based on a survey of small-molecule crystal structures, as described in

[Three-dimensional hydrogen-bond geometry and probability information from a crystal survey](#). Mills JE, Dean PM. *J Comput Aided Mol Des*. 1996 Dec;10(6):607-22.

There are many different sets of geometric criteria, corresponding to the many different donor-acceptor combinations (see Tables 5-8 in the reference). There is an upper bound on distance and one or more angular range criteria for each category of H-bond. It is generally useful to relax the criteria since most structures are not as high-resolution as those used in the survey. User-specified [tolerances](#) are added to each criterion's upper bound and/or subtracted from its lower bound (depending on the particular criterion).

Chimera uses atom and residue names, or if these are not "standard," the coordinates of atoms, to determine connectivity and [atom types](#), which in turn determine which geometric criteria are appropriate for detecting a hydrogen bond between specific atoms. Possible donor groups are hydrogen-bearing nitrogen, oxygen, and sulfur atoms, and possible acceptor groups are nitrogen, oxygen, and sulfur atoms with a lone pair. H-bonds involving other types of atoms are not considered. Not all of the possible donor-acceptor combinations were listed in the Mills and Dean parameter tables; some categories of sulfur were not surveyed, and several combinations of the surveyed types were not observed frequently enough to collect good statistics on their geometries. In such cases, **FindHBond** applies estimated H-bond criteria.

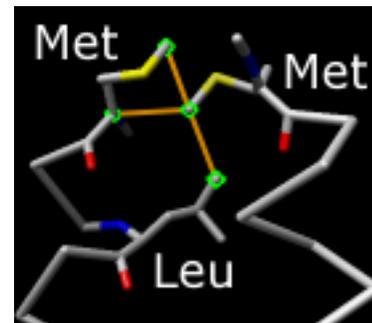
When there are no explicit hydrogens on a potential donor atom, **FindHBond** will use the donor [atom type](#) to infer their presence. For functional groups where the hydrogen positions are well-determined by the positions of the nonhydrogen atoms, the H-bonds detected in the presence and absence of explicit hydrogen atoms should be virtually identical. For functional groups where the hydrogen position is not well-determined (such as rotatable hydroxyls), the presence of an explicit hydrogen atom will limit the detected H-bonds to those appropriate for the observed position, instead of all of its possible positions.



## Find Clashes/Contacts

**Find Clashes/Contacts** identifies interatomic clashes and contacts based on [VDW radii](#) and user-specified [criteria](#). Terminology:

- *clashes* - unfavorable interactions where atoms are too close together; close contacts
- *contacts* - all kinds of direct interactions: polar and nonpolar, favorable and unfavorable (including clashes)



During continuous monitoring, such interactions can be shown with [selection](#), coloring, and [pseudobonds](#). Discontinuous monitoring additionally allows writing out information and assigning the largest [overlap](#) per atom as an [attribute](#) named **overlap**. See also: [FindHBond](#), [Intersurf](#), [Rotamers](#), [RR Distance Maps](#), [Crystal Contacts](#)

There are [several ways to start Find Clashes/Contacts](#), a tool in the **Structure Analysis** and **Surface/Binding Analysis** categories. It is also implemented as the command [findclash](#).

**Atoms to Check** are specified by [selecting](#) atoms, clicking **Designate**, and indicating which interactions should be found:

- **themselves** - interactions among the designated atoms
- **all other atoms** (default) - interactions between the designated atoms and all other atoms
- **other atoms in same model** - intra-model interactions between the designated atoms and all other atoms
- **second set of designated atoms** - interactions between the designated atoms and a second designated set (specified by [selecting](#) atoms and clicking **Designate selection as second set**)

Interactions between atoms in different submodels of the same model are ignored (note the [findclash](#) command has an option to include such interactions, however). Changing the [selection](#) will not automatically change any designated sets.

The **Clash/Contact Parameters** control what will be considered a clash or contact. The *overlap* between two atoms is defined as the sum of their [VDW radii](#) minus the distance between them and minus an [allowance](#) for potentially hydrogen-bonded pairs:

$$overlap_{ij} = r_{VDW_i} + r_{VDW_j} - d_{ij} - allowance_{ij}$$

- **Find atoms with VDW overlap  $\geq$  [ cutoff ] angstroms** - pairs of atoms with *overlap*  $\geq$  *cutoff* will be identified. A larger positive *cutoff* restricts the results to more severe clashes, whereas a negative *cutoff* can also identify favorable contacts.
- **Subtract [ allowance ] from overlap for potentially H-bonding pairs** - an *allowance*  $>$  0 reflects

the observation that atoms sharing a hydrogen bond can come closer to each other than would be expected from their [VDW radii](#). The *allowance* is only subtracted for pairs comprised of a donor (or donor-borne hydrogen) and an acceptor. This is equivalent to using smaller radii to characterize hydrogen-bonding interactions (for example, see [Li and Nussinov, Proteins 32:111 \(1998\)](#)). As in [FindHBond](#), possible donor groups are hydrogen-bearing nitrogen, oxygen, and sulfur atoms, and possible acceptor groups are nitrogen, oxygen, and sulfur atoms with a lone pair.

For detecting [clashes](#), *cutoff* values of 0.4-1.0 Å and *allowance* values of 0.2-0.6 Å are generally reasonable (default **clash** criteria **0.6** and **0.4** Å, respectively).

For detecting [contacts](#), negative *cutoff* values of 0.0-(-1.0) Å with an *allowance* of 0.0 Å are generally reasonable (default **contact** criteria **-0.4** and **0.0** Å, respectively).

- **Ignore contacts of pairs [ *N* ] or fewer bonds apart** (*N*=4 by default)
- **Include intra-residue contacts** (default off)
- **Include intra-molecule contacts** (default on) - whether to include intramolecular interactions, where a molecule is defined as a covalently bonded set of atoms

#### Options for the Treatment of Clash/Contact Atoms:

- **Select** - whether to [select](#) atoms meeting the clash/contact [criteria](#) (and deselect all other atoms)
- **Color [ [color well](#) ] (and color all other atoms [ [color well](#) ])** - whether to color atoms meeting and not meeting the clash/contact [criteria](#) (the default colors are **red** and **No color**, respectively; using **No color** reveals [model-level colors](#))
- **Draw pseudobonds of color [ [color well](#) ] and width [ *linewidth* ]** - whether to draw [pseudobonds](#) of the specified color (default **yellow**) and linewidth (default **2.0**) between pairs meeting the clash/contact [criteria](#). These pseudobonds can be removed with the command [~findclash](#). The [PseudoBond Panel](#) can also be used to close the pseudobond group (named **contacts**) or alter its display.
- **If endpoint atom hidden, show endpoint residue** - turn on the display of residues containing a contact atom if that atom is not displayed initially (By default, if the atom on either end of a [pseudobond](#) representing a contact is not shown, the pseudobond itself is not shown, although it still exists; displaying the atom allows the contact pseudobond to be shown.)

The following are only available for discontinuous checking ([when OK/Apply clicked](#)):

- **Assign 'overlap' attribute** - whether to assign the largest [overlap](#) per atom as an [attribute](#) named **overlap** and open the [Render by Attribute](#) tool
- **Write information to file** - whether to write clash/contact information to a text file. The [dialog](#) for saving the file includes a **Naming style** option to control [how atoms will be listed](#):
  - **simple** (for example, **HIS 16.A ND1**)
  - [command-line specifier](#) (for example, **:16.A@ND1**)
  - **serial number** (for example, **126**)
- **Write information to reply log** - whether to write clash/contact information to the [Reply Log](#). The **Atomspec display style** setting in the [Labels preferences](#) controls [how atoms will be listed](#).

### Options for the Frequency of Checking:

- **when OK/Apply clicked** (default) - this option is compatible with all [treatment options](#), including attribute assignment and writing out clash/contact information. Clicking **OK** runs the calculation and dismisses the dialog, whereas clicking **Apply** runs the calculation without dismissing the dialog.
- **after relative motions (until dialog closed)** - whenever relative motions are paused; only selection and display [treatments](#) are available. Relative motions include [bond rotations](#) and translations/rotations of one model relative to another, as opposed to translations/rotations of all models collectively.
- **continuously (until dialog closed)** - during motions; only selection and display [treatments](#) are available

## Structure Measurements

The **Structure Measurements** panel has four sections:

- [Distances](#) - monitor distances
- [Angles/Torsions](#) - monitor angles and torsions
- [Adjust Torsions](#) (now located in [Build Structure](#)) - rotate bonds
- [Axes/Planes/Centroids](#) - show axes, planes, and centroids based on sets of atoms, perform related measurements

Only one section is shown at a time, and clicking the tab for another brings it to the front. Each section is listed as a tool in the **Structure Analysis** category. Tools can be started in [several ways](#). See also: [Metal Geometry](#), [Ramachandran Plot](#), [CASTp Data](#), [measure](#)

**Atomspec display style** in the [Labels preferences](#) controls [how atoms are listed](#) in the **Structure Measurements** dialog. The [Distances figure](#) shows the simple style (residue name, residue specifier, and atom name). For an example of [command-line specifiers](#), see the [Adjust Torsions figure](#).

**Save** saves measurements to a file. **Close** closes the measurements panel, and **Help** opens this manual page in a browser window.

### Distances Distances

The **Distances** section of [Structure Measurements](#) is a table of distance monitors (measurements that update if there are changes). Distance monitors are saved in [sessions](#). See also: [FindHBond](#), [distance](#)

There are [several ways to start Distances](#), a tool in the **Structure Analysis** category. Distance monitors can be created in several ways:

- [Selecting](#) exactly two atoms, then clicking the **Create** button on the dialog. With default [mouse settings](#), two atoms can be [picked](#) (selected from the graphics window) by Ctrl-clicking the first and Shift-Ctrl-clicking the second.
- Doubleclick-[picking](#) an atom when one other atom is already [selected](#), then choosing **Show Distance** from the resulting [context menu](#).
- Doubleclick-[picking](#) an atom when 0 or >3 other atoms are already [selected](#), then choosing **Show Distances to Nearby Residues** from the resulting [context menu](#).
- Using the [distance](#) command.

The distance list can be sorted by the values in any column by clicking the header. Clicking the header once sorts the entries in order of increasing value and places an up

Distances - simple atom listings

arrowhead (triangle) in the header. Clicking again sorts the entries in decreasing order and places a down arrowhead (inverted triangle) in the header.

A distance monitor can be *chosen* by clicking on its line, and more than one can be chosen at a time. Chosen lines are highlighted in the dialog. **Ctrl**-click toggles the status of a line, while clicking on the first (or last) line of a contiguous block and then **Shift**-clicking on the last (or first) chooses all of the lines in the block.

Choosing a distance listing [selects](#) the corresponding distance [pseudobond](#). The **Choosing in table...** option allows additionally [selecting](#) the corresponding atoms.

**Remove** deletes the chosen

distance monitors. If there is only one, it is not necessary to choose it before using **Remove**.

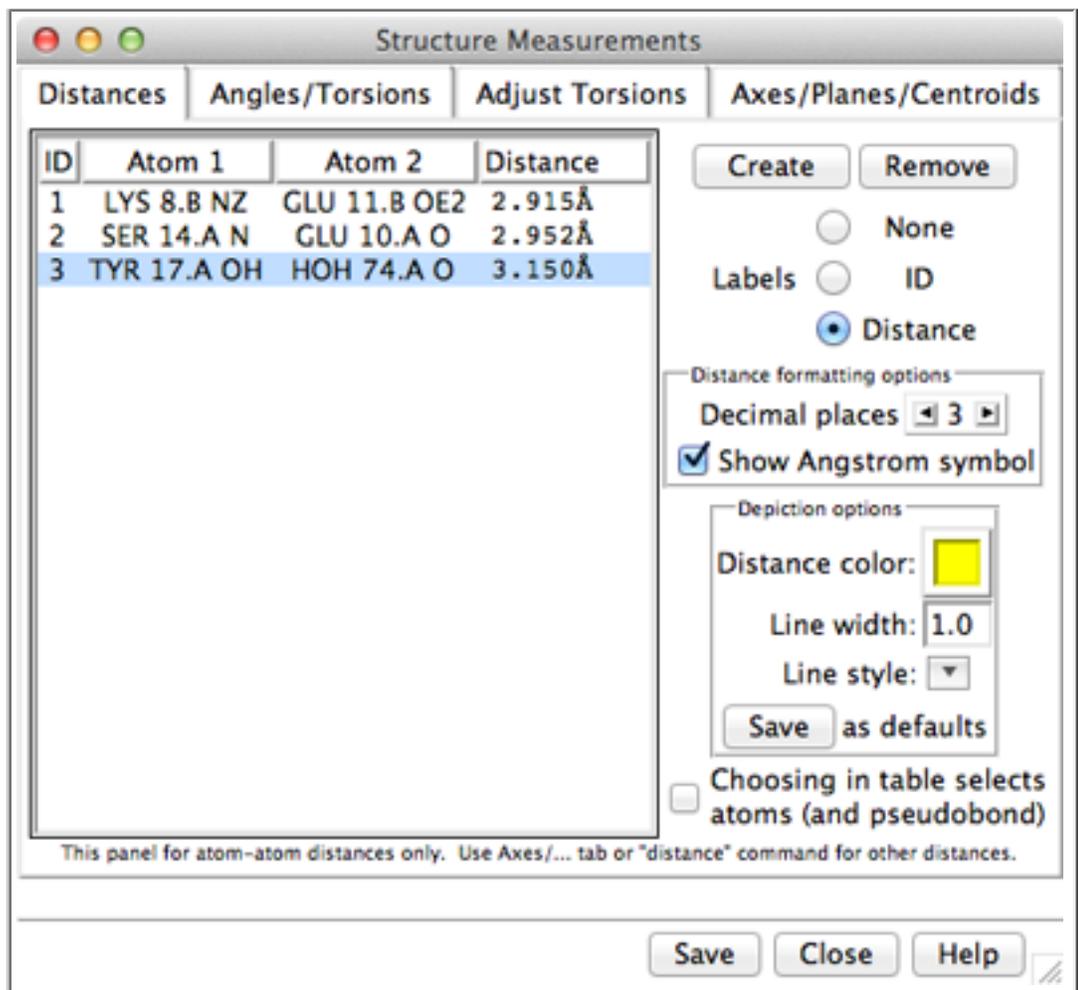
The **Labels** setting applies to all distance monitors and can be switched among:

- **None** - no label
- **ID** - distance monitor ID number (the first column in the table)
- **Distance** - distance value in the same units as the atomic coordinates, normally Å

The formatting options **Decimal places** and **Show Angstrom symbol** control how distance values are shown in the table and (when **Labels** is set to **Distance**) in labels. The **Decimal places** can be changed by clicking or holding down one of the arrows on either side of the value. Of course, decimal places beyond those present in the input coordinates are not very meaningful. Changes in distance formatting options are automatically saved in a user's [preferences file](#) and also apply to distances shown with [FindHBond](#).

The lines representing distance monitors are [pseudobonds](#) in a group named **distance monitor**. The following depiction options apply to the whole group:

- **Distance color** (a [color well](#)) - pseudobond group color, see [coloring hierarchy](#)
- **Line width** - linewidth in pixels
- **Line style** - whether solid, dashed, dotted, *etc.*
- **Save as defaults** - save the current distance color, linewidth, and line style settings in the user's [preferences file](#)



Settings of individual pseudobonds (as well as pseudobond groups) can be changed using the the [Selection Inspector](#) or the command [setattr](#).

## Angles/Torsions

The **Angles/Torsions** section of [Structure Measurements](#) is a table of angle monitors (measurements that update if there are changes). A “bond angle” is measured for three atoms and a “torsion angle” for four atoms; however, it is not necessary for the atoms to be contiguous or even bonded to one another. Angle monitors are saved in [sessions](#).

**Angles/Torsions** reports changes but cannot be used to change the angles. Angles can be changed using [Adjust Bond Angles](#) and [Adjust Torsions](#).

There are [several ways to start Angles/Torsions](#), a tool in the **Structure Analysis** category. Angle monitors can be created in various ways:

- [Selecting](#) exactly three atoms for a bond angle or four atoms for a torsion angle, then clicking the **Create** button on the dialog. With default [mouse settings](#), atoms can be [picked](#) (selected from the graphics window) by Ctrl-clicking the first and Shift-Ctrl-clicking additional atoms.
- Doubleclick-[picking](#) an atom when two other atoms are already [selected](#), then choosing **Measure Angle** from the resulting [context menu](#).
- Doubleclick-[picking](#) an atom when three other atoms are already [selected](#), then choosing **Measure Torsion** from the resulting [context menu](#).

The angles/torsions list can be sorted by the values in any column by clicking the header. Clicking the header once sorts the entries in order of increasing value and places an up arrowhead (triangle) in the header. Clicking again sorts the entries in decreasing order and places a down arrowhead (inverted triangle) in the header.

A measurement in the list can be *chosen* by clicking on its line, and more than one can be chosen at a time. Chosen lines are highlighted in the dialog. **Ctrl**-click toggles the status of a line, while clicking on the first (or last) line of a contiguous block and then **Shift**-clicking on the last (or first) chooses all of the lines in the block.

Choosing an angle monitor also [selects](#) the corresponding atoms and bonds when the **Choosing in table...** option is on. **Remove** deletes the chosen measurement(s). If there is only one measurement, it is not necessary to choose it before clicking **Remove**.

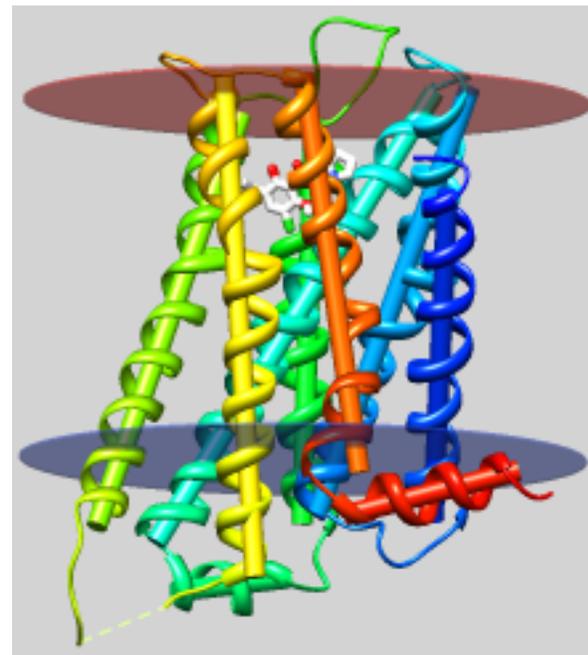
The **Decimal places** setting controls how angles are reported in the table; the number of digits shown after the decimal can be changed by clicking or holding down one of the arrows on either side of the value.

The command [angle](#) can also be used to measure bond angles and torsions; however, it yields a static measurement rather than a continuously updating monitor.



## Axes/Planes/Centroids

The **Axes/Planes/Centroids** section of [Structure Measurements](#) allows defining geometric objects (axes, planes, centroids) based on sets of atoms and performing measurements involving the objects. Axes can also be calculated from centroids or some combination of atoms and centroids. Axes are displayed as rods, planes as disks, and centroids as spheres. Objects to be listed in the [Axes/Planes/Centroids table](#) can also be created with the command [define](#). Each object is created as a [surface model](#) (hidden from the [Model Panel](#)) in the coordinate system of the lowest-numbered model involved in its calculation. **Axes/Planes/Centroids** objects and information are saved in [sessions](#). See also: [distance](#), [angle](#), [align](#), [shape](#), [measure inertia](#), [PipesAndPlanks](#), [Thermal Ellipsoids](#), [geometric objects](#)



dopamine D3 receptor  
helix axes, membrane planes  
(see [Image Gallery entry](#))

There are [several ways to start Axes/Planes/Centroids](#), a tool in the **Structure Analysis** category.

Clicking **Define axes...** opens a dialog for specifying atom/centroid sets and other axis parameters. Eigenvectors/values are calculated from the coordinates of each set of atoms/centroids after subtracting the position of their collective centroid. Each axis is anchored at the corresponding collective centroid and aligned with the first eigenvector of the corresponding coordinates (prior to any [helical correction](#)).

- **Each helix in [models]** - define an axis for each peptide/protein helix in the molecule model(s) chosen in the list. Peptide/protein helix assignments are taken from the input structure file or generated with [ksdssp](#). Only the backbone atoms N, CA, C are used to define the axes. Sometimes two helices are adjacent in sequence (not separated by any other residues), and the integer-valued residue attribute **ssld** is used to distinguish them. Although such cases are detected automatically, **ssld** can also be set manually with [setattr](#).
- **Selected atoms/centroids (axis name [name])** - define a single axis named *name* using all [selected](#) atoms and/or centroid objects
- **Axis Parameters:**
  - **Mass weighting** (only available for the **Selected atoms/centroids** option) - calculate major inertial axis by mass-weighting the atoms/centroids; cannot be combined with [helical correction](#). The radius of a centroid (Å) is used as its “mass” (daltons).
  - **Use helical correction** - adjust axis orientation to reduce the spread of atom-axis distances; cannot be combined with [mass weighting](#). While usually small, helical correction improves results for short helices.
  - **Replace existing axes** - whether to remove any/all pre-existing axis objects
  - **Color** (a [color well](#), default **No Color**, meaning to match the structure as much as possible)
  - **Radius**
    - **average axis-atom distance** (default) - if this setting is chosen but an axis is

based on only two points (atoms and/or centroid objects), the last used constant value (below) will be applied instead

- **angstroms** (a constant value)

Only one [color well](#) setting and one constant radius value can be specified at a time, but axes with different colors and radii can be created in successive stages when [Replace existing axes](#) is turned off. Clicking **Apply** (or **OK**, which also closes the dialog) calculates the axes, adds them to the [table](#), and generates the corresponding objects. Axis radii are reported in the [Reply Log](#).

Clicking **Define plane...** opens a dialog for defining a plane based on the [selected](#) atoms. Eigenvectors/values are calculated from the atomic coordinates after subtracting the position of their non-mass-weighted centroid. The plane is anchored at the centroid and aligned with the first two eigenvectors (the third eigenvector is normal to the plane).

- **Plane name**
- **Replace existing planes** (true/false) - whether to remove any/all pre-existing plane objects
- **Color** (a [color well](#), default **No Color**, meaning to match the source atoms as much as possible)
- **Set disk size to enclose atom projections** (true/false) - whether the disk radius should be set to span the projections of the defining atoms back onto the plane. If true, an **Extra radius (padding)** can be specified (default **0.0 Å**). If false, a **Fixed radius** can be specified (default **10.0 Å**). The disk center is the non-mass-weighted centroid of the atoms.
- **Disk thickness** (Å)

Clicking **Apply** (or **OK**, which also closes the dialog) calculates a plane based on the [selected](#) atoms, adds it to the [table](#), and generates the corresponding object. Plane radius is reported in the [Reply Log](#).

Clicking **Define centroid...** opens a dialog for defining a centroid based on the [selected](#) atoms:

- **Centroid name**
- **Mass weighting** (true/false) - whether to mass-weight atoms when calculating the centroid
- **Replace existing centroids** (true/false) - whether to remove any/all pre-existing centroid objects
- **Color** (a [color well](#), default **No Color**, meaning to match the source atoms as much as possible)
- **Radius** (Å)

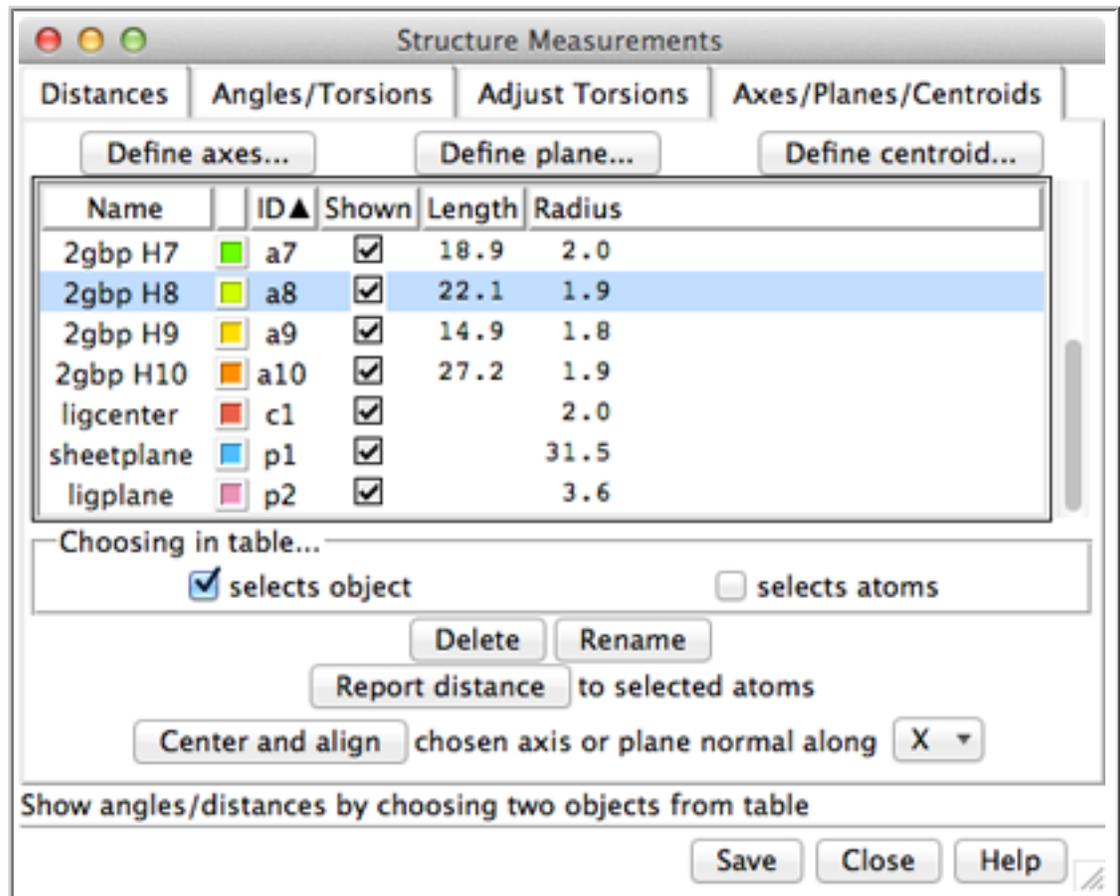
Clicking **Apply** (or **OK**, which also closes the dialog) calculates a centroid based on the [selected](#) atoms, adds it to the [table](#), and generates the corresponding object. Centroid coordinates are reported in the [Reply Log](#).

Columns in the **Axes/Planes/Centroids** table of objects:

- **Name**
- **color** - an object can be recolored by clicking its [color well](#) and using the

[Color Editor](#), or by [selecting](#) the object and using the [Actions... Color menu](#)

- **ID** - object identifier, aN for axes, pN for planes, cN for centroids
- **Shown** - whether the object is displayed
- **Length** - axis length, end-to-end distance needed to span the projections of the defining atoms back onto the axis
- **Radius** - object radius



The table can be sorted by a column by clicking the column header.

An object can be *chosen* by clicking on its line, and more than one can be chosen at a time. Chosen lines are highlighted in the dialog. **Ctrl**-click toggles the status of a line, while clicking on the first (or last) line of a contiguous block and then **Shift**-clicking on the last (or first) chooses all of the lines in the block.

Choosing any two objects reports their applicable geometric relationships below the table and in the [Reply Log](#). For angle measurements, axes and planes are treated as infinite. For distance measurements, axes are treated as finite line segments, while planes are treated as infinite. Optionally, choosing objects also:

- **selects object** - [selects](#) each object (currently nothing can be done to such a selection, but the objects are highlighted in the graphics window)
- **selects atoms** - [selects](#) the atoms used to define each object

Conversely, [picking](#) an object from the screen will choose it in the table.

Clicking **Delete** removes the chosen objects. In addition, closing a molecule model or deleting atoms removes any objects defined using those atoms. **Rename** allows changing the names of the chosen objects; this will change their order in the table if sorting is currently by name.

**Report distance** measures distances between the chosen objects and [selected](#) atoms; results are sent to the [Reply Log](#). If a single atom is selected, the distance to each chosen object is reported; if multiple atoms are selected, the minimum, maximum, and mean distances are reported.

**Center and align** is available when a single axis or plane object is chosen. It centers and aligns the chosen axis, or centers the chosen plane and aligns its normal vector, along the specified direction in the laboratory coordinate system:

- **X** - horizontal in the plane of the screen
- **Y** - vertical in the plane of the screen
- **Z** - perpendicular to the screen, along the line of sight

There is no control over which end of the axis or normal vector goes in the positive direction, but the view can be flipped 180° with a command, for example:

### [turn y 180 center view](#)

Axis/plane/centroid information written out with **Save** includes name and *untransformed* center coordinates  $(x_c, y_c, z_c)$ . Axis information also includes length and orientation expressed as a unit vector  $(x_u, y_u, z_u)$ . Axis endpoint coordinates  $(x_1, y_1, z_1)$   $(x_2, y_2, z_2)$  can be generated as follows:

$$x_1 = x_c - 0.5(\text{length})(x_u)$$

$$y_1 = y_c - 0.5(\text{length})(y_u)$$

$$z_1 = z_c - 0.5(\text{length})(z_u)$$

$$x_2 = x_c + 0.5(\text{length})(x_u)$$

$$y_2 = y_c + 0.5(\text{length})(y_u)$$

$$z_2 = z_c + 0.5(\text{length})(z_u)$$

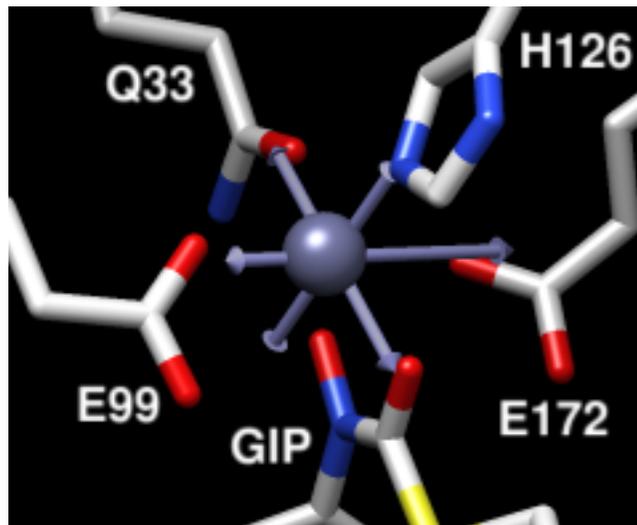
Plane information also includes radius and orientation expressed as a normal unit vector.



## Metal Geometry

Metal Geometry is a tool for analyzing metal ion coordination geometries. It also allows adding or deleting coordination [pseudobonds](#). See also: [Structure Measurements](#), [FindHBond](#), [Build Structure](#), [PseudoBond Reader](#)

There are [several ways to start Metal Geometry](#), a tool in the **Structure Analysis** category. If more than one metal ion is present, the ion of interest should be chosen from the pulldown menu at the top of the dialog. **Metal Geometry** can also be started by by double-[picking](#) a metal ion from the graphics window (doubleclicking it with the button assigned to [picking](#)) and choosing **Coordination Geometry** from the resulting [context menu](#).



The view in the graphics window will focus on the specified metal ion and nearby atoms. The **Metal transparency** slider allows adjusting the transparency of the displayed metal ion.

The **Coordination Table** lists atoms that might be coordinated to the metal ion, sorted by distance. Each row represents a *ligation set* comprised of not only the atom named in that row, but also all of the atoms in the rows above it. Clicking a row chooses that ligation set as the *current ligation set* and [selects](#) its atoms. The rows cannot be reordered, but atoms can be added to or removed from the list by [selecting](#) them in the graphics window and clicking the **Add** or **Remove** buttons on the right. Only one location of an atom with alternate locations can be listed at a time, and which location is listed can be controlled with the **Alt loc** menu to the right of the table. Columns:

Coordinator	Distance (Å)	Distance RMSD	Best Geometry
GLU 99.A OE1	1.91	0.000	N/A
GIP 400.B OF2	2.04	0.058	tetrahedral
HIS 126.B NE2	2.05	0.266	trigonal bipyramid
GLN 33.A OE1	2.05	0.251	trigonal bipyramid
GIP 400.B OZ1	2.13	0.322	octahedron
GLU 172.B OE1	3.25	0.523	octahedron

Geometry	Coord. Number	Distance RMSD
octahedron	6	0.523
trigonal prism	6	0.692

- **Coordinator** - a potentially metal-ligating atom, listed in the **Atomspec display style** specified in the [Labels preferences](#)
- **Distance (Å)** - distance in Å from that atom to the metal ion
- **Distance RMSD** - root-mean-square deviation (RMSD) of the [ligation set](#) and metal ion from their idealized positions in the **Best Geometry** for that set ([details...](#))
- **Best Geometry** - the coordination geometry for which the [ligation set](#) has the lowest **Distance**

**RMSD** (additional geometries can be shown in the [Geometry Table](#))

Clicking **Create/Update** will generate or delete coordination [pseudobonds](#) to the metal as needed to represent the [current ligation set](#). Clicking **Display options...** brings up a list of [pseudobond attributes](#) for this group, allowing changes in properties such as color, linewidth, and whether the lines are dashed. When an individual [pseudobond](#) is [selected](#), its attributes can be altered with the [Selection Inspector](#).

The **Geometry Table** lists possible metal coordination geometries for the [current ligation set](#), subject to certain [screens](#). Columns:

- **Geometry** - coordination geometry
- **Coord. Number** - total number of metal-ligating atoms needed to fill the **Geometry**
- **Distance RMSD** - RMSD of the [ligation set](#) and metal ion from their idealized positions in the **Geometry** ([details...](#)); the **Geometry** for which this is lowest is reported as the **Best Geometry** in the [Coordination Table](#)

Clicking a row in the **Geometry Table** chooses that metal coordination geometry. **Depict idealized geometry** shows the chosen geometry with arrow objects, and water oxygens can be **Added** to occupy any unfilled sites. The distance from the metal at which to add water oxygens is automatically set to the average for the [current ligation set](#), but can be edited.

Unlikely geometries can be screened out of the tables:

- **Screen unfilled geometries** (default **off**) - disregard coordination geometries with [coordination number](#) greater than the number of atoms in the [current ligation set](#)
- **Screen "overcrowded" geometries** (default **on**) - disregard coordination geometries that would place ligating atoms too close together, given their likely distance from the metal ion. Clicking **Show parameters** expands the dialog to reveal additional settings for this screen:

A geometry is considered overcrowded if hypothetical metal-ligating atoms placed at the ideal angles for that geometry, at a distance equal to the [average | maximum | minimum] of the actual chosen ligating atoms, would be less than *mindist* (default 2.8) Å apart.

Clicking **Hide parameters** hides the overcrowding settings.

**Close** closes the **Metal Geometry** dialog. **Help** opens this manual page in a browser window.

## Distance RMSD

In an idealized coordination geometry, the ligating atoms fall along vectors that emanate from the metal ion. The angles between the vectors are characteristic of the specific **Geometry**. The radiating set of vectors is repositioned to coincide as much as possible with the real positions of the metal ion and ligating atoms, and then for each real ligating atom, an idealized ligating atom is placed along the corresponding vector at the same distance from the idealized metal ion as the real atom is from the real metal ion. The best-fit root-mean-square deviation between the real atoms (including the metal ion) and their idealized

positions is computed.

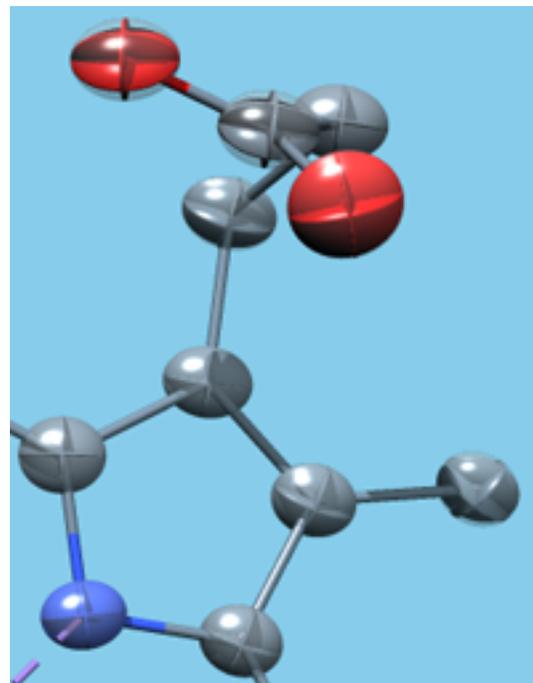


## Thermal Ellipsoids

**Thermal Ellipsoids** shows atomic anisotropic B-factors as ellipsoids, their principal axes, and/or their principal ellipses. These special depictions are created only for atoms with anisotropic B-factor information. Anisotropic B-factors are read from the input coordinate file (from ANISOU records in a PDB file or the analogous in CIF/mmCIF) and are included with only certain high-resolution structures. Ellipsoid axes and radii correspond to the eigenvectors and eigenvalues of the atomic mean-square displacement matrix. The radii are proportional to the root-mean-square displacements (RMSDs), the square roots of the eigenvalues.

Anisotropic B-factor depictions are [surface models](#) and are included in saved [sessions](#). See also: [Render by Attribute](#), [Axes/Planes/Centroids](#), [geometric objects](#)

There are [several ways to start Thermal Ellipsoids](#), a tool in the **Structure Analysis** category. It is also implemented as the command [aniso](#).



- **Scale factor** (default 1, no scaling) - controls ellipsoid size, where the unscaled ellipsoid radii equal the atomic RMSDs along the ellipsoid axes
- **Smoothing level** (default 3) - controls how many planar facets are used to draw ellipsoids and ellipses; higher values give smaller facets and a smoother appearance
- **Set scale factor for probability (%) [P]** - automatically fills in the **Scale factor** field for a specified % probability *P* (press return after typing in a value; *P* must be  $\geq 0\%$  and  $< 100\%$ )

The **Presets** menu in **Thermal Ellipsoids** provides quick access to several [predefined combinations](#) of the following options:

- **Depict ellipsoids** - whether to show ellipsoids
  - **Color** (a [color well](#), default **No Color**, meaning to match the corresponding atoms)
  - **Transparency** (a menu including **same as color** and several percentage transparency options)
- **Depict principal axes** - whether to show ellipsoid principal axes
  - **Color** (a [color well](#), default **No Color**, meaning to match the corresponding atoms)
  - **Length factor** (default 1.5) - length of principal axis depictions relative to ellipsoid size
  - **Thickness** (default 0.01 Å) - thickness of principal axis depictions (dimensions perpendicular to length)
- **Depict principal ellipses** - whether to show ellipsoid principal ellipses
  - **Color** (a [color well](#), default **No Color**, meaning to match the corresponding atoms)
  - **Size factor** (default 1.0) - size of principal ellipse depictions relative to ellipsoid size
  - **Thickness** (default 0.02 Å) - thickness of principal ellipse depictions

Clicking **Show Ellipsoids** or pressing return after entering a value creates the depictions; clicking **Hide Ellipsoids** removes them. When the **Restrict Show/Hide...** option is on and atoms are [selected](#), showing and hiding will affect only the depictions of the [selected](#) atoms. Otherwise, clicking **Show Ellipsoids** removes any pre-existing depictions and generates new ones for just the currently displayed atoms.

**Close** dismisses the dialog, and **Help** opens this manual page in a browser window.

## Thermal Ellipsoid Presets

The **Presets** menu in **Thermal Ellipsoids** includes several predefined combinations of [style settings](#):

- **Simple ellipsoid** - ellipsoids only, color from atoms
- **Principal axes** - axes only (length factor 1.0)
- **Principal ellipses** - ellipses only (size factor 1.0)
- **Ellipsoid and principal axes** - ellipsoids with axes (length factor 1.5)
- **Octant lines** - ellipsoids with black ellipses (size factor 1.01)
- **Snow globe axes** - white half-transparent ellipsoids with axes (length factor 0.99)
- **Snow globe ellipses** - white half-transparent ellipsoids with ellipses (size factor 0.99)

Presets do not include [scaling and smoothing settings](#), which are specified independently. Choosing a preset from the menu applies its settings immediately. Users can apply a built-in preset and then make further adjustments, or define and use their own custom presets:

- **Preset from current settings...** save a custom preset with the current [style settings](#) to the [preferences file](#)
- **Delete user preset...** remove a previously user-defined preset

## Display Tip

Since the ellipsoids may be obscured when atoms are shown as balls or spheres, using a wire or thin stick [representation](#) is recommended. Sticks can be made thinner by decreasing the molecule stick scale attribute, for example with the command:

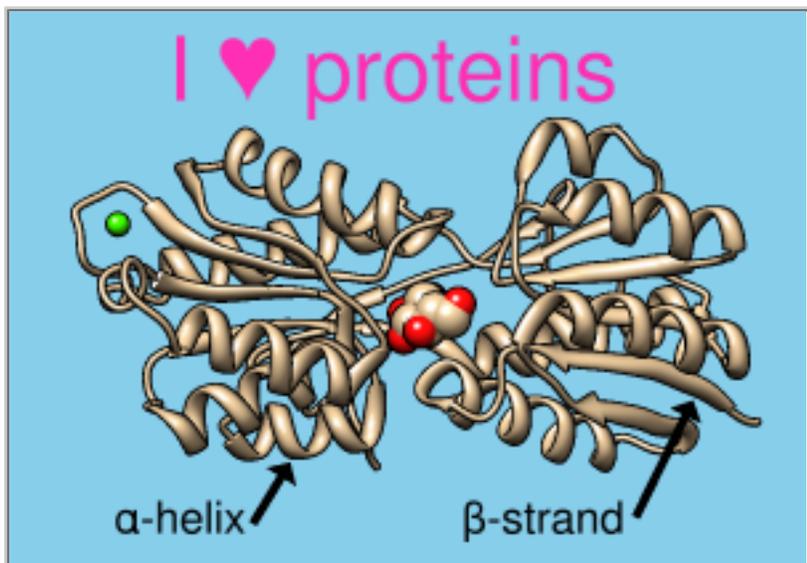
```
setattr m stickScale 0.5
```



## 2D Labels

2D Labels allows adding text, [symbols](#), and arrows to the display. This is useful for annotating [images](#) and [movies](#). The labels exist in the X,Y plane (the plane of the screen) and do not have a “depth” (Z-coordinate); they appear in front of any graphics displays and will not move when models are moved. Multiple colors and sizes of 2D labels can coexist. Text and symbols are laid out horizontally, whereas arrows can be oriented in any direction in the plane. (Symbols also include various arrows in different orientations, but cannot be oriented arbitrarily.) For 3-dimensional arrows, see [BILD format](#).

2D Labels information is saved in [sessions](#). 2D labels can also be saved to and read from a separate [label file](#).



There are [several ways to start 2D Labels](#), a tool in the **Utilities** category. 2D Labels is also implemented as the command [2dlabels](#). Starting 2D Labels opens a dialog with the following tabbed sections:

- [Labels](#)
- [Arrows](#)
- [Color Key](#) (appears in the same dialog, but is handled as a separate tool in the **Utilities** category)

Clicking the tab for a section brings it to the front. **Close** dismisses the 2D Labels dialog, and **Help** opens this manual page in a browser window.

### Labels:

**Use mouse for label placement** (near the bottom of the dialog) is automatically turned on. This option reassigns the left mouse button to labeling: clicking in the [graphics window](#) starts a new 2D label, and previously created 2D labels can be repositioned by dragging. Unchecking the option or closing the 2D Labels dialog returns the left mouse button to its previous function (by default, [rotation of models](#)).

The top part of the dialog lists the existing 2D labels and their (lower left-hand corner) X,Y coordinates. The X axis is horizontal and the Y axis is vertical, and both range from 0 to 1 in the visible portion of the window. It is possible to place a label partially or completely outside the visible area. The **Shown** checkboxes allow hiding/showing individual labels. All 2D labels collectively can be hidden and shown with the **Hide All** and **Show All** buttons.

The 2D label most recently created, moved, or clicked within the **2D Labels** dialog is the *active label*.

Typing into the **Text** area defines the content of the [active label](#). A label can include letters, numbers, and various symbols present on the keyboard, as well as spaces and carriage returns. Many other symbols such as Greek letters can also be included, as described below.

The **Insert symbol** menu provides easy access to several commonly used symbols. Choosing a symbol from the menu inserts it into the [active label](#). Choosing **more...** from the bottom of the menu shows [many more symbols](#) in a browser window, from which a symbol can be copied and then pasted into the **Text** area of the **2D Labels** dialog. Symbols from other sources can also be incorporated using standard copy-and-paste operations.

As text is entered, it will appear in the graphics window according to the current settings:

- **Color** (a [color well](#))
- **Font size** (default **24**)
- **Font style**
  - **normal** (default)
  - **italic**
  - **bold**
  - **bold italic**
- **Font typeface**
  - **sans serif** (default)
  - **serif**
  - **fixed**
- **Use solid label background** - whether to place a colored rectangle under the label text, and if so:
  - **Label background color** (a [color well](#), default semitransparent black) - color of the rectangle
  - **Margin around text** (default **9**) - how far beyond the label text in each direction (left, right, top, and bottom) to extend the rectangle, in pixels
  - **Outline around margin** (default **0**) - pixel width of an additional outline around the rectangle, set automatically to **white** for dark rectangle colors or **black** for light rectangle colors

The label background settings of the [active label](#) can be applied to all labels by clicking **Apply**.

The properties of an existing label can be changed by first making it the [active label](#) and then changing the setting. The color or font of *part* of an existing label can be changed by making it the [active label](#), highlighting the portion to be changed within the **Text** area, and then changing the setting.

Clicking **Delete** deletes the [active label](#).

For purposes of manipulation with the [2dlabels](#) command, 2D labels created with the graphical interface are assigned names **label2d\_id\_0**, **label2d\_id\_1**, **label2d\_id\_2**, *etc.* in

order of creation.

## Arrows:

**Use mouse for arrow placement** (near the bottom of the dialog) is automatically turned on. This option reassigns the left mouse button so that clicking and holding the button down starts a new 2D arrow, dragging extends the arrow, and releasing defines the location of the tip. A previously created arrow can be reoriented or its length changed by dragging from either end, or it can be moved without reorientation by dragging from its middle third. Unchecking the mouse option or closing the **2D Labels** dialog returns the left mouse button to its previous function (by default, [rotation of models](#)).

The top part of the dialog shows the existing 2D arrows and their start and end (tip) X,Y coordinates. The X axis is horizontal and the Y axis is vertical, and both range from 0 to 1 in the visible portion of the window. The color of an existing arrow can be changed by clicking its [color well](#) and using the [Color Editor](#). The **Shown** checkboxes allow hiding/showing individual arrows. All 2D arrows collectively can be hidden and shown with the **Hide All** and **Show All** buttons.

- **Arrow weight** (default **1.0**) - a scale factor for the overall thickness of the arrow
- **Arrowhead style**
  - **solid** (default)
  - **pointy**
  - **blocky**
  - **pointer**

The 2D arrow most recently created, moved, or clicked within the top part of the dialog is the *active arrow*. The weight or arrowhead style of an arrow can be changed by first making it the active arrow and then changing the setting. **Delete** deletes the active arrow.

2D labels (including text, symbols, and 2D arrows) can be written to or read from a file using the **File** menu in the **2D Labels** dialog. 2D label files allow transferring labels from one Chimera session to another, saving label work without saving an entire [session file](#), and hand-editing the files to obtain more precise label positions than easily afforded by interactive dragging. A simple text format is used (see the [example file](#)):

- **Label** indicates the start of information for a label, and the label identifier (if any) appears on the same line
- **Arrow** indicates the start of information for each arrow, and the arrow identifier appears on the same line
- label or arrow attributes (size, color, *etc.*) are given in subsequent lines that start with tabs
- symbols in label text can be indicated with [unicode](#) character codes or character names
- blank lines and comment lines (those starting with #) are ignored

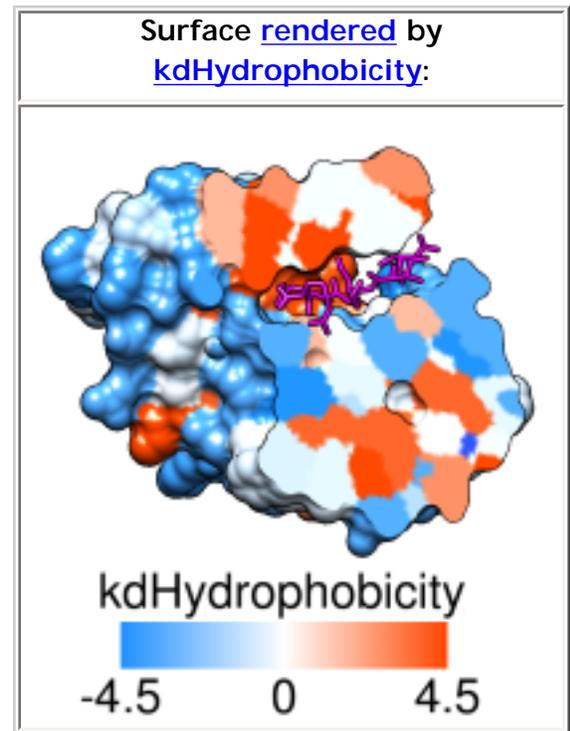


Color Key

**Color Key** allows creation and placement of a color key bar and associated labels in two dimensions. The labels can include [symbols](#). **Color Key** information is saved in [sessions](#). See also: [2D Labels](#), [Scale Bar](#)

There are [several ways to start Color Key](#), a tool in the **Utilities** category. In addition, there are buttons in [Render by Attribute](#), [Surface Color](#), and [Coulombic Surface Coloring](#) to start this tool and populate it to match the current coloring scheme. It is also implemented as the command [colorkey](#).

**Use mouse for key placement** (near the bottom of the dialog) is automatically turned on when **Color Key** is started. This option reassigns the left mouse button to color key creation and placement: clicking and dragging sets the location of opposing corners of the color key rectangle, or *color bar*. A key can be vertical or horizontal. A previously created color key can be moved by grabbing near its center and dragging, but clicking anywhere else will delete the color key and start over. Only one color key can exist at a time. Unchecking the mouse option or closing the **Color Key** dialog returns the left mouse button to its previous function (by default, [rotation of models](#)).



Any desired changes to the **Number of colors/labels** should be applied first, as changing the number will erase prior color/label specifications. A color can be changed by clicking the corresponding [color well](#) and using the [Color Editor](#). Label text can be entered directly or pasted (including [symbols](#)) into the corresponding fields.

The color/label pairs from top to bottom in the dialog will be arranged from top to bottom in a vertical key and from left to right in a horizontal key. **Reverse ordering of above** repopulates the color/label values in the reverse of the current order.

Additional options:

- **Color range depiction**
  - **blended** (default) - continuous
  - **distinct** - discontinuous
- **Label positions**
  - **left/top** - to the left of a vertical key, above a horizontal key
  - **right/bottom** (default) - to the right of a vertical key, below a horizontal key
- **Label color** (a [color well](#), default **white**) - if **No color**, the labels will be colored to match the individual colors in the key
- **Label justification** - how to align the labels in a vertical key; labels in horizontal keys are always center-justified
  - **left** - left-justify
  - **decimal point** (default) - align decimal points, if any
  - **right** - right-justify
- **Numeric label separation** - how to space numeric labels along the key; regardless of this setting, equal spacing will be used if any label is non-numeric or the labels are not in numerical order

- **proportional to value** (default)
- **equal**
- **Label offset** (default 0) - additional space in pixels between the labels and the color bar
- **Font size** (default 24)
- **Font style**
  - **normal** (default)
  - **italic**
  - **bold**
  - **bold italic**
- **Font typeface**
  - **sans serif** (default)
  - **serif**
  - **fixed**
- **Border color** (a [color well](#)) - color of outline around the color bar; if **No color**, no outline is shown
- **Border width** (default 3) - pixel width of outline around the color bar
- **Show tick marks** - whether to display short lines extending from the color bar toward the labels; if a border is shown, the border color is also used for the tick marks
  - **false** (default)
  - **true**
- **Tick length** (default 10) - pixel length of tick marks
- **Tick thickness** (default 4) - pixel width of tick marks; to make flush with the border (if any), use a value at least twice the border width

While all of the color-key-associated labels are of the same color and size, labels of different colors and sizes can be created with [2D Labels](#).

**Delete** deletes the color key display without changing the color/label settings in the dialog. **Close** dismisses the **Color Key** dialog, while **Help** opens this manual page in a browser window.

## Symbols for 2D Labels

[2D Labels](#) allows adding text, symbols, and arrows to the display. Symbols and special characters from the [free UCS outline fonts](#) can be incorporated by copying (such as from this page) and pasting into the **Text** field of the **2D Labels** dialog. They can also be pasted into [Color Key](#) labels.

Many of the supported symbols are provided here. Whether they are properly rendered in this page depends on the web browser and display font, but a symbol can be copied even when it is not rendered properly; it will be shown correctly in the 2D label!

The arrow symbols in this page can only be drawn in the orientations shown, but the **Arrows** section of [2D Labels](#) allows drawing arrows in any (X,Y) orientation.

Greek			Ligatures, Diacritical Marks		Superscripts	Miscellaneous	Math Symbols
α	A	alpha			1	○	±
β	B	beta	à	À	2	□	÷
γ	Γ	gamma	á	Á	3	◦	×
δ	Δ	delta	â	Â	<b>Fractions</b>	·	~
ε	E	epsilon	ã	Ã	¼	•	≡
ζ	Z	zeta	ä	Ä	½	◊	≈
η	H	eta	å	Å	¾	-	≠
θ	Θ	theta	æ	Æ	<b>Card Suits</b>	—	≡
ϑ			ç	Ç	♠	'	≤
ι	I	iota	è	È	♣	,	≥
κ	K	kappa	é	É	♥	“	Σ
λ	Λ	lambda	ê	Ê	♦	”	Π
μ	M	mu	ë	Ë	<b>Arrows</b>	'	∫
μ			ì	Ì	←	”	∂
ν	N	nu	í	Í	→	i	∇
ξ	Ξ	xi	î	Î	↑	ı	∅
ο	O	omicron	ï	Ï	↓	†	∇
π	Π	pi	ð	Ð	↔	‡	∈
	Π		ñ	Ñ	↔	§	∉
ρ	P	rho	ò	Ò	↵	¶	∋
σ	Σ	sigma	ó	Ó	⇐	%	*
ς	Σ		ô	Ô	⇐	<	√
			œ	Œ	⇑	>	∞
			õ	Õ	⇒	«	∞
			ö	Ö	⇓	»	∠
			ø	Ø		¢	^
			š	Š		£	v

τ	T	tau	ù	Û	↔	α	∩
υ	Υ	epsilon	ú	Ù	↶	¥	∪
	Υ		û	Û	↷	€	∴
			ü	Ü	&cularr; (not in sans serif)	©	⊃
φ	Φ	phi	ý	Ÿ	&curarr; (not in sans serif)	®	♀
χ	X	chi	þ	Þ	&olarr; (not in sans serif)	™	⊆
ψ	Ψ	psi	ÿ	Ÿ	&orarr; (not in sans serif)	•	⊇
ω	Ω	omega			• (only in serif)	•	⊕
						•	⊗
						•	⊥

---

UCSF Computer Graphics Laboratory / August 2013

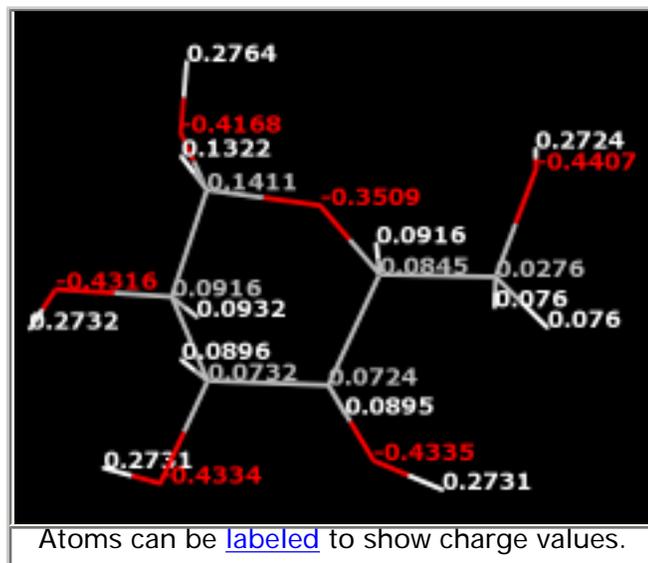


## Add Charge

**Add Charge** assigns atomic partial charges and Amber/GAFF atom types as the [attributes](#) `charge` and `gaffType`, respectively.

- Atoms in *standard residues* (water, standard amino acids, standard nucleic acids, and a few common variants and capping groups) are assigned charges and types from [Amber](#).
- Except for *monatomic ions*, which are simply assigned their net charges, atoms in *nonstandard residues* are assigned charges and types determined by [Antechamber](#), which is included with Chimera.

Atomic partial charges (and optionally, Amber/GAFF atom types) are included in saved [Mol2 files](#). The charges and Amber/GAFF atom types are used by other Chimera tools such as [Minimize Structure](#).



Charge assignment requires explicit hydrogens, which can be added beforehand with [AddH](#). Structures often have problems such as truncated sidechains that can lead to non-integral net charges. The [Dock Prep](#) tool includes options for fixing such problems prior to running [AddH](#) and **Add Charge**. See also: [PDB2PQR](#), [Coulombic Surface Coloring](#), [APBS](#), [Add Ions](#), [Write Prmtop](#), [reading charges from files](#)

There are [several ways to start Add Charge](#), a tool in the **Structure Analysis** and **Structure Editing** categories. **Add Charge** is also implemented as the command [addcharge](#).

Models to process can be chosen from the list with the left mouse button. Ctrl-click toggles the status of an individual model. To choose a block of models without dragging, click on the first (or last) and then Shift-click on the last (or first) in the desired block.

**Standard residues** can be assigned parameters from any of the following force fields ([details](#)):

- AMBER ff14SB (default)
- AMBER ff12SB
- AMBER ff03ua - treats amino acid sidechain aliphatic hydrogens as if collapsed onto the corresponding carbons; these “extra” hydrogens will be removed (however, the united-atom approach is not supported by [Minimize Structure](#), see [note](#))
- AMBER ff03.r1
- AMBER ff02pol.r1 (however, polarizability is not supported by [Minimize Structure](#), see [note](#))
- AMBER ff99SB (deprecated)
- AMBER ff99bsc0 (deprecated)

Monatomic ions are simply assigned their net charges. **Other residues** can be handled with either of the following charge calculation methods ([details](#)):

- **AM1-BCC** - semi-empirical (AM1) with bond charge correction (BCC)
- **Gasteiger** - faster method based on atom types and connectivity

The partial atomic charges can be shown as labels in **nonstandard residues** and/or **standard residues**. Labels will be shown for displayed atoms only. If charge labels are not shown with **Add Charge**, they can still be shown at some later point with [Actions... Label... other](#) (label with attribute **charge**) or commands:

*Command:* [labelopt](#) info charge

*Command:* [label](#)

Some subset of the atoms could be specified with the [label](#) command, whereas the above would label all displayed atoms.

**OK** initiates processing and dismisses the dialog, **Apply** initiates processing without dismissing the dialog, and **Close** simply dismisses the dialog. **Help** opens this manual page in a browser window.

[AddH](#) will be called as needed to add hydrogens. Potentially ambiguous or rare (shifted-pKa) protonation states, especially in binding sites and nonstandard residues, should be verified and corrected before charges are assigned. For example, extra hydrogens can be [deleted](#), and [atom types](#) can be edited (before hydrogen addition) with [setattr](#) or [Build Structure](#).

If there are any nonstandard residues, a dialog will appear for specifying their net charges. Names of the form RES1+RES2 indicate covalently bonded residues that will be treated as a single unit, and the net charge is for the combined entity. The [charge calculation method](#) can be changed at this point, if desired.

If a nucleic acid chain has a 5' terminal phosphate, the user will be asked whether this group should be deleted; otherwise, its atoms will be assigned charges of zero (the charge sets lack parameters for 5' phosphates).

A warning will appear if the name of any atom in a standard residue is not recognized or a model's net charge is not an integer; details will be reported in the [Reply Log](#). Cases of unrecognized atoms in standard residues and/or incorrect net charges should be examined and [resolved](#). Note that chain-terminal nucleotide residues will normally have non-integral charges, but the 5' and 3' charges sum to an integer.

## Details for Standard Residues

Atoms in standard residues (water, standard amino acids, standard nucleic acids, and a few common variants and capping groups) are assigned charges and atom types taken from Amber parameter files. Lookup tables are also used for [certain cofactors](#).

**Charge model** corresponds to force field version (see the [Amber](#) documentation for further details):

- **AMBER ff14SB** (default; used when **Add Charge** is called from [Coulombic Surface Coloring](#) or by the command [minimize](#) with **nogui true**) - charges from amino12.lib, amino12nt.lib, amino12ct.lib, nucleic12.lib

D.A. Case, V. Babin, J.T. Berryman, R.M. Betz, Q. Cai, D.S. Cerutti, T.E. Cheatham, III, T.A. Darden, R.E. Duke, H. Gohlke, A.W. Goetz, S. Gusarov, N. Homeyer, P. Janowski, J. Kaus, I. Kolossváry, A. Kovalenko, T.S. Lee, S. LeGrand, T. Luchko, R. Luo, B. Madej, K.M. Merz, F. Paesani, D.R. Roe, A. Roitberg, C. Sagui, R. Salomon-Ferrer, G. Seabra, C.L. Simmerling, W. Smith, J. Swails, R.C. Walker, J. Wang, R.M. Wolf, X. Wu, and P.A. Kollman (2014), AMBER 14, University of California, San Francisco.

Improvements in **ff14SB** relative to **ff12SB** include further sidechain dihedral corrections for proteins. The **ff14SB** force field is also recommended for RNA.

- **AMBER ff12SB** - charges from amino12.lib, amino12nt.lib, amino12ct.lib, nucleic12.lib. Improvements in **ff12SB** relative to [ff99 versions](#) include backbone and sidechain dihedral corrections for proteins and the **ff10** parameters for nucleic acids.
- **AMBER ff03ua** - charges from uni\_amino03.lib, uni\_aminont03.lib, uni\_aminoct03.lib, all\_nucleic94.lib

[New-generation amber united-atom force field](#). Yang L, Tan CH, Hsieh MJ, Wang J, Duan Y, Cieplak P, Caldwell J, Kollman PA, Luo R. *J Phys Chem B*. 2006 Jul 6;110(26):13166-76.

This “united-atom” force field treats amino acid sidechain aliphatic hydrogens as if they were collapsed onto the corresponding carbons. All nucleic acid hydrogens and amino acid backbone, polar, and aromatic hydrogens are still explicit. **Add Charge** will remove the “extra” hydrogens (those which are to be treated implicitly). \*\* [Minimize Structure](#) does not support the united-atom force field; this option is mainly useful for assigning charges and types before [writing a prmtop file](#) for [Amber](#). \*\*

- **AMBER ff03.r1** - charges from all\_amino03.lib, all\_aminont03.lib, all\_aminoct03.lib, all\_nucleic94.lib (**r1** refers to updated charge sets for N- and C-terminal amino acids; the original **ff03** used all\_aminont94.lib and all\_aminoct94.lib). Original **ff03** reference:

[A point-charge force field for molecular mechanics simulations of proteins based on condensed-phase quantum mechanical calculations](#). Duan Y, Wu C, Chowdhury S, Lee MC, Xiong G, Zhang W, Yang R, Cieplak P, Luo R, Lee T, Caldwell J, Wang J, Kollman P. *J Comput Chem*. 2003 Dec;24(16):1999-2012.

- **AMBER ff02pol.r1** - charges from all\_amino02.lib, all\_aminont02.lib, all\_aminoct02.lib, all\_nucleic02.lib (**r1** uses atom type H0 in glycine). \*\* [Minimize Structure](#) does not support the polarizable force field; this option is mainly useful for assigning charges and types before [writing a prmtop file](#) for [Amber](#). \*\*

Molecular mechanical models for organic and biological systems going beyond the atom centered two body additive approximation. Cieplak P, Caldwell J, Kollman P. *J Comput Chem*. 2001 Jul 30;22(10):1048-57.

[Strike a balance: optimization of backbone torsion parameters of AMBER polarizable force field for simulations of proteins and peptides.](#) Wang ZX, Zhang W, Wu C, Lei H, Cieplak P, Duan Y. *J Comput Chem.* 2006 Apr 30;27(6):781-90.

- **AMBER ff99SB** (deprecated) - charges from all\_amino94.lib, all\_aminont94.lib, all\_aminoc94.lib, all\_nucleic94.lib, same as the original **ff99**:

A second generation force field for the simulation of proteins, nucleic acids, and organic molecules. Cornell WD, Cieplak P, Bayly CI, Gould IR, Merz Jr KM, Ferguson DM, Spellmeyer DC, Fox T, Caldwell JW, Kollman PA. *J Am Chem Soc.* 1995 May;117(19):5179-97.

- **AMBER ff99bsc0** (deprecated; same charges as above, but uses atom type Cl in nucleic acids)

[Refinement of the AMBER force field for nucleic acids: improving the description of  \$\alpha/\gamma\$  conformers.](#) Pérez A, Marchán I, Svozil D, Sponer J, Cheatham TE 3rd, Laughton CA, Orozco M. *Biophys J.* 2007 Jun 1;92(11):3817-29.

The parameters include alternative sidechain protonation states for some standard residues. By default, **AddH** adds hydrogens to generate the states shown in **bold** below, although it does not remove hydrogens already present. The nondefault states can be attained by [deleting](#) hydrogens and/or editing [atom types](#) (with [setattr](#) or [Build Structure](#)) before adding hydrogens:

- aspartic acid **negative** or neutral
  - charges for the neutral ASH residue are assigned if one of the carboxylic acid oxygens (preferably OD2 for the proper charge distribution) has [atom type O3](#) and an attached hydrogen
- glutamic acid **negative** or neutral
  - charges for the neutral GLH residue are assigned if one of the carboxylic acid oxygens (preferably OE2 for the proper charge distribution) has [atom type O3](#) and an attached hydrogen
- cysteine **neutral reduced**, thiolate (negatively charged), or **half-cystine** (if there is a disulfide bond)
  - charges for the negative CYM residue are assigned if there is no disulfide bond and no atom named HG
- lysine **positive** or neutral
  - charges for the neutral LYN residue are assigned if the atom named NZ has [atom type N3](#) and thus two attached hydrogens rather than three
- histidine protonated at  $\delta$ -nitrogen (neutral),  $\epsilon$ -nitrogen (neutral), or both (positive)
  - in **AddH**, the [histidine protonation states](#) can be user-specified or guessed based on the local environment, and in [Coulombic Surface Coloring](#), they can be user-specified; **Add Charge** recognizes the state from the presence of hydrogen atom HD1, HE2, or both

Water charges correspond to the TIP3P model:

Comparison of simple potential functions for simulating liquid water. Jorgensen WL, Chandrasekhar J, Madura JD. *J Chem Phys.* 1983 Jul 15;79(2):926-35.

Note different solvent charge sets can be obtained using [Solvate](#). Charges previously assigned to solvent atoms by [Solvate](#) are not overwritten by **Add Charge**. However, [Minimize Structure](#) will still use the TIP3P model for the remaining water parameters.

Charges for the cofactors ATP, ADP, GTP, and GDP are simply taken from the [Amber parameter database](#) (Bryce Group, University of Manchester).

Cases of unrecognized atoms in standard residues and/or incorrect net charges should be examined and resolved. Approaches include:

- editing the input structure file to contain standard atom names that will be recognized
- using [Dock Prep](#) to perform various fixes on a structure before charge assignment
- adding missing atoms with [Build Structure](#)
- [deleting](#) unrecognized atoms that are deemed unnecessary for later calculations
- deciding it is acceptable for the unrecognized atoms to have zero charge
- assigning charges directly with [setattr](#), [defattr](#), or [Define Attribute](#)

## Nonstandard Residues and Antechamber

Monatomic ions are simply assigned their net charges. For other nonstandard residues (those not classified as [standard](#)), charges and GAFF atom types are determined using Amber's [Antechamber](#) module, which is included with Chimera. The available charge calculation methods are:

- **AM1-BCC** - semi-empirical (AM1) with bond charge correction (BCC), parametrized to reproduce *ab initio* (HF/6-31G\*) electrostatic potentials
- **Gasteiger** - iterative partial equalization of orbital electronegativity

While both methods are much faster than *ab initio* calculations, the Gasteiger method is the faster and more approximate of the two.

Note that Antechamber/GAFF are meant to handle most small organic molecules, but not metal complexes, inorganic compounds, or unstable species such as radicals, and may not work well on highly charged molecules.

Publications involving [Antechamber](#) use should cite:

[Automatic atom type and bond type perception in molecular mechanical calculations.](#) Wang J, Wang W, Kollman PA, Case DA. *J Mol Graph Model*. 2006 Oct;25(2):247-60.

GAFF atom types and associated parameters are described [online](#) and in:

[Development and testing of a general amber force field.](#) Wang J, Wolf RM, Caldwell JW, Kollman PA, Case DA. *J Comput Chem*. 2004 Jul 15;25(9):1157-74.

## Additional Parameter Sources

Additional sources of charges and other parameters include:

- [Amber parameter database](#) (Bryce Group, University of Manchester)
- [REDDB](#) and the [RED Server](#) (Université de Picardie Jules Verne and Burnham Institute for Medical Research)

## Reading Charges from Files

**Attribute assignment.** Charges can be specified arbitrarily in a simple text [attribute assignment file](#), read in with [Define Attribute](#) or the command [defattr](#).

**PQR.** The [PQR variant](#) of [PDB format](#) includes atomic partial charges and radii, which Chimera assigns as the atom [attributes charge](#) and [radius](#).

**Mol2.** When a structure is read from a Mol2 file, the [Sybyl atom types](#) and any partial charges are stored as the atom [attributes mol2type](#) and [charge](#).

**PubChem SDF.** When a structure is read from a 3D SDF obtained directly from [PubChem](#) (see [more](#) about these files), values in the section marked <PUBCHEM\_MMFF94\_PARTIAL\_CHARGES> are stored as the atom [attribute](#) named [charge](#). Note: SDFs [fetched](#) by PubChem CID from within Chimera are from a different database (**Pub3D**, provided by the [CICC](#) at Indiana University) and do not contain charges.

**Topology files.** Partial charges in some of the [topology formats](#) (Amber, PSF) input to [MD Movie](#) are stored as the atom [attribute](#) named [charge](#).



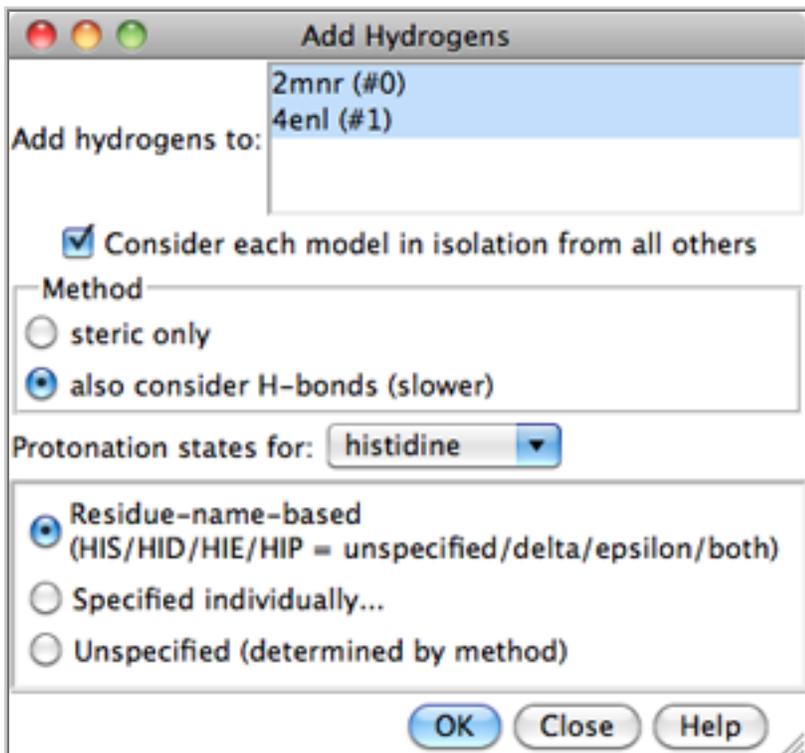
## AddH

**AddH** adds hydrogen atoms to molecules, as well as OXT atoms where missing from peptide C-termini. Chimera uses atom and residue names, or if these are not "standard," atomic coordinates, to determine connectivity and [atom types](#); **AddH** then uses the atom types to determine the number of hydrogens to be added and their positions. The positions of pre-existing atoms are not changed, but any lone pairs and unidentifiable-element atoms are deleted. See also: [FindHBond](#)

There are [several ways to start AddH](#), a tool in the **Structure Editing** category (including using it via [Dock Prep](#)). **AddH** is also implemented as the command [addh](#).

Models to which hydrogens should be added can be chosen from the list with the left mouse button. Ctrl-click toggles the status of an individual model. To choose a block of models without dragging, click on the first (or last) and then Shift-click on the last (or first) in the desired block.

**Consider each model in isolation from all others** - whether hydrogen placement should be affected by atoms within the same model only. Otherwise, other models in the vicinity (except submodels of the same model) may affect hydrogen placement, regardless of whether they were chosen for hydrogen addition.



The **Method** for adding hydrogens can be:

- **steric only** - based on [atom types](#) and clash avoidance
- **also consider H-bonds (slower)** (default) - based on [atom types](#), clash avoidance, and hydrogen bond formation. Considering H-bonds increases calculation time, and the method is still under development. Although hydrogens are placed to avoid clashes and form hydrogen bonds where possible, they are not energy-minimized, and a globally optimal network in terms of the number of H-bonds or total H-bonding energy is not necessarily found.

**Protonation states** of certain ionizable sidechains can be specified. By default, if amino acids have standard residue names, each histidine sidechain will be protonated based on its local environment, whereas the sidechains of other residue types will be assigned [protonation states](#) reasonable at physiological pH, regardless of the local environment: negative glutamic acid and aspartic acid, positive lysine and arginine, and neutral cysteine and tyrosine. Alternative protonation states of histidine, glutamic acid, aspartic acid, lysine, and cysteine can be specified interactively or with special residue names in the input coordinate file:

- Choices for **histidine**:
  - **Residue-name-based** (default) - residue names will be used to determine which histidine sidechain nitrogens should be protonated: the  $\delta$ -nitrogen in residues named HID, the  $\epsilon$ -nitrogen in HIE, and both nitrogens in HIP. Residues named HIS will be treated as unspecified, and may end up with either or both sidechain nitrogens protonated, depending on the [method](#) and the local environment.
  - **Specified individually...** regardless of which of the names above are used, the desired sidechain protonation state of each residue will be specified in a dialog by the user
  - **Unspecified (determined by method)** - regardless of which of the names above are used for histidine residues, all will be treated as unspecified, and may end up with either or both sidechain nitrogens protonated, depending on the [method](#) and the local environment.
- Choices for **glutamic acid**:
  - **Residue-name-based** (default) - residue names will be used to determine sidechain charge state: GLU negatively charged and GLH neutral, OE2-protonated
  - **Specified individually...** regardless of which of the names above are used, the desired sidechain protonation state of each residue will be specified in a dialog by the user
  - **Charged** - negatively charged
- Choices for **aspartic acid**:
  - **Residue-name-based** (default) - residue names will be used to determine sidechain charge state: ASP negatively charged and ASH neutral, OD2-protonated
  - **Specified individually...** regardless of which of the names above are used, the desired sidechain protonation state of each residue will be specified in a dialog by the user
  - **Charged** - negatively charged
- Choices for **lysine**:
  - **Residue-name-based** (default) - residue names will be used to determine sidechain charge state: LYS positively charged and LYN neutral
  - **Specified individually...** regardless of which of the names above are used, the desired sidechain protonation state of each residue will be specified in a dialog by the user
  - **Charged** - positively charged
- Choices for **cysteine**:
  - **Residue-name-based** (default) - residue names will be used to determine sidechain charge state: CYS neutral and CSM negatively charged
  - **Specified individually...** regardless of which of the names above are used, the desired sidechain protonation state of each residue will be specified in a dialog by the user
  - **Neutral**

Clicking **OK** initiates hydrogen addition and dismisses the dialog, while **Close** merely dismisses the dialog. **Help** opens this manual page in a browser window.

If any atoms cannot be assigned a [type](#), another dialog will appear. It is necessary to click on the line for each unassigned atom and then indicate its proper substituent geometry and number of substituents.

Added hydrogens are colored the [element color](#) (default white) if the attached atom is colored by element, otherwise the same as the attached atom.

The [default VDW radii](#) of carbon, nitrogen, oxygen, and sulfur atoms depend on whether hydrogen atoms are present. Therefore, *the radii of some atoms will change when hydrogens are added.*

## Protonation States

**AddH** aims to generate protonation states reasonable at physiological pH. For example, hydrogens are not added to the phosphodiester moieties of DNA and RNA. By default, aspartic acid and glutamic acid sidechains are assumed to be negatively charged, arginine and lysine sidechains positively charged (although [other states](#) can be attained). Two chemical moieties are treated as ambiguous at biological pH:

- imidazoles such as histidine sidechains; histidine protonation states can be [specified by the user](#) or guessed by the [method](#)
- terminal phosphates (the third ionization); if one P-O bond is at least 0.05 Å longer than the others around that same phosphorus atom, that oxygen will be protonated

Potentially ambiguous or rare (shifted-pKa) protonation states, especially in binding sites and nonstandard residues, should be verified and corrected as needed. For example, extra hydrogens can be [deleted](#), and [atom types](#) can be edited (before hydrogen addition) with [setattr](#) or [Build Structure](#).

Residues at the ends of connected peptide chains are inspected to determine whether they are real termini, based on any SEQRES information in the input PDB file (or the mmCIF equivalent) and the presence or absence of additional chains with the same IDs. Real N-termini are assumed to be positively charged (+H<sub>3</sub>N-) and real C-termini are assumed to be negatively charged (-CO<sub>2</sub><sup>-</sup>). If a C-terminal carboxylate is missing an oxygen (OXT), it will be added. End residues that are not real termini are terminated like other chain-internal residues, with N(H)- and -C(=O). The position of the N-end "amide" hydrogen in such cases is not fully determined by the positions of the existing atoms; **AddH** places this hydrogen to produce a  $\varphi$  angle equal to that of the subsequent residue.

## Bond Lengths

Bond lengths for X-H (X = C/N/O/S) are taken from the [Amber](#) parm99 parameters:

X	<a href="#">atom types</a>	X-H bond length (Å)
<i>sp</i> <sup>3</sup> carbon	C3	1.0900
<i>sp</i> <sup>2</sup> carbon	C2,Car	1.0800
<i>sp</i> carbon	C1	1.0560
nitrogen	N3+,N3,Npl,Ng+	1.0100
<i>sp</i> <sup>3</sup> oxygen	O3 (except water)	0.9600
<i>sp</i> <sup>3</sup> oxygen	O3 (water)	0.9572
sulfur	S3	1.3360

Bond lengths to other X are approximate, obtained by adding the covalent bond radii of element X and H.

## Recommended Alternative

When a more intensive approach is desired, the program **Reduce** (developed by the [Richardson Laboratory](#)) is a good alternative. **Reduce** places hydrogens to optimize local H-bonding networks and avoid steric overlaps, while flipping certain sidechains 180 degrees as deemed appropriate to fulfill these criteria. Asparagine and glutamine sidechains may be flipped to switch their terminal N and O atoms, and the imidazole ring of histidine may be flipped to switch N and C identities. The protonation state of histidine is adjusted based on the local environment.

**Reduce** is available free at <http://kinemage.biochem.duke.edu>:

- for on-line use as part of the **MolProbity** service, on a file uploaded or chosen by PDB code (individual sidechain flips can be accepted or rejected)
- for download (from the **Software** section) to run on most platforms

and is described in:

[Asparagine and glutamine: using hydrogen atom contacts in the choice of side-chain amide orientation](#). Word JM, Lovell SC, Richardson JS, Richardson DC. *J Mol Biol.* 1999 Jan 29;285(4):1735-47.



**Add Ions** adds monatomic counterions around molecule models using [AmberTools](#). Thanks to Wei Zhang (The University of Texas Health Science Center at Houston) for implementing this tool. See also: [Solvate](#), [Write Prmtop](#)

There are [several ways to start Add Ions](#), a tool in the **Structure Editing** and **Amber** categories.

The model of interest should be chosen from the list. Multiple models can be chosen, but they will be considered individually rather than as a combined system.

The ions will be placed in electrostatically favorable locations according to a Coulombic potential on a grid. The molecule model must already include hydrogens and charge assignments; otherwise, dialogs will appear for running [AddH](#) and [Add Charge](#) as needed prior to adding ions. The potential grid is calculated using a distance-dependent dielectric and ignoring any solvent molecules. A solvent molecule found to overlap with an ion will be removed.

#### Ion types:

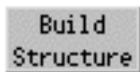
- **Cl<sup>-</sup>**
- **Cs<sup>+</sup>**
- **K<sup>+</sup>**
- **Li<sup>+</sup>**
- **MG2 (Mg<sup>++</sup>)**
- **Na<sup>+</sup>**
- **Rb<sup>+</sup>**

The number of ions can be set to **neutralize** the structure or to a **specific number**. To determine whether a positive or negative ion type would be needed to neutralize a structure, one could run [AddH](#) and [Add Charge](#) (which reports net charge) beforehand.

**OK** initiates adding ions and dismisses the panel, while **Apply** adds ions without dismissing the panel. Addition may take several seconds; progress is reported in the [status line](#).

**Close** dismisses the panel without adding any ions. **Help** brings up this manual page in a browser window.

See the [AmberTools addIons](#) documentation for further details.



## Build Structure

**Build Structure** can generate atomic structures “from scratch” or modify existing molecules. See also: [addaa](#), [swapaa](#), [swapna](#), [AddH](#), [Rotamers](#), [Ramachandran Plot](#), [Metal Geometry](#), [Minimize Structure](#), [Unit Cell](#), [Multiscale Models](#), [Change Chain IDs](#), [Renumber Residues](#), [Chimera interface to Modeller](#), [atom types](#), [modifying and saving data](#)

There are [several ways to start Build Structure](#), a tool in the **Structure Editing** category. Different sections are available from the menu near the top of the dialog:

- [Start Structure](#) - add an atom, fragment, or molecule not bonded to existing atoms
- [Modify Structure](#) - change or delete existing atoms, build outward step-by-step from an existing structure
- [Adjust Bonds](#) - add bonds, delete bonds, change bond lengths
- [Adjust Torsions](#) - rotate bonds (change dihedral angles)
- [Join Models](#) - bond and merge two models, moving one of them to form the appropriate bond
- [Invert](#) - swap substituents, potentially changing chirality
- [Adjust Bond Angles](#) - modify bond (valence) angles

**Atomspec display style** in the [Labels preferences](#) controls how atoms are listed in the **Build Structure** dialog: in the **simple** style (for example, **HIS 16.A ND1**, see also the [Distances figure](#)) or as [command-line specifiers](#) (for example, **:16.A@ND1**, see also the [Adjust Torsions figure](#)).

**Close** dismisses the **Build Structure** tool; **Help** opens this manual page in a browser window.

Outside of **Build Structure**, atoms and bonds can also be deleted:

- by [selecting](#) them and using the menu item [Actions... Atoms/Bonds... delete](#)
- with the command [delete](#)

### ← Start Structure

The **Start Structure** section of [Build Structure](#) creates atoms, fragments, and molecules independent of any pre-existing atoms. See also: [open](#) ([SMILES](#) and [PubChem](#)), [rna](#)

Options are to **Add**:

- **atom** - a single atom; helium is used as a dummy type not bonded to other atoms. Placement options:
  - **Center of view**
  - specified coordinates **x**, **y** and **z** in the [model](#) coordinate system

**Select placed atom** indicates the new atom should be [selected](#). The [Modify Structure](#) section can

then be used to specify the [selected](#) atom's element, valence, and geometry, and to append further atoms.

- **fragment** - the specified fragment, placed in the center of view. Several planar ring systems are available; a small diagram of the chosen **Fragment** is displayed.
- **SMILES string** - a modeled structure for the specified [SMILES](#) string. The structure is generated using the [smi23d web service](#) provided by the [Chemical Informatics and Cyberinfrastructure Collaboratory](#) at Indiana University, or if that does not produce a structure, the [SMILES translator](#) provided by the [National Cancer Institute CADD group](#). The [smi23d service](#) deploys the same procedure used to populate the **Pub3D** database (below). Hydrogens are included, although it may be necessary to [add](#) or [delete](#) hydrogens to generate the desired protonation state.
- **PubChem CID** - a modeled structure specified by [PubChem](#) compound ID (CID), fetched from the **Pub3D** database using a [web service](#) provided by the [Chemical Informatics and Cyberinfrastructure Collaboratory](#) at Indiana University. **Pub3D** is described in [Willighagen et al., BMC Bioinformatics 8:487 \(2007\)](#). About 99% of the compounds in [PubChem](#) are available; the structure generation pipeline generally handles organic compounds, but not inorganic, metallo-, or highly unstable species such as radicals. Hydrogens are included, although it may be necessary to [add](#) or [delete](#) hydrogens to generate the desired protonation state.

For any of the options above, clicking **Apply** will add the specified atom(s) and give them the specified **Residue name** (normally three characters long in [PDB format](#)).

- **peptide** - a peptide specified by one-letter amino acid codes (capitalization does not matter)

Clicking **Apply** brings up another dialog for specifying backbone  $\varphi$  (phi) and  $\psi$  (psi) angles and other parameters. One or more rows can be [chosen](#) with the mouse and **Set** to values either entered manually or supplied for various types of secondary structure:

description	$\varphi$ (°)	$\psi$ (°)
$\alpha$ helix	-57	-47
antiparallel $\beta$ strand	-139	135
parallel $\beta$ strand	-119	113
$3_{10}$ helix	-49	-26
$\pi$ helix	-57	-70

Rows in the dialog can be chosen with the left mouse button. **Ctrl**-click toggles the state (chosen or not) of single line. A block can be chosen by dragging, or by clicking on the first (or last) line in the desired block and then **Shift**-clicking on its last (or first) line.

Sidechain conformations will be taken from the specified **Rotamer library**:

- **Dunbrack** - [Dunbrack backbone-dependent rotamer library](#)
- **Richardson (common-atom)** - common-atom values (author-recommended) from the [Richardson backbone-independent rotamer library](#)
- **Richardson (mode)** - mode values from the [Richardson backbone-independent rotamer library](#)

The rotamer at each position will be chosen as described for the command [swapaa](#) with the [criteria](#) **cp**: by fewest number of clashes, and if a tie, then the highest probability according to the rotamer library. The residues are added in N→C order, so only clashes with more N-terminal residues are evaluated. The peptide will be assigned the specified **chain ID**.

Clicking **Apply** (or **OK**, which also dismisses the dialog) creates the peptide. Hydrogen atoms are not included. Backbone bond lengths and angles are taken from the [Amber](#) ff99 parameters. Sidechain bond lengths and angles are taken from the [Amber](#) parameter files **all\*94.lib**.

- **helical DNA/RNA** - double-helical nucleic acid specified by the one-letter residue codes of one strand (capitalization does not matter); the complementary strand will also be built to generate a double helix. Possible types:
  - **DNA** (sequence containing A,T,G,C)
  - **RNA** (sequence containing A,U,G,C)
  - **Hybrid DNA/RNA (enter DNA)** (sequence containing A,T,G,C)

Possible conformations:

- **A-form**
- **B-form**

The geometries were derived by surveying known structures of double-helical nucleic acids. The same torsion angle values are used for a conformation regardless of the composition of the double helix (specific sequence and whether DNA, RNA, or DNA/RNA hybrid).

- **more RNA...** simply refers users to [Assemble2](#), a Chimera plugin for designing 2D RNA structures and generating corresponding 3D structures. Executables, source code, and documentation are available for [download](#). *Assemble2* is developed by Fabrice Jossinet (Institut de Biologie Moléculaire et Cellulaire, Strasbourg).

The new atom, fragment, or molecule can be [colored by element](#) and placed in an existing molecule model or a **new model** with a specified name. A new model will be assigned the lowest available model number.

## ← Modify Structure

The **Modify Structure** section of [Build Structure](#) can be used to change the element, valence (number of directly attached atoms), and/or geometry (spatial arrangement of attached atoms) of one or more [selected](#) atoms; hydrogens are appended as needed to fill the valence. Building outward can be done by successive cycles of modifying a hydrogen attached to the previously modified atom.

The **Modify Structure** dialog can be opened directly by doubleclick-[picking](#) an atom when 0 or >3 other atoms are already [selected](#), then choosing **Modify Atom** from the resulting [context menu](#).

Clicking **Apply** will change the [selected](#) atoms as specified:

- **Element** - desired element
- **Bonds** - desired total number of substituents on the atom (0-4 allowed). Hydrogens will be added to the atom to generate the indicated total number of **Bonds**. Pre-existing atoms are not removed. If the atom is already bonded to one (and only one) other atom, the bond will be adjusted to an *approximate* length depending on the elements involved. The positions of any other atoms already present will not be changed. New hydrogens will be placed to form the [target \(idealized\) bond](#)

[angles](#), or if the atom already has two or more substituents, to maximally avoid those substituents. Bond lengths for X-H (X = C/N/O/S) are taken from the [Amber](#) parm99 parameters (see [table](#)). Bond lengths can be adjusted to specific values using [Set Bond Length](#), or allowed to change along with other degrees of freedom during [energy minimization](#).

- **Geometry** - target arrangement of bonds around the atom
  - **linear** - allowed for 2 bonds, target bond angle 180°
  - **trigonal** - allowed for 2-3 bonds, target bond angle(s) 120° in a plane
  - **tetrahedral** - allowed for 2-4 bonds, target bond angle(s) 109.5°

In [PDB format](#), **atom names** can be up to four characters long and should be unique within a residue. An atom name normally starts with the element symbol. If any element assignment would be changed, a new atom name or sequentially numbered range of names must be specified. When only atoms matching the specified element are [selected](#), however, it is possible to retain their current names.

- **Connect to pre-existing atoms if appropriate** - if a newly added hydrogen would be very close to an existing atom in the same model as the [selected](#) atom, discard the hydrogen and form a bond to the existing atom instead
- **Focus view on modified residue** - [focus](#) the view on the the residue containing the modified atom
- **Color new atoms by element** - color the modified atom and any newly added hydrogens [by element](#)
- **Residue Name**
  - **Leave unchanged**
  - **Change modified residue's name to [resname]** - change the residue name to *resname* for all atoms in the residue, not just the modified atom (appropriate when a standard residue is modified; for example, a methylated lysine should no longer be named LYS)
  - **Put just changed atoms in new residue named [resname] in chain [chainID]** - put the modified atom and any new hydrogens in a new residue named *resname* in chain *chainID*; otherwise, they will be included in the atom's current residue

In [PDB format](#), residue names are normally three characters long and chain identifiers a single character (or blank). However, Chimera will tolerate four-character residue names, and the chain identifier can be specified as **het** or **water** (in [PDB output](#), these translate to use of HETATM records with a blank chain identifier). Otherwise, [PDB output](#) will contain ATOM records for standard residues, HETATM records for nonstandard residues, and the specified chain identifier(s).

To build out further, [select](#) one of the new hydrogens and use **Modify Structure** again.

Clicking **Delete** removes the [selected](#) atoms and bonds.

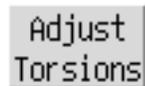
## ← Adjust Bonds

The **Adjust Bonds** section of [Build Structure](#) allows adding bonds, deleting bonds, and changing bond lengths. This dialog can be opened directly by doubleclick-[picking](#) a bond when no other bonds are already [selected](#), then choosing **Adjust Bond** from the resulting [context menu](#). Bonds can also be created

and deleted with the command [bond](#), and bond lengths changed with the command [adjust](#).

- **Delete selected bonds** - remove any bonds that are [selected](#)
- **Add [option] bonds between selected atoms** - add bonds to a [selected](#) set of atoms, where the *option* can be:
  - **reasonable** (default) - bond each pair in the set for which the interatomic distance is no greater than the sum of their [covalent bond radii](#) plus a tolerance of 0.4 Å
  - **all possible** - bond all pairs in the set, regardless of whether the bonds are reasonable
- **Set length of selected bonds to [length]** - adjust the lengths of the [selected](#) bonds sequentially to *length* Å, moving atoms on the:
  - **smaller side** (default)
  - **larger side**

Entering a new value or moving the slider will change the length. The order of bond adjustment may affect the final position of the structure (but not the structure itself) and cannot be controlled. If the bond is in a ring, only the flanking atoms will be moved, by equal distances in opposite directions. Clicking **Revert** restores the [selected](#) bonds to their lengths prior to selection. Once the selection has been changed, however, the bonds can no longer be reverted, even if re-selected.



## ← Adjust Torsions

The **Adjust Torsions** section of [Build Structure](#) is a table of active (rotatable) torsions. Active torsions are saved in [sessions](#). See also: [rotation](#)

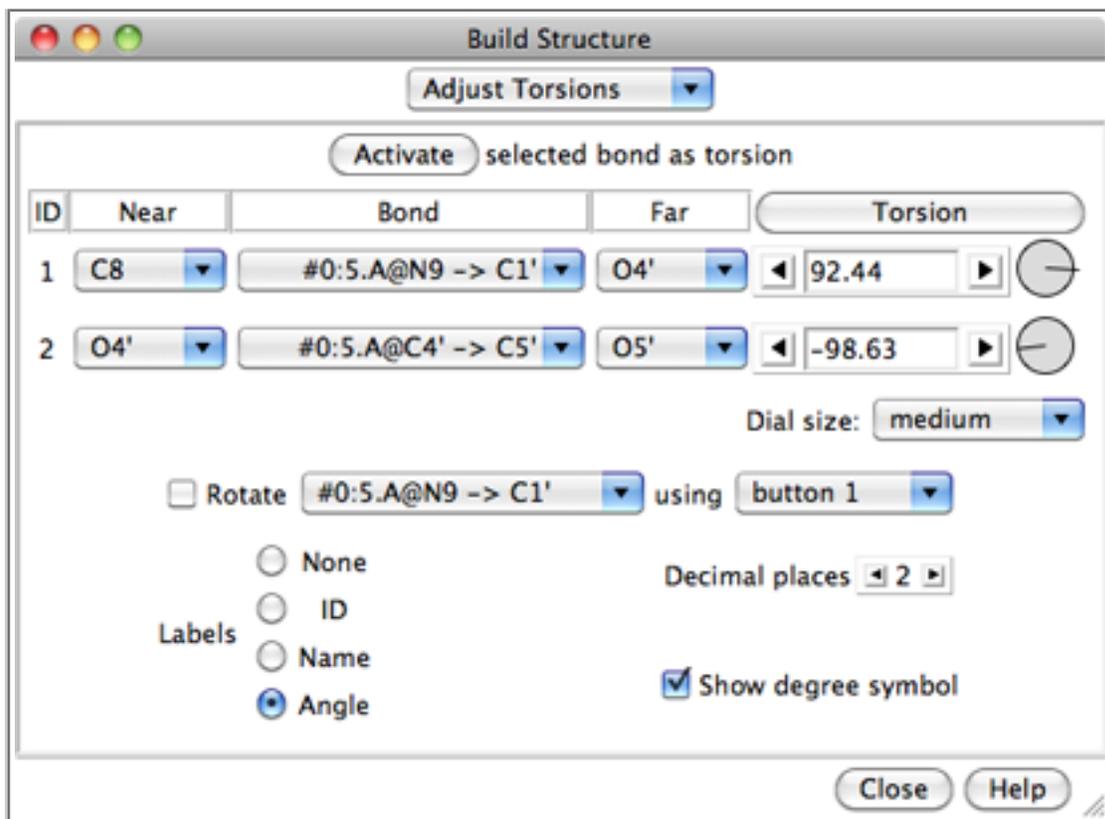
There are [several ways to start Adjust Torsions](#), a tool in the **Structure Editing** category. Torsions can be activated in three ways:

- [Selecting](#) exactly one bond, then clicking the **Activate** button on the dialog. With default [mouse settings](#), a bond can be [picked](#) (selected from the graphics window) with Ctrl-click.
- Doubleclick-[picking](#) a bond when no other bonds are already [selected](#), then choosing **Rotate Bond** from the resulting [context menu](#).
- Using the [rotation](#) command.

### Adjust Torsions - command-line specifiers

An error message will appear if an attempt is made to activate a bond that is within a ring or already rotatable.

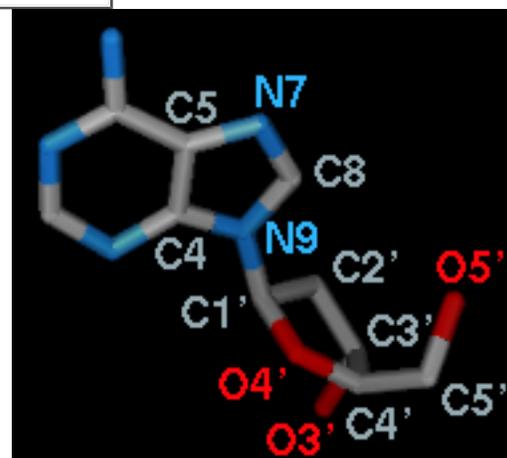
If the four atoms defining a torsion are called 1-2-3-4, 1 is the **Near** atom and 4 is the **Far** atom, which will move when the bond is rotated. The angle in degrees as defined by the current **Near** and **Far**



atoms is shown in the **Torsion** column. **Torsion** can be toggled to **Delta**; the reported value is then the angle in degrees relative to the starting angle, and there are no **Near** and **Far** columns. For a terminal bond (lacking additional atoms on one end), the value shown is always a delta, even if the column header is toggled to **Torsion**.

A bond can be rotated by entering a new angle value (and pressing return), clicking the arrowheads flanking the angle value, or manipulating the dial. The **Dial size** can be set to **small**, **medium**, or **large**. Further, torsions can be manipulated in the graphics window with the mouse. This can be done by checking **Rotate [torsion] using [button]** and choosing the desired torsion and mouse button from the pulldown menus.

The **Bond** column contains a pulldown menu for each active rotation, labeled with identifiers for atoms 2->3 (those flanking the rotatable bond):



- **Revert** - restore the original torsion angle
- **Reverse** - change which side of the torsion moves when the bond is rotated (change the torsion definition from A-B-C-D to D-C-B-A)
- **Deactivate** - deactivate the bond rotation and remove it from the table. Note that the torsion will not automatically revert to its original value when deactivated; **Revert** should be used before **Deactivate** to retain the original torsional angle. Even if the same rotation is reactivated later, **Revert** will not work to restore the angle, since the original value has not been saved.
- **Select** - [select](#) the rotatable bond and deselect any others; when in the [Torsion mode](#), also select the two flanking bonds if the rotatable bond is nonterminal

When atom 2 is bonded to more than two atoms, there is more than one possible **Near** atom, and alternatives (if any) are available in a pulldown menu from the current **Near** atom name. Likewise, when atom 3 is bonded to more than two atoms, there is more than one possible **Far** atom, and alternatives (if any) are available in a pulldown menu from the current **Far** atom name. For the first torsion in the example (figures above), there are two choices for the **Near** atom, **C8** and **C4**, and two choices for the **Far** atom, **O4'** and **C2'**. For the second torsion, there are two choices for the **Near** atom, **O4'** and **C3'**, and only one

possible **Far** atom, **O5'**. Of course, if **Reverse** is used, the **Near** and **Far** choices are interchanged, and in the [Delta mode](#), there are no **Near** and **Far** columns.

The **Labels** setting applies to all active torsions and can be switched among:

- **None** - no label
- **ID** - torsion ID number (the first column in the listing)
- **Name** - identifiers for atoms 2->3 (as shown in the **Bond** column)
- **Angle** - the angle in degrees

The **places** and **Show degree symbol** settings control how angle values are displayed when **Labels** is set to **Angle**. The **Decimal places** can be changed by clicking the arrowheads flanking the value.

## ← Join Models

The **Join Models** section of [Build Structure](#) forms a bond between two [selected](#) atoms. The two atoms must be in two different models, which will be merged into a single model when the bond is formed. See also: [combine](#)

- **C-N peptide bond** - specialized case of joining two peptides. The N-terminal N of one peptide and the C-terminal (carbonyl or carboxyl) C of the other should be [selected](#). Each selected atom must be bonded to only one carbon (except N in proline or hydroxyproline can be bonded to two carbons); however, it may also be bonded to hydrogen and/or OXT, and if so, these atoms will be replaced as appropriate by the new peptide bond.
  - **C-N length** (default 1.33 Å)
  - **C $\alpha$ -C-N-C $\alpha$  dihedral ( $\omega$  angle)** (default 180.0°)
  - **C-N-C $\alpha$ -C dihedral ( $\varphi$  angle)** (default -120.0°)  
The existing  $\psi$  angle (N-C $\alpha$ -C-N dihedral) of the same residue is simply reported; it can be changed as needed using [Adjust Torsions](#).
  - **Move atoms on [ N / C ] side** - which of the two models to reposition when forming the bond
- **other bond** - the more general case. One terminal atom (only bonded to one other atom) in each model must first be [selected](#). These two atoms will be removed and replaced with the new bond. Usually these atoms will be hydrogens, so it may be useful to add hydrogens beforehand, to whole models with [AddH](#) or in a more local fashion with the [Modify Structure](#) section of [Build Structure](#).
  - **length** - length for the new bond in Å
  - **[menu of available dihedrals] dihedral** - torsion angle in degrees for the four atoms in the dihedral chosen from the menu of possibilities
  - **Move atoms on [model #] side** - which of the two models to reposition when forming the bond

Clicking **Apply** replaces the two [selected](#) atoms with a bond and merges the models as specified. The moved model will cease to exist as its atoms are merged into the other model. Chain identifiers and residue numbers may also be adjusted, in two phases. First, the two models are merged, and chain identifiers in the incoming (moving) model will be changed as needed to avoid duplication. Next, when the bond is formed, if only one side has a chain ID (excluding [het](#)), it will also be assigned to the other side, or if both already have an ID, the ID of the nonmoving side will be used for both. The chain whose ID changes

will be renumbered as needed to avoid duplicate residue identifiers.

## ← Invert

The **Invert** section of [Build Structure](#) allows exchanging the positions of two substituents of an atom, potentially inverting a chiral center. Substituents of atoms that are not chiral centers can also be swapped. See also: [invert](#), [chirality](#)

- If a single atom is [selected](#), clicking **Swap** exchanges the positions of its two smallest substituents based on number of atoms, or if those are the same, atomic weights of the atoms directly bonded to the selected atom. Implicit hydrogens on the selected atom are considered, but not those on its substituents. These rules are not meant to reproduce the much more complex “priority” calculations used in chirality determination.
- If two atoms directly bonded to the same central atom are [selected](#), clicking **Swap** will exchange the positions of the substituents rooted at those atoms.

Any unintended results can be reversed by clicking **Swap** again without changing the [selection](#).

## ← Adjust Bond Angles

The **Adjust Bond Angles** section of [Build Structure](#) allows changing bond (valence) angles. Bond angles can also be changed with the command [adjust](#).

First, two adjacent bonds defining an angle to be modified must be [selected](#). The movement will be in the plane defined by the two selected bonds, and the **Move [bond] side** menu controls which set of atoms will move. While the selection exists, the bond angle can be modified by entering a new angle value (and pressing return), clicking the arrowheads flanking the value, or manipulating the dial. **Revert** restores the most recently changed bond angle to its original value, or if the moving side was switched after the angle was changed, the value at the time the moving side was switched. To preserve the ability to restore the original value, **Revert** to the original value before switching the side that moves.



## Align Chain Sequences

**Align Chain Sequences** generates a multiple sequence alignment (MSA) of structure chains in Chimera using a [Clustal Omega](#) or [MUSCLE](#) web service provided by the [UCSF Resource for Biocomputing, Visualization, and Informatics \(RBVI\)](#). The MSA web service can also be called from [Multalign Viewer](#) to [realign](#) an existing alignment. The coordinates of the structures are not used, only the sequences. By contrast, the [Match -> Align](#) tool generates an MSA using only the  $\alpha$ -carbon proximities in a 3D superposition. See also: [MatchMaker](#)

Clustal Omega users should cite:

[Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega](#). Sievers F, Wilm A, Dineen D, Gibson TJ, Karplus K, Li W, Lopez R, McWilliam H, Remmert M, Söding J, Thompson JD, Higgins DG. *Mol Syst Biol*. 2011 Oct 11;7:539.

MUSCLE users should cite:

[MUSCLE: multiple sequence alignment with high accuracy and high throughput](#). Edgar RC. *Nucleic Acids Res*. 2004 Mar 19;32(5):1792-7.

There are [several ways to start Align Chain Sequences](#), a tool in the **Sequence** category.

Two or more **Chains to align** should be chosen from the list of structure chains currently open in Chimera. A contiguous block can be chosen by click-and-drag, or by clicking on the first (or last) line in the block and then **Shift**-clicking on its last (or first) line. **Ctrl**-click toggles the status of a single line.

Options (default settings **bold**):

- **Use only selected part of chains (if any)** (true/false) - whether to include only the [selected](#) parts of the [chosen chains](#); if **false**, or if **true** but no part of a chain is [selected](#), the entire sequence will be used, including residues missing from the coordinates but present in SEQRES records
- **Alignment program (Clustal Omega/MUSCLE)** - whether to use [Clustal Omega](#) (default) or [MUSCLE](#)

If [Clustal Omega](#) [see the [README file](#) at the Clustal Omega website for details]:

- **Number of guide-tree/HMM iterations (1)** - how many combined guide tree/HMM iterations to perform after the initial alignment; can be 0 [--iterations]
- **Use full distance matrix during initial alignment (true/false)** - whether to use the full distance matrix for guide-tree calculation during the initial alignment [--full]
- **Use full distance matrix during alignment iteration (true/false)** - whether to use the full distance matrix for guide-tree calculation during iteration, if any [--full-iter]

If [MUSCLE](#) [see the [command-line reference](#) at the MUSCLE website and [BMC Bioinformatics 5:113 \(2004\)](#) for details]:

- **Maximum number of iterations (16)** - maximum number of iterations [-maxiters]
- **Maximum time to iterate in hours (0 means no limit) (0.0)** - maximum run time [-maxhours]
- **Find diagonals (faster execution if sequences are similar) (true/false)** - whether to use segments of high sequence similarity to constrain the alignment; may improve speed but at the expense of some accuracy [-diags]

**OK** initiates the calculation and dismisses the dialog, whereas **Apply** initiates the calculation without dismissing the dialog. The job will be run in the background; clicking the information icon  in the Chimera [status line](#) will bring up the [Task Panel](#), in which the job can be canceled if desired. **Cancel** dismisses the dialog, and **Help** opens this manual page in a browser window.

The output sequence alignment will be opened automatically in [Multalign Viewer](#).

---

UCSF Computer Graphics Laboratory / January 2014


[Tools Index](#)
**Animation**
[Introduction](#)
[Scenes](#)
[Keyframes, Timeline](#)
[Playback](#)
[Recording](#)
[Storyboard](#)
**Animation** Animation

\* **under development** \* — please send any comments and suggestions to [chimera-users@cgl.ucsf.edu](mailto:chimera-users@cgl.ucsf.edu)

The **Animation** tool allows saving and restoring [scenes](#) in Chimera, placing scenes along a [timeline](#), and [recording](#) a movie of the resulting animation. See also: [Movie Recorder](#), [Demos](#), [Rapid Access](#), [Chimera sessions](#), [making movies](#)

There are [several ways to start Animation](#), a tool in the **Utilities** category. Clicking **Preferences** opens the [Animation preferences](#). **Close** dismisses the **Animation** dialog, and **Help** opens this manual page in a browser window.

**Scenes**

A Chimera *scene* includes:

- locations and orientations of models and clipping planes (*etc.*, as in a saved [position](#))
- display styles, colors, and visibilities of molecule models, molecular surfaces, and their parts
- [volume](#) displays
- [axis, plane, and centroid](#) representations
- 3D labels, [2D labels](#), [color keys](#)
- [CASTp](#) pocket display
- certain global parameters such as [background](#), [lighting](#), and [effects](#)

Scenes are under development and do not include all aspects of Chimera. Scenes increase [session](#) file size.

Clicking the plus icon  in the **Scenes** section of the the [Animation](#) dialog saves the current state of Chimera as a scene and generates a thumbnail image to represent it. Clicking the thumbnail restores the corresponding scene. Scenes can also be saved and restored with the [scene](#) command, or restored from [Rapid Access](#).

Right-clicking a scene thumbnail raises a context menu with the options:

- **Add to timeline** - append the scene to the [timeline](#)
- **Properties** - open a dialog for editing the scene name and description
- **Update** - overwrite the scene with the current state of Chimera
- **Delete** - delete the scene<sup>\*</sup>

\* Deleting a scene also removes its occurrences from the [timeline](#).

Clicking a scene thumbnail not only restores the scene in Chimera but also chooses the scene (highlights the thumbnail) in the dialog. Multiple scenes in a contiguous block can be chosen by clicking the first (or last) thumbnail in the block and then **Shift**-clicking on the last (or first) thumbnail. **Ctrl**-click toggles the state of a single scene. The chosen scene(s) can be collectively:

- deleted<sup>\*</sup> by clicking the minus icon  in the **Scenes** section of the dialog
- appended to the [timeline](#) by clicking the plus icon  in the **Timeline** section of the dialog

## Keyframes, Actions, and Timeline

A *keyframe* is simply a [scene](#) in the context of a *timeline*. Each keyframe is associated with a preceding *transition* of one or more display frames. The timeline may also contain *actions*, or operations in Chimera other than transitions between keyframes.

Available **Actions**:

- **Rock** - oscillating rotation
- **Roll** - continuous rotation

In [Animation](#), a [scene](#) or an [action](#) can be inserted into the **Timeline** by drag-and-drop, or a scene can be appended with **Add to timeline** in its [right-click menu](#).

One or more [chosen scenes](#) (or similarly, an action chosen by clicking its

thumbnail) can be appended to the timeline by clicking the plus icon  above the timeline.

Keyframes and actions can be repositioned along the timeline by:

- dragging, where Shift-dragging also moves any subsequent (rightward) keyframes and actions in parallel
- editing [duration](#), which will also move any subsequent (rightward) keyframes and actions in parallel

Giving the first (leftmost) keyframe a duration > 0 frames indicates repeating its

contents accordingly, since there is no preceding keyframe to define a transition; this is indicated with a gray rectangle extending from the start of the timeline to the keyframe position.

The timeline contents can be [played back](#) to preview the movie that would result from [recording](#).

Right-clicking a keyframe or action thumbnail raises a context menu with the options:

- **Properties** - edit [properties](#) of the action, or for a keyframe, the preceding transition
- **Delete** - remove the keyframe or action from the timeline

Keyframe/action properties:

- **Duration**
  - **Duration in frames** - number of frames; if changed, all subsequent (rightward) keyframes and actions will shift accordingly
  - **Nth frame for discrete transitions** - frame at which discontinuous changes should occur

Properties that change continuously include: model orientations and scale; [clipping plane](#) positions; colors and display states (faded in/out with transparency) of atoms, ribbons, and [molecular surfaces](#); colors, display states, and positions of [2D labels](#). Discontinuous changes include those in [atom/bond styles](#) and [volume displays](#). Transitions between different frames of a [trajectory](#) will linearly interpolate the trajectory frame number but use only integer values, thus duplicating or skipping trajectory frames as needed to produce the specified duration.

- **Parameters** - action-specific settings
  - for **Rock**:
    - **Angle to rock through** (default 60°)
    - **Axis of rotation** (x/y/z)
    - **Times to rock** (default 1)
  - for **Roll**:
    - **Degrees to rotate** (default 360°)
    - **Axis of rotation** (x/y/z)
    - **Precess around axis** (true/false) - whether to also rotate around a moving axis that is carried along by the main rotation (see the [Wobble motion](#) movie in the Chimera Animation Gallery)
    - **Precession tilt** (default 10°) - the angle between the moving axis and the main axis of rotation

Clicking a thumbnail in the timeline chooses the keyframe or action (highlights the thumbnail). Multiple keyframes/actions in a contiguous block can be chosen by

clicking the first (or last) thumbnail in the block and then **Shift**-clicking on the last (or first) thumbnail. **Ctrl**-click toggles the state of a single keyframe or action. The chosen keyframes/actions can be collectively:

- removed from the timeline by clicking the minus icon  in the **Timeline** section of the dialog
- dragged in parallel along the timeline

Timeline display can be:

- expanded by clicking the zoom-in icon 
- subsequently restored to the original size by clicking the zoom-out icon



## Playback

In [Animation](#), a red vertical line (the *scrubber*) indicates playback position on the [timeline](#). Double-clicking a keyframe thumbnail moves the scrubber to that keyframe and restores the corresponding scene in Chimera. Other playback features are provided by clickable icons:

icon	meaning
	play animation; switches to a pause icon  during playback, and if playback is paused, to a resume-play icon 
	advance playback by a single display frame
	toggle looping during playback (brighter icon when on)

During playback, the target display rate is set to 25 frames per second, to match as closely as possible the speed of the movie that would result from [recording](#) with default encoding options ([details...](#)).

## Recording

Clicking the recording icon  in the [Animation](#) dialog opens a dialog with movie settings. After the necessary inputs have been specified, clicking **Record** initiates playing and recording the animation (as defined in the [timeline](#)) from start to finish. Recording in progress is indicated by a brighter icon, which can be clicked to abort the process; otherwise, the resulting series of images will be encoded into a movie file. Settings:

- **File name** - name for the resulting movie file
- **File type** (movie format) choices:
  - H.264 [.mp4]
  - VP8/WebM [.webm]
  - Theora [.ogv]
  - Quicktime [.mov]
  - AVI MSMPEG-4v2 [.avi]
  - MPEG-4 [.mp4]
  - MPEG-2 [.mpg]
  - MPEG-1 [.mpg]
  - WMV2 [.wmv]
- **Rendering:**
  - **Chimera** (default) - Chimera rendering, normally offscreen ([details...](#)). Images can be supersampled, that is, initially generated at a higher resolution and then sampled down to the final size.
    - **Supersample** (1x1/2x2/3x3/4x4) - how many pixels to sample in the X and Y dimensions for each pixel in the final saved image; thus, 1x1 corresponds to no supersampling. Higher values increase the smoothness of edges in saved images and increase calculation time with little effect on file size. 3x3 is generally recommended when supersampling is done.
  - **POV-Ray** - [raytrace with POV-Ray](#). This rendering option is the slowest but includes fancier effects such as high-quality shadows. The **POV-Ray Options** button opens the corresponding [preferences](#).

#### Advanced Options:

- **Quality** (highest/higher/high/good/medium/fair/low) - higher quality corresponds to higher (variable) playback bit rates and a larger movie file, assuming the same window size and movie frame rate
- **Image format** - image file format (however, PNG will be used regardless of this setting if [raytracing](#) is done):
  - JPEG [.jpeg]
  - PNG [.jpeg]
  - PPM [.jpeg] (default)
- **Additional recording options** - options are the same as for the command [movie record](#); if this field is left blank, image frames will be saved with default names in a default location (but normally deleted after movie encoding, depending on the encoding options)
- **Additional encoding options** - options are the same as for the command [movie encode](#); if this field is left blank, the movie will be encoded to play at 25 frames per second, and image frames will be deleted after the movie has been encoded

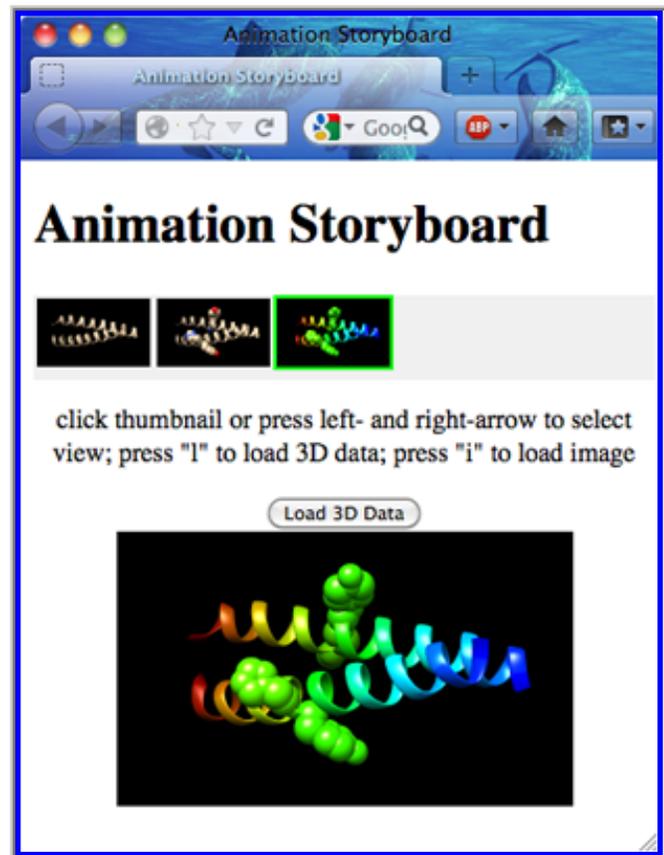
**Close** closes the dialog without initiating recording. **Image Tips** shows the [tips on preparing images](#), and **Help** opens this manual page in a browser window.

#### Animation Storyboard

Clicking the storyboard icon  in [Animation](#) opens a dialog for specifying a directory in which to save storyboard files. The files include **index.html**, with HTML title as specified in the save dialog, and three files for each [scene](#) in the [timeline](#):

- *scene\_name.html* - [WebGL export](#)
- *scene\_name.png* - full-sized image
- *scene\_name\_tn.png* - thumbnail image

The storyboard can be viewed by opening **index.html** (which embeds the remaining files) in a browser window. A small example is shown at right.



Across the top are thumbnails of the keyframes in the order in which they appear in the timeline, with consecutive keyframes that are the same scene collapsed into a single entry. The central panel shows either the static image or the 3D-manipulable [WebGL](#) rendering of a scene. Clicking a thumbnail or navigating the series of thumbnails with the keyboard left and right arrows updates the central panel to the static image of the corresponding scene. Pressing the **1** (letter ell) key or clicking **Load 3D Data** loads the corresponding WebGL data into the central panel. When WebGL is shown, the corresponding static image can be obtained again by clicking **Load Image**, or by clicking in the browser window outside of the central panel and then pressing the **i** key.

---

UCSF Computer Graphics Laboratory / July 2013

## APBS APBS

The **APBS** tool is an interface for running [APBS](#) (Adaptive Poisson-Boltzmann Solver) electrostatics calculations, using either a web service provided by the [National Biomedical Computation Resource \(NBCR\)](#) or a locally installed copy of the program. Users should cite:

[Electrostatics of nanosystems: application to microtubules and the ribosome.](#) Baker NA, Sept D, Joseph S, Holst MJ, McCammon JA. *Proc Natl Acad Sci USA*. 2001 Aug 28;98 (18):10037-41.

A structure should be prepared for APBS calculations by reconstructing missing heavy atoms, adding hydrogens, and assigning atomic charges and radii. These tasks can be done with [PDB2PQR](#) alone or in combination with parts of [Dock Prep](#). Atomic charges can be assigned with [Add Charge](#) or [PDB2PQR](#), although the latter may be preferred because it includes force fields developed specifically for Poisson-Boltzmann calculations. By [default](#), any explicit [solvent](#) (typically water) will be omitted from the APBS calculation.

The resulting [electrostatic potential map](#) will be opened as a new model in Chimera and the [Electrostatic Surface Coloring](#) tool for coloring [molecular surfaces](#) by potential will appear. Alternatively, the map can be shown as isopotential surfaces; these are not displayed automatically, but can be shown by starting [Volume Viewer](#) and clicking the eye icon or by using the [volume](#) command.

See also: [Coulombic Surface Coloring](#), [DelPhiController](#)

There are [several ways to start APBS](#), a tool in the **Surface/Binding Analysis** category. It is also implemented as the command [apbs](#).

- **Molecule** - the structure of interest (choose from pulldown menu of models in Chimera) \*\* If charges were assigned with [PDB2PQR](#), the model output by that step should be used, not the original model.\*\*
- **DX output file (optional)** - name and location of output [electrostatic potential map](#); if not specified, a temporary filename and location will be used.

### Options:

Focusing will be performed automatically; that is, there will be an initial electrostatics calculation on a larger grid with relatively coarse divisions, followed by another calculation on a smaller grid with finer divisions, for which the boundary conditions are determined from the first run [keyword **mg-auto**, [details](#) at the APBS site]. Default grid sizes (see **dime**, **cglen**, and **fglen** below) are based on the dimensions of the input structure.

- **Grid dimensions (dime):**  $[nx][ny][nz]$  - grid points per processor; dimensions in integer grid units along the molecule X, Y, and Z axes; commonly used values are 65, 97, 129, and 161 [[details](#) at the APBS site]

- **Coarse grid lengths (cglen):** [*xlen*][*ylen*][*zlen*] - dimensions in Å of the coarse grid along the molecule X, Y, and Z axes; the coarse grid should completely enclose the biomolecule [[details](#) at the APBS site]
- **Use molecule center for coarse grid center (cgcent) (true/false)** - whether to center the coarse grid on the molecule [[details](#) at the APBS site]
- **Coarse grid center coordinates:** [*xcen*][*ycen*][*zcen*] - if not centering on the molecule, coordinates of the center of the coarse grid in the molecule coordinate system
- **Fine grid lengths (fglen):** [*xlen*][*ylen*][*zlen*] - dimensions in Å of the fine grid along the molecule X, Y, and Z axes; the fine grid should enclose the region of interest in the molecule [[details](#) at the APBS site]
- **Use molecule center for fine grid center (fgcent) (true/false)** - whether to center the fine grid on the molecule [[details](#) at the APBS site]
- **Fine grid center coordinates:** [*xcen*][*ycen*][*zcen*] - if not centering on the molecule, coordinates of the center of the fine grid in the molecule coordinate system
- **Boundary condition for coarse grid (bcfl)** - how to initialize potential at the boundary of the coarse grid (results from the coarse run are then used to initialize potential at the boundary of the fine grid) [[details](#) at the APBS site]
  - **zero** - boundary potential set to zero; generally not recommended
  - **single Debye-Hückel** (default) - potential set to values prescribed by a Debye-Hückel model for a single sphere with a point charge, dipole, and quadrupole; the boundary should be sufficiently far from the molecule
  - **multiple Debye-Hückel** - potential set to values prescribed by a Debye-Hückel model for multiple noninteracting spheres with point charges; works better than the single approximation when the boundary is near the molecule, but can be very slow for large molecules
- **Solute dielectric constant (pdie)** (default 2.0) - dielectric constant inside the molecule [[details](#) at the APBS site]
- **Solvent dielectric constant (sdie)** (default 78.54) - dielectric constant outside the molecule [[details](#) at the APBS site]
- **Charge mapping method (chgm)** - how atomic partial charges are mapped to grid points [[details](#) at the APBS site]
  - **trilinear interpolation** - linear spline, charges mapped onto nearest-neighbor grid points; resulting potentials are very sensitive to the grid setup
  - **cubic B-spline discretization** (default) - charges spread out to two layers of grid points (nearest- and next-nearest neighbors); intermediate sensitivity to the grid setup
  - **quintic B-spline discretization** - charges spread out to three layers of grid points; lowest sensitivity to the grid setup
- **Include mobile ions (ion) (true/false)** - whether to include mobile ions in the calculation [[details](#) at the APBS site]
- **Positive ion charge (e), conc. (M), and radius:** [*q*][*c*][*r*] - if including mobile ions, positive ion charge in electron units (the value should be positive), molar concentration, and radius in Å. The total system of mobile ions must be electroneutral; for example, if the positive ion has twice the charge magnitude of the negative ion, its concentration should be half as high.
- **Negative ion charge (e), conc. (M), and radius:** [*q*][*c*][*r*] - if including mobile ions, negative ion charge in electron units (the value should be negative), molar concentration, and radius in Å
- **Poisson-Boltzmann equation:**
  - **linearized (lpbe)** (default)
  - **nonlinear (npbe)** - more computationally expensive to solve than the linearized equation, but more accurate for highly charged systems such as nucleic acids or phospholipid membranes
  - **size-modified (smpbe)** [[details](#) at the APBS site]

- **How to map dielectric values and ion accessibility (srfm)** [[details](#) at the APBS site]
  - **molecular surface** - partition space into regions of solute and solvent dielectric by the molecular surface (solvent-excluded surface calculated with the specified [solvent radius](#) and [density of points](#)), and into regions of ion inaccessibility or accessibility by the VDW surface inflated by the [ion radius](#)
  - **smoothed molecular surface** (default) - as above, except smooth the dielectric and accessibility values to reduce sensitivity to the grid setup
  - **cubic-spline surface** - use a cubic-spline surface to partition regions of solute dielectric and ion inaccessibility from regions of solvent dielectric and ion accessibility; the spline window width is set to 0.3 Å [[swin details](#) at the APBS site]
  - **7th-order polynomial** - use a seventh-order polynomial to partition regions of solute dielectric and ion inaccessibility from regions of solvent dielectric and ion accessibility
- **Surface density (sdens)** (default 10.0 points/Å<sup>2</sup>) - density of points used to calculate a molecular surface for [mapping values](#) [[details](#) at the APBS site]
- **Solvent radius (srad)** (default 1.4 Å) - solvent (probe) radius used to calculate a molecular surface for [mapping values](#) [[details](#) at the APBS site]
- **System temperature (temp)** (default 298.15 K) - temperature to use in the Poisson-Boltzmann equation [[details](#) at the APBS site]
- **Include explicit solvent (solvent)** (true/false) - whether to include any explicit [solvent](#) (typically water molecules) present in the [input model](#)

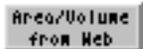
#### Executable location:

- **Opal web service** (default)
  - **Server** - URL of web service implemented using [Opal](#); clicking **Reset** restores the URL for the service provided by the [NBCR](#)
- **Local**
  - **Path** - pathname of locally installed executable

**OK** initiates the calculation and dismisses the dialog, whereas **Apply** initiates the calculation without dismissing the dialog. The job will be run in the background; clicking the information icon  in the Chimera [status line](#) will bring up the [Task Panel](#), in which the job can be canceled if desired. **Close** dismisses the dialog, and **Help** opens this manual page in a browser window.

The [electrostatic potential map](#) will be opened as a new model in Chimera, and the [Electrostatic Surface Coloring](#) tool for coloring [molecular surfaces](#) by potential will appear. Alternatively, the map can be shown as isopotential surfaces; these are not displayed automatically, but can be shown by starting [Volume Viewer](#) and clicking the eye icon or by using the [volume](#) command.

## Area/Volume from Web



**Area/Volume from Web** uses the [StrucTools server](#) to calculate atomic surface areas and Voronoi volumes from molecular coordinates. The calculated values are assigned as atom [attributes](#). See also: [Measure Volume and Area](#), [CASTp Data](#), [surface](#), [measure buriedArea](#)

Note that when a [molecular surface](#) is generated in Chimera, the total analytical solvent-accessible and solvent-excluded areas are automatically reported in the [Reply Log](#), and the values per atom and residue are assigned as [attributes](#) named **areaSAS** and **areaSES**. These values correspond directly to the molecular surface in Chimera and depend on the same parameters ([VDW radii](#) and various [molecular surface settings](#)). By contrast, only the settings in the **Area/Volume from Web** dialog are sent to the [StrucTools server](#), and other parameters such as VDW radii are determined solely by the process underlying the server.

There are [several ways to start Area/Volume from Web](#), a tool in the **Surface/Binding Analysis** category. In the resulting dialog, one or more molecule models can be chosen from the **Molecules** list, or chains from the **Chains** list.

Calculation options:

- **Accessible Surface (Gerstein)** [[details](#) at StrucTools]

- **Surface probe size** (1.3/1.4/1.5/1.6)
- **Atoms to use** (No Hetatms/Atoms + Hetatms/**All atoms except waters**)  
\*hydrogens are ignored

generates atom attribute **accessibleSurface** (default name)

- **Surface Area (MSMS)** [[details](#) at StrucTools]

- **Surface probe size** (1.3/1.4/1.5/1.6)
- **Atoms to use** (No Hetatms/Atoms + Hetatms/**All atoms except waters**)

generates atom attributes with default names **msmsArea\_MS** (solvent-excluded surface area) and **msmsArea\_SAS** (solvent-accessible surface area)

- **Voronoi Volume (Gerstein)** [[details](#) at StrucTools]

- **Method** (Normal Voronoi/**Method B**/Radical Plane/Modified Method B)
- **Radii** (**Chothia Radii**/Richards Radii)
- **Atoms to use** (No Hetatms/Atoms + Hetatms/**All atoms except waters**)  
\*hydrogens are ignored

generates atom attribute **voronoiVolume** (default name)

Different attribute names can be specified, keeping in mind that an attribute name should be an alphanumeric string without spaces (although underscores are allowed), should not begin with a digit,

underscore, or capital letter, and should not be the same as the name of a [built-in attribute](#).

Further options:

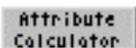
- **Open Render/Select by Attribute** - whether to start [Render/Select by Attribute](#), which shows a histogram of attribute values and allows mapping them to color or using them as [selection](#) criteria. The atomic values and totals per residue will be listed automatically in the dialog as atom and residue attributes, respectively. Even if this option is not used, the attribute(s) will still be available in [Render/Select by Attribute](#) if it is started later in the session.
- **Save server output to file** - whether to save the HTML output from the server as a file
- **Show server output in browser** - whether to show the HTML output in a browser window

**OK** initiates the calculation and dismisses the dialog, whereas **Apply** initiates the calculation without dismissing the dialog. **Close** dismisses the dialog, and **Help** opens this manual page in a browser window.

The transformed coordinates of atoms in the [chosen](#) model(s) or chain(s) are sent to the **StrucTools** server, where they are handled collectively. For example, if proteins in two different models are positioned to form a complex, the results obtained for each protein when both models are chosen will differ from the results obtained for each protein separately. The coordinates will include modifications made in Chimera, such as [bond rotations](#). However, other inputs such as atomic and probe radii are determined only by the server and the settings in the **Area/Volume from Web** dialog. The [atomic radii](#) and [molecular surface settings](#) in Chimera are not used.

For each attribute, the atoms are assigned the values calculated by the server (surface area in Å<sup>2</sup>, volume in Å<sup>3</sup>). The total area or volume and the number of atoms assigned values are reported in the [Reply Log](#) and [status line](#). When the output contains data for fewer atoms than were sent to the server, a warning message will report the number of “missing” values. This is quite common and rarely a cause for concern; for example, if the **No Hetatms** option is chosen, there will be no output for such atoms. Atoms missing from the server output will not be assigned attribute values, and atoms with uncalculatable Voronoi volumes will not be assigned a volume value. Attribute values can be summed over sets of atoms with the [Attribute Calculator](#).

## Attribute Calculator



**Attribute Calculator** generates new numerical [attributes](#) from existing ones. The calculated attribute values, or even simply the values of an existing attribute, can be written to a file.

There are [several ways to start Attribute Calculator](#), a tool in the **Structure Analysis** category.

One must specify a name for the attribute to be created and whether it will be an attribute of **atoms**, **residues**, or **molecules** (molecule models). The name must be an alphanumeric string without spaces; it can include underscores, but should not begin with an underscore, digit, or capital letter. Atom attributes with names containing **area**, **volume**, or **charge** (after word separation by camel case and/or underscores) will be totaled automatically to give the corresponding residue attributes.

The **Formula** is an arithmetic expression. Supported operators are + (addition), - (subtraction), \* (multiplication), and / (division). Parentheses can be used for grouping.

Terms in the expression can be:

- **numbers** - simple integers and floating-point numbers only, *i.e.*, no scientific notation
- **function calls** - currently, only the functions **sum** and **average** are supported. Averages are calculated considering only the atoms (or residues) with values for the attribute. For example, the average [kdHydrophobicity](#) of a model containing amino acid and water residues is calculated considering only the amino acid residues.
- **attributes** - attributes are of the form *scope.attribute*. The *scope* can be **atom**, **residue**, or **molecule**, and *attribute* is the name of a numerical attribute of the appropriate scope (for example, see the lists of attributes automatically defined for [atoms](#), [residues](#), and [molecule models](#)). When defining a larger scope and referring to values from a smaller scope, the attribute is actually a list of values. Typically, the value list must be reduced to scalar using a function.

For example, to define a residue (or molecule model) attribute equal to the average B-factor of its constituent atoms:

```
average(atom.bfactor)
```

Several options further affect the calculation and attribute assignment:

- **Restrict attribute assignment to current selection, if any** - whether, if a [selection](#) exists, to assign values of the new attribute to the selected items (atoms, residues, or molecule models) only
- **Restrict formula domain to current selection, if any** - whether, if a [selection](#) exists, only the attribute values of selected items should be used in the calculation
- **Open Render/Select by Attribute** - whether to open the [Render/Select by Attribute](#) tool after assigning attribute values. Even when this option is off, the attribute(s) will be available within the [Render/Select by Attribute](#) tool if it is opened later in the session.
- **Show calculation results in Reply Log** - whether to report results in the [Reply Log](#)
- **Save calculation results to file** - whether to write the results to a text file, in the [format](#) used as

input to [Define Attribute](#)

**OK** performs the calculation/assignment and dismisses the dialog. **Apply** performs the calculation/assignment without dismissing the dialog. **Close** dismisses the dialog, and **Help** opens this manual page in a browser window.

The [formula](#) is not reevaluated automatically; if any constituent attribute values change, **OK** or **Apply** must be used again to recalculate the result with the new values.

---

UCSF Computer Graphics Laboratory / March 2014

## Define Attribute

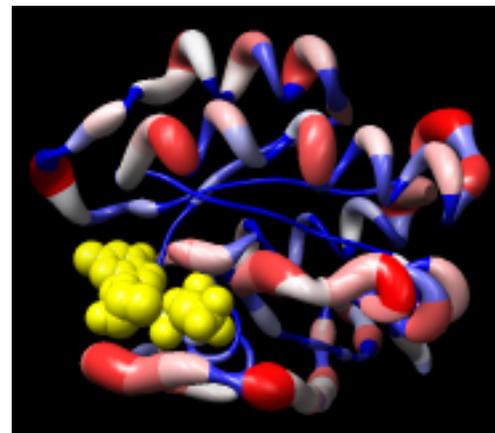
Items in Chimera such as atoms, bonds, residues, molecule models, and [segmentation regions](#) have *attributes*: properties with [names](#) and values.

Values of many attributes can be viewed and changed in the [attribute inspectors](#).

Attributes of atoms, residues, and molecule models can be created:

- automatically by Chimera [[atoms](#)] [[residues](#)] [[molecule models](#)]
- arbitrarily (or changed) with [Define Attribute](#), [defattr](#), or [setattr](#)
- by combining other attributes using the [Attribute Calculator](#)
- by various other Chimera tools such as [Add Charge](#), [Values at Atom Positions](#), and [Multalign Viewer](#)

H-ras (121p) rendered by residue percentExposed  
(see [examples](#))



and then:

- rendered visually with [Render by Attribute](#) or [rangecolor](#)
- displayed as labels (see command [labelopt](#) or custom labeling options under [Actions... Label](#))
- used in [selection](#) and command-line [atom specification](#)
- [saved](#) to an [assignment file](#)
- reported with [list](#)

**Define Attribute** reads atom, residue, and molecule model attributes from [assignment files](#). Values of existing attributes can be reassigned, or entirely new attributes created. The resulting attributes are included in saved [sessions](#). **Define Attribute** is also implemented as the command [defattr](#). See also: the [Attributes tutorial](#), [segmentation region attributes](#)

There are [several ways to start Define Attribute](#), a tool in the **Structure Analysis** category. Starting **Define Attribute** brings up a dialog for [opening](#) an input [attribute assignment file](#).

One or more molecule models should be chosen from the **Restrict to models** list; only these will be considered during attribute assignment. Individual models or blocks of models can be chosen with the left mouse button. Ctrl-click toggles the status of an individual model. To choose a block of models without dragging, click on the first (or last) and then Shift-click on the last (or first) in the desired block.

Additional options:

- **Open Render/Select by Attribute** - whether to open the [Render/Select by Attribute](#) tool after assigning attribute values. Even when this option is off, the attribute(s) will be available within the [Render/Select by Attribute](#) tool if it is opened later in the session.

- **Send match info to Reply Log** - whether to send details about the attribute assignments to the [Reply Log](#)

**OK** performs the assignment and dismisses the dialog, whereas **Apply** performs the assignment without dismissing the dialog. **Close** simply dismisses the dialog. **Help** opens this manual page in a browser window.

Assignments can only affect structures that are open when the attribute is defined/assigned. For example, atoms in models opened *after* radii have been changed using **Define Attribute** will have the Chimera [default radii](#).

## ATTRIBUTE ASSIGNMENT FILES

Attribute assignment files are simple user- or program-generated text files that can be read by [Define Attribute](#) or the command [defattr](#) to create attributes and assign their values. Examples are provided [below](#).

A hash symbol # at the beginning of a line indicates a *comment*. Basic properties of an attribute are described in *control lines* of the form:

*identifier: value*

Possible control line contents:

identifier	value	description
<b>attribute</b> (required)	alphanumeric string without spaces; can include underscores; cannot begin with a digit, underscore, or capital letter; atom attributes with names containing <b>area</b> , <b>volume</b> , or <b>charge</b> (after word separation by camel case and/or underscores) will be totaled automatically to give the corresponding residue attributes	attribute name, <a href="#">existing</a> or new
<b>match mode</b> (optional)	<b>any</b> (default) non-zero 1-to-1	expected matches per <a href="#">assignment line</a> (an error message will be sent to the <a href="#">Reply Log</a> when a line's match behavior differs from the match mode)
<b>recipient</b> (optional)	<b>atoms</b> (default) residues molecules	level of attribute assignment
<b>none handling</b> (optional)	<b>None</b> (default) string delete	treatment of values given as <b>none</b> or <b>None</b> : as the Python value None, as a string, or as removal of any assignment of that attribute from the specified item

The attribute assignments are described in *assignment lines* of the form:

(Tab)[atom-spec](#)(Tab)*attr-value*

Any kind of [atom specification](#) can be used.

[Control lines](#) and [comments](#) can be interspersed with assignment lines. However, if multiple attributes are defined in the same file, the **attribute** control line should precede all other lines for the corresponding attribute.

Allowed *attr-value* types are:

- real number (float)
- integer
- boolean (**true** or **false**, case-independent)
- color - an attribute whose [name](#) ends with **color** (case-independent) will be interpreted as a color-valued attribute. A color value can be expressed as:
  - a color name (a [built-in name](#) or one defined previously with [colordef](#))
  - RGBA values, four space-separated numbers each ranging from 0 to 1 (inclusive) that represent the red, green, blue, and opacity components of the color; if the last number is omitted, the opacity is set to 1 (completely opaque)
- string - a string of any characters except tabs; values that might be interpreted as one of the other types should be enclosed in double quotes

New numerical attributes will appear in the attribute lists of [Render/Select by Attribute](#); boolean and string attributes will be listed only in the [Select by Attribute](#) portion; color attributes will not be listed.

## ATTRIBUTE ASSIGNMENT EXAMPLES

- [percentExposed.txt](#) creates a new residue attribute, **percentExposed**, and assigns values previously calculated by [Getarea](#) for the protein structure 121p (see [raw results](#)). The [figure](#) at the top of the page was generated by opening the structure, defining the attribute, and then using [Render by Attribute](#) to color the structure and render it as a worm. Atoms are undisplayed except for the ligand (yellow spheres).
- [areaSESgxxg.txt](#) creates a new residue attribute, **areaSESgxxg**, solvent-excluded surface area of each type of amino acid in the context of a Gly-X-Gly tripeptide. The values can be used to normalize residue **areaSES** (automatically generated when a [molecular surface](#) is shown) to give fractional exposures of amino acids in protein structures. Thanks to the Murgita laboratory at McGill University for the data. The [literature reference](#) is provided in the file, along with instructions for calculating fractional exposures with [Attribute Calculator](#).
- [midasrad.txt](#) assigns atomic radii, *i.e.*, sets values for the existing [atom attribute radius](#). The first specification in the file is **#**, which assigns a value to all atoms. Subsequent lines reassign the radii of atoms of particular elements. Remember that atoms in models opened *after* the assignment will have the [default radii](#) instead of those in [midasrad.txt](#).
- [rescol.txt](#) sets values for the existing [atom attribute color](#) and the existing [residue attribute ribbonColor](#) according to amino acid type. Non-amino-acid residues are colored dark gray. The colors are specified with [built-in names](#). Note that the same results can be achieved with a

Chimera [command file](#) ([rescol.com](http://rescol.com)).

- [wwHydrophobicity.txt](#) creates a new residue attribute, **wwHydrophobicity**, with values according to the “interface” [amino acid hydrophobicity](#) scale of Wimley and White ([literature reference](#) provided in the file).
- [hhHydrophobicity.txt](#) creates a new residue attribute, **hhHydrophobicity**, with values according to the “biological” [amino acid hydrophobicity](#) scale of Hessa *et al.* ([literature reference](#) provided in the file).



## AutoDock Vina

The **AutoDock Vina** tool allows running ligand-receptor docking calculations with [AutoDock Vina](#), using either a web service provided by the [National Biomedical Computation Resource \(NBCR\)](#) or a locally installed copy of the program. Users should cite:

[AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading.](#) Trott O, Olson AJ. *J Comput Chem.* 2010 Jan 30;31(2):455-61.

Calculations are for a single ligand; database screening is not enabled. Docking results are shown automatically in [ViewDock](#). See also: [AddH](#), [Dock Prep](#)

There are [several ways to start AutoDock Vina](#), a tool in the **Surface/Binding Analysis** category. It is also implemented as the command [vina](#).

The receptor and ligand structures should be opened as separate models in Chimera. If the receptor structure contains MSE (selenomethionine) residues, incomplete side chains, or atoms with alternate locations, running [Dock Prep](#) beforehand to correct those issues is recommended. The **AutoDock Vina** tool runs [AutoDock accessory scripts](#) locally to (further) prepare the structures, such as to add hydrogens if they have not been added already with Chimera.

**Output file** - location and filename prefix for output files. If the prefix is *name*, the files generated by a successful run will be:

- *name* (or *name.pdbqt* if indicated in the file browser) - docking results in [PDBQT](#) format, automatically read into [ViewDock](#) when the calculation finishes
- *name.receptor.pdb* - receptor PDB file from Chimera, input to the AutoDock [receptor preparation script](#)
- *name.receptor.pdbqt* - processed receptor in [PDBQT](#) format, input to AutoDock Vina
- *name.ligand.pdb* - ligand PDB file from Chimera, input to the AutoDock [ligand preparation script](#)
- *name.ligand.pdbqt* - processed ligand in [PDBQT](#) format, input to AutoDock Vina
- *name.conf* - AutoDock Vina configuration file

**Receptor** - the model to use as the receptor (choose from pulldown menu)

**Ligand** - the model to use as the ligand (choose from pulldown menu)

**Receptor search volume options** - definition of the box in which to sample ligand positions

- **Resize search volume using [button]** - reassign the indicated mouse button to defining a search volume box. When there is no pre-existing box:
  - Dragging with the assigned button sweeps out a box aligned with the X, Y, and Z axes, the smallest possible box containing all voxels under the initial and final cursor positions.

Depending on the viewing angle, the box may be surprisingly large.

After a box has been created, it can be erased or further adjusted.

- Clicking the assigned button at a point not over the box erases the box.
- Dragging from a point over the box moves the frontmost face; also holding down Shift moves the the rearmost face instead of the frontmost.
- Dragging from a point not over the box translates the whole box in X and Y; Shift-dragging translates the box in the Z dimension.

**Center:** [x][y][z] - box center in the receptor coordinate system; can be edited directly

**Size:** [x][y][z] - box dimensions along X, Y, and Z in the receptor coordinate system; can be edited directly

**Receptor options** - settings for the [receptor preparation script](#):

- **Add hydrogens in Chimera (true/false)** - whether to add hydrogens in Chimera (see [addh](#)) before calling the script. The receptor prep script will check for hydrogens and add them if they are missing. AutoDock Vina needs the polar (potentially H-bonding) hydrogens to identify atom types for scoring purposes.
- **Merge charges and remove non-polar hydrogens (true/false)** - note AutoDock Vina does not use charges or nonpolar hydrogens, so this setting is not expected to affect results except for the presence or absence of nonpolar hydrogens in the processed receptor
- **Merge charges and remove lone pairs (true/false)** - note AutoDock Vina does not use charges or lone pairs, so this setting is not expected to affect results except for the presence or absence of lone pairs in the processed receptor (and there may not have been any lone pairs to start with)
- **Ignore waters (true/false)**
- **Ignore chains of non-standard residues (true/false)** - ignore chains composed entirely of residues other than the 20 standard amino acids
- **Ignore all non-standard residues (true/false)** - ignore all residues other than the 20 standard amino acids

**Ligand options** - settings for the [ligand preparation script](#):

The ligand prep script will check for hydrogens and add them if they are missing. AutoDock Vina needs the polar (potentially H-bonding) hydrogens to identify atom types for scoring purposes.

- **Merge charges and remove non-polar hydrogens (true/false)** - note AutoDock Vina does not use charges or nonpolar hydrogens, so this setting is not expected to affect results except for the presence or absence of nonpolar hydrogens in the ligand output files
- **Merge charges and remove lone pairs (true/false)** - note AutoDock Vina does not use charges or lone pairs, so this setting is not expected to affect results except for the presence or absence of lone pairs in the ligand output files (and there may not have been any lone pairs to start with)

**Advanced options** - docking parameters

- **Number of binding modes (1-10, default 9)** - maximum number of binding modes to generate
- **Exhaustiveness of search (1-8, default 8)** - thoroughness of search, roughly proportional to time
- **Maximum energy difference (kcal/mol) (1-3, default 3)** - maximum score range; binding modes with scores not within this range of the best score will be discarded

**Executable location:**

- **Opal web service** (default)
  - **Server** - URL of web service implemented using [Opal](#); clicking **Reset** restores the URL for the service provided by the [NBCR](#)
- **Local**
  - **Path** - pathname of locally installed executable

**OK** initiates the calculation and dismisses the dialog, whereas **Apply** initiates the calculation without dismissing the dialog. The calculation is run as a background job. Clicking the information icon  in the Chimera [status line](#) will bring up the [Task Panel](#), in which the job can be canceled if desired. **Close** dismisses the dialog, and **Help** opens this manual page in a browser window.

The docking results will be opened as a new model (with multiple submodels) and shown automatically using [ViewDock](#). Please see the [AutoDock Vina manual](#) for a description of the output values.



## Benchmark

**Benchmark** measures graphics card and CPU performance on standard Chimera rendering tasks. It facilitates comparisons among hardware systems, and for users with access to different systems, indicates which will give the best performance with Chimera. Results can be sent to the Chimera developers for inclusion in the [benchmark web page](#).

There are [several ways to start Benchmark](#), a tool in the **Utilities** category.

For accurate results, any models open in Chimera should be closed and any other processes running on the system should be halted before benchmarks are run. Other windows should not be placed on top of the Chimera graphics window during benchmark calculations.

The top part of the **Benchmark** panel contains explanatory text. Clicking **Run Benchmarks** performs all of the benchmark tasks (which may take several minutes) and appends the results to this text. The tasks include rendering [atomic coordinates](#) and [volume data](#). Clicking **Halt** aborts benchmark calculations in progress. After the full set of benchmarks has been run, the **Report Scores** button can be clicked. **Report Scores** uses the bug-reporting system to send the results to the Chimera developers for inclusion in the [benchmark web page](#). **Show scores reported by others** opens that page in a browser window.

Typically, **Measure frame rate continuously** (every second) or **one time** (when that button is clicked) would not be used during a benchmark run, but instead to evaluate the *actual* frame rate during some manually executed display task. On some systems, **Benchmark** results are affected by the monitor refresh rate. The actual frame rate is additionally subject to the fixed delay introduced by Chimera (upper limit ~30 frames/sec).

**Close** dismisses the **Benchmark** interface, and **Help** opens this manual page in a browser window.

## Individual Benchmarks

**Show individual test controls** displays buttons for running the various benchmarks individually.

- **molecule** - render a macromolecular structure, PDB entry 1f4h. The structure is opened and shown in various [representations](#). The values reported are:
  - frames per second in each representation, including **Null** (open but not displayed)
  - operations per second (**Ops**), where an operation consists of flipping the display status of each atom computationally but not visibly; this is a measure of CPU performance

The remaining benchmarks use volume data, a grid of numbers.

- **surface** - measure the OpenGL rendering rate for triangles by drawing the surface of the cube
- **mesh** - measure the OpenGL rendering rate for lines by drawing a mesh surface of the cube
- **contour** - measure CPU and memory performance by contouring a Gaussian bump in the cube at a

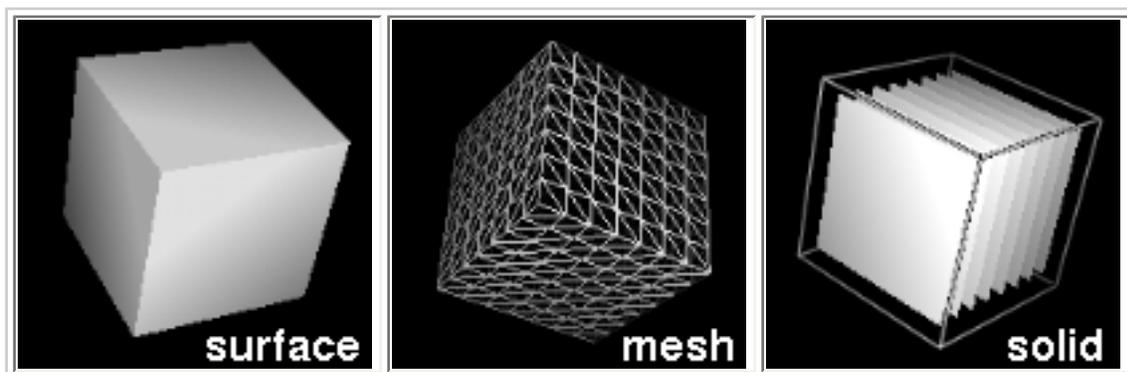
threshold that produces a spherical surface occupying nearly the whole volume (not displayed)

- **solid** - measure two-dimensional texture mapping performance by making each point in the cube opaque white
- **recolor** - measure bandwidth to the graphics card by repeatedly updating the color at each point in the cube

Each volume benchmark finds and reports the edge size  $N$  of an  $N$  by  $N$  by  $N$  cube on which the task can be performed at 10 frames per second (fps). An iterative approach is used to find the data size that can be handled at 10 fps; if this fails to converge, the **Halt** button can be used to stop the calculation. This edge size is also called the *score* for the task. Data that can be handled at 10 fps is easily manipulable in Chimera.

Pressing one of the buttons next to **Show standard model** (**surface**, **mesh**, or **solid**) will open the specified model type of the specified **size** in the graphics window. Depending on the size and viewing angle, a **solid** model may look like a stack of planes; this is an artifact of the rendering procedure. The models can be closed with the [Model Panel](#) or the command [close](#).

Rendering time depends on window size and model position within the view; these can be set as in the benchmark runs using the **Set standard view** buttons **camera** (position and orientation of model within the view) and **window size** (512 x 512 pixels).



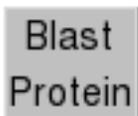
## TECHNICAL NOTES

**Chimera window comes to top.** When the Chimera graphics window is covered by other windows, some systems will shortcut the drawing process. Running any of the benchmarks brings the graphics window to the top.

**Timing method.** The benchmark timings use wall clock time (the Python `time.time()` call) instead of CPU time (Python `time.clock()`) because the latter does not measure the time used by the graphics card.

**Insufficient memory.** The **solid** volume benchmark can run out of memory on a machine with fast graphics relative to main memory size.

**Convergence method.** Iterative bisection is used to find the 10 fps size for volume data. An alternative is to get the time for a given size and use a scaling law. However, there is no simple relationship between rate and data size; pixel fill rate, number of primitives, texture memory size, and CPU cache size can all cause hard-to-predict scaling behavior.



## Blast Protein

**Blast Protein** performs protein sequence searches using a [BLAST](#) web service hosted by the [UCSF Resource for Biocomputing, Visualization, and Informatics \(RBVI\)](#). Corresponding structures can be retrieved and automatically superimposed, and the [pseudo-multiple alignment](#) from BLAST can be shown in [Multalign Viewer](#). The search results are included in saved [sessions](#). See also: [mda](#)

There are [several ways to start Blast Protein](#), a tool in the **Sequence** category. When it is started from the [Info menu](#) in [Multalign Viewer](#), any of the sequences in the alignment can be specified as the query. Otherwise, the query sequence can be:

- **From Structure** - a protein chain from a structure open in Chimera
- **Plain Text** - a protein sequence entered or pasted in the **Sequence** field as plain text

Search parameters:

- **Program**
  - **blast** (default) - protein-protein search
  - **psiblast** - iterated protein-protein search, where each pass after the first uses a PSSM (position-specific scoring matrix) based on the results of the previous pass
- **Database**
  - **pdb** (default) - sequences of structures in the [Protein Data Bank \(PDB\)](#). Whereas completely identical sequences are collapsed to a single entry in the **pdb** sequence database from NCBI, the [RBVI](#) web service uses a customized version of this database in which the identical sequences are separate entries. The **List only...** option controls whether multiple hits with the same PDB identifier or only the best-matching one should be included in the results.
  - **nr** - all non-redundant GenBank CDS translations + RefSeq Proteins + PDB + SwissProt + PIR + PRF. This database is much larger than **pdb** and will take longer to search.
- **E-value (1e-X)** - significance cutoff; only matches with E-values  $\leq 10^{-X}$  will be returned (X=3 by default)
- **Matrix** (BLOSUM45/**BLOSUM62**/BLOSUM80/PAM30/PAM70) - amino acid substitution matrix to use for alignment scoring
- **Passes** (default 1) - number of **psiblast** iterations; 1 pass is equivalent to **blast**, whereas multiple passes will find more distantly related sequences
- **List only best-matching chain per PDB entry** (default) - whether multiple hits with the same PDB identifier (matches to different chains in that PDB entry) or only the best-matching one should be included in the results

Clicking **OK** launches the search and dismisses the dialog. **Apply** launches the search without dismissing the dialog, **Cancel** simply dismisses the dialog, and **Help** opens this manual page in a browser window.

The search is a background task. Clicking the information icon  in the Chimera [status line](#) will bring up the [Task Panel](#), in which the job can be canceled if desired.

## Blast Results

Results are returned as a table of sequences with several columns of information. Which columns are displayed can be set in the [Column configuration](#) panel.

Some columns of information are available for **nr** sequences regardless of whether they are in the **pdb** subset:

- **GI** - sequence GI number (for the query, **model name** is shown instead)
- **Evalue** - significance value
- **Score** - alignment score
- **Description** - text description (truncated)

Additional columns are available for **pdb** sequences:

- **PDB** - corresponding PDB identifier, including chain identifier
- **Authors** - structure authors
- **Chain names** - chain identifiers and descriptions of polymer chains in the structure
- **Copies** - number of copies of the hit chain in the structure
- **Date** - structure deposition date
- **Ligand formulas** - chemical formulae of ligand chemical components
- **Ligand names** - names of ligand chemical components
- **Ligand smiles** - [SMILES](#) strings of ligand chemical components
- **Ligand symbols** - residue names of ligand chemical components
- **Ligand weights** - molecular weights of ligand chemical components
- **Method** - method of structure determination
- **Polymers** - number of different polymer chains in the structure (not counting multiple copies of the same sequence)
- **PubMed** - [PubMed](#) identifier of literature reference, if any
- **Residues** - number of residues in the hit chain
- **Resolution** - crystallographic resolution
- **Species** - scientific name of source organism
- **Title** - structure title
- **Total atoms** - total number of atoms in the structure (all chains)
- **Total residues** - total number of residues in the structure (all chains)
- **UniProt** - [UniProt](#) identifier, if any, for hit chain
- **Weight** - molecular weight of hit chain

Other than **PDB** identifier, the information in these columns is retrieved using a web service provided by the [RCSB PDB](#).

One or more of the hits (lines in the table) can be chosen with the left mouse button. A block of lines can be chosen by dragging, or by clicking on the first (or last) line in the desired block and then **Shift**-clicking on its last (or first) line. **Ctrl**-click toggles the state (chosen or not) of single line.

Buttons at the bottom of the results dialog:

- **Show in MAV** - display the [pseudo-multiple alignment](#) of the chosen sequences in [Multalign Viewer](#)
- **Load Structure** - retrieve and open the PDB structures corresponding to the chosen sequences (as described for [Fetch by ID](#)); they will be superimposed automatically according to the [pseudo-multiple alignment](#). Entire PDB entries are retrieved, not just the matching chains.
- **PDB Web Site** - show [RCSB PDB](#) entries for the chosen sequences in a browser window
- **Columns** - show/hide the **Column** configuration panel, which contains checkboxes to set which [columns](#) are displayed. A maximum column width (default **40** characters) can be specified. Buttons within the panel:
  - **All** - show all columns in the table
  - **Default** - restore the previously saved configuration
  - **Standard** - use the factory default configuration
  - **Set Default** - save the current configuration to the Chimera [preferences file](#)
- **Export...** save the chosen rows, in the current sort order and including the currently displayed [columns](#), as a file in CSV (comma-separated values) format. If no rows are chosen, all will be exported. Note that some columns may contain commas or newline characters (multiple lines), but these are handled according to CSV standard by enclosing the values in quotes. Spreadsheet software and other programs built to work with the CSV standard will interpret the exported files correctly, but user scripts need to account for the potential complexities, and viewing the file contents directly may be confusing.
- **Hide** - hide the table without deleting the results; it can be shown again using the **Raise** option of its [instance in the Tools menu](#). This is also useful when the table has become obscured by other windows.
- **Quit** - delete the results
- **Help** - show this manual page in a browser window

## Notes

**Basic Local Alignment Search Tool (BLAST).** The [BLAST](#) software is provided by the [NCBI](#) and described in the following:

[Gapped BLAST and PSI-BLAST: a new generation of protein database search programs.](#)

Altschul SF, Madden TL, Schäffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ. *Nucleic Acids Res.* 1997 Sep 1;25(17):3389-402.

[Basic local alignment search tool.](#) Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. *J Mol Biol.* 1990 Oct 5;215(3):403-10.

**Pseudo-multiple alignment.** The pseudo-multiple alignment from BLAST is not a true multiple alignment, but a consolidation of the pairwise alignments of individual hits to the query. This output corresponds to the BLAST formatting option ([alignment view](#)) "flat query-anchored without identities," in which identities and conservative changes relative to the query are shown in upper case, nonconservative changes in lower case.

## UCSF Chimera Web Data

[Chimera](#) can be used as a helper application for several types of files linked to web pages:

Web Data File Types		
type	suffix (extension)	contents
<a href="#">Chimera web data</a>	.chimerax	instructions on data files to open, commands and code to execute (can have a Chimera <a href="#">demo</a> embedded)
Mol2	.mol2	molecular structure
<a href="#">PDB</a>	.pdb .ent	molecular structure
Python	.py .pyc .pyo .pyw	Python code
VRML	.vrml .wrl	graphical object

While the other file types are somewhat standard, [Chimera web data \(chimerax\)](#) files are specifically meant to be interpreted by Chimera.

For Chimera to show the data when a link to a file is clicked,

1. the web browser must be [configured](#) to send the file to Chimera
2. except on Mac OS X, Chimera must be [enabled to accept](#) such files

Users will be asked to confirm whether files containing Chimera commands and/or Python code should be accepted. By default, this will happen the first time such a file is encountered during each Chimera session; other options are to ask each time and never (controlled in the [Web Access preferences](#)). Because commands and code written with malicious intent may harm a user's computer, **only files from a trusted source should be accepted.**



### Browser Configuration

For Chimera to act as a helper application for [data linked to web pages](#), the browser must recognize which files should be sent to Chimera. Except on Mac OS X, this can be set up automatically using the **Browser Configuration** tool (under **Tools... Utilities** in the Chimera menu). The alternative is to configure the browser [manually](#).

### Manual Configuration

The [automated approach](#) should be used where possible. The manual configuration procedure varies with

browser and system, and the examples below address only the Chimera web data (chimerax) file type. A similar approach can be used to specify Chimera as the helper application for additional file types.

### On Mac OS X:

After Chimera has been installed, the Mac automatically “knows” to use it as the helper application for the [registered file types](#). Browsers like **Safari** that cannot be configured to call Chimera directly can still be used to download such files, and opening any of the files (for example, by doubleclicking a file icon) will then call Chimera.

The simplest way to configure **Firefox** for file types not already assigned a helper application is to attempt to open the file, then instruct the browser to

**Open with** (browse to locate Chimera)

(the default location is **/Applications/Chimera.app**); check the option to do this automatically for such files in the future. If the file type is already assigned a helper application, the assignment can be changed using the browser preferences.

### On UNIX (Linux):

Browser preferences generally include a section for helper applications. A new type should be added with the following information:

```
Description: Chimera web data
File extension (suffix): chimerax
MIME type: application/x-chimerax
Application:
  CHIMERA_ROOT/bin/chimera_send
      - or sometimes -
  CHIMERA_ROOT/bin/chimera_send %s
```

where the Chimera installation directory should be substituted for CHIMERA\_ROOT.

In addition, UNIX systems use **.mime.types** to identify types of files downloaded from the web and **.mailcap** to associate these types with applications; both files reside in a user's home directory. Both files should be created if they do not already exist. The line

```
application/x-chimerax chimerax
```

should be added to the end of **.mime.types** and the lines

```
application/x-chimerax
CHIMERA_ROOT/bin/chimera_send %s
```

should be added to the end of **.mailcap**. The Chimera installation directory should be substituted for CHIMERA\_ROOT.

## On Windows:

**Internet Explorer** uses the system registry to determine which application should handle a particular type of file. Only users with administrative privileges can register a file type. The general procedure for registering the Chimera file type will be something like: from the Control Panel menu, choose (depending on the system) Tools... Folder Options or View... Options, then click the File Types tab. Create a new type with

**File extension:** chimerax

and (if asked)

**Description:** Chimera web data

**MIME type:** application/x-chimerax

Add a new action (it may be necessary to click on the line for the chimerax type and then the Advanced button to add this information):

**Action:** open

**Application used to perform action:**

```
"CHIMERA_ROOT\bin\chimera.exe" "--send" "%1"
```

where the Chimera installation directory should be substituted for CHIMERA\_ROOT (C:\Program Files\Chimera is the default). Check the box to "Always show extension" before closing the File Types panel.

## Accepting Web Data

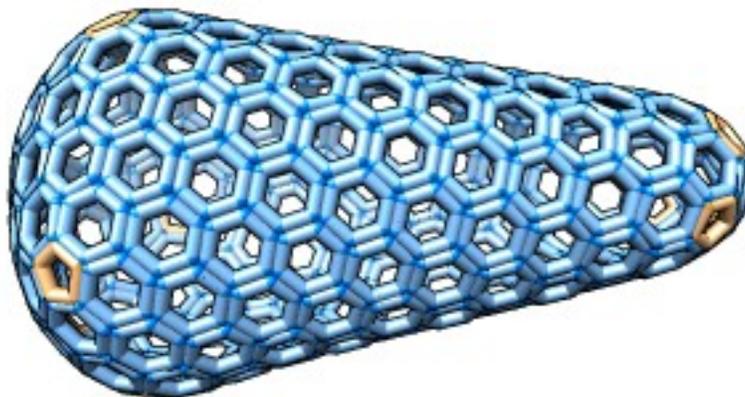
After a browser has been [configured](#) to send Chimera the appropriate files, setting **Accept web data** to **true** in the [Web Access preferences](#) enables the running instance of Chimera to accept such files. On Mac OS X, this preference is not shown, since Chimera will always be enabled to accept such files. If there is no running instance of Chimera enabled to accept web data, another instance of Chimera will be started and used to open the data. If there are multiple running instances of Chimera set to accept web data, the file will be sent to the instance that most recently had focus (was most recently clicked into).

Even when Chimera is enabled to accept web data, it is necessary to [confirm acceptance of certain file types](#) (once per session by default; controlled in the [Web Access preferences](#)).



## Cage Builder

**Cage Builder** creates polyhedral cages composed of hexagons, pentagons, squares and other polygons. The cages can be decorated with atomic models to create oligomeric molecular assemblies using the [sym](#) command (see [examples](#)). Cages are created as [marker sets](#). See also: [Icosahedron Surface](#), [hkcage](#), [meshmol](#)



There are [several ways to start Cage Builder](#), a tool in the **Higher-Order Structure** category.

- **Attach polygons with [ 3 ][ 4 ][ 5 ][ 6 ][ 7 ] sides** - clicking a number attaches a new polygon of the specified shape to each currently [selected](#) polygon edge, or if no such edges exist, creates a single new polygon. All polygons are part of a single Chimera [marker model](#) named **Cage**. The attached edges are automatically *joined*. Joined edges are shown in lighter colors than unjoined edges.

Clicking **Minimize** improves the alignment of joined edges. This is typically done after some rounds of new polygon attachment. Clicking **Delete** removes entire polygons that contain any [selected](#) edge or vertex. **Close** simply dismisses the dialog, and **Help** opens this manual page in a browser window.

Clicking **Options** reveals additional settings (clicking the close button  on the right hides them again):

- **[ Join ] or [ Unjoin ] selected polygon edges** - clicking **Join** when exactly two edges are [selected](#) will join those edges. The two edges should belong to different polygons, and a polygon will be relocated as needed to attach and join the edges. Their previous join partners, if any, will become unjoined; an edge can only be joined to one other edge at a time. Clicking **Join** with fewer or more than two edges selected has no effect. **Unjoin** will unjoin all joined pairs with one or both edges selected.
- **Join polygons so that [N] edges meet at a vertex** (default **on**) - whether to automatically join additional edges during new polygon attachment or when **Join** is used, with the constraint that  $N$  edges (default **3**) should meet at each vertex. For example, if a lone pentagon has all of its edges selected and the **6** button is clicked, not only will a new hexagon be attached (adjacent edges joined) to each side of the pentagon, but also each hexagon will be attached its neighboring hexagons. Polygon positions may be adjusted to accommodate joining. If this option is off, only the edges of new polygon attachment will be joined automatically.
- **[ Scale ] cage by [f]** - clicking **Scale** or changing the value and pressing Enter (return) resizes all polygons and moves them radially from the geometric center of the cage model by a factor  $f$

(default 2)

- **[ Set ] edge length [e] thickness [t] • inset [i]** (default on) - clicking **Set** or changing a value and pressing Enter (return) updates all polygons in the cage model to the specified edge length (default 1.0) and stick thickness (default 0.2) in physical units, typically Å. With inset on, the edge lengths are reduced by the inset value (default 0.1) so that the edges of attached polygons will not overlap completely. For well-aligned joined edges, using an inset equal to half the thickness makes the sticks representing the edges lie side by side. With inset off, the polygons are shown at the nominal size with attached edges right on top of each other (to the extent allowed by geometric constraints). The length, thickness, and inset settings will also be used for subsequently created polygons.
- **[ Expand ] cage by [g] times edge length** - clicking **Expand** or changing the value and pressing Enter (return) fattens the cage by moving each polygon outward along its normal by a factor  $g$  (default 1) times edge length, keeping polygon sizes the same. Subsequently clicking **Minimize** will bring the polygons back together, often leaving the whole cage with a rounder shape. This option has rather limited usefulness.
- **Create [ Mesh ] model from cage polygons, color** (a [color well](#), by default a dark gray) - clicking **Mesh** generates a new [marker model](#) from the existing polygons, in which attached edges are replaced by a single edge and attached vertices are replaced by a single vertex. The mesh model cannot be edited or resized with the **Cage Builder** dialog; all editing should be done with the polygon cage model.

## Placing Atomic Models on Cages

Copies of molecular models can be placed on each polygon using the [sym](#) command.

For example, one might want to put a single copy of the structure at each pentagon (ignoring any other types of polygons). First, the molecular model would need to be positioned as desired relative to one of the pentagons in the cage model. Models can be positioned relative to one another “manually” by [activating/deactivating](#) them for motion and [using the mouse](#).

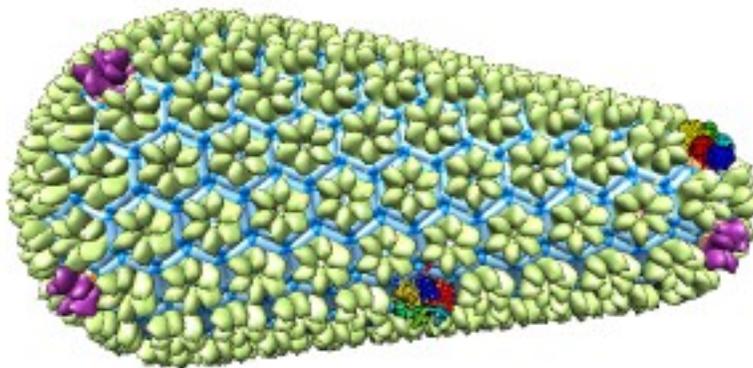
Then, if the molecule model is #1 and the cage model is #0, the following would place a copy of the molecule model at an equivalent position relative to each pentagon:

*Command:* [sym](#) #1 group #0,p5

The copies can be removed with [~sym](#).

Another possibility is to place multiple copies per polygon. For example, after a molecule model #1 has been positioned relative to (1/6) of a hexagon in cage model #0, the following would use  $C_6$  symmetry about the hexagon center to place six copies per hexagon:

*Command:* [sym](#) #1 group #0,pn6 surface true update true

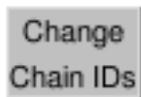


The additional arguments in the second example specify creating low-resolution [multiscale surfaces](#) instead of atomic copies (for faster rendering) and updating their positions when the original copy of the molecular model is moved relative to the cage.

Currently there is no good way to refine the molecular positions so that interfaces between molecules are physically realistic.

---

UCSF Computer Graphics Laboratory / July 2011



## Change Chain IDs

There are [several ways to start Change Chain IDs](#), a tool in the **Structure Editing** category. It is also implemented as the command [changechains](#). See also: [Build Structure](#), [Renumber Residues](#), [modifying and saving data](#)

The chain(s) to be assigned new identifiers (chain IDs) should be chosen from the list. For each chosen chain, an entry box will be supplied for the new ID, where case is important.

Clicking **Apply** or **OK** (which also dismisses the dialog) reassigns chain IDs as indicated, if possible. If the reassignment would give one or more duplicate residue numbers within a chain, an error message will appear and the reassignment will not occur.

**Close** dismisses the dialog; **Help** opens this manual page in a browser window.

**Handling multiple chains with the same ID.** Multiple chains with the same character ID present a problem because the dialog lists them as a single chain. However, if the ID is blank, disconnected chains will be detected and listed separately. Thus, the workaround is to:

1. assign them (collectively) a blank ID, *i.e.*, enter a space character and click **Apply**
2. assign unique IDs to the chains, now listed separately

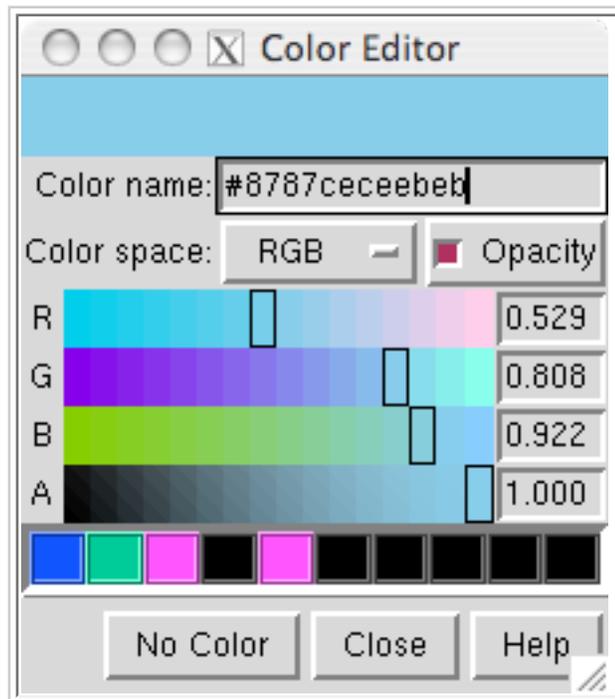
## Color Editor

The **Color Editor** allows colors to be created interactively. See also: the [Palette Editor](#), [colordef](#), [coloring](#)

There are [several ways to start](#) the **Color Editor**, a tool in the **Utilities** category. In addition, clicking a [color well](#) opens the **Color Editor** (if it is not already open) and *activates* the well so that it reflects any color changes within the **Color Editor**. The border of the color well turns white to signify activation.

The color defined in the **Color Editor** can be:

- applied using [Actions... Color... from editor](#)
- applied to any [color well](#) in a Chimera dialog by dragging and dropping with the mouse
- specified as **colorpanel**, **fromeditor**, or **editor** in [coloring](#) commands
- saved to a new color name with [colordef](#)



A color may be defined in the RGB, HLS, HSV, CMYK, or Gray (grayscale) **Color space**, and may include transparency (opacity < 1). In the RGB color space, for example, there are sliders for the red, green, and blue components. If **Opacity** is checked, a slider for the opacity A is also shown. Sliders can be moved or values can be entered directly. The bar under a slider shows what colors would be obtained for different positions of that slider, if the others were left unchanged. The currently defined color is shown in the top part of the **Color Editor** and applied to the items described by the receiving color well (active color well or drag/drop target color well).

Each color component is expressed as a value in the range 0-1. Color components are sometimes specified with integers 0-255 instead; if a value > 1 and ≤ 255 is entered, it is interpreted as coming from this scale and automatically divided by 255 to convert it to the 0-1 range.

The **Color name** field shows a **Tk color code** consisting of "#" and 12 digits, 4 each for the red, green and blue (RGB) components of a color. Each component is expressed as a hexadecimal number (allowable characters 0123456789abcdef) ranging from **0000** (0 in the decimal system) to **ffff** (65,535 in the decimal system). Tk color codes can have different numbers of digits, but always a multiple of 3: if 3 digits total, then there is 1 digit per color component with **f** meaning saturation; if 6 total, there are 2 digits per component with **ff** meaning saturation ([examples](#)); and so on. If a 3-, 6-, or 9-digit code is entered in the **Color name** field, it will be expanded to 12 digits. Rounding effects may be evident when a shorter, less precise code is expanded. The Tk code reflects the RGB definition of the color shown at the top of the **Color Editor**, regardless of the **Color space** setting. Tk color codes do not include transparency information.

The **Color Editor** can be set to one of the [built-in colors](#) by typing the name into the **Color name** field and

hitting return. The field will then revert to the corresponding Tk code.

Clicking the **No Color** button deletes the color assignments of the items described by the active color well. The ultimate result depends on the situation, because visible color is determined by a [hierarchy](#). Assigning "no color" at an overriding level reveals the colors at the next level in the hierarchy. For example, individual atom and bond colors overrule color assigned at the model level; if "no color" is assigned to a set of atoms, they will assume their model-level colors.

Colors can be saved in the *palette*, the bar of rectangles near the the bottom of the **Color Editor**. The currently defined color can be dragged from the top bar to a palette slot; definitions of colors in the palette are saved in the [preferences file](#). Clicking a color in the palette makes it the currently defined color.

The **Help** button brings up this manual page in a browser window. **Close** dismisses the **Color Editor**.



## ResProp

**ResProp** is a tool for assigning the 20 standard amino acids to categories. These categories appear in the [Select menu](#) (under **Select... Residue... amino acid category**) and can be used in [command-line specifications](#). There are several [predefined categories](#). **ResProp** allows changing category assignments and creating new, custom categories.

There are [several ways to start ResProp](#), a tool in the **Structure Analysis** category.

Under **Residue Category** is a pulldown menu of categories, with the name of the current (most recently chosen) category shown. Checkboxes control which of the 20 standard amino acids are assigned to the current category. **Changes are applied instantly**, so the category of interest should be created as needed and designated as the current category before changes are made. The [default category assignments](#), however, can be restored with the **Revert to defaults** button.

### Category Editing:

- **New category...** create a new category with a user-specified name and make it the current category
- **Delete...** delete the current category
- **Rename...** rename the current category
- **Revert to defaults** - restore the [default category assignments](#) without affecting any custom categories

Changes and additions are stored in `.chimera/ResProp/schemaData.py` in the user's home directory and will be carried through to subsequent Chimera sessions.

**Help** brings up this manual page in a browser window, while **Close** dismisses the **ResProp** dialog.

## Amino Acid Categories

Amino acid categories appear in the [Select menu](#) and can be used in [command-line specifications](#). Category assignments can be changed and new categories created with [ResProp](#). The default categorizations are taken from:

[Prediction of protein secondary structure and active sites using the alignment of homologous sequences.](#) Zvelebil MJ, Barton GJ, Taylor WR, Sternberg MJ. *J Mol Biol.* 1987 Jun 20;195(4):957-61.

Default Amino Acid Category Assignments								
hydrophobic	positive	negative	polar	charged	small	tiny	aromatic	aliphatic

ALA	X				X	X		
ARG		X		X	X			
ASN				X		X		
ASP			X	X	X	X		
CYS	X					X		
GLU			X	X	X			
GLN				X				
GLY	X					X	X	
HIS	X	X		X	X		X	
ILE	X							X
LEU	X							X
LYS	X	X		X	X			
MET	X							
PHE	X						X	
PRO						X		
SER				X		X	X	
THR	X			X		X		
TRP	X			X			X	
TYR	X			X			X	
VAL	X					X		X



## MatchMaker

**MatchMaker** superimposes protein or nucleic acid structures by first creating pairwise sequence alignments, then fitting the aligned residue pairs. Residue types and/or secondary structure information can be used to create the initial sequence alignments. Fitting uses [one point](#) per residue. Optionally, a structure-based multiple sequence alignment can be computed after the structures have been superimposed.

Note: if it is already known which residue numbers in one structure should be paired with which residue numbers in the other, another possibility is to use the command [match](#). See [superimposing structures](#) for a discussion of the different methods available in Chimera. See also: [Match -> Align](#), [Multalign Viewer](#), [Align Chain Sequences](#), the [Superpositions and Alignments tutorial](#), and

[Tools for integrated sequence-structure analysis with UCSF Chimera](#). Meng EC, Pettersen EF, Couch GS, Huang CC, Ferrin TE. *BMC Bioinformatics*. 2006 Jul 12;7:339.

There are [several ways to start MatchMaker](#), a tool in the **Structure Comparison** category. **MatchMaker** is also implemented as the command [mmaker](#) (or [matchmaker](#)).

The **MatchMaker** dialog is organized by the main steps to be performed:

1. [generating pairwise sequence alignments](#)
2. [matching](#), *i.e.*, superimposing the structures according to those pairwise alignments
3. optionally, [creating a multiple sequence alignment](#) from the structural superposition

**Save settings** writes the current **MatchMaker** parameters to the [preferences file](#). **Reset to defaults** resets the dialog to the factory default parameter settings without changing any preferences.

Clicking **OK** or **Apply** will start the calculations with or without closing the dialog, respectively. Sequence alignment scores, parameter values, and structure RMSDs will be reported in the [Reply Log](#).

**Cancel** simply closes the dialog, while **Help** opens this manual page in a browser window.

### ← Initial Pairwise Sequence Alignments

**Further restrict matching to current selection** allows ignoring residues of the reference and/or match structures that are not [selected](#). In general, restriction should only be used in specific cases to suppress results that would otherwise be obtained. For example, two chains that would otherwise align on their N-terminal domains can be forced to align on their C-terminal domains by selecting the C-terminal domains and using the restriction option. Otherwise, restriction is not recommended, because full-length alignments tend to be of higher quality, and [iteration](#) already serves to exclude poorly superimposed regions from the final fit. Although unselected parts of matched chains will appear in the resulting

sequence alignment (if [shown](#)), they have simply been added back in as “filler,” without consideration of how the characters align, after alignment and matching of only the [selected](#) residues.

### Chain pairing options:

- **Best-aligning pair of chains between reference and match structure** (default) - One reference structure and one or more structures to match should be chosen. For each structure to be matched, the reference-match pair of chains with the highest [sequence alignment score](#) will be used.
- **Specific chain in reference structure with best-aligning chain in match structure** - One reference chain and one or more structures to match should be chosen. Individual lines or blocks of lines can be chosen with the left mouse button; Ctrl-click toggles the status of a line. For each structure to be matched, the chain that aligns to the reference chain with the highest [sequence alignment score](#) will be used.
- **Specific chain(s) in reference structure with specific chain(s) in match structure** - One or more reference chains should be chosen from the list. For each reference chain chosen, one chain to be matched should be chosen from the corresponding pulldown menu. If multiple chains are to be matched to the same reference chain, it is necessary to match them in separate steps (by choosing the chain to match and then clicking **Apply**). A given chain cannot be matched to two different reference chains simultaneously, and chains from the same structure (molecule model) cannot simultaneously serve as a reference chain and a chain to match.

### Alignment algorithm:

- **Needleman-Wunsch** (default) - global
- **Smith-Waterman** - local

Sequence alignment scoring can include a residue similarity term, a secondary structure term, and gap penalties.

- **Matrix** (default **BLOSUM-62**) - what [substitution matrix](#) to use for the residue similarity part of the score. If an amino acid matrix is chosen, only peptide sequences will be aligned; if a nucleic acid matrix is chosen, only nucleic acid sequences will be aligned. An error message will appear if there are no reference-match pairs of the appropriate type.
- **Gap penalties**: When [secondary structure scoring](#) is included, the secondary-structure-specific **Gap opening penalties** (**Intra-helix**, **Intra-strand**, **Any other**) are used instead of the single **Gap opening penalty**. The same **Gap extension penalty** is used, however.
- **Include secondary structure score (N%)** (default **on** and  $N=30$ ) - whether to include a secondary structure term in the score, and at what weight relative to the residue similarity term. **Show parameters** reveals the secondary structure scoring parameters.  $N$  reflects the relative weights of the terms, which can be adjusted by moving the slider. If the weight is 30%, for example,

$$\text{total score} = 0.70(\text{residue similarity score}) + 0.30(\text{secondary structure score}) - \text{gap penalties}$$

Setting the weight to 0% is [not the same](#) as turning the option off, however. The values in the secondary structure **Scoring matrix**(for all pairwise combinations of **H**helix, **S**strand, and **O**other)

and the secondary-structure-specific **Gap opening penalties** can be adjusted. **Reset secondary structure scoring parameters to defaults** can be used to restore the default values of all secondary structure scoring parameters.

- **Compute secondary structure assignments** (available when [secondary structure scoring](#) is used; default **on**) - whether to first identify helices and strands by running the [ksdssp](#) algorithm, overwriting any pre-existing secondary structure assignments (except for CA-only structures, which are automatically skipped). This option may improve superposition results by generating consistent assignments, whereas pre-existing assignments may reflect the use of different criteria on different structures. [Ksdssp](#) parameter defaults can be adjusted with the [compute SS](#) dialog (opened from the [Model Panel](#)).
- **Show pairwise alignment(s)** (default **off**) - whether to display the resulting pairwise reference-match sequence alignments; each will be shown in a separate [Multalign Viewer](#) window. When [fit iteration](#) is employed, the pairs used in the final fit will be shown in the alignment as a [region](#) (colored boxes) named **matched residues**. The "RMSD: ca" [header](#) is automatically shown above the sequences, with histogram bar heights representing the [single-point](#) spatial variation among residues associated with a column. In the pairwise case, the value per column is simply the distance between the two associated residues.

\*When the fit has been [restricted](#) to [selected](#) residues, the unselected residues of matched chains will still appear in the alignment, but merely as a convenient compact representation; how they are aligned is not meaningful.

\*\*These pairwise sequence alignments can be considered a by-product of superposition. Successful superposition only requires these alignments to be partly correct, as incorrect portions tend to be excluded from the fit during [iteration](#). If the sequences are easy to align (highly similar), the sequence alignments are likely to be correct throughout. However, if the sequences are more distantly related, parts of the alignments may be incorrect even when a successful superposition is produced. In those cases, a [structure-based alignment](#) should be superior.

## ← Matching

Fitting uses one point per residue: CA atoms in amino acids and C4' atoms in nucleic acids. If a nucleic acid residue lacks a C4' atom (some lower-resolution structures are P traces), its P atom will be paired with the P atom of the aligned residue.

**Iterate by pruning long atom pairs until no pair exceeds [x] angstroms** (default **on** and  $x=2.0$ ) - whether to iteratively remove far-apart residue pairs from the "match list" used to superimpose the structures. This does not change the initial sequence alignment, but restricts which columns of that alignment will be used in the final fit. Otherwise, all of the columns containing both sequences (i.e. without a gap) will be used. In each cycle of iteration, atom pairs are removed from the match list and the remaining pairs are fitted, until no matched pair is more than  $x$  Å apart. The atom pairs removed are either the 10% farthest apart of all pairs or the 50% farthest apart of all pairs exceeding the cutoff, whichever is the lesser number of pairs. Iteration tends to exclude sequence-aligned but conformationally dissimilar regions such as flexible loops, allowing a tighter fit of the best-matching "core" regions.

Regardless of which chain(s) in a model to be matched are aligned in sequence to the reference, the entire

model will be reoriented.

## ← Final Structure-Based Sequence Alignment

If **MatchMaker** is used simply to superimpose structures, this step can be omitted. However, if one also wants a corresponding structure-based sequence alignment, this step is recommended, especially if the sequences are dissimilar.

**After superposition, compute structure-based multiple sequence alignment** (default off) - call [Match -> Align](#) to generate a sequence alignment consistent with the superposition. If not called with this option, [Match -> Align](#) can still be started later independently.

Calculating a structure-based alignment can take several minutes, depending on the number of structures, but there are advantages:

- it can provide better statistics for describing structural similarity (RMSD, *etc.*) because more alignment columns are correct
- it can produce a multiple sequence alignment, whereas the initial sequence alignments are only pairwise

The output sequence alignment is automatically shown in [Multalign Viewer](#) and can be [saved to a file](#) from that tool. The fully populated columns are highlighted as a [region](#) (colored boxes). Clicking the region will select the corresponding parts of the structures, in effect their common cores. The “**RMSD: ca**” [header](#) shows the spatial variation per column.

## Notes

**Meaning of 0% secondary structure score.** Turning off [Include secondary structure score](#) is not the same as moving the slider to zero with the option turned on. When the option is on:

- The secondary-structure-specific gap opening penalties are used regardless of the slider position.
- If [Compute secondary structure assignments](#) is also turned on, [ksdssp](#) is run and may change pre-existing secondary structure assignments.

Match  
-> Align

## Match -> Align

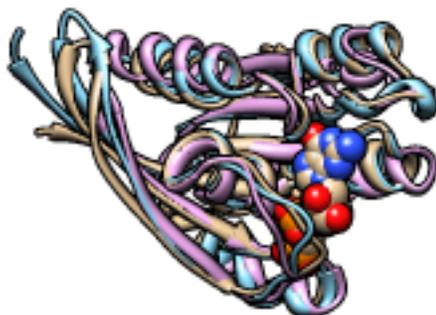
**Match -> Align** creates a sequence alignment from a [structural superposition](#) of proteins or nucleic acids in Chimera. Residue types are not used, only their spatial proximities. Iterations of refitting the structures using the sequence alignment and generating a new sequence alignment can be performed.

The output sequence alignment is automatically shown in

### [Multalign Viewer](#)

and root-mean-square deviations (RMSDs) over the fully

populated columns of the alignment and other [structural similarity scores](#) (SDM, Q-score) are reported in the [Reply Log](#).



		91	101	111
RMSD: ca				
121 p, chain A	87	T . K S F E D I H Q	Y R E Q I K . R V K	. D S . D D V P M V
1r2q, chain A	105	E . E S F A R A K N	W V K E L Q R Q . A	S . P N . . I V I A
1j2j, chain A	97	R E R . V N E A R E	E L M R M L A E D E	L . R D . . A V L L
RMSD: ca		121	131	141
121 p, chain A	113	L V G N K C D L . A	A R T . V E S . R Q	A Q D L . . A R S .
1r2q, chain A	130	L S G N K A D L A N	K R A . V D F . Q E	A Q S Y . . A D D .
1j2j, chain A	123	V F A N K Q D L . P	. N . A M . N A A E	I T D K L G L . H S
RMSD: ca		151	161	171
121 p, chain A	137	Y . . . G I P Y I E	T S A K T R Q G V E	D A F Y T L V R E I
1r2q, chain A	155	N . . . S L L F M E	T S A K T S M N V N	E I F M A I A K K L
1j2j, chain A	148	L R H R N W Y I Q A	T C A T S G D G L Y	E G L D W L S N Q L

For an informal introduction, see the [Superpositions and Alignments tutorial](#). See also:

[Align Chain Sequences](#), [MatchMaker](#), [Multalign Viewer](#), and

[Tools for integrated sequence-structure analysis with UCSF Chimera](#). Meng EC, Pettersen EF, Couch GS, Huang CC, Ferrin TE. *BMC Bioinformatics*. 2006 Jul 12;7:339.

There are [several ways to start Match -> Align](#), a tool in the **Structure Comparison and Sequence** categories.

Chains to be included in the sequence alignment should be chosen from the top section of the panel.

- **Residue-residue distance cutoff (angstroms)** (default 5.0) - maximum CA-CA distance (C4'-C4' for nucleic acids) for defining membership in a column of the output sequence alignment
- **Residue aligned in column if within cutoff of:**
  - at least one other (default)
  - all others

- how the [cutoff](#) should be applied; equivalent for alignments of only two chains
- **Gap character** - how to show gaps in the output sequence alignment
  - . (period)
  - - (dash)
  - ~ (tilde)
- **Allow for circular permutation** - whether to double sequences as needed to simultaneously align the N-terminal region of one protein with the C-terminal region of the other and vice versa. Doubling the sequence of one of a pair of proteins related by *circular permutation* is required because **Match->Align** enforces N → C chain directionality. Information on any permutations will

be sent to the [Reply Log](#).

- **Iterate superposition/alignment...** whether to perform one or more cycles of refitting the structures using the sequence alignment and generating a new sequence alignment from the adjusted superposition. Superpositions will not be adjusted unless iteration is turned on. Only the final superposition and final sequence alignment will be shown. The number of fully populated columns in any intermediate sequence alignments and corresponding match statistics will be reported in the [Reply Log](#).

#### Iteration Parameters:

- **Iterate alignment:**
  - **at most [M] times** (default 3) - refit and then generate a new sequence alignment *N* times (or fewer, if convergence is reached)
  - **until convergence** - refit and then generate a new sequence alignment until the number of fully populated columns no longer increases
- **Superimpose full columns:**
  - **across entire alignment** - refit the structures using all fully populated columns of the sequence alignment
  - **in stretches of at least [L] consecutive columns** (default 3) - refit the structures using only the fully populated columns in consecutive stretches of *L* or more
- **Reference chain for matching** [*chain*] - which structure should remain fixed as the others are matched to it

**Save settings** writes the current **Match -> Align** parameters to the [preferences file](#). **Reset to defaults** resets the dialog to the factory default parameter settings without changing any preferences.

Clicking **Apply** (or **OK**, which also dismisses the dialog) initiates the calculation.

The output sequence alignment is automatically shown in [Multalign Viewer](#) and can be [saved to a file](#) from that tool. The fully populated columns are highlighted as a [region](#) (colored boxes). Clicking the region will select the corresponding parts of the structures, in effect their common cores. The “**RMSD: ca**” [header](#) shows the spatial variation per column.

The number of fully populated columns in the alignment and the corresponding pairwise and overall RMSDs are reported in the [Reply Log](#). The [structural similarity measures](#) SDM and Q-score are also given. Structures are not refit using the final sequence alignment; rather, the existing superpositions are simply evaluated over the fully populated columns of that alignment.

**Close** dismisses the dialog without generating an alignment. **Help** opens this manual page in a browser window.

## Measures of Structural Similarity

**Match -> Align** reports root-mean-square deviations (RMSDs) calculated using one atom per residue: CA in amino acids, C4' in nucleic acids. However, RMSD values alone are not meaningful, because a lower RMSD can be achieved simply by using fewer residues to calculate it, even if the superposition stays exactly the same. When an RMSD value is used to describe structural similarity, the number of positions used in the calculation (the alignment length) should also be reported.

Several measures that normalize RMSD by alignment length and overall sequence length have been developed. Besides RMSDs, **Match -> Align** calculates and reports the **Structural Distance Measure (SDM)** and the **Q-score** (Q for quality). These scores depend not only on the structural similarity, but also on how well the structures are superimposed and the [cutoff](#) distance used to equivalence residues. A larger cutoff will increase both the alignment length and the RMSD.

The **SDM** is described in:

[Molecular anatomy: phyletic relationships derived from three-dimensional structures of proteins.](#) Johnson MS, Sutcliffe MJ, Blundell TL. *J Mol Evol.* 1990 Jan;30(1):43-59.

SDM values were found to be linearly related to sequence-derived distances and to yield similar phylogenetic trees. As a distance measure, SDM is zero for identical structures and increases as the similarity decreases. The [cutoff](#) used for residue pairing is included in the SDM equation; values to be compared with any previously published values should be calculated with the same cutoff distance. The original publication used a cutoff of 3.5 Å.

The **Q-score** is described in:

[Secondary-structure matching \(SSM\), a new tool for fast protein structure alignment in three dimensions.](#) Krissinel E, Henrick K. *Acta Crystallogr D Biol Crystallogr.* 2004 Dec;60(Pt 12 Pt 1):2256-68.

Q-scores were found to agree fairly well for superpositions (of the same structures) from different servers. Values range from zero for completely dissimilar or unsuperimposed structures to 1 for identical structures. The Q-score equation includes an empirical constant of 3.0 Å regardless of the [cutoff](#) used for residue pairing.

Note that SDM and Q-score were developed as pairwise measures. Their formulas easily generalize to the multiple case, and this generalization has been implemented in **Match -> Align**, but keep in mind that only pairwise scores (obtained by running **Match -> Align** on two structures at a time) can be compared to previously published values.

## Sequence Alignment Derivation

In most cases, a semi-heuristic algorithm is used. However, a modified Needleman-Wunsch procedure (dynamic programming) is used for the case of two chains and no allowance for circular permutation:

The score for aligning a pair of residues is:

- (cutoff - distance) for distances no greater than the [cutoff](#)
- -1 for distances greater than the [cutoff](#)

The gap penalty is zero, since for this application the spatial proximity should be more important than adjacency in sequence; that is, residues farther apart than the distance cutoff should not be aligned.

This process determines the sequence alignment that best represents the structural alignment.



## Morph Conformations

**Morph Conformations** creates a trajectory that morphs between two or more structures. A few [example systems](#) are listed below. [MD Movie](#) is automatically used to show the trajectory and can [record](#) it as a movie. See also: [Morph Map](#), [RR Distance Maps](#), [coordset](#), [making movies](#), the [ParM filament tutorial](#) at the Chimera web site

**Morph Conformations** is under development (see [limitations](#)).

There are [several ways to start Morph Conformations](#), a tool in the **Structure Comparison** category. It is also implemented as the command [morph](#).

The different structures should be opened as separate models or submodels in Chimera and [superimposed](#). The apparent motion across a morph trajectory depends on how the input structures are superimposed; matched regions will remain approximately steady. The structures can have different numbers of residues or different sequences (homologs or mutants can be compared), but currently they must contain [equal numbers of chains](#). Chains are paired by chain ID if the sets of IDs are identical, otherwise by order of occurrence in the input files.

The next step is to populate the **Conformations** list with structures; the order in the list corresponds to the order in which they will be visited in the output trajectory. The same model can be listed more than once to produce a morph trajectory that visits the same conformation more than once, for example: A → B → C → A.

Clicking **Add...** brings up a dialog that lists the open molecule models. Clicking a model's name and then the **Add** button (or double-clicking the model's name) puts it in the **Conformations** list. Multiple models can be chosen and added simultaneously. A block of models can be chosen by dragging, or by clicking on the first (or last) line in the desired block and then **Shift**-clicking on its last (or first) line. **Ctrl**-click toggles the state (chosen or not) of single line. After the desired conformations have been added, the model-choosing dialog can be dismissed by clicking **Close**.

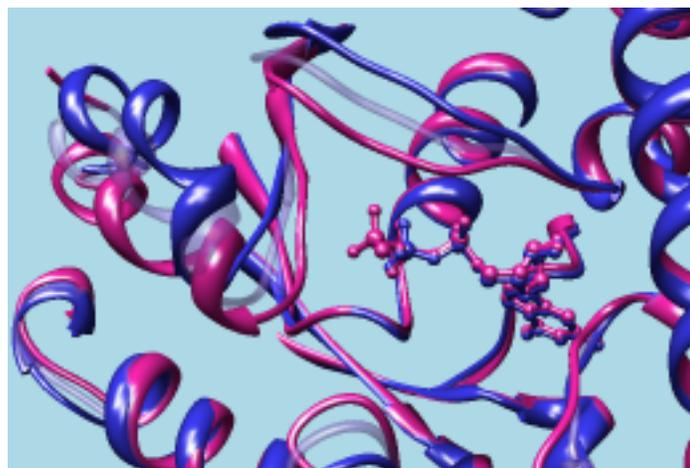
Clicking a line in the **Conformations** list designates that entry as the target of subsequent button actions:

- **Remove** - remove the conformation
- **Up** - move the conformation higher in the list (earlier in the trajectory)
- **Down** - move the conformation lower in the list (later in the trajectory)

Each sequential pair of structures in the **Conformations** list will serve as the starting and ending points of

### GTP-binding switch

(1tagA, 1tndA, morph intermediate)



one *segment* of a morph trajectory. A morph trajectory can have multiple segments. Within each segment, *intermediates* are generated by interpolating the positions of the [atoms in common](#). Interpolation includes:

1. rigid-body transformations of atom groups partitioned by hinge regions. Hinges are identified as described in [Krebs and Gerstein, Nucleic Acids Res 28:1665 \(2000\)](#).
2. coordinate changes within the atom groups

How interpolated coordinates are generated, and in how many steps, can be specified independently for each [segment](#). The following four settings apply to the segment preceding the structure currently highlighted in the **Conformations** list.

- **Interpolation method** - how the [rigid-body transformations](#) will be calculated
  - **corkscrew** (default) - Each group's starting and ending positions are related by a rotation about a center chosen to describe as much of the movement as possible, and a translation along the axis of rotation. These two components are interpolated.
  - **independent** - Each group's starting and ending positions are related by a rotation about the group's center of mass, and a translation. These two components are interpolated.
  - **linear** - Each group's starting and ending positions are related by a translation, which is interpolated.
- **Interpolation rate** - how conformational changes will be distributed across the [segment](#)
  - **linear** (default) - coordinate changes will be distributed approximately evenly
  - **ramp down** - coordinates will change most rapidly near the starting conformation
  - **ramp up** - coordinates will change most rapidly near the ending conformation
  - **sinusoidal** - coordinates will change most rapidly halfway between the starting and ending conformations
- **Interpolation steps [K]** - the starting and ending conformations of the [segment](#) will be *K* steps apart (default **20**); *K*-1 [intermediates](#) will be generated
- **Force Cartesian intermediates** (off by default) - whether [within-group coordinate changes](#) should be interpolated strictly in Cartesian space. Otherwise, internal coordinates will be used for the interpolation where possible. Using internal coordinates is slower but produces less distortion. A trajectory made with Cartesian forcing may be acceptable if few atomic details will be shown (for example, if only ribbons will be displayed).

The **Minimize** setting applies collectively to all [segments](#) of the morph trajectory:

- **Minimization steps [M]** - *M* steps of minimization (default **60**) will be applied to each [intermediate](#)

Minimization requires the correction of structural inconsistencies, addition of hydrogens, and association of atoms with [force field parameters](#). [Dock Prep](#), [AddH](#), [Add Charge](#), and [Minimize Structure](#) are called in no-GUI mode to perform these tasks; that is, the dialogs will not appear, but each tool will execute with default settings. When minimization is turned on, interpolation to generate an [intermediate](#) will use the minimized coordinates of the prior intermediate.

Clicking **Create** hides the dialog (unless the option to **Keep dialog up after Create** is checked) and initiates the calculation. The [MD Movie](#) tool will be called to display the morph trajectory, and any specified **Action on Create** will be performed:

- **none**
- **show Model Panel** - open the [Model Panel](#) (allowing the input structures to be hidden or closed)
- **hide Conformations** - hide the input structures

The trajectory will be opened as a separate model. Coordinates for the [atoms in common](#) from the [segment](#) endpoints (input structures) and the [intermediates](#) comprise the frames of the trajectory. The [MD Movie](#) tool can be used to [record](#) the trajectory as a movie file. If the movie includes ribbons, consider re-evaluating secondary structure at each frame as described [below](#). After the trajectory has been viewed, it can be [saved as a PDB file](#).

**Hide** hides the **Morph Conformations** dialog, **Quit** exits from the tool, and **Help** opens this manual page in a browser window. If the **Morph Conformations** dialog has been hidden or becomes obscured by other windows, it can be resurrected with the **Raise** option for its [instance in the Tools menu](#). Multiple copies of the **Morph Conformations** dialog can coexist, possibly with different settings and/or listed conformations.

## Atoms in Common

[Intermediates](#) are generated by interpolating between starting and ending structures. Interpolation requires a pairing of atoms in the starting structure with atoms in the ending structure. Only atoms common to both [segment](#) endpoints are included in the morph trajectory.

The structures must contain equal numbers of biopolymer chains (see [limitations](#)). If the sets of chain IDs are identical (for example, each structure contains chains A and B), the IDs will be used to indicate pairing; if they differ (for example, one structure contains chains A and B, whereas another contains chains A and D), the chains will be paired by their order of occurrence in the input files. Residues are paired by aligning the chain sequences. The sequence alignment is performed using the [matchmaker](#) defaults (Needleman-Wunsch algorithm, BLOSUM-62 matrix, secondary structure reassignment with [ksdssp](#), 30% secondary structure scoring, *etc.*), except that the Nucleic matrix is used for nucleic acids. Only the sequence alignment stage of [matchmaker](#) is performed, not the superposition of structures. **Morph Conformations** does not change how the input structures are superimposed.

HET residues such as ligands and ions are only included if they are present in both structures and attached to the same atom(s) in the paired chains by at least one “covalent” bond (which can be an unrealistic bond added manually, *e.g.*, with the command [bond](#), and subsequently undisplayed) or ion coordination [pseudobond](#).

Once residues are paired, atoms in common within those residues are paired. In paired residues of the same type, atom pairing is straightforward. In paired residues of different types, only atoms with the same names are paired, and only a single connected fragment is kept per residue. For example (disregarding hydrogens), phenylalanine and tyrosine have in common all atoms of phenylalanine.

## Example Systems for Morphing

A few example systems are listed here, including chain IDs. Currently it is necessary to delete other peptide chains before morphing between the structures.

Different conformations of the same or nearly the same protein:

- alpha-transducin GTP-binding switch (see the [figure](#) and the [Superpositions and Alignments tutorial](#)):
  - 1tagA - complex with GDP
  - 1tndA - complex with GTP analog
- thioredoxin reductase ball-and-socket movement (see the [Ball-and-socket motion](#) movie and script in the Chimera Animation Gallery):
  - 1tdeA - complex with FAD
  - 1f6mA - complex with FAD and NADP+ analog
- glucose/galactose binding protein:
  - 2fw0A - uncomplexed
  - 2gbpA - complex with glucose

Different but homologous proteins:

- phosphomannomutases, pairwise sequence identity ~50-70%:
  - 2i55C - *Leishmania* PMM complex with glucose bisphosphate
  - 2fucA - human PMM1
  - 2amyA - human PMM2
- pectate lyases from different organisms, sequence identity ~35%:
  - 1jtaA
  - 1bn8A

As structures become less similar in sequence, a morph trajectory between them is more likely to contain backbone “breaks,” corresponding to gaps in the sequence alignment that defines the [atoms in common](#).

For more complicated examples (simultaneously morphing molecules and density maps, morphing symmetric copies of a molecule), see the [ParM filament tutorial](#) at the Chimera web site.

## Limitations

**MD Movie does not automatically re-evaluate secondary structure.** [MD Movie](#) does not automatically recompute secondary structure assignments as coordinates change across a trajectory. This is relevant when [ribbons](#) are displayed and the conformational changes are large enough to alter secondary structure assignments. To recompute secondary structure at each frame, use a [per-frame script](#) in [MD Movie](#) that includes the Chimera command [ksdssp](#).

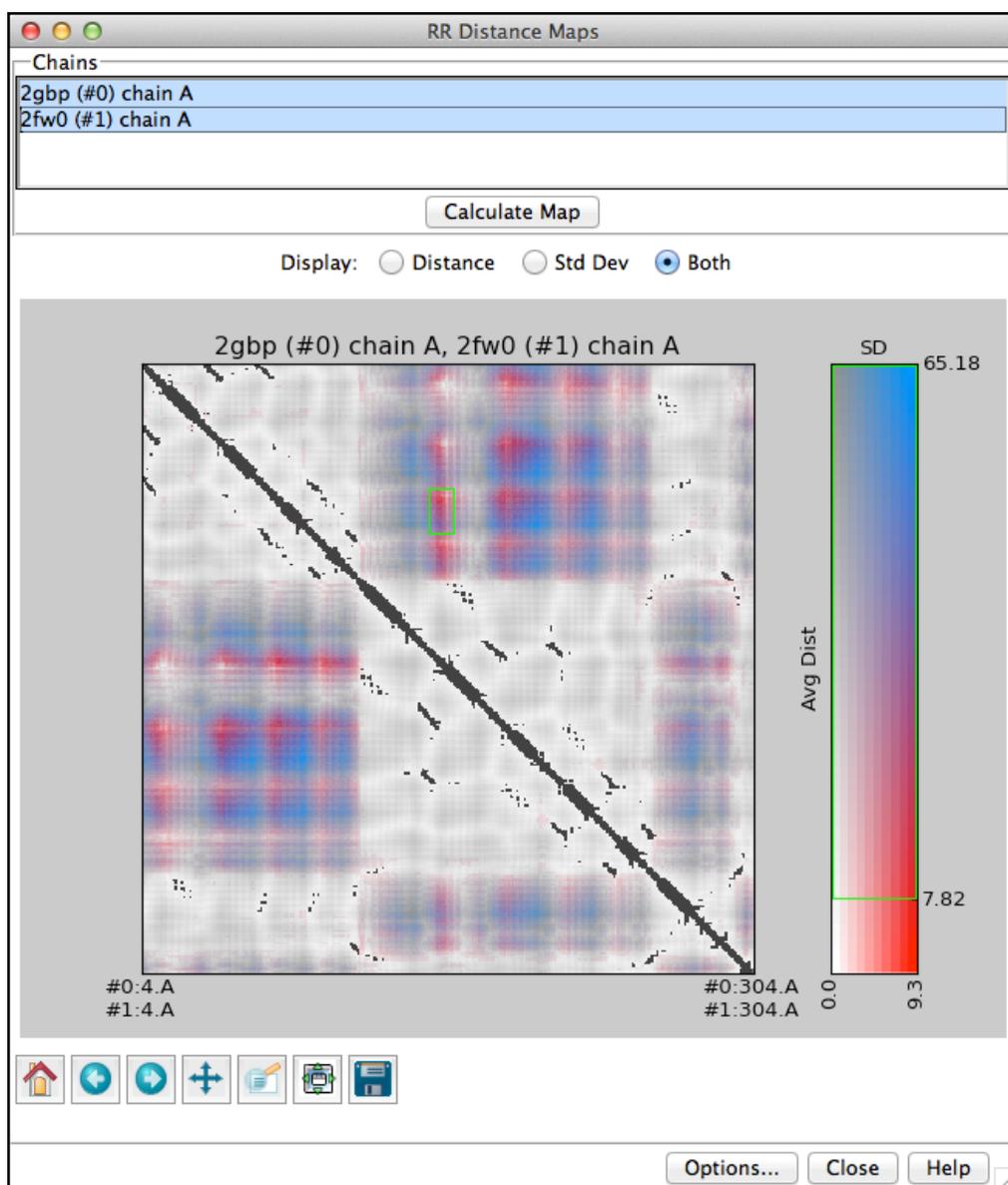
**Structures with different numbers of chains are not handled.** Currently, a morph trajectory can only be generated from input structures with equal numbers of biopolymer chains. Extra chains in the input models should be [deleted](#) beforehand or [split](#) into separate models not used in morphing.

**Sequences should be easy to align.** The sequences of the structures must be aligned to determine the [atoms in common](#) for interpolation. When the sequences are dissimilar, parts of the sequence alignment may be wrong, leading to a jumbled and unattractive morph trajectory. A possible future improvement is to allow users to specify residue pairings with an input sequence alignment.

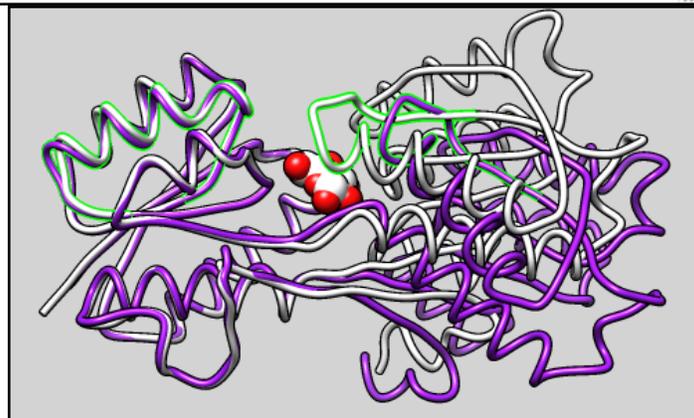
**Minimization limitations.** The [Minimize Structure](#) tool has its own set of [limitations](#).



## RR Distance Maps



**RR Distance Maps** creates a *distance map*, a generalization of a [protein contact map](#) in which residue-residue distances are shown with color gradations. In a protein contact map, a pair of residues is simply marked as contacting or not contacting based on some criterion such as a cutoff distance. **RR Distance Maps** generates a color-coded map of the C $\alpha$ -C $\alpha$  distances within an individual protein chain or a combined map for two or more related chains. In a combined map, the average (intrachain) distances and/or their standard deviations can be shown. Both quantities can be shown on the same map with different dimensions of color. See also: [Ramachandran Plot](#), [Find Clashes/Contacts](#), [Morph](#)



## [Conformations](#)

There are [several ways to start RR Distance Maps](#), a tool in the **Structure Comparison** category.

Individual chains or blocks of chains can be chosen from the **Chains** list with the left mouse button. Ctrl-click toggles the status of an individual chain. Only protein chains containing  $\alpha$ -carbons will be handled. If a single chain is chosen, an individual distance map will be generated. If multiple chains are chosen, they must have sequences similar enough to be aligned. The sequences will be aligned to obtain residue equivalences, and mean distances and standard deviations will be calculated for equivalenced residue pairs.

Clicking **Calculate Map** aligns the sequences (if multiple chains) and creates the map. The sequence alignment is shown in a [Multalign Viewer](#) window.

Choices for map **Display**:

- **Distance** - C $\alpha$ -C $\alpha$  distances within a single chain or average C $\alpha$ -C $\alpha$  distances among equivalenced residue pairs in multiple chains
- **Std Dev** (multiple chains only) - standard deviations of distances among equivalenced residue pairs
- **Both** (multiple chains only) - both the average distances and standard deviations among equivalenced residue pairs

The min/max colors can be changed in the [options](#), and values outside of the range of interest can be [masked](#) by adjusting the green outline within the color key.

**Close** exits from **RR Distance Maps**, and **Help** opens this manual page in a browser window.

## Mouse-Map Interactions

When the distance map window has mouse focus:

- Placing the cursor over the map reports the corresponding value(s) and structure residue IDs in the status area near the bottom of the window.
- Dragging to select a rectangle within the map [selects](#) the corresponding structure residues in the main Chimera window.
- The green outline within the color key to the right of the map defines the range of colormapped values. The color key always contains the full range of values, but the edges of the green rectangle can be dragged smaller and the whole rectangle can be dragged to reposition it within the key. Values outside the range of the green rectangle are *masked* to a single color in the map (the [excluded color](#), default almost-black). The green rectangle cannot be resized continuously, but the color key can be subdivided more finely using the [options](#).

If the map window becomes obscured by other windows, it can be resurrected with the **Raise** option for the **RR Distance Maps** [instance in the Tools menu](#).

Several standard [navigation icons](#) below the plot are provided by [matplotlib](#). If a mouse mode such as zooming is activated by clicking one of these icons, it is necessary to click the icon again to turn the mode off.

## Colormap Options

Clicking **Options...** opens the **Colormap Options**:

- **Vertical resolution** (default **10**) - number of vertical subdivisions in the color key
- **Horizontal resolution** (default **10**) - number of horizontal subdivisions in the color key, applicable when both distances and standard deviations are shown
- **Low D, low SD color** (a [color well](#), default ) - color for minimum distance (and minimum standard deviation, if both are shown)
- **High D, low SD color** (a [color well](#), default ) - color for maximum distance (and minimum standard deviation, if both are shown)
- **Low D, high SD color** (a [color well](#), default ) - color for maximum standard deviation (and minimum distance, if both are shown)
- **High D, high SD color** (a [color well](#), default ) - color for maximum standard deviation and maximum distance when both are shown
- **Excluded color** (a [color well](#), default ) - color for [masked](#) values (those beyond the range of the green outline rectangle in the color key)

Clicking **Apply** updates the map display; it also removes any [masking](#), but subsequently the green rectangle can be resized smaller again to resume masking. **Reset** restores the default settings in the dialog, and **Cancel** simply dismisses the dialog.

Note: a simple binary contact-map-like appearance can be obtained by masking low (contact) distances after setting the excluded color to black and all other colors to white, or using some other similar dark-light scheme.

## Technical Notes

**Sequence alignments.** Pairwise alignments are calculated with the Needleman-Wunsch algorithm. Multiple alignments are calculated with a MUSCLE web service, with settings as described for [Align Chain Sequences](#).

**Combined (multiple-chain) maps only include residues from fully populated alignment columns.** Only sequence alignment columns containing residues from all chosen chains are used to define the residue equivalences in a combined map.

**Coloring is applied across the full value range.** Although high and/or low values can be masked, colors can only be specified for the minimum and maximum values in the entire map. It might be useful to specify colors for the minimum and maximum values of a narrower range of interest to emphasize variations within that range.

**Combined (multiple-chain) maps omit terminal residues.** Related to the preceding point, the N-terminal two positions and C-terminal two positions are omitted from a combined map, because it was

found that large standard deviations arising from these positions made it difficult to see variations on a smaller, more generally useful scale.

---

UCSF Computer Graphics Laboratory / September 2014

## Ensemble Cluster

## Ensemble Cluster

**Ensemble Cluster** clusters members of a conformational ensemble (where each member contains the same atoms) and determines cluster representatives. It is a reimplementaion of the method described in:

[An automated approach for clustering an ensemble of NMR-derived protein structures into conformationally related subfamilies.](#) Kelley LA, Gardner SP, Sutcliffe MJ. *Protein Eng.* 1996 Nov;9(11):1063-5.

See also: [Ensemble Match](#), [MD Movie](#)

First, the ensemble of interest should be opened in Chimera from a single PDB file in which MODEL and ENDMDL records delimit the different structures (for example, [1plx](#)).

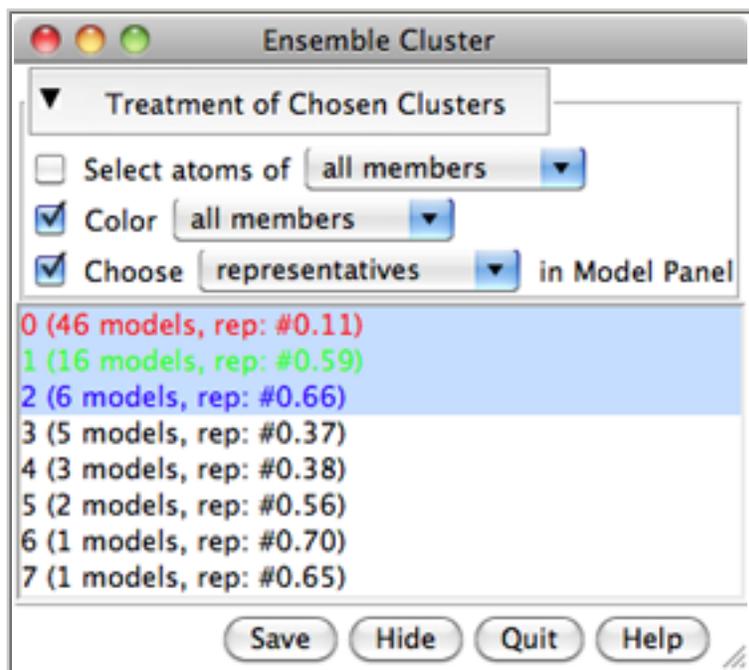
There are [several ways to start Ensemble Cluster](#), a tool in the **Structure Comparison** and **MD/Ensemble Analysis** categories. When it is started, a list of the open models will appear. The model containing the ensemble should be designated as the **Ensemble to Cluster**. Which atoms to consider (**Parts to Match**) can be indicated with a command-line-style [atom specification](#), or the field can be left blank to include all atoms.

**OK** dismisses the model list and starts the clustering calculation, while **Apply** starts the calculation without dismissing the list.

The calculation may take several minutes, depending on the size and number of structures in the ensemble. Pairwise best-fit RMSD values are computed (without moving the structures) and written to the [Reply Log](#). When the calculation finishes, a dialog will appear listing the clusters, their sizes, and the [model IDs](#) of their conformational representatives.

In the list, one or more clusters can be *chosen* with the left mouse button. Chosen lines are highlighted in the dialog. **Ctrl**-click toggles the status of a line, while clicking on the first (or last) line of a contiguous block and then **Shift**-clicking on the last (or first) chooses all of the lines in the block.

Clicking the black arrowhead reveals/hides options for the **Treatment of Chosen Clusters**. Each option applies to a *target* of either **all members** or just the **representatives** of the chosen clusters:



- **Select atoms of** [*target*] - [select](#) all atoms
- **Color** [*target*] - color the structures and list text to match, using a different color for each chosen

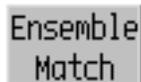
cluster

- **Choose [target] in Model Panel** - also choose the structures in the [Model Panel](#) for further operations. The **Model Panel functions** include **show only**, **tile**, **match**, **focus**, and **write PDB**; in most further dialogs, the same models will be chosen automatically.

These option settings are saved in the [preferences file](#) for subsequent uses of **Ensemble Cluster**.

If the cluster list is hidden (by clicking the **Hide** button) or becomes obscured by other windows, it can be resurrected using the **Raise** option for its [instance in the Tools menu](#).

**Save** allows writing information (cluster membership and representatives) for the chosen clusters to a text file. **Quit** closes the table and exits from **Ensemble Cluster**. **Help** opens this manual page in a browser window.



## Ensemble Match

An *ensemble* contains multiple sets of coordinates for exactly the same atoms, for example, snapshots along a simulation trajectory or different solutions of NMR data.

**Ensemble Match** calculates superpositions and RMSD values for every pairwise comparison between two ensembles. Each ensemble must contain at least one structure (ensemble member), and the members of the two ensembles must contain the same set of atoms. **Ensemble Match** can also be used to compare an ensemble to itself. However, if the ensemble has many members, [MD Movie](#) may be more appropriate for obtaining all-by-all pairwise RMSD values (see [RMSD analysis](#)). See also: [Ensemble Cluster](#), [Tile Structures](#), [superimposing structures](#)

First, each ensemble should be opened in Chimera from a single PDB file (with MODEL and ENDMDL records delimiting individual members, if more than one). There are [several ways to start Ensemble Match](#), a tool in the **Structure Comparison** and **MD/Ensemble Analysis** categories.

One ensemble should be designated as the **Reference** and the other as the **Alternative**. In the resulting table of comparisons, the reference structures will be listed vertically (the alternatives horizontally) and any matching operations will move the alternative structure onto the reference. **Parts to Match** are indicated with a command-line [atom specification](#); if this field is left blank, all atoms will be used. The [atom specification](#) describes the atoms to be used in *each* structure. If atom names are given, they should specify equal numbers of atoms occurring in the same order in the different structures (if @ca is entered, the first CA in a structure is matched with the first CA in other structures, the second CA with the second in other structures, *etc.*).

**OK** performs the calculation and dismisses the dialog, while **Apply** performs the calculation without dismissing the dialog. **Close** dismisses the dialog without performing any calculation. **Help** brings up this manual page in a browser window.

The resulting RMSD values are shown in a table. The figure at right shows comparisons between a reference ensemble with three members (models 0.8, 0.9, and 0.11) and an alternative ensemble with two members (models 1.1 and 1.9). In the table:

- Checking the option button next to an RMSD value performs the corresponding pairwise superposition. The number of atom pairs used to calculate the RMSD is reported in the [status line](#) (the atoms to use were specified previously as the **Parts to Match**).

	#1.1	#1.9
#0.9	3.828	2.131
#0.10	2.576	2.128
#0.11	2.396	3.065

The dialog box also includes a 'Save...' button, a 'Hide' button, a 'Quit' button, and a 'Help' button.

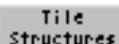
More than one alternative structure may be matched to the same reference structure, but a particular alternative structure cannot be matched to more than one reference structure at a time. It may be helpful to first use [Tile Structures](#) to spread out the members of the reference ensemble

and facilitate inspection of the matches.

- Toggling the **A** checkbox [activates/deactivates](#) a model for motion. A deactivated model will not move in response to [mouse manipulations](#) in the main window or movement commands such as [move](#) and [turn](#).
- Toggling the **D** checkbox hides/shows a model (enables or disables display at the model level, see [display hierarchy](#)).
- Clicking an "x" button (in column header) both deactivates and hides the corresponding alternative model (unchecks **A** and **D**).
- Clicking a "globe" button (to the left of a row) activates the corresponding reference model and all alternative models currently matched to it while deactivating all other models. The "stick man" button does all the things the "globe" button does, *plus* zooms in on the corresponding reference model.

The RMSD values can be exported collectively to a text file using the **Save** button. If the RMSD table is hidden (by clicking the **Hide** button) or becomes obscured by other windows, it can be resurrected using the **Raise** option for its [instance in the Tools menu](#). **Quit** closes the table and exits from **Ensemble Match**.

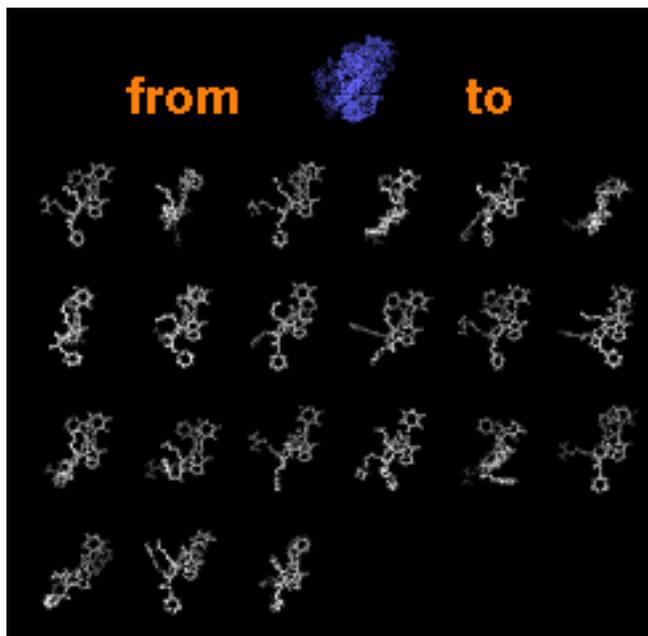
## Tile Structures



**Tile Structures** "tiles" models: it separates and spaces them evenly in the viewing plane. This is one way to show a set of related structures, such as snapshots from a simulation. See also: [Ensemble Match](#), [Ensemble Cluster](#)

There are [several ways to start Tile Structures](#), a tool in the **Structure Comparison** and **MD/Ensemble Analysis** categories. It is also implemented as the command [tile](#), which has more options than the graphical interface.

Starting **Tile Structures** opens a dialog for specifying which models should be tiled. Individual models or blocks of models can be chosen from the **Models** list with the left mouse button. Chosen lines are highlighted in the dialog. **Ctrl**-click toggles the status of a line, while clicking on the first (or last) line of a contiguous block and then **Shift**-clicking on the last (or first) chooses all of the lines in the block.



**Border scale** indicates how far apart the structures should be spaced, with larger values giving greater separations. The default of **1.0** spaces the models so that their bounding spheres abut, while **0.0** superimposes the bounding sphere centers.

**OK** tiles the structures and dismisses the dialog, while **Apply** tiles the structures without dismissing the dialog. **Close** dismisses the dialog without performing any tiling. **Help** brings up this manual page in a browser window.

The command [set independent](#) can be used to make models rotate about individual centers rather than a collective center.

Default or previously saved positions can be restored with the command [reset](#), while previous model transformations (but not scale or camera center) can be restored with [Undo Move](#).

## Minrms Plot

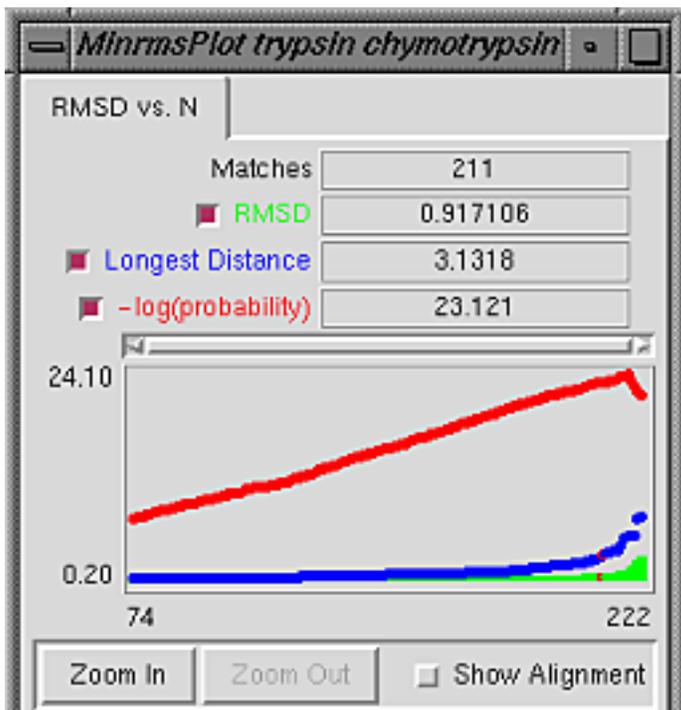
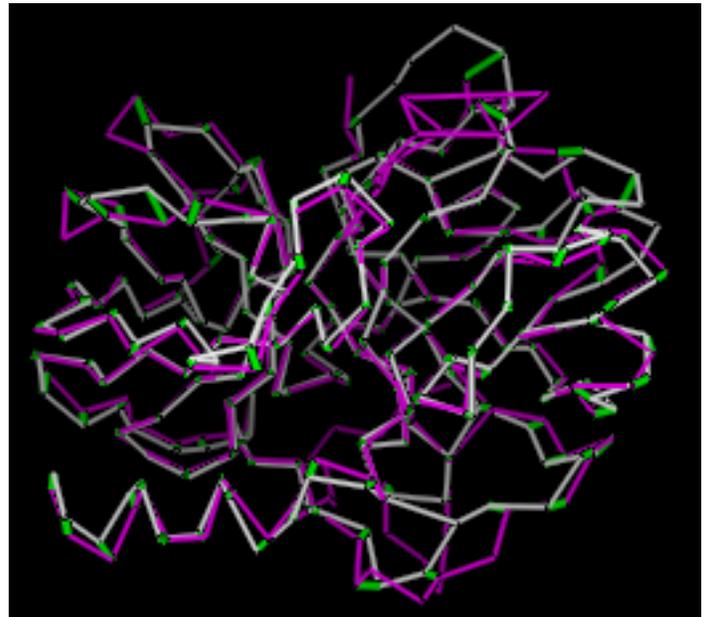
### Minrms Plot

**Minrms Plot** displays results from the separate program [MinRMS](#), which generates a series of structural alignments (superpositions) of two protein structures. **MinRMS** is a stand-alone, nongraphical program that is run from the system command line, not from within Chimera. It is not included with Chimera, but is [available](#) for download in source code form.

Briefly, **MinRMS** creates many superpositions of the same two structures differing in the number of residues paired; for a given number of pairings, the superposition is that which minimizes the RMSD. The  $n$ th entry of the "lowest-RMSD list" consists of a structural alignment and the  $n$  residue pairings within it used to calculate the RMSD. Each list item can be expressed as a sequence alignment in MSF format (file `alignn.msf`); the corresponding rotation/translation information is included in the comments. The  $n$  residue pairings are aligned in sequence within the MSF file.

**Minrms Plot** graphs the numerical results, shows the structural alignments, and calls [Multalign Viewer](#) to show the corresponding sequence alignments.

There are [several ways to start Minrms Plot](#), a tool in the **Structure Comparison** category. Starting **Minrms Plot** brings up a dialog box requesting the file `align_chimera.info` previously created by **MinRMS**. This file lists `pdb1` and `pdb2` (the names of the two PDB files), the name of the plot data file (currently just `align_chimera.plot`), and the number of alignment files to be read.



Upon reading the file, Chimera will display the molecules in `pdb1` and `pdb2` in the graphics window and open an additional **Minrms Plot** window containing a graph. Initially, the molecules are not superimposed (and may not both be visible); their positions depend on the coordinates in the input PDB files. Showing just the backbone or chain trace of proteins is recommended to simplify the display.

The **RMSD vs. N graph** depicts the members of the "lowest-RMSD list" of structural alignments that have been written out as MSF files. RMSD is shown in green, the longest distance between paired residues in blue, and an adaptation of the [Levitt-Gerstein probability score](#) in red versus increasing number of residue pairings, or "matches." The  $-\log(\text{probability})$  is shown, where probability represents the likelihood that such a favorable superposition could be attained by chance (proteins not



structurally related); larger values of  $-\log(\text{probability})$  indicate greater significance. **Clicking with the mouse within this graph determines which structural alignment is displayed in the graphics window.** In the figure, the graph has been clicked to choose the

alignment with 211 residue pairs (matches). **Zoom In** expands the graph to show a smaller range of N in the same area; **Zoom Out** reverses this.

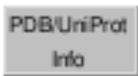
**Show Alignment** indicates that the sequence alignment corresponding to the current structural alignment should be shown in [Multalign Viewer](#). When the structural alignment is changed by clicking on the graph, [Multalign Viewer](#) will automatically switch to the corresponding sequence alignment.

**Hide** dismisses the interface without exiting from **Minrms Plot**. When the **Minrms Plot** interface is hidden or obscured by other windows, it can be resurrected using the **Raise** option for the **Minrms Plot** [instance in the Tools menu](#). **Close** closes the **Minrms Plot** window, and **Help** opens this manual page in a browser window.

The structure in *pdb1* is static (unless moved interactively by the user) and the structure in *pdb2* is reoriented relative to the first. Residue pairings are indicated with green [pseudobonds](#) in the graphics window.

## REFERENCES

1. [MINRMS: an efficient algorithm for determining protein structure similarity using root-mean-squared-distance](#). Jewett AI, Huang CC, Ferrin TE. *Bioinformatics*. 2003 Mar 22;19(5):625-34.
2. [A unified statistical framework for sequence comparison and structure comparison](#). Levitt M, Gerstein M. *Proc Natl Acad Sci U S A*. 1998 May 26;95(11):5913-20.



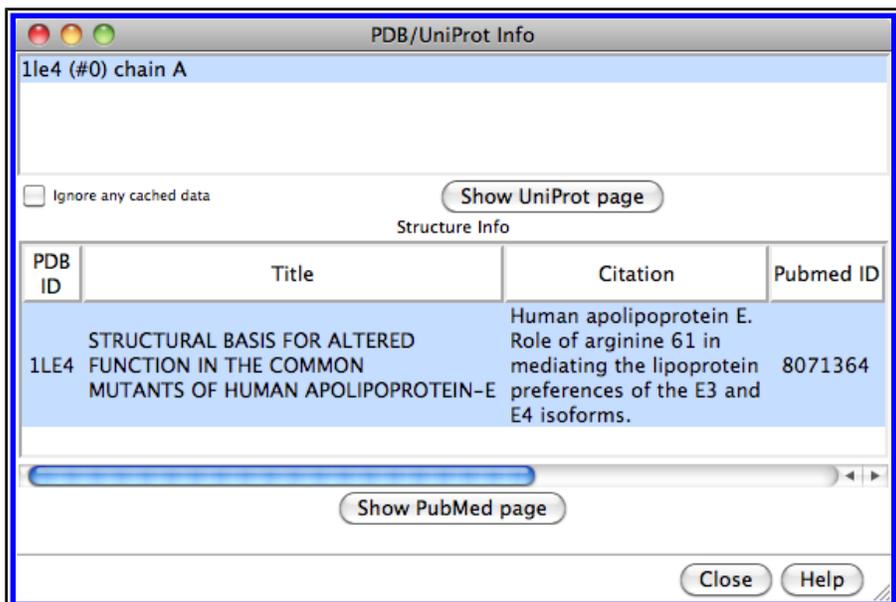
## PDB/UniProt Info

**PDB/UniProt Info** retrieves annotations for Protein Data Bank (PDB) entries using a web service provided by the [RCSB PDB](#).

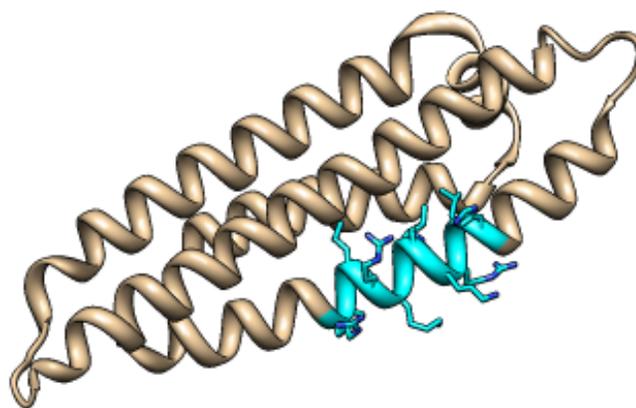
Sequences are displayed in [Multalign Viewer](#), and feature annotations from [UniProt](#) are mapped onto the sequences as [regions](#). See also: [fetching](#) UniProt data

There are [several ways to start PDB/UniProt Info](#), a tool in the **Sequence** category. It is also available as the [UniProt info...](#) function in the [Model Panel](#).

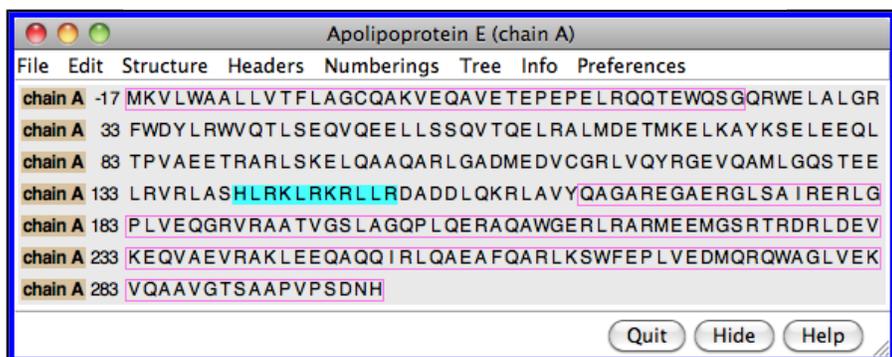
The top part of the initial dialog lists the PDB chains currently open in Chimera. One or more chains can be chosen with the left mouse button. Choosing a chain uses its PDB ID to retrieve structure annotations and any PDB-UniProt mapping information (UniProt ID and residue number correspondences) from the [RCSB PDB](#). The information is fetched as an XML file that can be cached and reused as needed depending on the [Fetch preferences](#) and whether the option to **Ignore any cached data** is checked. A dialog will appear for entering the PDB ID if it cannot be determined automatically. If the chain does not correspond to an entry in the PDB, no information can be retrieved. Sequence annotations are fetched from [UniProt](#); these are not cached.



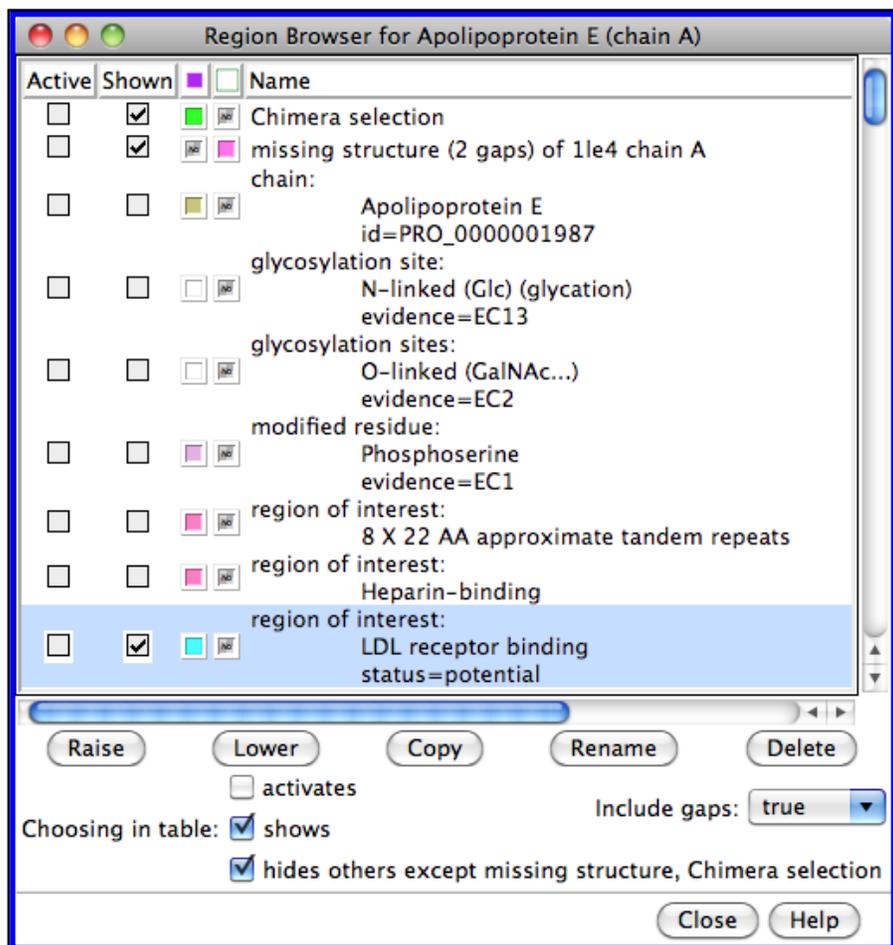
- **Structure annotations.** For valid PDB entries, the bottom part of the initial dialog will also show entry title, primary literature citation, structure determination method, source organism, and release date annotations from the RCSB PDB. One or more entries can be chosen in the bottom part of the dialog. Clicking **Show PubMed page** will display (in a browser window) the primary citation abstract at [PubMed](#) for each chosen entry.



- **Sequence annotations.** Additional windows will appear for showing sequence feature annotations from [UniProt](#), if available. Not all PDB chains are annotated with a UniProt ID, and a given PDB entry may contain both chains with and without UniProt IDs. For each chain with a UniProt ID:



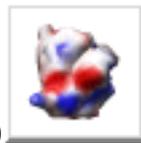
- the amino acid sequence from UniProt will be displayed in the [Sequence tool](#) (also known as [Multalign Viewer](#)), numbered according to the structure
- the UniProt sequence features will be listed as [regions](#) in the **Multalign Viewer Region Browser**; this dialog controls the display of regions in the [sequence window](#). Full details are in the [Region Browser](#) documentation, but briefly:



- any parts of the sequence not present in the structure will be shown as a pinkish outline region named **missing structure...**

- making a region **Active** [selects](#) the corresponding structure residues for subsequent operations with the [Actions menu](#) (in the example, the **LDL receptor binding** region was selected, its atoms/bonds were shown, it was colored cyan and then by heteroatom, and then the selection was cleared)
- making a region **Shown** displays it in the [sequence window](#)
- region colors can be changed by clicking the square [color wells](#) and using the [Color Editor](#); one column of wells is for interior colors, the other for outline colors
- clicking **Show UniProt page(s)** will display the corresponding web page(s) at UniProt ([example](#))

Other aspects of [sequence display](#) can be controlled in [Multalign Viewer](#).



## Surface Color (Electrostatic Surface Coloring)

Surface Color colors [surface models](#) and their [caps](#) by:

- [electrostatic potential](#) (not calculated on the fly, but read from a pre-existing [grid](#); however, see [Coulombic Surface Coloring](#))
- [volume data](#) value or gradient norm
- distance from a point, axis, or plane

Surface Color information is included in saved [sessions](#).

See also: [APBS](#), [DelPhiController](#), [Render by Attribute](#), [Values at Atom Positions](#), [Rainbow](#), [Color Zone](#), [Color Key](#)

Thanks to Steve Ludtke (Baylor College of Medicine) for developing the original version of this tool, Isosurface Colorizer (distributed with the [EMAN package](#)). The Chimera implementation uses C++ code to accelerate the color calculations.

There are [several ways to start Surface Color](#), a tool in the **Volume Data** category (including from the [Volume Viewer Tools](#) menu). It is also implemented as the command [scolor](#).

The same tool is available as **Electrostatic Surface Coloring** in the **Surface/Binding Analysis** category, and simply [opening](#) an [electrostatic potential](#) file will start this tool automatically.

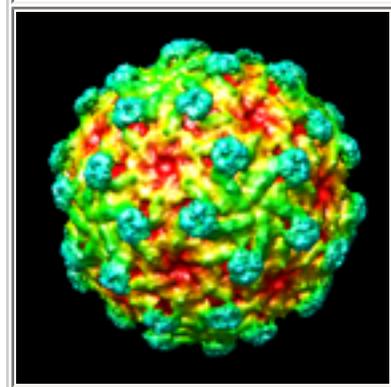
The surface of interest should first be displayed and chosen from the menu next to **Color surface**. How to display a surface depends on its type. For example, a [molecular surface](#) can be displayed with **Actions... Surface... show** or the command [surface](#), whereas a GRASP surface is displayed by [opening](#) a GRASP surface file. Any [surface cap](#) will be treated as part of the surface for coloring purposes. There is an [option](#) to color only the cap and not the rest of the surface.

Surface vertices are associated with values, either [distances](#) or values based on [separate data](#). Clicking the **Color** button applies the specified [value-to-color mapping](#). A surface will be recolored automatically if it changes shape. **Uncolor** reverts the [chosen surface](#) to its color in single-color mode. Caps will revert to the same color as the surface being capped, unless the [Surface Capping](#) option to [use a separate color](#) is turned on.

Choices for coloring by distance:

- **radius** - distance from a point. The point is defined by the **origin** coordinates.

radial coloring (by distance from a point)



- **cylinder radius** - distance from an axis. The axis is defined by any point on the axis and a direction. The point is defined by the **origin** coordinates. The three values in the **axis** field specify direction. For example, (1 0 0), (0 1 0), and (0 0 1) specify directions along the X, Y, and Z axes, respectively. Any three values can be entered, however (the axis need not be parallel to the X, Y, or Z axis).
- **height** - topographic height, distance from a plane. The plane is defined by any point on the plane and a vector normal to the plane. The point is defined by the **origin** coordinates. The vector is defined by the values in the **axis** field, as described [above](#).

The **origin** and **axis** are specified in the untransformed coordinate system of the [chosen surface](#), not the viewing coordinate system. The **origin** coordinates can be entered directly, or **center** can be clicked to set the origin to the center of the bounding box of the [chosen surface](#).

Choices for coloring by separate data:

- **electrostatic potential** - values from a [potential file](#)
- **volume data value** - values from [volume data](#)
- **volume gradient norm** - how steeply [volume data](#) values change in space

If the **potential file** or **volume file** has not been opened already, click **browse...** to locate and open the file.

If first started as **Surface Color**, the tool initially contains five colors in rainbow order; started as **Electrostatic Surface Coloring**, it contains red, white, and blue.

Clicking any [color well](#) allows the color to be adjusted interactively, and the value associated with the color can be edited. There are [options](#) to set the colors and values automatically and to control the number of wells. If a value is blank, the corresponding color will not be used, and at least two values must be specified. However, if all values are blank when **Color** is clicked, the fields will be filled in as if the [option](#) to set the full range of values had been used.

Clicking the **Options** button reveals additional settings that can be hidden again by clicking the small button  on the right.

- The number of **Colors** can be 2, 3, 5, 10, 15, or 20.
- Choosing a **Palette** populates the [color wells](#) with a set of colors:
  - **Rainbow** - standard rainbow colors, starting with red and ending with blue
  - **Red-Blue** - from red to white to blue
  - **Cyan-Maroon** - from a medium cyan to white to maroon
  - **Blue-Red** - from blue to white to red
  - **Gray** - grayscale, starting with black and ending with white

Clicking **Reverse** puts the colors in the opposite order.

- **Create color key** - open the [Color Key](#) dialog and populate it with the current **Surface Color** colors and values; a color key can then be created interactively with the mouse
- **Set full range of surface values** - set the color-associated values to range linearly from the

minimum to the maximum found in the surface

- **Surface offset** (for coloring by [electrostatic potential](#) only; default 1.4 Å) - how far out from each surface vertex, along its normal, to evaluate the electrostatic potential. The rationale for looking outward is that the values at the centers of any interacting atoms are more relevant than those at their surfaces. A [molecular surface](#) is *solvent-excluded*; it shows where the surface of a spherical probe (typically of radius 1.4 Å) can lie. Thus, 1.4 Å out from the molecular surface is about as close as the probe center can get, the *solvent-accessible* surface.
- **Color outside volume** (for coloring by [data](#) only; a [color well](#)) - color to use for vertices outside the bounds of the [data](#)
- **Only color sliced surface face** - color only the [surface cap](#), not the rest of the surface
- **Report value at mouse position** - whether mousing over the colored surface should give the corresponding values and coordinates in the [status line](#)
- **Per-pixel coloring** - determine color separately for each surface pixel instead of interpolating across surface triangles (see [technical notes](#)). This tends to give smoother color gradations. Per-pixel coloring only applies to coloring by [data values](#), not by gradients or [distances](#).

## TECHNICAL NOTES

**Per-pixel coloring.** In [per-pixel coloring](#), the [data](#) and the specified color/value pairs are used to set up a 3D texture (array of colors). OpenGL interpolates the texture to determine a color for each surface pixel. If it is not possible to create a 3D texture equal to the size of the volume, per-pixel coloring will not be performed. Some graphics cards/drivers limit 3D texture size to  $512^3$ , and often a smaller limit is imposed by the graphics card memory. When per-pixel coloring is not done, the coloring is per-vertex: data are interpolated and colors mapped as described below.

**Data value interpolation.** The transformed coordinates of the surface and the [data](#) are used to map data values to surface vertices.

- For coloring by data values directly, the value for each surface vertex is found by trilinear interpolation from the eight corners of the enclosing data grid cell.
- For coloring by gradient norm, the gradient at each volume grid point is computed by taking the difference in data values between the next and preceding grid points along each axis and dividing by twice the grid spacing. The gradient at a surface vertex is found by trilinear interpolation from the eight corners of the enclosing data grid cell. The norm of the interpolated gradient (the length of the gradient vector) is used for assigning colors.

**Color mapping.** The color mapping is defined by the specified color/value pairs, or thresholds. The value associated with each surface vertex is compared to the thresholds. Vertices with values lower than any threshold are assigned the color of the lowest-value threshold, while vertices with values higher than any threshold are assigned the color of the highest-value threshold. The colors of the remaining vertices are obtained by linear interpolation between the nearest lower and higher thresholds. Finally, each surface triangle is colored by linearly interpolating its vertex colors. Colors are defined by red, green, blue and opacity/transparency components.

## LIMITATIONS

**Subsequent coloring may erase molecular surface custom colors.** Unless explicitly limited to non-surface items, subsequent use of [Actions... Color](#) or the command [color](#) on the molecule model corresponding to a [molecular surface](#) will reset the surface's color source to **atoms** and wipe out the

**Surface Color** coloring. The custom surface coloring will be erased even when only parts of the molecule model that do not contribute to the molecular surface are recolored.

**Independent centers of rotation should not be used.** If the [center of rotation method](#) is set to **independent**, the relative positions of a molecular surface and an [electrostatic potential](#) map (or other [volume data](#)) will change as rotations are performed. Because the surface and map motions are not properly synchronized, the resulting coloring will be wrong. The other [center of rotation methods](#) do not cause this problem.



## Coulombic Surface Coloring

**Coulombic Surface Coloring** calculates electrostatic potential according to Coulomb's law:

$$\varphi = \sum [q_i / (\epsilon d_i)]$$

$\varphi$  is the potential (which varies in space),  $q$  are the atomic partial charges,  $d$  are the distances from the atoms, and  $\epsilon$  is the dielectric, representing screening by the medium or solvent. A distance-dependent dielectric ( $\epsilon = Cd$  where  $C$  is some constant) is sometimes used to approximate screening by implicit solvent.

**Coulombic Surface Coloring** colors [molecular surfaces](#) by the potential values and can handle structures with or without explicit [hydrogens](#). It can also [generate a grid](#) of the potential values.

Whereas **Coulombic Surface Coloring** can only color [molecular surfaces](#) based on the charges of the residues they enclose, the [coulombic](#) command has an option to use the charges of an arbitrary set of atoms. This allows coloring the surface of one molecule by the potential from another, for example, or coloring nonmolecular surfaces such as density isosurfaces.

The [Electrostatic Surface Coloring](#) tool is similar, but uses a previously computed [electrostatic potential grid](#). The grid could be from **Coulombic Surface Coloring** or from any of several separate (not included with Chimera) programs that solve the Poisson-Boltzmann equation. Chimera does include interfaces to such programs: [DelPhiController](#) requires a local (user-installed) copy of [DelPhi](#), and the [APBS](#) tool can use either a web service or a locally installed copy of [APBS](#) (Adaptive Poisson-Boltzmann Solver).

Poisson-Boltzmann calculations are more complex and, if done correctly, more accurate than simple Coulomb's law approaches. However, a Coulombic potential may suffice for visualization. (See <http://tinyurl.com/mzopva> for an informal comparison of images made in Chimera using **Coulombic Surface Coloring** with published figures of Poisson-Boltzmann electrostatic potential.)

See also: [Render by Attribute](#), [AddH](#), [Add Charge](#), [PDB2PQR](#), [Values at Atom Positions](#), [Color Key](#)

There are [several ways to start Coulombic Surface Coloring](#), a tool in the **Surface/Binding Analysis** category. It is also implemented as the command [coulombic](#).

The [molecular surface\(s\)](#) should first be displayed (using **Actions... Surface... show** or the command [surface](#)) and then chosen from the list of **Surfaces to color by ESP**. Only the residues enclosed by a surface will be used to calculate the potential on that surface. For example, nearby ions, solvent, or ligand molecules will not affect the results for a surface that encloses only protein. **Coulombic Surface Coloring** does not show the potential from one molecule on the surface of another, although that can be done in an additional step after [generating a grid](#).

## Parameters:

- **Number of colors/values** (default **3: red, white, and blue**) - up to 11 pairs can be used. Each color can be adjusted by clicking its [color well](#) and using the [Color Editor](#). The values can be edited directly, and are in units of kcal/(mol·e) at 298 K.

The specified color/value pairs or thresholds define a color mapping. The value calculated for each surface vertex will be compared to the thresholds. Surface vertices with values lower than any threshold will be assigned the color of the lowest-value threshold, while vertices with values higher than any threshold will be assigned the color of the highest-value threshold. The colors of the remaining vertices will be obtained by linear interpolation between the nearest lower and higher thresholds. Finally, each surface triangle will be colored by linearly interpolating its vertex colors. Colors are defined by red, green, blue, and opacity/transparency components.

- **Distance-dependent dielectric (true/false)** - whether  $\epsilon$  should vary in proportion to the distance from each charge
- **Dielectric constant** (default **4.0**) - value of  $C$ , where  $\epsilon = Cd$  if distance-dependent,  $\epsilon = C$  if not distance-dependent
- **Distance from surface** (default **1.4 Å**) - how far out from each surface vertex, along its normal, to evaluate the electrostatic potential. The rationale for looking outward is that the values at the centers of any interacting atoms are more relevant than those at their surfaces. A [molecular surface](#) is *solvent-excluded*; it shows where the surface of a spherical probe (typically of radius 1.4 Å) can lie. Thus, 1.4 Å out from the molecular surface is about as close as the probe center can get, the *solvent-accessible* surface.

The **Implicit Histidine Protonation** section controls the charge states of histidines in structures without hydrogens. Although hydrogens will not be added explicitly, the implicit protonation states will affect the results because the charges differ throughout the residue for different [protonation states](#), and the charges of the implicit hydrogens will be merged with those of the attached heavy atoms.

- **Residue name-based** - residue names will be used to determine which histidine sidechain nitrogens should be implicitly protonated: the  $\delta$ -nitrogen in residues named HID, the  $\epsilon$ -nitrogen in HIE, and both nitrogens in HIP. The **HIS** = setting specifies how residues with the standard histidine name (HIS) should be treated:
  - **estimated from H-bonds** (default) - choose protonation state based on local H-bonding environment
  - **delta** - neutral sidechain, implicit hydrogen at  $\delta$ -nitrogen
  - **epsilon** - neutral sidechain, implicit hydrogen at  $\epsilon$ -nitrogen
  - **both** - positive sidechain, implicitly protonated at both sidechain nitrogens
- **Specified individually...** the desired protonation state of each histidine residue will be specified with checkboxes in the dialog

Turning on **Compute grid...** reveals additional options for for generating a three-dimensional grid of the values (see [why this might be useful](#)):

- **Grid spacing** (default **1.0 Å**) - grid resolution; separation of grid points along X, Y, and Z
- **Padding** (default **5.0 Å**) - how far to extend the grid box in every direction from the minimum needed to enclose the atoms
- **Volume name** (default **Coulombic ESP**) - dataset name

Clicking **Apply** (or **OK**, which also dismisses the dialog) calculates the electrostatic potential and applies the coloring.

**Create corresponding color key** opens the [Color Key](#) dialog and populates it with the current colors and values; a color key can then be created interactively with the mouse.

**Close** simply dismisses the dialog, while **Help** opens this manual page in a browser window.

## GENERATING VOLUME DATA

The [Compute grid...](#) option generates a grid of the electrostatic potential values (a volume dataset) and starts the [Electrostatic Surface Coloring](#) and [Volume Viewer](#) tools. This allows:

- showing the Coulombic potential from one molecule on the surface of a different molecule, or on a nonmolecular [surface model](#)
- coloring planar [caps](#) where surfaces are [clipped](#)
- showing the potential as isosurfaces (however, this is generally less useful for Coulombic than for Poisson-Boltzmann ESP)
- saving the grid to a file for later use

The first two of these tasks can be accomplished using [Electrostatic Surface Coloring](#) (or [scolor](#)), the latter two with [Volume Viewer](#) (or [volume](#)).

## HYDROGENS AND CHARGES

The Coulombic potential calculation requires charge assignments, which in turn require hydrogens. An existing structure lacking hydrogens is not changed, but a copy is created in memory, [protonated](#), and [assigned charges \(details\)](#), which are then transferred to the existing structure. Selenomethionine (MSE) residues are treated as methionines (MET) for purposes of charge assignment. Where hydrogens are missing from the existing structure, their charges are collapsed onto the adjacent heavy atom: such hydrogens are *implicit*.

A structure may already have *explicit* hydrogens, or they can be added beforehand in Chimera with [AddH](#). A structure may also have pre-existing charge assignments, from prior use of [Add Charge](#), [PDB2PQR](#), or [Coulombic Surface Coloring](#), or [read from a file](#). If all of the atoms corresponding to the chosen surface already have charges, those values will be used rather than assigned anew the first time Coulombic coloring is applied to that surface. In subsequent applications, the existing charges will be used unless a setting in the [Implicit Histidine Protonation](#) section is changed, which forces the charges to be assigned anew. Another way to force reassignment is to remove the charges with the command [~setattr a charge](#).

While implicit hydrogens are appropriate for most uses, collapsing the hydrogen charges onto the adjacent heavy atoms tends to decrease positive potential magnitudes at the surface. This usually has little effect on the qualitative picture, but if it is of concern, the structure can be [protonated](#) beforehand. A disadvantage of explicit hydrogens, however, is that they make surfaces more rugged (bumpier) and complicated to view. To circumvent this problem, one could first add hydrogens, then [generate a grid](#) of the Coulombic potential values, then delete the hydrogens to make the molecular surface less rugged (or show the surface for a separate copy of the structure that does not have hydrogens), and finally, color

that surface with [Electrostatic Surface Coloring](#) using the Coulombic grid that had been generated in the presence of hydrogens.

## LIMITATIONS

**Subsequent coloring operations may erase surface custom colors.** Unless explicitly limited to non-surface items, subsequent use of [Actions... Color](#) or the command [color](#) on the molecule model corresponding to a [molecular surface](#) will reset the surface's color source to **atoms** and wipe out the Coulombic potential colors. The custom surface coloring will be erased even when only parts of the molecule model that do not contribute to the molecular surface are recolored.

**Subsequent recomputation of the molecular surface erases custom colors.** Anything that triggers surface recalculation, such as deleting atoms from the molecule model or changing certain [molecular surface parameters](#), will erase the Coulombic potential colors.

**Surface caps not colored.** This tool does not color [caps](#) on [clipped](#) surfaces, but it can be done in an additional step after [generating a grid](#).



## Surface Zone

**Surface Zone** restricts the display of [surface models](#) and their [caps](#) to zones around [selected](#) atoms or [markers](#). For [Volume Viewer](#) surfaces, the [Zone feature](#) in the volume dialog is more efficient; it limits contouring calculations to the region containing the zone. **Surface Zone** information is included in saved [sessions](#). See also: [Color Zone](#), [bondzone](#), [zonesel](#)

There are [several ways to start Surface Zone](#), a tool in the **Surface/Binding Analysis** category. It is also implemented as the command [sop zone](#).

The **Surface** of interest should be chosen from the list of available surface models. Surfaces from [Multiscale Models](#) will be named for the PDB file from which the surfaces were generated.

Clicking **Zone** limits the display of the chosen surface to areas within a specified distance (**Radius**) of any [selected](#) atoms. Surfaces are composed of triangles; only triangles with all three vertices in the zone will be shown.\* By default, a zone is based only on distances to any [selected](#) atoms (and [Volume Tracer](#) markers); the [bondzone](#) command indicates that points along any [selected](#) bonds (and [Volume Tracer](#) links) should also be used.

The **Radius** can be adjusted by moving the slider or by typing in a new value. There is no way to use different radii for different atoms. Although the maximum radius attainable by moving the slider is 30.0, larger values can be typed. If the set of selected atoms is changed, **Zone** must be clicked again to move the zone accordingly. Otherwise, the zone will remain centered on the formerly selected atoms. Clicking **Zone** will not update the surface display when no atoms are selected.

Clicking **No Zone** unlimits the display of the chosen surface model (shows the entire surface).

**Close** dismisses the dialog, and **Help** opens this manual page in a browser window.

\* To hide the visual clutter of disconnected patches of surface that fall within the zone, try the **maxComponents** option of [sop zone](#).


[Tools Index](#)

## ViewDock

[Background](#)
[Startup and Input](#)
[The ListBox](#)
[Compound Status](#)
[Hydrogen Bonds](#)
[Choosing by Value](#)
[Compound Display](#)
[ListBox Menu](#)
[Example Files](#)

## ViewDock

**ViewDock** facilitates the interactive analysis of receptor-ligand docking results. Users can click through a list of docked compounds (or different poses of a single compound) to view them individually in the context of a binding site. The dialog shows chemical names, docking scores, and other information read from the file. Docking results can be sorted, classified by interest level, and saved for future reference. **ViewDock** was originally developed to handle output from [DOCK](#), but has since been enhanced to support results from other programs.

For a step-by-step example, see the [ViewDock tutorial](#).

The state of **ViewDock** is included in saved [sessions](#). Scoring grids from DOCK (versions 4, 5, 6) can be visualized with [Volume Viewer](#). See also: [Dock Prep](#), [AutoDock Vina](#), [Write DMS](#), [FindHBond](#), [Sphgen sphere files](#)

## STARTUP AND INPUT

There are [several ways to start ViewDock](#), a tool in the **Surface/Binding Analysis** category. Starting the tool brings up a [dialog](#) for opening the file of docked molecules. Format options include:

- **Dock 4, 5 or 6** [.pdb, .mol2] (default) - from recent versions of [DOCK](#)
- **Dock 3.5.x search** [.pdb] - search mode results from the variant developed in the [Shoichet laboratory](#)
- **Dock 3.5.x single** [.pdb] - single mode results from the variant developed in the [Shoichet laboratory](#)
- **Dock 3 or 3.5** [.pdb] - from older versions of [DOCK](#)
- **AutoDock** [.pdbqt] - [PDBQT](#) format from [AutoDock](#) or [AutoDock Vina](#) (atomic partial charges and AutoDock4 atom types are assigned as the [attributes](#) charge and autodockType); see also the Chimera [AutoDock Vina](#) interface
- **Maestro/Glide** [.mae] - from [Maestro](#)
- **GOLD** [.mol2] - from [GOLD](#)
- **Mordor** [.ind] - from [MORDOR](#), MOlecular Recognition with a Driven

- dynamics OptimizeR
- all (guess type)
- all (ask type)

Another way to start **ViewDock** is with the [viewdock](#) command, which loads a specified file of docking results.

## THE LISTBOX

When the file of docked molecules has been read, the **ViewDock ListBox** will appear and the first molecule will be displayed in the main Chimera window. In most cases, a user will also display the receptor structure, which can be opened either before or after the file of docked molecules. **ViewDock** is controlled via the **ListBox** and its [menu](#).

The **upper panel** lists the compounds along with their [status](#)

settings (**S**) and the values of various *descriptors*.

Available descriptors can be shown or hidden using the

[Column](#)

[menu](#). Most

descriptors are simply read from the file of docked ligands; what descriptors are present and available

depends on the docking program and options used. In addition:

S	Number	Name
V	1	test_22
V	2	test_27
V	3	test_10
V	4	test_19
V	5	test_28
V	6	test_04
V	7	test_21
V	8	test_07
V	9	test_26

```

Chimera Model #2.1
##### Number      : 1
##### Source num  : 22
##### Name       : test_22
##### Description : ribose-monophosphate
##### Reflect    : 0
##### Energy score      : -42.64
#####  intermolecular van der Waals : -21.10
#####  intermolecular electrostatic : -21.54
##### RMSD from input orientation (A) : 40.86
Change Compound State
Viable Deleted Purged
Hide Quit Help

```

- the [HBonds... Add Count...](#) options can be used to identify [hydrogen bonds](#) and add their counts as new descriptors
- [Column... Diagram...](#) can be used to generate a column of 2D chemical diagrams with a web service provided by the [UCSF Resource for Biocomputing, Visualization, and Informatics \(RBVI\)](#)

The listing can be sorted by the values in any column (except diagrams) by clicking the column header. Clicking the header once sorts the entries in order of increasing value and places an up arrow in the header. Clicking again sorts the entries in decreasing order and places a down arrow in the header.

A compound can be *chosen* by clicking on its line in the upper panel. Multiple compounds may be chosen at once. Chosen lines are highlighted and only the chosen compounds are displayed. **Ctrl**-click adds to an existing choice rather than replacing it. To highlight a block of compounds without having to hold down the mouse button, click on the first (or last) and then **Shift**-click on the last (or first) in the desired block. Depending on the docking program and version used, the **lower panel** may show more detailed information when a single compound is chosen.

Compound [descriptors](#) read from the input file are assigned as [attributes](#) of **molecules** (molecule models). The attribute name is based on the descriptor name with **dock** prepended. Such attribute assignments allow docked compounds to be [rendered](#) (with colors or radii) or [selected](#) by their descriptor values.

**Hide** dismisses the **ListBox** without exiting from **ViewDock**. **Quit** exits from **ViewDock** and closes the docked molecules (like [File... Close](#) in the **ListBox** menu). **Help** brings up this manual page in a browser window. When the **ListBox** is hidden or obscured by other windows, it can be resurrected using the **Raise** option for the **ViewDock** [instance in the Tools menu](#). The instance's **Hide** and **Quit** options have the same effect as the corresponding **ListBox** buttons.

## COMPOUND STATUS

Three mutually exclusive states are possible for each compound: **Viable**, **Deleted**, or **Purged**. The initial state is viable, but if a molecule is deemed uninteresting, it can be changed to deleted or purged by clicking on the checkboxes near the bottom of the **ListBox**. Clicking a status checkbox changes the status of all [chosen](#) compounds.

In general, **Viable** compounds are interesting (or have not been looked at yet), **Deleted** compounds are less interesting but may deserve another look, and **Purged** compounds are meant to be discarded. The [Compounds menu](#) controls (based on status) which compounds are listed in the upper panel.

When structures are written out using [File... Rewrite](#), viable and deleted but not purged structures are included in the output file. In contrast, structures in all three states are included in a file written out using [File... Save](#) or [File... Save As](#). The states of the molecules are also recorded in these output files, which can be input to later sessions of **ViewDock**. No matter how the molecules are sorted in the **ListBox**, they remain in their original order in a saved PDB file.

There are other ways to change status besides clicking a status checkbox:

- if [Selection... Prune](#) is activated and part of a compound is [picked](#) in the graphics window, the compound is undisplayed and marked as deleted
- [HBonds... Change Compound State](#) allows compound status to be changed based on ligand-receptor [hydrogen bonds](#)

## HYDROGEN BONDS

The [HBonds menu](#) allows incorporation of hydrogen bonding information calculated with [FindHBond](#). If **FindHBond** has not already been run, its interface will appear. If **FindHBond** has been run, but one wishes to run it again with different parameters, it can be started from the main Chimera **Tools** menu or by using another of the [standard ways to start tools](#). When **FindHBond** is used to help screen compounds in **ViewDock**, the **inter-model** setting is most efficient.

The [HBonds... Add Count...](#) options add hydrogen-bonding statistics as [descriptor](#) columns:

- the number of hydrogen bonds between a compound and either the entire receptor [**HBonds (all)**] or the currently [selected](#) atoms of the receptor [**HBonds (sel)**]
- the number of ligand atoms participating in these hydrogen bonds
- the number of receptor atoms participating in these hydrogen bonds

The hydrogen bond descriptors can then be used to [choose compounds by value](#) or guide manual changes in [compound status](#).

[HBonds... Change Compound State](#) allows changes in [compound status](#) based on the number of hydrogen bonds to the receptor or to a [selected](#) portion of the the receptor. To use different hydrogen-bonding criteria, remember to change the compounds back to their previous states before re-running the hydrogen bond calculation. (If many status changes have been performed manually, it may be helpful to use [File... Save](#) or [File... Save As](#) *before* making further changes.)

## CHOOSING BY VALUE

Compounds can be [chosen](#) based on the values of any (or all) of the available [descriptors](#), including hydrogen-bond statistics calculated with [FindHBond](#) (see the [HBonds menu](#)).

[Compounds... Choose by Value](#) brings up a dialog for defining value ranges. Whichever descriptors are shown in the **ListBox** are available for choosing compounds. To add a descriptor to the **Choose by Value** dialog, simply show the descriptor in the **ListBox** using the [Column menu](#). Any descriptor shown in the **ListBox** but not of interest for choosing compounds can be disregarded by unchecking the corresponding checkbox.

Checkboxes under **Choose from** designate which compounds should be considered, on the basis of their [status](#). The values displayed in the histograms and/or lists only come from this designated set of compounds.

Numerical descriptors are depicted in a histogram. Within a histogram, two *markers* are shown as vertical bars. Clicking on a marker shows its **Value** in black. Clicking elsewhere within the histogram shows the **Value** (X-coordinate) of the mouseclick in gray. A marker can be moved by changing its **Value** and then pressing Enter (return) or by dragging it horizontally with the left mouse button. Holding the Shift key down reduces the speed (mouse sensitivity) of marker dragging tenfold, allowing finer control. Only compounds with values between the markers (inclusive) will be [chosen](#).

String-valued descriptors are displayed as a list. Clicking a line highlights just that line; **Ctrl**-click adds to an existing choice rather than replacing it. To highlight a block of lines without having to hold down the mouse button, click on the first (or last) and then **Shift**-click on the last (or first) in the desired block. Only compounds with highlighted values will be [chosen](#).

A message of the form *N of M compounds applicable* reports how many of the compounds (N) of those [designated](#) to choose from (M) meet all current criteria. These compounds will be [chosen](#) when **Apply** or **OK** (which also dismisses the dialog) is pressed.

## COMPOUND DISPLAY

Display can be controlled in many ways. Because of the potential for confusion, however, it is recommended that the **ViewDock ListBox**, not commands, be used to control the display of the docked compounds.

The remainder of this section explains the interplay among display controls, but can be skipped if only the **ViewDock ListBox** will be used to control compound display.

Individual compounds are submodels of a single model (model 0, for example) and are specified #0.1, #0.2, etc. (see [atom specification syntax](#)). Even though they are termed submodels, the display of each compound can be independently disabled or enabled at the model level (see [display hierarchy](#)). Within **ViewDock**, there are two ways to control model-level display:

- [choosing](#) one or more compounds within the **ListBox** upper panel enables their display and disables the display of any compounds previously [chosen](#) in the **ListBox**
- the **Hide** and **Show** options in the [Chimera menu](#) disable and enable the display as indicated, regardless of what is [chosen](#) in the **ListBox**

Compounds whose display has been disabled at the model level are unresponsive to the command [display](#). Ways of enabling display at the model

level (and restoring responsiveness to the command [display](#)) include choosing the compound in the **ListBox** upper panel, using one of the [Chimera... Show](#) options, and using the command [modeldisplay](#).

For example, if only the first five compounds are display-enabled,

```
Command: ~disp #0.2  
and
```

```
Command: disp #0.2  
but not
```

```
Command: disp #0.8
```

will work. In contrast, other commands like

```
Command: label #0.7  
and
```

```
Command: color pink #0.6
```

are executed, although their effects may not be visible until the corresponding molecules are displayed. The command

```
Command: disp #0
```

will display only the compounds whose display is enabled at the model level.

---

## LISTBOX MENU

### File

- **Save** - write all compounds and their states to a file with the same name as the input file (overwrites the input file)
- **Save As...** write all compounds and their states to a user-named file
- **Rewrite...** write all viable and deleted (but not purged) compounds and their states to a user-named file
- **Close** - close the **ListBox** and remove the docked molecules

### Compounds

The **Compounds** menu controls which compounds are listed in the top panel of the **ListBox**. Any combination of the first three entries may be checked to indicate which [status](#) types should be listed.

- **List Viable**
- **List Deleted**
- **List Purged**

- **List Chosen Only** - show only the [chosen](#) compounds of the [indicated](#) status types
- **List All** - show all compounds of the [indicated](#) status types (regardless of whether they are [chosen](#))
- **Invert Choices** - [choose](#) the current unchosen compounds and *vice versa*
- **Choose by Value...** open a dialog for [choosing compounds by descriptor value](#)

## Column

- **Show** - add a [descriptor](#) (many possibilities) to the listing in the top panel of the **ListBox**
- **Hide** - remove a [descriptor](#) from the listing in the top panel of the **ListBox**
- **Read...** read a text file specifying which [descriptor](#) columns should be shown after the **S** (status) column. The format is one descriptor name per line, entered exactly as shown in the **Show/Hide** menus, in the desired order. The file should not contain any blank lines. Any existing columns of descriptors will be replaced by the those specified in the file.
- **Diagram...** generate 2D chemical diagrams of the structures with a web service provided by the [UCSF Resource for Biocomputing, Visualization, and Informatics \(RBVI\)](#), or hide/show previously generated diagrams.

Options:

- **Diagram width** (default **96**) - pixel width of generated PNG image
- **Diagram height** (default **96**) - pixel height of generated PNG image

A compound's diagram will be blank until results are returned, and image generation may not work for all compounds.

## Selection

The following checkboxes are mutually exclusive and control how [picking](#) from the graphics window is handled by **ViewDock**.

- **Ignore** - ignore [picking](#) in the graphics window
- **Identify** - use the molecule containing the [picked](#) atom or bond as the current choice in the **ListBox**
- **Prune** - undisplay the molecule containing the [picked](#) atom or bond and change its status to deleted

## Chimera

The **Hide** and **Show** options toggle the display status of compounds in the main Chimera window.

- **Hide All [Viable, Deleted, Purged]**

- **Show All [Viable, Deleted, Purged]**

## HBonds

The **HBonds** menu allows incorporation of H-bond information calculated with [FindHBond](#).

- **Add Count to Entire Receptor** - add [descriptor](#) columns listing the number of ligand-receptor H-bonds and the numbers of ligand atoms and receptor atoms, respectively, involved in these H-bonds (a given atom may participate in more than one H-bond)
- **Add Count to Selected Atoms** - add [descriptor](#) columns listing the number of H-bonds between a ligand and the currently [selected](#) atoms of the receptor and the numbers of ligand atoms and receptor atoms, respectively, involved in these H-bonds (a given atom may participate in more than one H-bond). If the [selection](#) is later changed, this menu item must be chosen again to refresh the numbers.
- **Change Compound State...** mark as [viable/deleted/purged] compounds without:
  - hydrogen bonds to [all/any] [selected](#) receptor heteroatoms
  - [n] or more hydrogen bonds to receptor - ( $n=2$  by default)

## Movie

- **Play** - automatically go through the list of compounds, displaying one at a time; all compounds currently listed in the top panel of the **ListBox** will be shown, regardless of status, in the order in which they are listed
- **Stop** - stop the automatic sequential display of compounds (only available after **Play** has been chosen)
- **Options...** Play or Stop the movie, control the length of time each compound is shown

---

## EXAMPLE FILES

Note that DOCK 4 interprets information about each molecule as **Name** or **Description** based on where this information occurs in the input database file. Here are examples of a single molecule from [\(A\)](#) a Mol2 database file input to DOCK 4, [\(B\)](#) a Mol2 output file from DOCK 4, and [\(C\)](#) a PDB output file from DOCK 4. Multiple-molecule files are simply concatenations of the data for single molecules. **Number** is sequential number in order of occurrence in the file from DOCK, and **Source num** is the sequential number in order of occurrence in the database file input to DOCK.

---

UCSF Computer Graphics Laboratory / November 2012



## Dock Prep

**Dock Prep** performs several tasks to prepare structures for [DOCK](#) or for other calculations, such as:

- deleting water molecules
- repairing truncated sidechains
- adding hydrogens
- assigning partial charges
- writing files in Mol2 format

Many of these steps can be performed separately in Chimera, but **Dock Prep** unites them for convenience. Thanks to P. Therese Lang (DOCK development team, see [references](#)) for helping to develop this tool. See also: [PDB2PQR](#), [ViewDock](#), [AutoDock Vina](#), [Write DMS](#), [Sphgen sphere files](#)

If extra molecules such as ligands or additional subunits are present but unwanted during docking, these should be [deleted](#) before **Dock Prep** is used. **Dock Prep** does not delete such molecules (other than water and certain ions, optionally) because they could be important for binding or for maintaining receptor structure. Conversely, the biologically relevant form of the receptor may contain more subunits than are present in the structure file. If important for downstream calculations, the relevant form should be generated before **Dock Prep** is run. For obtaining multimers, see: [fetching](#) PDB-biounit and PQS files, the command [sym](#), [Multiscale Models](#), [Unit Cell](#)

There are [several ways to start Dock Prep](#), a tool in the **Structure Editing** category (including using it via [Minimize Structure](#)).

Under **Molecules to prep**, the structure(s) of interest should be chosen from the list of open molecule models. Individual models or blocks of models can be chosen with the left mouse button. Ctrl-click toggles the status of an individual model. To choose a block of models without dragging, click on the first (or last) and then Shift-click on the last (or first) in the desired block.

Several operations can be performed on the chosen structures:

- **Delete solvent** - delete any [solvent](#) molecules (usually waters). This is generally done to prepare a receptor structure for docking. If any solvent molecules are thought to be important for ligand binding, however, one should manually delete the *other* solvent residues beforehand and deactivate this option in **Dock Prep**.
- **Delete non-complexed ions** - delete any [ions](#) that are not participating in covalent or coordination bonds (by default, the latter are shown as dashed lines). This bonded-or-not distinction is based solely on input bond specifications such as CONECT and LINK records in PDB files; it is not inferred from the chemistry of the system.
- **If alternate locations, keep only highest occupancy** - for atoms with alternate locations, retain only the highest-occupancy set (if a tie, the set with a lower B-factor where the alternate locations

branch from the common structure)

**Change** - convert modified residues to standard residues for which parameters are available:

- **selenomethionine (MSE) to methionine (MET)** - change [MSE](#) residues to MET by changing the selenium atom to a sulfur atom named SD and adjusting the CG-SD and SD-CE bond lengths to 1.81 and 1.78 Å, respectively
- **bromo-UMP (5BU) to UMP (U)** - change 5-bromouridine-5'-monophosphate (PDB chemical component [5BU](#)) to RNA residue uridine-5'-monophosphate (U) by deleting the bromine atom
- **methylselenyl-dUMP (UMS) to UMP (U)** - change 2'-methylselenyl-2'-deoxyuridine-5'-phosphate (PDB chemical component [UMS](#)) to RNA residue uridine-5'-monophosphate (U) by replacing the methylselenyl moiety with an oxygen atom named O2' and adjusting the bond length to 1.430 Å
- **methylselenyl-dCMP (CSL) to CMP (C)** - change 2'-methylselenyl-2'-deoxycytidine-5'-phosphate (PDB chemical component [CSL](#)) to RNA residue cytidine-5'-monophosphate (C) by replacing the methylselenyl moiety with an oxygen atom named O2' and adjusting the bond length to 1.430 Å

- **Incomplete side chains:**

- **Replace using Dunbrack rotamer library**
- **Replace using Richardson (common-atom) rotamer library**
- **Replace using Richardson (mode) rotamer library**
- **Mutate residues to ALA (if CB present) or GLY**

The [rotamer library options](#) replace each truncated sidechain with a complete sidechain of the same residue type, as if using the command [swapaa](#) with the specified [rotamer library](#) and **preserve true** (default settings of other [options](#)). If different settings are desired, use [swapaa](#) separately before using **Dock Prep**. The [mutation option](#) changes each residue with a truncated sidechain to glycine or alanine, the latter if a CB atom is present.

Using one of these options to generate complete sidechains is recommended because in incomplete residues, the partial charges will not sum to integer values, and extra hydrogens (unrecognized in the charge addition step) will be added where the missing atoms would have been attached.

- **Add hydrogens** - call [AddH](#) for hydrogen addition. This tool aims to generate protonation states reasonable at physiological pH. For example, hydrogens are not added to the phosphodiester moieties of DNA and RNA. By default, aspartic acid and glutamic acid sidechains are assumed to be negatively charged, while arginine and lysine sidechains are assumed to be positively charged (although [other states can be attained](#)). Ambiguous groups are terminal phosphates (the third ionization) and imidazoles such as histidine sidechains. Histidine protonation states can be [user-specified](#) or guessed by the [method](#). Whether a terminal phosphate is triply ionized is guessed using bond lengths. [Additional rules](#) are used to identify and handle chain termini.

Potentially ambiguous or rare (shifted-pKa) protonation states, especially in binding sites and nonstandard residues, should be verified and corrected before charges are assigned. For example,

extra hydrogens can be [deleted](#), and [atom types](#) can be edited (before hydrogen addition) with [setattr](#) or [Build Structure](#).

- **Add charges** - call [Add Charge](#) to assign partial charges to atoms. Partial charges are assigned as an atom [attribute](#) named **charge** and will be included in [Mol2 output](#).

Charges for *standard* residues (water, standard amino acids, standard nucleic acids, and a few common variants and capping groups) are taken from [Amber \(details\)](#). If any atoms in standard residues are not recognized, a warning will appear and information on the atoms will be sent to the [Reply Log](#). Cases of unrecognized atoms in standard residues and/or incorrect net charges should be [examined and resolved](#).

Charges for *nonstandard* residues, if any, are calculated using Amber's [Antechamber](#) module (included with Chimera; publications involving its use should cite the [reference](#)). It is necessary to specify the formal charge of each nonstandard residue and which [charge calculation method](#) should be used.

- **Write Mol2 file** - open a dialog for [saving Mol2 files](#); options include saving coordinates relative to the untransformed coordinates of a particular model, generating an @SET section containing the [selected](#) atoms (as used to specify the rigid portion of a ligand in [DOCK](#)), and saving multiple models in a single file or multiple files

**OK** performs the specified tasks and dismisses the **Dock Prep** dialog, while **Cancel** simply dismisses the dialog. **Help** opens this manual page in a browser window.

Tasks are performed in the order listed on the **Dock Prep** dialog. If any individual step is canceled, subsequent steps will not be performed; for example, if charge assignment is canceled, a Mol2 file will not be written. To skip a particular step, uncheck its option before initiating the **Dock Prep** calculation.

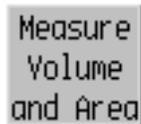
## LIMITATIONS

**Does not build missing segments.** Structures may have missing residues or atoms where coordinates could not be determined because of disorder or flexibility. **Dock Prep** can fix [truncated sidechains](#), but it will not build missing backbone segments.

## REFERENCES

The following describe aspects of **Dock Prep** and exemplify its use in docking applications:

- [DOCK 6: combining techniques to model RNA-small molecule complexes](#). Lang PT, Brozell SR, Mukherjee S, Pettersen EF, Meng EC, Thomas V, Rizzo RC, Case DA, James TL, Kuntz ID. *RNA*. 2009 Jun;15(6):1219-30.
- [Development and validation of a modular, extensible docking program: DOCK 5](#). Moustakas DT, Lang PT, Pegg S, Pettersen E, Kuntz ID, Brooijmans N, Rizzo RC. *J Comput Aided Mol Des*. 2006 Oct-Nov;20(10-11):601-19.



## Measure Volume and Area

**Measure Volume and Area** reports the total surface area and enclosed volume of a [surface model](#). To measure individual disconnected blobs within a single surface model, use [Measure and Color Blobs](#) instead. These tools compute surface area and enclosed volume from surface triangles rather than analytically. See also: [measure](#), [Area/Volume from Web](#), [CASTp Data](#)

Note that when a [molecular surface](#) is generated, the total analytical solvent-accessible and solvent-excluded areas are automatically reported in the [Reply Log](#), and the values per atom and residue are assigned as [attributes](#) named `areaSAS` and `areaSES`.

There are [several ways to start Measure Volume and Area](#), a tool in the **Surface/Binding Analysis** and **Volume Data** categories (including from the [Volume Viewer Tools](#) menu).

The surface of interest should be chosen from the **Surface** menu. The surface area and enclosed volume are reported in the dialog and written to the [Reply Log](#). Values reflect the physical units of the data, usually  $\text{\AA}^3$  for volume and  $\text{\AA}^2$  for area.

Surfaces are composed of triangles. Anything that changes the positions of the vertices in a surface will change its area and enclosed volume, for example:

- adjusting the contour level of an isosurface shown with [Volume Viewer](#)
- smoothing a [Multiscale Models](#) surface
- changing the **probe radius** or **vertex density** [attribute](#) of a [molecular surface](#)

**Update automatically** (on by default) indicates volume and area should be recomputed automatically whenever such a change occurs. Otherwise, a new calculation can be triggered by clicking **Update**. A current limitation is that changes in [molecular surfaces](#) are not detected automatically (the **Update** button must always be used).

### TECHNICAL NOTES

**Clipping and hiding are ignored.** The calculation uses the full surface, even if part of it is hidden by clipping planes or zoning (with [Surface Zone](#) or [zoning](#) in [Volume Viewer](#)).

**Surface holes.** The volume enclosed by a surface with holes is calculated by assuming each hole has a planar cap. These caps are not displayed or included in the surface area determination. The number of holes is reported, as planar caps may not represent missing data very well. [Multiscale Models](#) surfaces should never have holes. Holes in a contour surface from [Volume Viewer](#) may occur at the boundary of the data, but by default these will already be covered with [planar caps](#). Such boundary caps will be included in the surface area calculation and will not be reported as holes.



## Measure and Color Blobs

**Measure and Color Blobs** enables measuring individual disconnected parts, or *blobs*, within a [surface model](#). Quantities reported for a blob include:

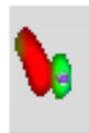
- the **area** of its triangulated surface
- the **volume** enclosed
- the **size** (dimensions from longest to shortest) of its [bounding box](#)

The surface area and enclosed volume are computed from surface triangles rather than analytically. To measure the *total* surface area and enclosed volume of a surface model (possibly containing multiple blobs), [Measure Volume and Area](#) or the command [measure](#) should be used instead. See also: [Segment Map](#), [Surfnet](#)

Note that when a [molecular surface](#) is generated, the analytical solvent-accessible and solvent-excluded areas (total and for each disconnected component) are automatically reported in the [Reply Log](#), and the values per atom and residue are assigned as [attributes](#) named **areaSAS** and **areaSES**. See also: [Area/Volume from Web](#), [CASTp Data](#)

There are [several ways to start Measure and Color Blobs](#), a tool in the **Surface/Binding Analysis** and **Volume Data** categories (including from the [Volume Viewer Tools](#) menu). The dialog includes the options:

- **Use mouse [button] to pick blobs** - which mouse button (1, 2, 3, Ctrl-1, Ctrl-2, or Ctrl-3) will be used to pick surface blobs from the screen. The icon representing this mouse mode in the [Mouse preferences](#) is shown at right.
- **Color blob [color well]** - color to use for the next picked blob
- **Change color automatically** - whether to set a new color (randomly) each time a blob is picked
- **Show principal axes box [color well]** - whether to show a bounding box in the specified color (default **green**) for each subsequently picked blob. A blob's bounding box is aligned with its principal axes of inertia, calculated assuming a constant mass per unit surface area. Box outlines are placed in a [surface model](#) named **Principal axes box**. Clicking **Erase** closes the box model. Even if the box is not shown, its size will be reported along with the other measurements described below.



Surfaces are composed of triangles. Anything that alters the positions of vertices in a surface, such as [smoothing](#) the surface or changing its [contour level](#) in [Volume Viewer](#), will change its area and enclosed volume.

When a blob is picked from the screen, the **area** of its triangulated surface, the **volume** enclosed, and its [bounding box size](#) (dimensions from longest to shortest) are reported in the **Measure and Color Blobs** dialog, the [status line](#), and the [Reply Log](#). Values reflect the physical units of the data, usually Å<sup>3</sup> for

volume, Å<sup>2</sup> for area, Å for length.

Blob surface area and volume are not affected by [clipping](#) (the clipped-away parts will still be included), but the planar [cap](#) can itself be measured. In that case, the volume will be near-zero, and the **boundary** (circumference) will also be reported.

Circumferences of any holes in the blob surface are also reported. The volume enclosed by a surface with holes is calculated by assuming each hole has a planar cap. These caps are not displayed or included in the surface area determination. The number of holes is reported, as planar caps may not represent missing data very well. [Multiscale Models](#) surfaces should never have holes. Holes in a contour surface from [Volume Viewer](#) may occur at the boundary of the data, but by default these will already be covered with [planar caps](#). Such boundary caps will be included in the surface area calculation and will not be reported as holes.

## LIMITATIONS

Colors applied with **Measure and Color Blobs** are lost when the surface contour level is changed.

After the tool is closed, it may be necessary to use the [Mouse preferences](#) to reassign mouse buttons to their previous functions.

## Intersurf Intersurf

Intersurf generates and displays [interface surfaces](#). See also: [FindHBond](#), [Find Clashes/Contacts](#), [Surfnets](#), [measure buriedarea](#)

The **Intersurf** tool in Chimera is an implementation of the method described in:

[Intersurf: dynamic interface between proteins](#). Ray N, Cavin X, Paul JC, Maigret B. *J Mol Graph Model*. 2005 Jan;23(4):347-54.

In this context, an *interface surface* is not the surface of either set of atoms, but a surface that divides the space between them. The division of space is based on Delaunay tetrahedralization.

There are [several ways to start Intersurf](#), a tool in the **Surface/Binding Analysis** category. **Intersurf** is also implemented as the command [intersurf](#).

The first step is to specify two sets of atoms, which can be different **Molecules** (molecule models) or different **Chains** within the same or different models. (To specify sets of atoms that are not whole models or whole chains, it is necessary to use the command [intersurf](#) instead, with [atom pairing](#).) Two entries should be chosen from the list of **Molecules** or **Chains** by clicking one and Ctrl-clicking the other, or by dragging if the two entries are adjacent.

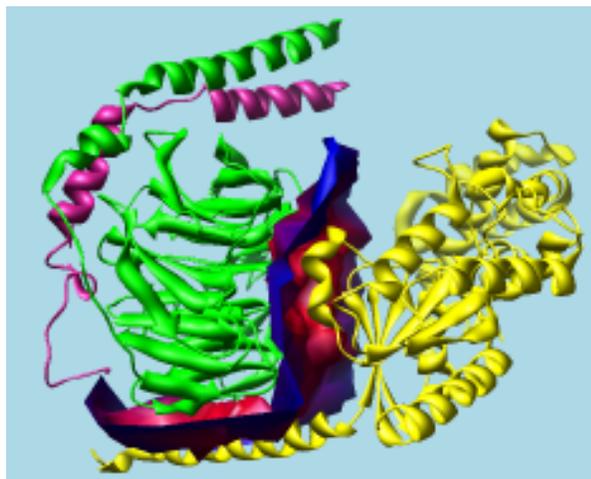
Options:

- **Solid surface (rather than mesh)** - whether to display the interface surface as solid or mesh
- **Reuse last surface (if any)** - whether to replace any pre-existing interface surface with the new one (otherwise, an additional [surface model](#) will be generated)
- **Select interface atoms** - whether to [select](#) the atoms that [define](#) the interface surface
- **Residue centroid distance pruning** - whether to ignore residues whose centroids are not within the **Prune distance** (default 30.0 Å) of any residue centroid in the other set of atoms

The **Molecule bias** determines how the space between the two sets of atoms should be divided. Each vertex in the interface surface lies between two atoms, one from each set. The **Molecule bias** ranges from 0 to 1, where a bias of 0.5 (default) places each interface surface point equidistant from the [VDW surfaces](#) of the two corresponding atoms. A bias of 0.0 places interface surface vertices at the VDW surface of the set higher in the list of **Molecules** or **Chains** (regardless of order chosen), and a bias of 1.0 places them at the VDW surface of the set lower in the list. Where the VDW surfaces of the two sets coincide, changes in bias will have little effect on interface surface vertex placement.

**OK** generates the interface surface and dismisses the dialog, whereas **Apply** generates the surface without dismissing the dialog. The surface is generated as a [surface model](#) and assigned the same model number and transformation as the corresponding molecule model (or, if atoms from more than one model were used, the lowest-numbered of those models). Models can be hidden or closed using the [Model Panel](#).

G protein heterotrimer (1gg2)  
alpha-beta interface surface



When an interface surface is generated, the histogram is populated with the corresponding atom-atom distances. The calculation can be repeated to adjust the [coloring](#) (preferably with the [Reuse last surface](#) option turned on).

**Close** dismisses the **Intersurf** dialog, and **Help** opens this manual page in a browser window.

## Coloring

The histogram with sliders (*thresholds*) allows the surface to be colored by the distance between atoms across the interface. Interface surfaces can also be colored according to volume data (with [Surface Color](#)) and to match atoms (with [Color Zone](#)).

Clicking on a threshold shows its distance **Value** in black. Clicking elsewhere within the histogram shows the **Value** (X-coordinate) of the mouseclick in gray. A threshold can be moved by changing its **Value** and then pressing Enter (return) or by dragging it horizontally with the left mouse button. Holding the Shift key down reduces the speed (mouse sensitivity) of threshold dragging tenfold, allowing finer control. Thresholds can be added by Ctrl-clicking with the left mouse button on the histogram. Ctrl-clicking on an existing threshold deletes it. Besides a **Value** (position on the histogram), each threshold has a **Color**. The **Color** applies to the threshold most recently moved or clicked and can be adjusted by clicking the adjacent [color well](#).

The thresholds define a function that maps distances to colors. For each surface point, the corresponding atom-atom distance is compared to the thresholds on the histogram. Color is defined by red, green, blue and opacity/transparency components. The color of the closest threshold at a lower value (to the left) and the color of the closest threshold at a higher value (to the right) are linearly interpolated. Points with distance values less than the leftmost threshold are colored according to the leftmost threshold, while points with distance values greater than the rightmost threshold are colored according to the rightmost threshold.

See also: [Color Zone](#), [Surface Color](#)

**Surfnnet** identifies molecular cavities and indentations and displays them as surfaces. The method is an adaptation of that described in:

[SURFNET: a program for visualizing molecular surfaces, cavities, and intermolecular interactions.](#) Laskowski RA. *J Mol Graph.* 1995 Oct;13(5):323-30, 307-8.

See also: [Measure and Color Blobs](#), [Intersurf](#), [CASTp Data](#)

There are [several ways to start Surfnnet](#), a tool in the **Surface/Binding Analysis** category.

**Surfnnet** finds cavities between two [specified](#) sets of atoms, **Atom Set 1** and **Atom Set 2**. The two sets of atoms can be identical, overlapping, or completely different, and in the same or different models. If **Atom Set 2** is left blank, it is assumed to be the same as **Atom Set 1**.

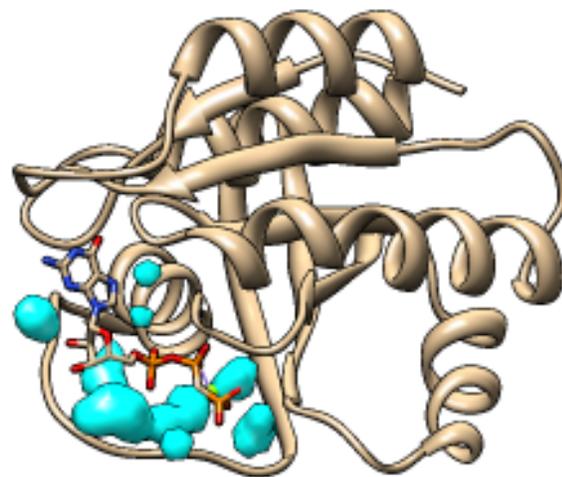
A brief description of the method is needed to explain the options:

1. Pairwise combinations of atoms, one from each set, are examined for intervening void space. Only pairs of atoms no farther apart than the **Distance Cutoff** (in Å) are examined. A "gap sphere" is placed directly between the two atoms and shrunk until it no longer intersects the VDW surface of *any* atom. [Atomic radii](#) are assigned by Chimera based on inferred atom types. Gap spheres with radii equal to or greater than the **Grid Interval** are retained.
2. The **Density** of each gap sphere is smeared out from its center according to a **Gaussian** or **Quadratic** function such that the value at its radius (100.0 in arbitrary units) is half the value at its peak (200.0).
3. At each point in a 3-dimensional grid with **Grid Interval** spacing (in Å), only the largest density contribution from a gap sphere is stored. This is not necessarily the contribution from the nearest gap sphere center, since the decrease in density with distance depends on radius.
4. The grid values are used to generate a contour surface at the level of 100.0, shown in the specified **Representation** (**Mesh** or solid **Surface**) and [Color](#) (default white).

**OK** executes a **Surfnnet** calculation and dismisses the dialog. **Apply** runs the calculation without closing the dialog. The resulting surface is generated as a [surface model](#) and assigned the same model number and transformation as the corresponding molecule model (or, if atoms from more than one model were used, the lowest-numbered of those models). Each run creates an additional surface model, which can be hidden or closed using the [Model Panel](#).

**Close** closes the dialog, and **Help** opens this manual page in a browser window.

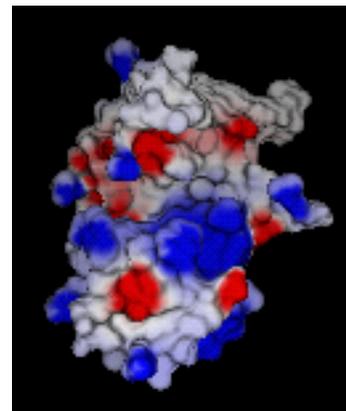
H-ras (121p)  
protein vs. ligand



## DelPhiController



**DelPhiController** is an interface to [DelPhi](#), which uses the finite-difference Poisson-Boltzmann equation to calculate electrostatic potentials in and around molecules. DelPhi is not included with Chimera; **DelPhiController** works with the academic version available from the [Honig Lab](#) or the [DelPhi development team](#). For more details, see the DelPhi documentation at either of those sites. See also: [APBS](#), [Coulombic Surface Coloring](#)



There are [several ways to start DelPhiController](#), a tool in the **Surface/Binding Analysis** and **Volume Data** categories (including from the [Volume Viewer Tools](#) menu). It has three sections organized like index cards: [Input](#), [Parameter](#), and [Output](#). Clicking a tab brings the corresponding card to the front.

Clicking **Run** initiates the DelPhi calculation; the time required depends on the system and parameters, but generally ranges from nearly instantaneous to minutes. **Cancel** dismisses the **DelPhiController** panel, while **Help** opens this manual page in a browser window.

The [Electrostatic Surface Coloring](#) tool for coloring [molecular surfaces](#) by potential will appear after the potential map (\*.phi file) has been created. Alternatively, the map can be shown as isopotential surfaces; these are not displayed automatically, but can be shown by starting [Volume Viewer](#) and clicking the eye icon or by using the [volume](#) command.

## Input

At a minimum, the required inputs are a structure of interest, a [radius file](#), a [charge file](#), and the location of the DelPhi executable. Any adjustments in [Parameter](#) settings and [Output](#) file names and locations must be made before the calculation is started.

The structure can be a model already open in Chimera or the contents of a local PDB file, specified using the pulldown menu in the **Molecule** section of the dialog.

- if **Chimera Model**, use **Browse...** to specify a model that has already been opened in Chimera
- if **PDB File**, use **Browse...** to specify the file or enter a pathname manually

Only the atoms to be included in the calculation should be present in the structure. It may be necessary to delete solvent, ions, ligands, and alternate crystallographic locations of atoms, and in some cases, to add hydrogens. Ultimately, which atoms should be included depends on the specific application and the availability of corresponding [radius](#) and [charge](#) parameters. Several tasks to prepare a model in Chimera can be performed with the [Dock Prep](#) tool (however, any charge assignments performed with [Dock Prep](#) will be ignored; DelPhi will only use the values in the [charge file](#) specified with **DelPhiController**).

The locations of other input files can be entered manually or filled in with **Browse**. When DelPhi has been run successfully with **DelPhiController**, the input file locations are saved in a user's [preferences file](#).

- **DelPhi Executable** (required) - name and location of the DelPhi executable. If a pathname is not already stored in the [preferences file](#), this will be filled in automatically provided the executable is on the user's path (and, on a Mac, provided Chimera is started from the terminal command line).
- **Atomic Radii File** (\*.siz, required) - a radius assignment file; see the [DelPhi development site](#) for examples
- **Atomic Charge File** (\*.crg, required) - a charge assignment file; see the [DelPhi development site](#) for examples
- **Potential File (for focussing)** (\*.phi) - a previously calculated electrostatic potential map, not required unless [focussing](#) is being performed
- **PDB Input for Site Potentials** (\*.pdb) - a PDB file, not required unless an [output](#) site potentials file is specified; however, if such an output is specified and this field is left blank, the coordinates of the structure for which the potential is being calculated will be used

[Radius](#) and [charge](#) files from the [DelPhi development site](#) or user-customized versions can be used. Each atom that does not match any specifications in the files will be assigned a radius of 0 and charge of 0. The [log file](#) should be checked to see if radii and charges have been assigned properly. It may also be helpful to generate a [modified PDB file](#). To achieve the correct assignments, it may be necessary to

- add or edit specifications in the assignment files so they will match atom/residue names in the structure
- edit atom/residue names in the structure so they will match specifications in the assignment files

## Parameter

DelPhi uses many parameters, and although reasonable defaults are supplied, what values are optimal will depend on the situation. Note that the defaults in **DelPhiController** may differ from those described in the DelPhi documentation.

### Basic:

- **Automatic Convergence** (on by default) - automatically calculate the number of iterations needed to attain convergence
- **Calculate GRASP Surface** (off by default) - generate a GRASP-format surface file named **grasp.srf** in the same location as the [output](#) electrostatic potential map
- **Box Fill (%)** (60 by default) - percentage of the grid that the largest dimension (X, Y, or Z) of the structure will fill. Smaller percentages will increase the accuracy of the boundary conditions, but yield a coarser representation of the molecule. For a single run (*i.e.*, when [focussing](#) is not done), reasonable compromise values are ~60% for viewing potentials outside a protein, ~90% for solvation energy calculations.
- **Grid Scale** (1.2 by default) - grid resolution, points per Å
- **Probe Radius** (1.4 by default) - radius (Å) of the solvent probe used to define the solvent-excluded surface; values of 1.4-1.8 Å are reasonable for water

### Advanced:

- **Fancy Charge Distribution** (off by default) - distribute charges to grid points using a spherical distribution (otherwise linear cubical interpolation will be used)

- **Log File Convergence** (off by default) - write convergence information to the output [log file](#)
- **Log File Potentials** (off by default) - write potentials to the output [log file](#)
- **Periodic Boundary (X)** (off by default)
- **Periodic Boundary (Y)** (off by default)
- **Periodic Boundary (Z)** (off by default)
- **Boundary Conditions** - how to set boundary potentials where [periodicity](#) is not used
  - **Zero**
  - **Debye-Huckel Dipole**
  - **Focussing** - use a potential map from a previous run to set the boundary conditions for the current run (requires an [input](#) electrostatic potential map)
  - **Debye-Huckel Total** (default) - Coulombic; recommended unless focussing is being performed
- **Interior Dielectric Constant** (2 by default) - dielectric constant inside the molecule; 2 represents electronic polarizability
- **Exterior Dielectric Constant** (80 by default) - dielectric constant outside the molecule; 80 approximates water
- **Grid Convergence** (0.0 by default)
- **Ionic Strength** (0.0 by default) - ionic strength (M) of the solvent; physiological ionic strength is approximately 0.145 M
- **Ion Radius** (0.0 by default) - thickness (Å) of the ion exclusion layer, relevant when ionic strength > 0.0; a value of 2.0 Å is reasonable for sodium chloride
- **Non-Linear Iterations** (0 by default)

## Output

The results of any requested **Energy** calculations will be written to the [log file](#).

- **Total Grid** - charge multiplied by potential, summed over all charges within the grid; contains the real electrostatic energy plus the grid self-energy
- **Reaction Field** - interaction of all charges with the induced surface charge within the grid; solvation energy is generally obtained by taking the difference in this value between calculations using different exterior dielectric constants
- **Coulombic** - energy calculated with Coulomb's law to bring all charges from infinity to their locations in the interior dielectric medium

Output files:

- **Potential File (phimap)** (\*.phi) - DelPhi electrostatic potential map; potentials are in kT/e (0.593 kcal/mol at 25°C)
- **Site Potentials File** (\*.frc) - file containing the potentials and fields at coordinates from an [input](#) file "for site potentials"
- **Dielectric Map File** (\*.eps) - binary dielectric bit map
- **Modified PDB File** (\*.pdb) - coordinates of the input structure in PDB format, except with the assigned radius and charge of each atom placed in columns 55-60 (replacing occupancy) and 61-67
- **Log File** (\*.log) - always produced; reports control parameter settings, calculation times, and other information

## PDB2PQR PDB2PQR

The **PDB2PQR** tool is an interface for running [PDB2PQR](#), which prepares structures for further calculations by reconstructing missing atoms, adding hydrogens, assigning atomic charges and radii from specified force fields, and generating [PQR](#) files. Several of the force field options were developed specifically for Poisson-Boltzmann calculations, and thus, a primary use is to prepare structures for [APBS](#) (Adaptive Poisson-Boltzmann Solver). The interface can use either a web service provided by the [National Biomedical Computation Resource \(NBCR\)](#) or a locally installed copy of the program. Users should cite:

[PDB2PQR: expanding and upgrading automated preparation of biomolecular structures for molecular simulations](#). Dolinsky TJ, Czodrowski P, Li H, Nielsen JE, Jensen JH, Klebe G, Baker NA. *Nucleic Acids Res.* 2007 Jul;35(Web Server issue):W522-5.

[PDB2PQR: an automated pipeline for the setup of Poisson-Boltzmann electrostatics calculations](#). Dolinsky TJ, Nielsen JE, McCammon JA, Baker NA. *Nucleic Acids Res.* 2004 Jul 1;32(Web Server issue):W665-7.

Results are opened as a new model in Chimera, with **charge** and **radius** [attributes](#) assigned to the atoms.

\*\* Any residues not handled by the designated [force field](#) will be omitted. Conversely, any unwanted residues such as waters should be [deleted](#) beforehand to ensure they do not appear in the result. \*\*

**PDB2PQR** and [Dock Prep](#) overlap in functionality, but only partially, and even the seemingly shared functions (*e.g.*, repairing truncated sidechains, adding hydrogens) are done differently. It may be useful to run certain parts of [Dock Prep](#) beforehand, for example, to delete [solvent](#). However, only the charge and radius assignments from **PDB2PQR**, not those from Chimera or other Chimera tools, can be written to a [PQR](#) file. See also: [AddH](#), [Add Charge](#), [FindHBond](#), [Coulombic Surface Coloring](#)

There are [several ways to start PDB2PQR](#), a tool in the **Structure Editing** category. It is also implemented as the command [pdb2pqr](#).

- **Molecule** - the structure of interest (choose from pulldown menu of models in Chimera)
- **Force field** - source of charge and radius information; **residues not handled by the designated force field will be omitted**
  - **AMBER** - AMBER ff99 ([Wang, Cieplak, and Kollman](#), *J Comput Chem* **21**:1049 (2000))
  - **CHARMM** - CHARMM27 ([MacKerell et al.](#), *J Phys Chem B* **102**:3586 (1998))
  - **PARSE** (default) - **PAR**ameters for **S**olvation **E**nergy ([Sitkoff, Sharp, and Honig](#), *J Phys Chem* **98**:1978 (1994) and [Tang et al.](#), *J Mol Biol* **366**:1475 (2007))
  - **PEOEPB** - a version of Gasteiger-Marsili Partial Equalization of Orbital Electronegativities, optimized for Poisson-Boltzmann calculations ([Czodrowski et al.](#), *Proteins* **65**:424 (2006))
  - **SWANSON** - AMBER ff99 charges with optimized radii ([Swanson et al.](#), *J Chem Theory Comput.* **3**:170 (2007))
  - **TYLO6** - a Poisson-Boltzmann-optimized force field ([Tang, Yang, and Luo](#), *J Phys Chem B* **110**:18680 (2006))

- **PQR output file (optional)** - name and location of output [PQR](#) file; if not specified, a temporary name and location will be used.

### Options:

- **Use PROPKA to predict protonation states (true/false)** - whether to use [PROPKA](#) (version 3.0) to predict the pKa values of ionizable groups in protein. Users should cite:

[Improved treatment of ligands and coupling effects in empirical calculation and rationalization of pKa values.](#) Søndergaard CR, Olsson MHM, Rostkowski M, Jensen JH. *J Chem Theory Comput.* 2011;7(7):2284-95.  
[Very fast empirical prediction and rationalization of protein pKa values.](#) Li H, Robertson AD, Jensen JH. *Proteins.* 2005 Dec 1;61(4):704-21.

- **pH value to use with PROPKA (default 7.0)** - set protonation states based on the PROPKA-predicted pKa values and the specified pH
- **Make protein N-terminus neutral (PARSE only) (true/false)** - only available for the PARSE [force field](#)
- **Make protein C-terminus neutral (PARSE only) (true/false)** - only available for the PARSE [force field](#)
- **Debump added atoms (true/false)** - ensure that new atoms are not rebuilt too close to existing atoms [[details](#) at the PDB2PQR site]
- **Optimize hydrogen bonds (true/false)** - adjust hydrogen positions and flip certain sidechains (His, Asn, Glu) as needed to optimize hydrogen bonds [[details](#) at the PDB2PQR site]
- **Hydrogen bond distance cutoff (default 3.4 Å)**
- **Hydrogen bond angle cutoff (default 30.0°)**
- **Report hydrogen bonds in Reply Log (true/false)** - whether to send hydrogen bond information to the [Reply Log](#)
- **Assign charges to ligands (true/false)** - use the PEOEPB approach (see [above](#)) to assign charges to any [ligand](#) residues, after protonation and conversion to Mol2 format (by Chimera) as currently required by the program [[details](#) at the PDB2PQR site]; the residue(s) will be renamed LIG and placed in chain L
- **Display APBS control file in Reply Log (true/false)** - whether to generate an example [APBS](#) control file and show it in the [Reply Log](#); however, such a file is not needed for running the Chimera [APBS](#) tool

### Executable location:

- **Opal web service (default)**
  - **Server** - URL of web service implemented using [Opal](#); clicking **Reset** restores the URL for the service provided by the [NBCR](#)
- **Local**
  - **Path** - pathname of locally installed executable

**OK** initiates the calculation and dismisses the dialog, whereas **Apply** initiates the calculation without dismissing the dialog. The job will be run in the background; clicking the information icon  in the

Chimera [status line](#) will bring up the [Task Panel](#), in which the job can be canceled if desired. **Close** dismisses the dialog, and **Help** opens this manual page in a browser window.

The processed structure will be opened as a new model in Chimera, with **charge** and **radius** [attributes](#) assigned to the atoms.

---

UCSF Computer Graphics Laboratory / October 2013


[Tools Index](#)

## Rotamers

[Startup and Input](#)
[Rotamer Lists](#)
[Evaluation](#)

- [Clashes](#)
- [H-Bonds](#)
- [Density](#)

[Rotamer List Menu](#)


## Rotamers

The **Rotamers** tool allows amino acid sidechain rotamers to be viewed, [evaluated](#), and [incorporated](#) into structures. A residue can be changed into a different rotamer of the same type of amino acid or mutated into a different type. The state of **Rotamers** is included in saved [sessions](#). See also: [Adjust Torsions](#), [Dock Prep](#), [Ramachandran Plot](#)

Although sidechains at multiple positions can be replaced simultaneously, the **Rotamers** tool is not recommended for predicting the conformations of multiple sidechains in an interacting cluster. Programs such as [SCWRL](#) are more appropriate for that purpose.

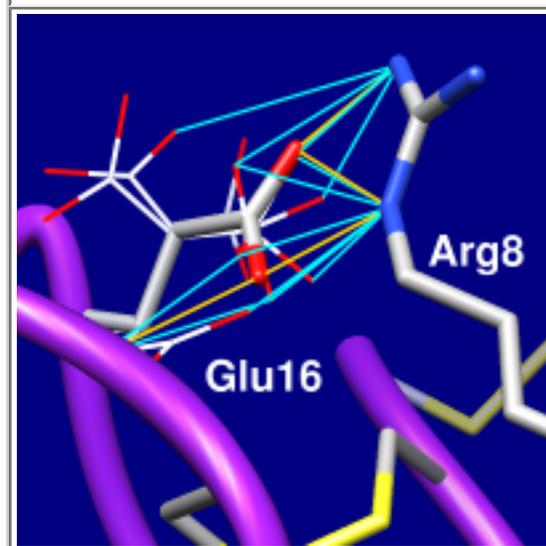
There are [several ways to start Rotamers](#), a tool in the **Structure Editing** category. **Rotamers** is also implemented as the command [swapaa](#).

One or more amino acid residues must be [selected](#) to indicate where the rotamers should be placed. Partly selected residues will be considered selected, and if more than one residue is selected, a rotamer set (of the same residue type) will be placed at each.

- **Rotamer type** - of the 20 standard amino acids, which type of rotamers to show at each [selected](#) position.

In the **Dynamics** [rotamer library](#) only, there are multiple choices of type for cysteine and histidine depending on the oxidation or protonation state of the sidechain:

- CYH - cysteine reduced free sulfhydryl
- CYS - cysteine oxidized disulfide-bonded (half-cystine)

3hj2 Glu16 rotamers (see [list](#))

- HID - histidine neutral  $\delta$ -protonated
- HIE - histidine neutral  $\epsilon$ -protonated
- HIS - histidine neutral (HID and HIE combined)
- HIP - histidine positive protonated on both sidechain nitrogens

These refer to the species for which conformational data were collected, but the rotamers do not include hydrogens and will be given standard residue names (CYS or HIS) if incorporated into a structure.

For nonstandard amino acids, see the [SwissSidechain Chimera plugin](#).

- **Rotamer library** - source of rotamer torsion values and probabilities
  - **Dunbrack** - [Dunbrack backbone-dependent rotamer library](#) (for chain-terminal residues, the Dunbrack backbone-independent version is used instead):

[Rotamer libraries in the 21st century](#). Dunbrack RL Jr. *Curr Opin Struct Biol*. 2002 Aug;12(4):431-40.

- **Dynameomics** - [Dyameomics rotamer library](#):

[The Dynameomics rotamer library: amino acid side chain conformations and dynamics from comprehensive molecular dynamics simulations in water](#). Scouras AD, Daggett V. *Protein Sci*. 2011 Feb;20(2):341-52.

- **Richardson (common-atom)** - common-atom values (author-recommended) from the [Richardson backbone-independent rotamer library](#):

[The penultimate rotamer library](#). Lovell SC, Word JM, Richardson JS, Richardson DC. *Proteins*. 2000 Aug 15;40(3):389-408.

- **Richardson (mode)** - mode values from the [Richardson backbone-independent rotamer library](#)

**Close** dismisses the **Rotamers** dialog, while **Help** opens this manual page in a browser window.

When **Apply** (or **OK**, which also dismisses the dialog) is clicked, phi and psi angles at each [selected](#) position are measured and reported in the [Reply Log](#). If a type without a rotamer set (glycine, alanine) has been specified, the residue(s) will simply be changed to the new type, and the change cannot be undone. Otherwise, for each [selected](#) position, the specified rotamer set will be displayed and a [rotamer list](#) will appear. Bond lengths and angles are taken from the [Amber](#) parameter files **all\*94.lib**, and hydrogens are not included. The rotamers are

placed on the existing residue by superimposing the CA, CB, and C atoms, or if the existing residue does not have a CB atom, the N, CA, and C atoms.

## ROTAMER LISTS

Each rotamer list includes sidechain torsion (**Chi**) and **Probability** values.

Probabilities are simply taken from the [rotamer library](#)

and are not affected by the structural environment, except by phi and psi angles when the Dunbrack library is used.

Rotamers are initially listed in order of decreasing probability, but the listing can be sorted by the values in any

column by clicking the header. Clicking the header once sorts the entries in order of increasing value and places an up arrowhead in the header. Clicking again sorts the entries in decreasing order and places a down arrowhead in the header.

The rotamer-list [Columns menu](#) controls which columns are shown and allows rotamers to be [evaluated](#) in terms of clashes or contacts, hydrogen bonds, and/or overlap with a density map. The rotamer-list [Select menu](#) allows rotamers to be [selected](#) for subsequent operations such as with the [Actions menu](#) or [commands](#).

One or more rotamers can be *chosen* from the list by clicking and dragging with the left mouse button; **Ctrl**-click toggles whether a rotamer is chosen. Only the chosen rotamers are displayed.

Clicking **Apply** (or **OK**, which also dismisses the list) will incorporate the chosen rotamer(s). When there will be multiple sidechains at a given residue position, the new sidechain(s) will be assigned different alternative location identifiers. The **Existing side chain(s)** setting controls what happens to the pre-existing sidechains. They can be replaced, retained, or selectively retained based on [selection](#). After a sidechain has been replaced, it cannot be recovered except by reopening the structure. The [display style](#) of the former sidechain will be used. Other characteristics, however, will not be transferred and may need to be regenerated (for example, with [AddH](#) and [Add Charge](#)). **Close** dismisses the list

**rotamer list (see [display](#))**

Chi 1	Chi 2	Chi 3	Probability	Clashes	H-Bonds	Density▼
-64.6	-177.9	-2.3	0.269889	0	4	6.84
-60.3	-175.3	-56.1	0.144294	0	3	5.91
-63.5	177.2	57.0	0.103382	2	3	4.68
-66.8	91.0	-54.1	0.002987	4	1	2.25
57.5	-175.8	2.0	0.039138	12	1	2.17
-60.3	83.2	4.4	0.019166	4	0	2.04
-59.9	-72.2	-12.0	0.107063	5	0	1.99
-53.2	-79.8	59.8	0.008870	3	0	1.97
-73.7	77.8	52.0	0.027550	1	0	1.95
-178.8	68.8	3.0	0.008467	17	0	1.91
-60.5	-67.7	-47.3	0.093982	3	0	1.71
-174.5	72.1	-57.9	0.004032	18	1	1.58
-178.8	179.0	8.7	0.044019	11	2	1.50

Existing side chain(s): **replace**

Buttons: **OK** **Apply** **Close** **Help**

and deletes the set of rotamers without further modifying the structure.

## ROTAMER EVALUATION

Rotamers can be evaluated in the context of the structural environment or [compared to density](#) (or other [volume data](#)). Only the sidechain atoms of a rotamer are evaluated. For [hydrogen bond](#) and [clash](#) detection, interactions with rotamers in the same set and the current residue at that position are disregarded, but all other atoms in the vicinity will be included unless **Ignore... other models** (see below) is used. That setting is useful for preventing superimposed related proteins or additional copies of the starting structure from contributing to the counts. In addition, atoms in the same model that are unwanted for such calculations (for example, [solvent](#)) should be deleted beforehand.

[Columns... Add... Clashes](#) brings up a dialog for finding clashes or contacts and showing the results in the [rotamer list](#). This uses capabilities from [Find Clashes/Contacts](#):

- **Find atoms with VDW overlap  $\geq$  [ *cutoff* ] angstroms** (default 0.6 Å)
- **Subtract [ *allowance* ] from overlap for potentially H-bonding pairs** (default 0.4 Å)

The *overlap* between two atoms is defined as the sum of their [VDW radii](#) minus the distance between them and minus an allowance for potentially hydrogen-bonded pairs:

$$overlap_{ij} = r_{VDW_i} + r_{VDW_j} - d_{ij} - allowance_{ij}$$

An *allowance*  $> 0$  reflects the observation that atoms sharing a hydrogen bond can come closer to each other than would be expected from their [VDW radii](#). It is only subtracted for pairs comprised of a possible hydrogen-bond donor atom (or donor-borne hydrogen) and possible acceptor atom.

The **Default criteria** are appropriate for clash detection. For detecting *clashes*, *cutoff* values of 0.4–1.0 Å and *allowance* values of 0.2–0.6 Å are generally reasonable. For detecting *contacts*, negative *cutoff* values of 0.0–(–1.0) Å with an *allowance* of 0.0 Å are generally reasonable.

- **Column Value** - what to show in the **Clashes** column of the [rotamer list](#)
  - **Number of clashes** (default) - how many interactions with [overlap](#)  $\geq$  [cutoff](#) are found for the rotamer
  - **Sum of overlaps** - the sum of rotamer [overlap](#) values  $\geq$  [cutoff](#)
- **Draw pseudobonds of color [ [color well](#) ] and width [ *linewidth* ]** - whether to show the interactions as [pseudobonds](#) of the specified color and linewidth

- **Ignore clashes with other models** - whether to count only interactions with the model for which the rotamer set is shown

[Columns... Add... H-Bonds](#) brings up a dialog for finding hydrogen bonds (H-bonds) and showing the results in the [rotamer list](#). This uses capabilities from [FindHBond](#):

- **Draw H-bonds of color** [ [color well](#) ] and **width** [ *linewidth* ] - whether to show the H-bonds as [pseudobonds](#) of the specified color and linewidth
- **Relax H-bond constraints** (on by default) - whether the tolerances to **Relax constraints by** (default 0.4 Å and 20.0 degrees) should be applied to the precise [geometric criteria](#) for hydrogen bonding
- **Color H-bonds not meeting precise criteria differently** [ [color well](#) ] - whether to use the specified color for H-bonds meeting the relaxed but not the precise [geometric criteria](#)
- **Ignore H-bonds with other models** - whether to count only interactions with the model for which the rotamer set is shown

[Columns... Add... Density](#) brings up a dialog for comparing rotamer sidechain atoms with a density map (or other [volume data](#)) and showing the results in the [rotamer list](#). The **Volume data** should be chosen from the list of open volume data sets. **Browse...** can be used to locate and open a volume data file if it is not already open. Map values at the rotamer sidechain atom positions (see [Values at Atom Positions](#)) will be summed and shown using the specified **Column name**.

Clicking **Apply** (or **OK**, which also dismisses the dialog) performs the calculation and shows the results in the [rotamer list](#). **Close** simply dismisses the dialog, while **Help** opens this manual page in a browser window.

## ROTAMER LIST MENU

### Select

- **Current Rotamers** - [select](#) all rotamers currently [chosen](#) in the [rotamer list](#)
- **All Rotamers** - [select](#) all rotamers in the [rotamer list](#)

### Columns

- **Add**
  - **Clashes...** [evaluate](#) rotamer [clashes/contacts](#) and show the results in a **Clashes** column
  - **H-Bonds...** [evaluate](#) rotamer [H-bonds](#) and show the results in an **H-Bonds** column
  - **Density...** [evaluate](#) rotamer [fit with density](#) or other [volume data](#) and show the results in a column

- [*currently available columns*] - show/hide column according to checkbox

---

UCSF Computer Graphics Laboratory / August 2013

## Chimera Interface to Modeller

Chimera provides a graphical interface to running the program [Modeller](#), either locally or via a web service hosted by the [UCSF RBVI](#). Two types of calculations are available:

- [Comparative \(homology\) modeling](#). Theoretical models of a protein are generated using at least one known related structure and a sequence alignment of the known and unknown structures. The protein to be modeled is the *target*, and a related known structure used for modeling is a *template*. The inputs for comparative modeling can be generated in [several ways](#).
- [Building](#) parts of a protein without using a template. Missing segments can be built *de novo*, or existing segments *refined* by generating additional possible conformations. Parts that need building or refinement are often loop regions.

Modeller is developed by the [Sali Lab](#). Use of Modeller, whether a previously [downloaded](#) copy or via web service, requires a *license key*. Academic users can [register](#) free of charge to receive a license key. (Commercial entities and government research labs, please see [Modeller licensing](#).) Modeller users should cite:

[Comparative protein modelling by satisfaction of spatial restraints](#). Šali A, Blundell TL. *J Mol Biol*. 1993 Dec 5;234(3):779-815.

See also: [Build Structure](#), [Rotamers](#), [Multalign Viewer](#), [mda](#), [ModBase fetch](#), [modifying and saving data](#), the [Comparative Modeling tutorial](#)

### ← Comparative Modeling with Modeller

Comparative modeling requires a structure to serve as a template, and a target-template sequence alignment. There are [several ways](#) to obtain these inputs using Chimera. Sequence alignments are shown in [Multalign Viewer](#). The Chimera interface to comparative modeling with Modeller can be started by choosing [Structure... Modeller \(homology\)](#) from the **Multalign Viewer** menu, or called with [mda](#). With this interface, **\*only a single chain or subunit can be modeled at a time.\*** Modeling a multimer or complex requires running Modeller outside of Chimera.

- **Choose the target (sequence to be modeled)** - the name of the target sequence should be chosen from the pulldown menu of all sequences in the current alignment
- **Choose at least one template** - at least one structure to use as a template should be chosen from the table, by clicking or dragging with the left mouse button to highlight the corresponding rows. **Ctrl**-click toggles the status of a single row. The choices are actually the sequences in the alignment, and choosing a sequence indicates using its associated structure(s). Some of the columns are blank initially, but clicking [Fetch Structures/Annotations](#) loads the structures into Chimera and fills in the table, where possible. The table can be sorted by the contents of any column by clicking the column header:
  - **Sequence** - sequence name; choosing a sequence indicates using its associated structure as a template (the table lists all of the sequences in the alignment except the target)
  - **Structure ID** - identifier for the structure associated with the sequence, if any; usually a 4-

character PDB ID, but could be a [SCOP domain ID](#) (which includes a PDB ID)

- **%ID** - percent sequence identity of the sequence as compared to the target, computed from the alignment
- **Title** - title of structure (from PDB entry)
- **Organism** - source organism of structure (from PDB entry)

#### Clicking **Fetch Structures/Annotations**:

1. fetches the structure for any sequence not already associated with a structure, if that structure can be deduced from the sequence name (as in the [Structure preferences](#) in [Multalign Viewer](#))
2. uses the structure IDs to look up additional information

#### • Choose where to run Modeller:

- **Run Modeller via web service**
  - **Modeller license key** - a [license key](#) is required to run the program; the [Modeller home page](#) includes links to register for a key and to download the program
- **Run Modeller locally**
  - **Location of Modeller executable** - the location of the Modeller executable file; the [license key](#) will have been entered somewhere already during local installation
  - **Modeller script file (optional, overrides dialog)** - use the specified Modeller script to control the calculation; this will override the settings in the dialog. The script corresponding to the current dialog settings can be viewed in [IDLE](#) by clicking **Get Current Modeller Script**, saved to a file using the IDLE menu, and edited by hand as desired. For more details on scripting Modeller, consult the [Modeller manual](#).

#### • **Advanced Options:**

- **Number of output models [N] (max 1000) (default 5)**
- **Include non-water HETATM residues from template** (off by default) - whether to include HETATM residues other than water (ligands, ions, detergent molecules, *etc.*) from the template in the output models. This option will propagate all qualifying residues, even from multiple templates; those not desired in the output should be [deleted](#) from the template(s) beforehand.
- **Include water molecules from template** (off by default) - whether to include water residues from the template in the output models. This option will propagate all qualifying residues, even from multiple templates; those not desired in the output should be [deleted](#) from the template(s) beforehand.
- **Build models with hydrogens (warning: slow)** (off by default)
- **Use fast/approximate mode (produces only one model)** (off by default) - use fast/approximate mode (~3 times faster) to get a rough idea of model appearance or to confirm that the alignment is reasonable. This mode does not randomize the starting structure (generates only a single model) and performs very little optimization of the target function.
- **Use thorough optimization (recommended with MDA)** - optimize more thoroughly than the default, as recommended for modeling large targets with **MultiDomain Assembler** (the [mda](#) command)
- **Temporary folder location (optional)** - use the specified location for temporary files; otherwise, a location will be generated automatically
- **Distance restraints file (optional)** - Specify an input file containing distance restraints (see example file [distres.txt](#)). Each line in the file should be of the format:

*res1 res2 dist stdev*

where *res1* and *res2* are residue numbers or ranges of residue numbers in the target sequence, *dist* is the distance in Å, and *stdev* is the standard deviation. If a single residue is specified, its C $\alpha$  will be used to anchor the restraint. If a residue range (e.g. 233–275) is specified, the range's center of mass will be used to anchor the restraint.

**OK** starts the calculation and dismisses the panel, while **Apply** starts the calculation without dismissing the panel. **Close** dismisses the panel without performing any calculation. **Help** brings up this manual page in a browser window.

Running Modeller is a background task. Clicking the information icon  in the Chimera [status line](#) will bring up the [Task Panel](#), in which the job can be canceled if desired.

After the calculation has finished, the comparative models are opened in Chimera and can be [saved](#) in the usual ways. The models are automatically superimposed onto the template (or the lowest-numbered of multiple templates) using [matchmaker](#) defaults, and the view is focused on that template. The models are [associated](#) with the target sequence, and the **RMSD header** displayed in the [sequence window](#). Model scores are shown in a [Model List](#), the same dialog used for comparative models [fetched](#) from [ModBase](#). Loops or other parts of a model can be subjected to further [refinement](#).

Running Modeller with identical inputs on different machines may give different (but equally valid) results, due to small numerical differences that can lead to finding different local optima of the modeling objective function.

## ← Building/Refinement with Modeller (Model/Refine Loops )

The only required input for Modeller building or refinement is a protein structure. Missing segments can be built *de novo*, or existing segments *refined* by generating additional possible conformations. Building and refinement can be applied to protein structures regardless of whether they were [modeled](#) or determined experimentally.

The Chimera interface to Modeller for building or refinement can be accessed by [starting Model/Refine Loops](#), a tool in the **Structure Editing** category, or by choosing [Structure... Modeller \(loops/refinement\)](#) from the **Multalign Viewer (Sequence tool)** menu. Using **Model/Refine Loops** is equivalent to using the [Sequence](#) tool to show the sequence of a chain, then using its menu to show the interface. If the structure chain is [associated](#) with a sequence in an alignment in [Multalign Viewer](#), however, the alignment sequence can be used instead of the individual sequence.

- The output models will include any water molecules and other HETATM residues from the input model; those not desired in the output should be [deleted](#) from the input beforehand.
- **Model/remodel** - which parts of the input model to build or refine; they will be considered collectively even if composed of multiple noncontiguous sequence segments
  - **active region** - residues in the [active region](#) in [Multalign Viewer](#) (the region most recently drawn or clicked); this [region](#) must contain only residues from one sequence, and the

model [associated](#) with that sequence will be used as the input

- **Chimera selection region** - the currently [selected](#) amino acid residues, as represented by the **Chimera selection region** in [Multalign Viewer](#); this [region](#) must contain only residues from one sequence, and the model [associated](#) with that sequence will be used as the input
- **non-terminal missing structure** - *missing structure* refers to residues that appear in PDB SEQRES records but are missing from the coordinates section of a PDB file. The model of interest should be chosen from the pulldown menu on the right. By default, missing structure is indicated in the sequence with red outline boxes, corresponding to the **missing structure... [region](#)** for the model. The *non-terminal* missing structure consists of only the missing segments that are constrained at both ends by existing structure.

\*\* Note: if the input model contains a non-terminal missing segment that the user opts not to build, inevitably Modeller will bring the existing structure on either side together to “close the gap” in the output models. \*\*

- **all missing structure** - all missing segments, including N-terminal and C-terminal parts only constrained at one end by existing structure; all residues in the **missing structure... [region](#)** for the model chosen from the pulldown menu on the right
- **Allow this many residues adjacent to missing regions to move** (default 1) - if one of the **missing structure** options is used, how many existing residues at each missing/existing junction to allow to move relative to the input structure
- **Number of models to generate** (default 5)
- **Loop modeling protocol**
  - **standard** (default)
  - **DOPE** - Discrete Optimized Protein Energy score (see [Shen and Sali](#), *Protein Sci* 15:2507 (2006)) with Lennard-Jones potential and GB/SA implicit solvent interaction; generally higher quality but more computationally expensive than the standard method, and more prone to calculation failure (resulting in fewer models than requested)
  - **DOPE-HR** - same as DOPE, except higher precision
- **Run modeller using**
  - **web service** (default)
  - **local installation**

If web service:

  - **Modeller license key** - a [license key](#) is required to run the program; the [Modeller home page](#) includes links to register for a key and to download the program

If local installation:

  - **Location of Modeller executable** - the location of the Modeller executable file; the [license key](#) will have been entered somewhere already during local installation
  - **Custom Modeller script file (optional, overrides dialog)** - use the specified Modeller script to control the calculation; this will override the settings in the dialog. The script corresponding to the current dialog settings can be viewed in **IDLE** by clicking **Get Current Modeller Script**, saved to a file using the IDLE menu, and edited by hand as desired. For more details on scripting Modeller, consult the [Modeller manual](#).
- **Temporary folder location (optional)** - use the specified location for temporary files; otherwise, a location will be generated automatically

**OK** starts the calculation and dismisses the panel, while **Apply** starts the calculation without dismissing the panel. **Close** dismisses the panel without performing any calculation. **Help** brings up this manual page in a browser window.

Running Modeller is a background task. Clicking the information icon  in the Chimera [status line](#) will bring up the [Task Panel](#), in which the job can be canceled if desired.

After the calculation has finished, the models (each including the unchanged parts of the protein in addition to what was built or refined) are opened in Chimera and can be [saved](#) in the usual ways. The models are [associated](#) with the sequence, and the **RMSD header** displayed in the [sequence window](#). Model scores are shown in a [Model List](#), the same dialog used for comparative models [fetched](#) from [ModBase](#).

---

UCSF Computer Graphics Laboratory / October 2014



## Movement Mouse Mode

**Movement Mouse Mode** enables moving part or all of a molecule model (and not other parts or other models) with the mouse regardless of [activation status](#). The set of atoms to move can be a [selection](#), chain, secondary structure element, or whole molecule model. This tool allows making coarse modifications to structures, after which their coordinates can be [saved](#).

There are [several ways to start Movement Mouse Mode](#), a tool in the **Structure Editing** and **Movement** categories.

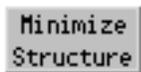
The dialog controls which items will rotate and translate in response to [mouse manipulations](#):

- **Normal** (default) - all models that are [activated for motion](#)
- **Move selection** - the currently [selected](#) atoms and/or [surface pieces](#); when nothing is [selected](#), as **Normal**
- **Move molecule** - the molecule model under the cursor when the mouse button was pressed; when no molecule model was under the cursor, as **Normal**
- **Move chain** - the chain under the cursor when the mouse button was pressed; when no chain was under the cursor, as **Normal**
- **Move helix, strand, or turn** - the helix, strand, or other peptide chain segment (comprised of residues containing CA) under the cursor when the mouse button was pressed; when no such segment was under the cursor, as **Normal**

The rotation and translation mouse button assignments (as shown in the [Mouse preferences](#)) are not changed, only which items respond to the rotations and translations.

*Nonstandard* movements (by mouse modes other than **Normal**) occur regardless of model [activation](#) status. There is no consideration of what is physically reasonable; any bonds between movable and immovable parts will simply stretch and distort. Nonstandard movements can be undone and redone by clicking the **Undo Move** and **Redo Move** buttons, respectively. A history of up to 100 nonstandard movements is stored. Positions are only logged after pauses in motion of at least 1 second. (The separate [Undo/Redo Move](#) tool handles only standard movements.)

The center of rotation is as described in the [Rotation](#) tool, except that only the movable atoms are considered when calculating the bounding box for the **center of models** method.



## Minimize Structure

**Minimize Structure** energy-minimizes molecule models, optionally holding some atoms fixed. Minimization routines are provided by [MMTK](#), which is included with Chimera. [Amber](#) parameters are used for standard residues, and Amber's [Antechamber](#) module (also included with Chimera) is used to assign parameters to nonstandard residues.

**Minimize Structure** is in development and has several [limitations](#). It is intended for cleaning up small molecule structures and improving localized interactions within larger systems. It may not be able to resolve large-scale distortions or widespread structural problems. By definition, energy-minimization simply moves the system toward a local minimum without crossing energy barriers, and does not search for the global minimum.

There are [several ways to start Minimize Structure](#), a tool in the **Structure Editing** category. It is also implemented as the command [minimize](#).

Models to minimize can be chosen from the list with the left mouse button. Ctrl-click toggles the status of an individual model. To choose a block of models without dragging, click on the first (or last) and then Shift-click on the last (or first) in the desired block. All chosen models are treated as a single system for energy calculations; other models are ignored. Within the chosen models, all atoms are included in energy calculations, regardless of whether they are [held fixed](#). (However, parts of models can be excluded from energy calculations using the [minimize](#) command with [fragment true](#).)

[Steepest descent](#) minimization is performed first to relieve highly unfavorable clashes, followed by [conjugate gradient](#) minimization, which is much slower but more effective at reaching an energy minimum after severe clashes have been relieved. Energies (kJ/mol) are reported in the [Reply Log](#). **\*\*Step numbers reported by MMTK are 2 greater than the actual numbers of minimization steps performed.** The additional "steps" are not minimization steps but operations required to obtain gradient values and updated coordinates.\*\*

- **Steepest descent steps** (default **100**) - number of steps of steepest descent minimization to perform before any conjugate gradient minimization
- **Steepest descent step size (Å)** (default **0.02**) - initial step length for steepest descent minimization
- **Conjugate gradient steps** (default **10**) - number of steps of conjugate gradient minimization to perform after finishing any steepest descent minimization
- **Conjugate gradient step size (Å)** (default **0.02**) - initial step length for conjugate gradient minimization
- **Update interval** (default **10**) - how frequently to update the display, in terms of minimization steps
- **Fixed atoms** - atoms to hold in place during minimization:
  - **none** (default) - all atoms will be allowed to move
  - **selected** - any [selected](#) atoms should be held fixed, all other atoms allowed to move
  - **unselected** - any [selected](#) atoms should be allowed to move, all other atoms held fixed
- **Memorize options chosen in subsequent dialogs** - as explained below, [Dock Prep](#) and further tools may be called to prepare structures for minimization; this option specifies saving their

settings in the [preferences file](#) for future uses of **Minimize Structure**

- **Use previously memorized options, if any** - use settings saved with the preceding option in a prior use of **Minimize Structure**
- **Neither memorize nor use memorized options** - do not use previously saved settings; show the dialogs so that settings can be chosen explicitly for this use of **Minimize Structure**, but do not save the settings

Clicking **Minimize** dismisses the dialog (unless the option to **Keep dialog up after Minimize** is checked) and may call [Dock Prep](#) to perform [several tasks](#) to prepare the system for energy calculations. In turn, [Dock Prep](#) may call additional tools:

- [AddH](#) to add hydrogens. If **Minimize** has already been clicked, the [selection](#) will be adjusted prior to minimization to include any newly attached atoms in the [fixed](#) and movable sets. Note that any other models in the vicinity will influence hydrogen placement even if hydrogens are not being added to those other models. If such interactions are not desired, the other models should be closed beforehand.
- [Add Charge](#) to associate atoms with partial charges and other [force field parameters](#). Required even when [alternative charges](#) will be used.

It is sometimes useful to run [Dock Prep](#) independent of **Minimize Structure** beforehand, then skip all tasks when it reappears after **Minimize** is clicked:

- adding hydrogens separately beforehand allows them to be deleted or repositioned as needed prior to minimization
- adding charges separately beforehand allows [alternative charges](#) to be specified prior to minimization

**Close** dismisses the **Minimize Structure** dialog. **Help** opens this manual page in a browser window.

## Force Field Parameters

Different procedures are used to assign parameters to [standard residues](#), [monatomic ions](#), and [nonstandard residues](#).

*Standard residues* include water, standard amino acids, standard nucleic acids, and a few common variants and capping groups.

1. [Add Charge](#) recognizes standard residues based on their atom and residue names and assigns Amber residue names, Amber atom types, and atomic partial charges from an Amber force field chosen by the user, default **ff14SB** ([details](#)).
2. **Minimize Structure** uses the Amber atom types to associate the atoms with other parameters from the chosen force field.

*Monatomic ions* are assigned user-specified net charges and Amber VDW parameters. The following ions are handled: Li<sup>+</sup>, Na<sup>+</sup>, K<sup>+</sup>, Rb<sup>+</sup>, Cs<sup>+</sup>, F<sup>-</sup>, Cl<sup>-</sup>, Br<sup>-</sup>, I<sup>-</sup>, Mg<sup>2+</sup>, Ca<sup>2+</sup>, Zn<sup>2+</sup>. In addition, Fe ion nonbonded parameters are taken from the heme residue in the [Amber parameter database](#). See [Limitations](#) for how to

add types.

*Nonstandard residues* are all residues not recognized as [standard residues](#) or [monatomic ions](#).

1. **Add Charge** uses Amber's [Antechamber](#) module (included with Chimera) to assign [GAFF types](#) and calculate atomic partial charges within each nonstandard residue. It is necessary to specify the formal charge of each nonstandard residue and which [charge calculation method](#) should be used. Publications involving [Antechamber](#) use should cite:

[Automatic atom type and bond type perception in molecular mechanical calculations.](#) Wang J, Wang W, Kollman PA, Case DA. *J Mol Graph Model.* 2006 Oct;25(2):247-60.

Note that Antechamber/GAFF are meant to handle most small organic molecules, but not metal complexes, inorganic compounds, or unstable species such as radicals, and may not work well on highly charged molecules.

2. **Minimize Structure** uses the GAFF types to associate nonstandard residues with parameters other than charges. The GAFF atom types and associated parameters are described [online](#) and in:

[Development and testing of a general amber force field.](#) Wang J, Wolf RM, Caldwell JW, Kollman PA, Case DA. *J Comput Chem.* 2004 Jul 15;25(9):1157-74.

## User-Specified Partial Charges

Arbitrary partial charges (such as obtained from the literature or [parameter databases](#)) can be specified. To do so:

1. run [Dock Prep](#) independent of **Minimize Structure** to perform any necessary tasks including charge addition (thus running [Add Charge](#), which is still needed to assign Amber/GAFF atom types)
2. reassign the **charge** attribute of the atoms to the desired values (using [Define Attribute](#), [defattr](#), or [setattr](#))
3. run **Minimize Structure** and turn off all options in the ensuing [Dock Prep](#) dialog, as the necessary tasks have already been performed

## Limitations

**Lack of access to many settings.** There is no way to specify several MMTK settings, including distance cutoffs. MMTK defaults are used. Evaluating all pairwise nonbonded interactions regardless of interatomic distance makes the calculation relatively slow.

**Limited ability to use arbitrary parameters.** It is difficult to change or add parameters. Arbitrary partial charges can be [specified](#). Experts can adjust parameters (other than charge) of [standard residues](#) and [monatomic ions](#) by editing files in `bin/amber14/dat/leap/parm/` within the Chimera installation. The following parameter files are used:

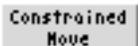
- **ff14SB**: parm10.dat + frcmod.ff14SB + frcmod.ionsjc\_tip3p
- **ff12SB**: parm10.dat + frcmod.ff12SB + frcmod.ionsjc\_tip3p
- **ff03ua**: parm99.dat + frcmod.ff03 + frcmod.ff03ua + frcmod.ionsjc\_tip3p
- **ff03.r1**: parm99.dat + frcmod.ff03 + frcmod.ionsjc\_tip3p

In addition, a custom frcmod file (not included with the Amber distribution) is used to specify Fe ion nonbonded parameters taken from the heme residue in the [Amber parameter database](#) (Bryce Group, University of Manchester).

To add an element that is not already handled, it may also be necessary to create a file for that element in the MMTK atom database within the Chimera installation. For example, to handle Li<sup>+</sup>, there is a file **lib/python\*/site-packages/MMTK/Database/Atoms/li** (where \* is the appropriate python version number) containing the following:

```
name = 'lithium'  
symbol = 'Li'  
mass = 6.941
```

## Constrained Move



**Constrained Move** provides mouse modes for rotation and translation relative to axes defined in a model's frame of reference. Such manipulations are helpful for aligning structures with an axis of symmetry. There are [several ways to start Constrained Move](#), a tool in the **Movement** category.



### CONSTRAINED MOUSE MODES

**Constrain mouse rotations** reassigns the button currently assigned to rotation to constrained rotation. Likewise, **Constrain mouse translations** reassigns the button currently assigned to XY translation to constrained translation. Deactivating these options restores the previous rotation/translation behavior.

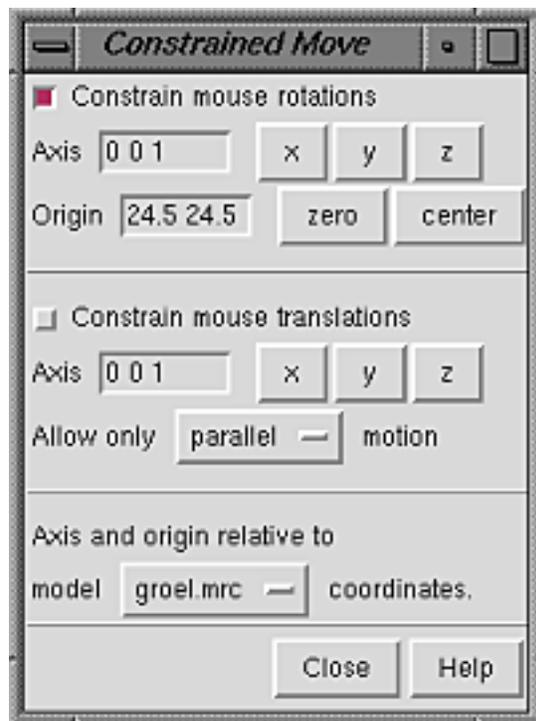
It is possible to have different buttons assigned to a constrained motion and the corresponding unconstrained motion at the same time. Buttons can be reassigned using the [Mouse preferences](#). The constrained mouse modes are not shown in the preferences panel unless **Constrained Move** has been started.

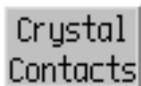
### AXIS DEFINITION

Different axes can be used for rotation and translation. The three values in the **Axis** field specify direction. Clicking **x**, **y**, or **z** sets the direction along the reference model's X, Y, or Z axis, directions (1 0 0), (0 1 0), or (0 0 1), respectively. Any three values can be entered, however, and should be separated by spaces, with no commas or parentheses.

An axis for *rotation* is defined by a direction and an anchor point, or **Origin**. Clicking **zero** sets the **Origin** to (0 0 0), while clicking **center** sets it to the center of the bounding box of the reference model. Only a direction is needed to constrain *translation*.

It is not necessary to turn off constrained modes before changing the reference model, an axis definition, or whether translation should be parallel or perpendicular to the specified axis.





## Crystal Contacts

The **Crystal Contacts** tool checks for clashes between copies of a structure when they are positioned according to symmetry information in the input coordinate file (PDB or mmCIF). This tool was created to validate symmetry information for virus capsid structures in the [Protein Data Bank](#). See also: [Unit Cell](#), [Find Clashes/Contacts](#)

There are [several ways to start Crystal Contacts](#), a tool in the **Higher-Order Structure** category. It is also implemented as the command [crystalcontacts](#).

The **Molecule** (presumably read from a file with symmetry information) should be chosen from the list of open molecule models. Clicking **Show Contacts** checks for atomic contacts within the specified **Contact distance** between copies. Atoms to exclude from the calculation (solvent, *etc.*) should be [deleted](#) beforehand.

The results are shown schematically. Each copy of the original structure is shown as a ball, and close contacts between copies are shown as cylinders connecting the balls:

- **green balls** = the original structure and copies related by noncrystallographic symmetry
- **blue balls** = copies related by crystallographic symmetry, together with the green balls representing one unit cell
- **yellow balls** = copies from neighboring unit cells
- **red cylinders** = close contacts

The steps are described in more detail below; the symmetry information is described as for PDB files, but the corresponding information from an mmCIF file can also be used.

- CRYST1 information is used to draw a unit cell box with one corner at (0,0,0).
- The structure in the PDB file is represented as a single green ball, even if it includes multiple chains.
- MTRIX records are used to generate copies related by noncrystallographic symmetry, also shown in green; the green balls collectively represent one *crystal asymmetric unit (CAU)*. If there are no MTRIX records, the starting structure is already the CAU and only one green ball will be shown (example: PDB entry **1bzm**). Icosahedral capsid structures typically contain a fraction of the CAU, a *noncrystallographic symmetry (NCS) asymmetric unit* that must be duplicated according to MTRIX records to generate the CAU (example: PDB entry **1v9u**). Each ball represents a single NCS asymmetric unit.
- CRYST1 or SMTRY records are used to generate copies related by crystallographic symmetry, shown in blue; the blue and green balls together represent the contents of one *crystal unit cell*.
- Copies with any atom within the **Contact distance** of a copy shown in green are identified. A red cylinder with radius proportional to the number of contacting atoms is drawn to represent the contact; the minimum and maximum cylinder radii are limited, with the minimum corresponding to 1% of atoms in contact and maximum corresponding to 5%. Copies in neighboring unit cells are

shown as yellow balls if any of them form close contacts with a copy shown in green. Copies right on top of each other are shown with a transparent green ball rather than a connecting cylinder.

- The balls and box are created as a [marker set](#) and opened as a separate model.

The **Create copies of contacting molecules** option indicates whether to load the full atomic coordinates of any contacting copies when **Show Contacts** is pressed.

Information on the contacts is listed in the [Reply Log](#):

Each line reports a pair of contacting NCS asymmetric units. While low numbers of contacting **Atoms** are fairly common, very high numbers (especially complete overlap of the structures) indicate there may be an error in the symmetry information. An index **MATRIXref** tells which MTRIX record positions one subunit, and indices **MTRIX**, **SMTRY**, and **Unit Cell** identify the other. The MTRIX and SMTRY matrix indices start at 0, and therefore do not exactly match the matrix numbers given in the PDB header.

Each unique relative orientation of contacting NCS asymmetric units is reported, and the number of occurrences of geometrically equivalent pairs with the same contacts is given in the **Copies** column of the table. Because of round-off errors in computing symmetry matrices, tolerances of 0.1 degrees and 0.1 Å are used to identify equivalent relative orientations.

**Close** dismisses the **Crystal Contacts** dialog. **Help** opens this manual page in a browser window.

## What is a Demo?

A *demo* is a presentation within Chimera. A demo can include several panels, where a *panel* generally consists of:

- a set of operations performed in Chimera (currently only those that can be expressed as [commands](#))
- a block of explanatory text that appears when the set of operations begins

Demos included with Chimera can be started from the **Tools... Demos** menu.

A *demo source file* contains the instructions to Chimera for [running](#) a demo. A demo source file and any associated data files together contain all of the demo information. Simply opening the demo source file in Chimera will start the demo. Any data files opened by the demo should be in the same directory as the source file, unless the source file specifies otherwise. The demo source [file type](#) can be designated by the suffix `.src` (part of the filename) or the prefix **demo:** (not part of the filename).

The [Demo Editor](#) is a graphical interface for creating demo source files. After [Chimera commands](#) and other demo information such as title and explanatory text have been entered into the editor, the resulting demo can be [saved](#) as a source file in XML format. An older (2004), non-XML source file format is also supported for the purposes of [running](#) a demo.

## Running a Demo

A [Chimera demo](#) can be started by opening a [demo source file](#) or a [Chimera web data](#) file that has a demo embedded. Demos included with Chimera can be started from the **Tools... Demos** menu.

At each step in a demo, changes may occur within the main Chimera window and text may be shown in a separate demo dialog. Buttons on the dialog control the demo:

- **Next** - proceed to the next panel
- **Back** - return to the previous panel
- **Close** - close the demo dialog and all models that were not already open when the demo was started (closing all models before starting a demo is recommended, however)
- **Help** - open this manual page in a browser window

The models in the demo can be [moved](#) as desired; changes in position and scale are automatically corrected for by the demo mechanism.

Within a demo, text shown as a link may be a standard link to a web page (URL) or may connect to a set of commands or code that will execute when the text is clicked. Right-clicking on the link text brings up a context menu to:

- show the URL, command, or code in the [status line](#) and [Reply Log](#)
- open the URL or execute the command or code (same effect as clicking the link text)
- paste the command in the [Command Line](#)

In addition, a linked URL will be shown in a balloon when the cursor is hovered over the link text.

## Demo dialog menu:

### File

- **Save Demo As...** bring up a [dialog](#) for saving the [source file](#) of the current demo. Unless a separate [location](#) was specified by the demo creator, any associated data files will be saved in the same directory as the source file.
- **Close Demo** - close the demo dialog and all models that were not already open when the demo was started
- **Open in Editor** - open the [source file](#) of the current demo in the [Demo Editor](#)

### Controls

- **Auto** - whether to advance automatically through the panels (with a [time delay](#) specified by the demo creator)
- **Loop** - whether the last panel should wrap around to the first
- **Next** - proceed to the next panel
- **Back** - return to the previous panel



## Creating a Demo

The **Demo Editor** is a graphical interface for creating [demo source files](#). After [Chimera commands](#) and other demo information such as title and explanatory text have been entered into the editor, the resulting demo can be [saved](#) as a source file in XML format. A demo can be created from scratch or by modifying an existing source file.

There are [several ways to start](#) the **Demo Editor**, including by choosing **Tools... Demos... Demo Editor** from the menu. This brings up the editor in an empty state. To have it show the contents of an existing demo, start [running](#) that demo and then choose **File... Open in Editor** from the [demo dialog menu](#).

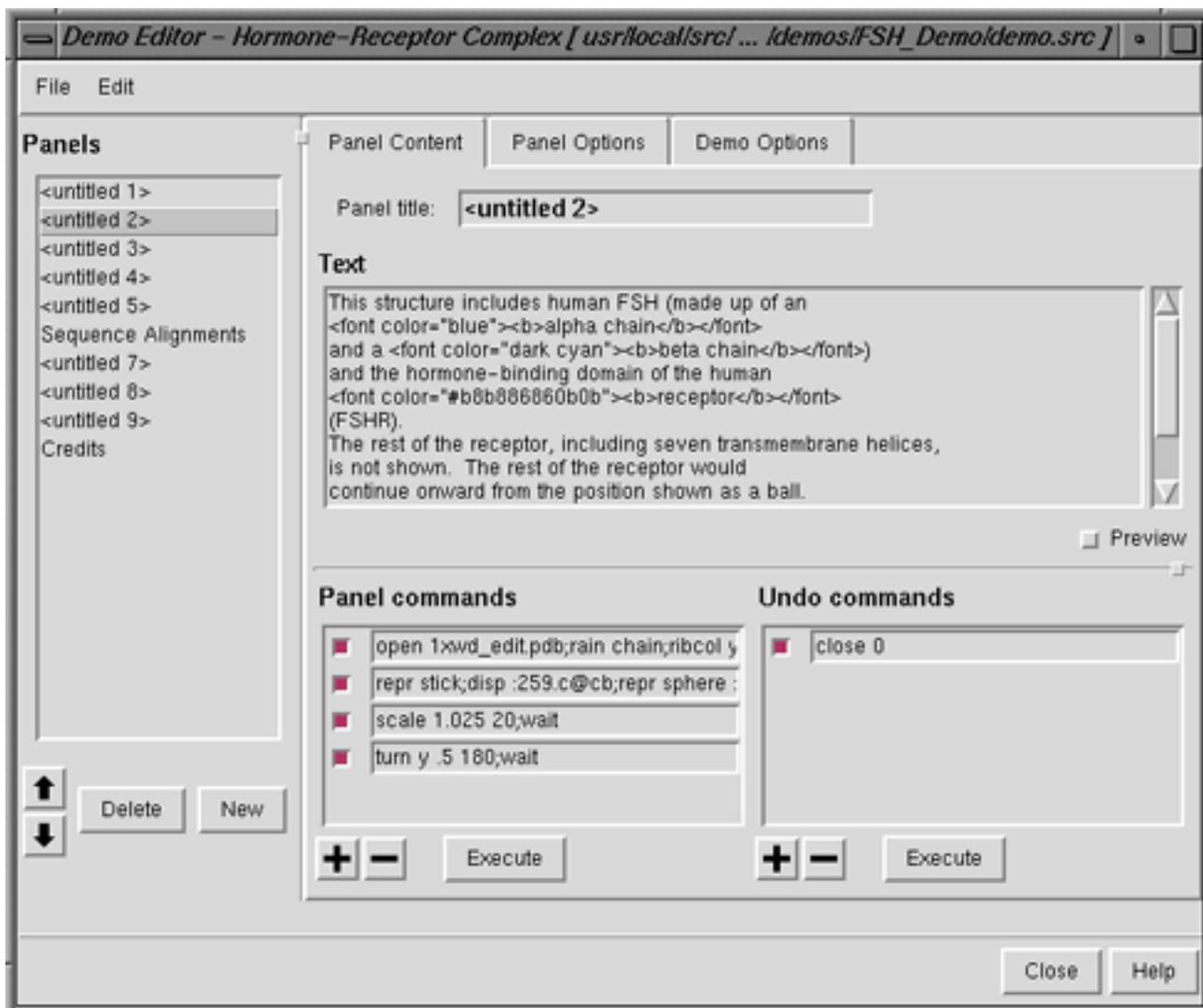
The **Demo Editor** has its own [menu](#), as well as the following sections:

- [Panels](#)
- [Panel Content](#)
- [Panel Options](#)
- [Demo Options](#)

The **Panels** section of the [Demo Editor](#) lists the [panels](#) in the demo. Clicking the name of a panel makes it the *current panel*: its name is highlighted and its information shown in the the **Demo Editor**. **New** adds a new panel below the current panel, if any (otherwise at the bottom of the list). **Delete** deletes the current panel. The up and down arrow buttons move the current panel up and down in the list. The order in the

list is the order in which the panels will be shown when the demo is [run](#).

### Demo Editor - Panel Content



The **Panel Content** section of the [Demo Editor](#) specifies the content of an individual [panel](#):

- **Panel title** (optional) - title for a given [panel](#); overrides any overall [demo title](#)
- **Text** (optional) - explanatory text for a given [panel](#); the use of several [HTML](#) tags is supported within this section:
  - headers (but h1, h2, and h3 give the same result)
  - **bold**, *italic*, and underlined text
  - font **color** specifications (using a [Tk color code](#) or [Chimera color name](#))
  - relative and absolute font size specifications
  - paragraphs and line breaks
  - ordered and unordered lists
  - preformatted text
  - links to web pages:
 

```
<a href="http://www.cgl.ucsf.edu/chimera/">Chimera home page</a>
```

In addition, links to Chimera [commands](#) or Python statements to be executed in Chimera can be constructed analogously to links to web pages:

- Chimera commands:
  - <a href="command:open 3fx2">open the structure</a>
  - <a href="command:color red; repr sphere">show as red spheres</a>
- Python: <a href="python:chimera.openModels.open('3fx2', type='pdb')">open the structure with Python</a>

Only single quote marks should be used within a command or statement, to avoid confusion with the double quote marks enclosing the command or statement.

Activating the **Preview** checkbox will show the text with any HTML-type tags interpreted. If there are no tags, the text will look exactly the same.

- **Panel commands** (optional) - [commands](#) for a given [panel](#) (see [rules and tips](#)). The plus (+) and minus (-) sign buttons add and delete entry fields for commands. If the cursor is in a particular panel command field, + will add another field below that field and - will delete that field. If the cursor is not in a panel command field, + will add a field at the bottom of the list and - will delete the field at the bottom of the list. The checkbox next to each field controls whether its contents will be executed when the panel is played. If the checkbox is deactivated, the information will still be present (if the demo is [saved](#), for example) but the contents will not be executed. **Execute** runs the panel commands with active checkboxes.
- **Undo commands** (optional) - [commands](#) that undo the [panel commands](#) (see [rules and tips](#)). The plus (+) and minus (-) sign buttons add and delete entry fields for commands. If the cursor is in a particular undo command field, + will add another field below that field and - will delete that field. If the cursor is not in an undo command field, + will add a field at the bottom of the list and - will delete the field at the bottom of the list. If undo commands are present, the demo dialog **Back** button will be enabled for the panel. The first panel is an exception (the **Back** button will not be enabled), but in that case, **Close** can be used to close the demo. The checkbox next to each field controls whether its contents will be executed when the panel's **Back** button is used. If the checkbox is deactivated, the information will still be present (if the demo is [saved](#), for example) but the contents will not be executed. **Execute** runs the undo commands with active checkboxes, but unlike the demo **Back** button, does not [regenerate the position](#) due to the commands of the *previous panel*.

Chimera demos include a built-in mechanism for resetting positions to correct for any manipulations by the user. Clicking **Next** regenerates the position directly due to the commands of the *current panel* (in case the user had subsequently moved anything) before executing the commands of the following one. Clicking **Back** executes any undo commands and then regenerates the position due to the commands of the *previous panel*. The demo developer is responsible for determining which commands are needed to undo all operations other than positioning/scaling. If the commands for the panel only change positioning and/or scaling, an empty entry in the **Undo commands** section can be used to enable the **Back** button.

The **Panel Options** section of the [Demo Editor](#) includes properties of an individual [panel](#):

- **Panel delay** - the time to pause in seconds after executing the commands in the panel when in [Auto](#) mode; overrides any [default delay](#)

The **Demo Options** section of the [Demo Editor](#) includes general properties of the [demo](#):

- **Demo title** (optional) - the title that will be shown on the demo dialog for any [panel](#) that lacks its

[own title](#)

- **Data directory** (optional) - where to look for data files opened by the demo; it is usually best to leave this blank and place any data files in the same directory as the [demo source file](#).
- **Default delay** - the time to pause in seconds (default 5) after executing the commands for a [panel](#) when in **Auto** mode; overridden by any individual [panel delay](#)
- **AutoRun on start** (true/false) - whether the the demo should start in **Auto** mode
- **Image** (optional) - a file containing an image to be shown on the demo dialog; possible formats are bitmap (\*.bmp), GIF (\*.gif), JPEG (\*.jpg), PNG (\*.png), and TIFF (\*.tif). The unscaled image will be placed in the upper left; dimensions of roughly 100 by 100-250 pixels (width by height) are recommended. If no image is specified, the dialog will show the Chimera logo on a black background.
- **Image background color** (a [color well](#), **No Color** by default) - the color used to fill in the left side of the demo dialog under the specified [image](#). The width of the rectangle of color will match the width of the image. If an image is specified but the background color is not, a light gray will be used (the same color as the rest of the demo dialog). If no image is specified, the dialog will show the Chimera logo on a black background.

For a demo to be listed in the **Tools... Demos** menu, there must be a corresponding directory under **share/Demo/demos/**. The directory should include the demo's [source file](#) (named **demo.src**) and any associated data files. For example, the file **share/Demo/demos/COX\_Demo/demo.src** contains the instructions for the demo that is started by choosing **Tools... Demos... Cyclooxygenase Demo** from the menu. The name that appears in the menu is set within the file **share/Demo/ChimeraExtension.py**.

**Demo Editor menu:****File**

- **Open Demo File...** bring up a [dialog](#) for opening an existing [demo source file](#) to be viewed or modified
- **Run Demo** - run the demo according to the current contents of the **Demo Editor**
- **Save Demo File** - save a [demo source file](#) representing the current contents of the **Demo Editor** to a pre-existing name/location
- **Save Demo File As...** save a [demo source file](#) representing the current contents of the **Demo Editor** to a new name/location

(**Save Demo File (As)** does not save any associated data files; to save data files along with the [source file](#), use **File... Save Demo As** in the [demo dialog menu](#).)

- **Close Demo File** - remove the contents of the **Demo Editor**
- **Quit Editor** - close the **Demo Editor**

**Edit**

- **Insert Duplicate Panel** - add a new [panel](#) that is a copy of the [current panel](#)



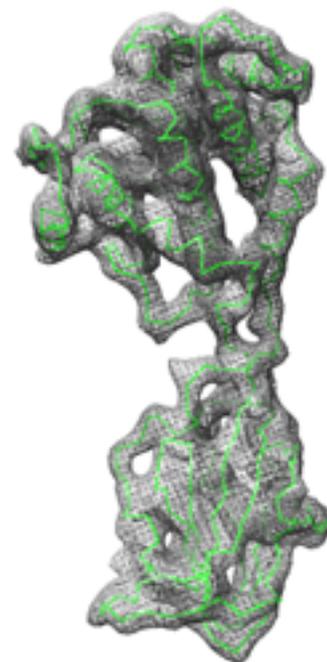
## Fit in Map

**Fit in Map** locally optimizes the fit of atomic coordinates into a density map or one density map into another. The maps usually represent electron density, but other types of [volume data](#) can also be used. See also: [Volume Viewer](#), [Fit to Segments](#), [Values at Atom Positions](#), [MultiFit](#), [measure](#), [molmap](#), [sym](#), [saving maps after fitting](#)

There are [several ways to start Fit in Map](#), a tool in the **Volume Data** category (including from the [Volume Viewer Tools](#) menu). It is also implemented as the command [fitmap](#). However, the command provides some notable features that are not available in **Fit in Map**:

- [global search](#) with random initial placement
- [sequential fitting](#) of multiple different structures
- [symmetric fitting](#) of copies of the same structure

The atomic structures and/or map(s) of interest should first be [opened](#) in Chimera. The *fit model* (atoms or map to fit) and *reference map* can then be chosen from the respective menus in the **Fit in Map** dialog. Since the [optimization](#) is local rather than global, the fit model should be placed in a trial position relative to the reference map before fitting. This usually involves [interactive manipulation](#) and toggling models between [active](#) and immovable states. When atoms are fit, an entire molecule model or just the currently [selected](#) atoms can be used.



Clicking **Fit** performs [local optimization](#) of the fit ([atoms-in-map](#) or [map-in-map](#)) using the current [options](#). The calculation will stop and the [fit model](#) will be repositioned after the earliest of:

- [convergence](#)
- 2000 steps
- when **Halt** is clicked

Clicking **Fit** again may further improve the results, especially if [convergence](#) was not reached. **Undo** backtracks through transformations of the [fit model](#) relative to the [reference map](#) applied by **Fit in Map**, whereas **Redo** reapplies them.

[Find Clashes/Contacts](#) can be used to monitor for close contacts between atomic structures as they are moved, and atomic coordinates can be [saved](#) relative to the [reference map](#).

Clicking **Update** gives the current [atoms-in-map](#) and/or [map-in-map](#) fit values without performing any fitting.

Clicking **Results** opens the [Reply Log](#), which along with the fit values reports the number of steps taken and [fit model](#) displacement (translational shift and angle of rotation) since **Fit** was clicked. The transformation of the [fit model](#) relative to the [reference map](#) is described with a [transformation matrix](#) and as an axis of rotation (a unit vector), point on the axis, degrees of rotation, and shift parallel to the axis.

Clicking **Options** reveals additional settings (clicking the close button  on the right hides them again):

- **Real-time correlation / average update** - update fit values continuously as a model is moved
- **Use map simulated from atoms, resolution [r]** - generate a density map from the [fit model](#) atomic coordinates and use [map-in-map](#) fitting (but report both [atoms-in-map](#) and [map-in-map](#) fit values). The map is generated by describing each atom as a Gaussian distribution of width proportional to  $r$  and amplitude proportional to the atomic number (other parameters set to [molmap](#) defaults).
- **Use only data above contour level from first map** - exclude data outside the [fit map](#) contour surface from [map-in-map](#) fitting
- whether to **Optimize** the [overlap](#) or the [correlation](#) during [map-in-map](#) fitting
- **Correlation calculated about mean data value** - whether the mean values or zero (default) should be used to calculate the [correlation](#) during [map-in-map](#) fitting
- whether to **Allow** [fit model](#) rotation and/or shift movements
- **Move whole molecules** - when fitting selected atoms, whether to move just those atoms or the entire molecule model(s) containing them. Regardless of this setting, only the selected atoms will be used to calculate the fit.

Atoms-in-map fitting:

The average map value at fit atom positions is maximized. For each atom within the bounds of the reference map, the map value is found by trilinear interpolation from the eight corners of the enclosing data grid cell. Atoms outside the bounds of the map are not used for computing averages.

Map-in-map fitting:

Either the overlap or the correlation can be maximized. The calculation can include all nonzero-valued fit map grid points or only those with values above the map's lowest contour level in [Volume Viewer](#) (see [options](#)). The *overlap* is the sum over fit map grid points of the product of the fit map value and the reference map value at that point, determined by trilinear interpolation. It can be expressed as the inner product of vectors  $\mathbf{u}$  and  $\mathbf{v}$  containing the fit map values and the corresponding interpolated reference map values:

$$\text{overlap} = \langle \mathbf{u}, \mathbf{v} \rangle$$

In similar notation,

$$\text{correlation} = \frac{\langle \mathbf{u}, \mathbf{v} \rangle}{|\mathbf{u}| |\mathbf{v}|}$$

or if the [correlation about mean](#) option is turned on,

$$correlation = \frac{\langle \mathbf{u} - \mathbf{u}_{ave}, \mathbf{v} - \mathbf{v}_{ave} \rangle}{|\mathbf{u} - \mathbf{u}_{ave}| |\mathbf{v} - \mathbf{v}_{ave}|}$$

where  $\mathbf{u}_{ave}$  is a vector with all components equal to the average of the components of  $\mathbf{u}$  and  $\mathbf{v}_{ave}$  is defined analogously. The correlation equals the cosine of the angle between the vectors (after subtraction of averages) and can range from -1 to 1, whereas the range of overlap values depends on the scaling of the maps.

**Close** closes the **Fit in Map** dialog. **Help** opens this manual page in a browser window.

## NOTES

**Local optimization algorithm.** If rotation and translation are both allowed, every even step is a translation and every odd step is a rotation. The center of rotation is the geometric center of the fit atoms or fit map grid points, whichever applies. Optimization is by steepest ascent. Map value gradients at atom positions or fit map points are calculated using trilinear interpolation of the gradients at the reference map points. Gradients at grid points are calculated by the center difference method. Atoms or fit map points outside the reference map or within one voxel of the edge of the data at a given step do not contribute to the optimal direction at that step. The initial step size is the largest (default 0.5 grid unit, where a grid unit is the spacing between reference map grid points). If after four steps the maximum cumulative displacement is less than half the displacement achievable if all steps were in the same direction (e.g., half of 2.0 grid units = 1 grid unit), the step size is halved. Successive rounds of four steps with fixed step size and halving the step size based on the maximum displacement criterion are repeated until *convergence* (when step size falls below some minimum, default 0.01 grid unit) or the number of steps reaches some maximum (default 2000). Values other than the defaults can be specified with the [fitmap](#) command.

**Transformation matrices.** The transformation matrix of the fit model relative to the reference map is reported in the [Reply Log](#). The first three columns of the matrix describe a rotation and the fourth describes a translation (performed after the rotation). The transformation is also described as an axis of rotation (a unit vector), point on the axis, degrees of rotation, and shift parallel to the axis.

**Comparing fits.** Because local optima rather than a global optimum are found, it is often beneficial to explore and compare different fits. This can be done by opening multiple copies of the fit model and fitting them from different starting positions. The [Model Panel](#) can be used to hide and show individual copies. Different positions of atomic coordinates can be compared with the command [rmsd](#).

**Clipping and hiding are ignored.** The calculation uses full maps even if partly hidden by clipping planes, zoning (with [Surface Zone](#) or [zoning](#) in [Volume Viewer](#)), or [subregion selection](#) in [Volume Viewer](#).

**Self-correlation  $\neq$  1.** The correlation value of a map with itself in the same position should equal 1. However, in such cases, **Fit in Map** can report values much less than 1 because of floating-point rounding errors in grid point positions. Grid points at the boundary of a map can be found to lie outside the map,

yielding interpolated values of 0.

**Speed.** Fitting 60,000 atoms using 76 steps takes about 15 seconds on a 2 GHz PC system (circa 2005). The time taken is proportional to the number of fit atoms or fit map grid points. [Selecting](#) just the CA atoms of a protein will allow substantially faster calculation than using all atoms. The optimization calculation is written partly in C++ for speed and partly in Python.

---

UCSF Computer Graphics Laboratory / June 2011



## Fit to Segments

**Fit to Segments** rigidly fits atomic structures into [segmentation regions](#) from [Segment Map](#). Along with that tool, it is part of the [Segger package](#) described in:

[Quantitative analysis of cryo-EM density map segmentation by watershed and scale-space filtering, and fitting of structures by alignment to regions](#). Pintilie GD, Zhang J, Goddard TD, Chiu W, Gossard DC. *J Struct Biol*. 2010 Jun;170(3):427-38.

See also: [Segger documentation](#) at NCMI, [Fit in Map](#), [Volume Viewer](#), [MultiFit](#), [fitmap](#), [measure](#), [molmap](#), [mask](#)

There are [several ways to start Fit to Segments](#), a tool in the **Volume Data** category (including from the [Volume Viewer Tools](#) menu).

The **Structure to fit** should be chosen from the pulldown menu of molecule models. Any atoms in the model that are not desired for fitting, such as solvent or extra chains, should be [deleted](#) before the fitting is done.

Clicking **Fit** performs the fitting. The default [settings](#) are to fit the structure to the [selected](#) regions using the principal axes method. The regions are made transparent; region transparency/opacity can be adjusted further using the [Regions menu](#) in [Segment Map](#) or the main Chimera menu [Actions... Surface](#).

Fit information is shown in a table:

- **Corr** - the [correlation](#), also known as the *cross-correlation score*, between the map within the [segmentation region\(s\)](#) used for fitting and the map [generated from the atomic structure](#)
- **Molecule** - the chosen structure
- **Map** - the density map from which the [segmentation region\(s\)](#) were derived
- **Region** - the [segmentation region\(s\)](#) used for fitting

One or more rows (fits) can be chosen with the mouse. The structure is repositioned as each fit is chosen. The chosen fit(s) can be saved to PDB files or removed from the table using the **Fit to Segments** [File menu](#).

Clicking **Options** reveals additional settings (clicking the close button  on the right hides them again):

- **Density map resolution [r] grid spacing [s]** - set parameters for generating a density map from the atomic structure, to allow calculating [correlation](#) values for fits. Each atom is described as a 3D Gaussian distribution of width proportional to the resolution  $r$  and amplitude proportional to the atomic number (as is done by the command [molmap](#)). The grid spacing  $s$  is the separation of points along the X, Y, and Z axes of the generated map. The resolution and spacing should be

approximately the same as for the map into which the structure is being fit (the map that was [segmented](#)). Clicking **Fit** or **Calculate Map** will generate the map from the structure if it does not already exist.

- **Which regions to use for fitting:**
  - **Combined selected regions** (default) - use the [selected](#) segmentation regions collectively
  - **Each selected region** - use each [selected](#) segmentation region for a separate fit
  - **Groups of regions including selected region** - generate groups of segmentation regions that include the [selected](#) one, try fitting to each group; save the best fits in the [table](#)
  - **Groups of regions including all regions** - combinatorially group subsets of all segmentation regions, try fitting to each group; save the best fits in the [table](#)
- **Alignment method:**
  - **Align principal axes (faster)** (default) - align centers, then principal axes from longest to shortest. The center and principal axes of a structure are calculated from its atomic coordinates, non-mass-weighted. The center and principal axes of a region or group of regions are calculated from the voxels they enclose; all grid points inside the surface(s) are weighted equally. The principal axes are the eigenvectors of a covariance matrix. An eigenvector gives the direction of each axis, but the signs of these directions are ambiguous. When performing the alignment, the signs are flipped to generate four possible transforms. Only non-reflecting transforms are considered, in which either none or two of the three axes are flipped. The fit with the highest [correlation](#) is kept.

The principal axes of the atomic structure can be shown/hidden with the [File menu](#) in **Fit to Segments**. The principal axes of segmentation regions can be shown with the [Regions menu](#) in **Segment Map**.

- **Rotational search (more reliable)** - align centers, then exhaustively rotate the structure to find the best fit. The alignment with the highest [correlation](#) is kept.
- **Optimize fits** (on by default) - whether to perform a [local optimization](#) of each fit. Optimization uses density both inside and outside the region(s) used in the initial fit, and can move the structure far from the region(s).

## Fit to Segments Menu

### File

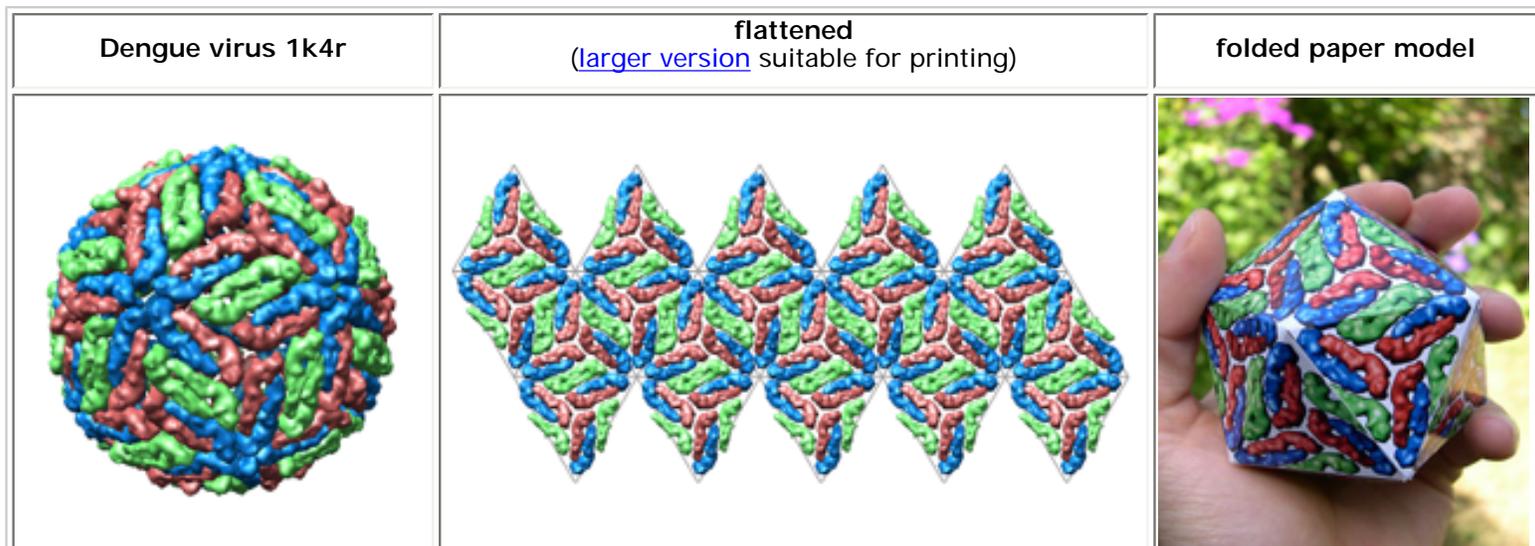
- **Save chosen fit molecules** - save a PDB file of the structure in its fit position for each chosen row in the [table of fits](#)
- **Place molecule copies** - add a copy of the structure in its fit position (as a new model) for each chosen row in the [table of fits](#)
- **Delete fits from list** - remove the chosen row(s) from the [table of fits](#)
- **Show molecule axes** - display the [principal axes](#) of the atomic structure as arrows
- **Hide molecule axes** - hide the [principal axes](#) of the atomic structure
- **Show overlapping regions** - display only the regions that overlap the atomic structure in its

current position (all regions can be displayed again using the [Regions menu](#) in [Segment Map](#))

---

## Flatten Icosahedron Flatten Icosahedron

**Flatten Icosahedron** rearranges the faces of an icosahedral virus capsid from [Multiscale Models](#) into a plane. This flat view can be [saved as an image](#), printed, and folded into a paper icosahedron model.



There are [several ways to start Flatten Icosahedron](#), a tool in the **Higher-Order Structure** category.

The capsid subunit coordinates should first be opened in Chimera and the complete capsid generated with [Multiscale Models](#) using **Icosahedral symmetry, xyz 2-fold axes (VIPER)**. This is done automatically when a structure is [fetched](#) from the [Virus Particle Explorer database](#) (VIPERdb) or a local [VIPER file](#) is read. This tool will not work on structures that are not multiscale models, even if they have the correct type of symmetry. All multiscale models present will be affected.

When **Flatten** is clicked, the twenty triangular faces of an icosahedral capsid are laid out in a plane. The orientations of the faces relative to the structure are assumed based on the [known symmetry](#). Each chain is associated with the triangular face whose center is closest to the geometric center of the chain. To generate the planar arrangement, a common rotation and translation are applied to all the chains of a given face. The chains and faces remain three-dimensional; each face is simply repositioned relative to the others. **Unflatten** reverses the process. **Close** dismisses the **Flatten Icosahedron** dialog, and **Help** opens this manual page in a browser window.

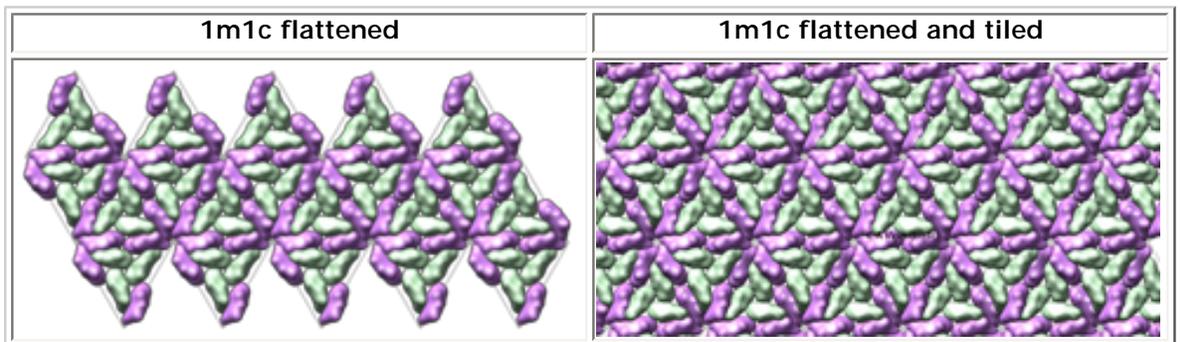
- The **Radius** controls the spacing of faces in the plane. If an ideal tetrahedron were flattened using a radius equal to that of the smallest sphere it fits within, the triangular faces would exactly abut. Larger radii would generate triangles in the flattened state larger than the faces of the original ideal tetrahedron. If the **Radius** field is blank, the largest distance between the capsid center and an atom in the capsid will be used. Changes in the value can be applied by clicking **Flatten** (it is not necessary to unflatten before reflattening).
- **Show triangle mesh** indicates whether the triangles should be outlined in the flattened state. The lines can be used as guides for cutting and folding to produce a paper model. The **color** of the lines can be specified by clicking the [color well](#) and using the [Color Editor](#). The lines can be moved in front of the plane (so that they are not obscured by parts of the structure) by entering a **z offset** value and pressing Enter (return).
- **Use orthographic camera mode** turns off perspective. This is important for generating an undistorted image of the flattened tetrahedron. The [projection mode](#) (perspective or orthographic) can also be controlled with the [Camera tool](#). Further, it is important to keep the plane of the flattened structure normal to the line of sight. If the structure was rotated after flattening, the correct orientation can be restored with the command [reset](#).

Before [saving an image](#), it may be desirable to adjust display parameters, in general (see the [tips on preparing images](#)) and in [Multiscale Models](#). For example, when a multiscale model is first created, the original chains may not be shown as surfaces like the rest of the chains. To remedy this, [select](#) the chains with loaded atoms and change them to the [surface](#)

[style](#) as well.

## NOTES

**Tiling.** Cutting and folding to make a paper model chops away parts of proteins that straddle triangle boundaries. Tiling the image allows parts of a cut protein to appear on both sides of the cut lines when taped together. Tiling is not necessary, but improves the appearance of the final model. Currently, Chimera cannot perform the tiling, so a separate image-editing program must be used. GIMP was used in the example shown here. Steps:



- Create a flat icosahedron image in Chimera with a transparent background. If system hardware permits, background transparency can be enabled in the [Effects tool](#). (Only high-end graphics cards support this feature; if not available, use the image-editing program to make the background transparent instead.)
- In the image-processing program, make four copies of the image and place them around the border of the original image. In GIMP, this is done using "duplicate layer" and the "move layer" mouse mode. The "stacking order" of the image copies may affect the appearance. In this example, the four surrounding copies were placed below the original image. The resulting tiling is not perfect; ideally, along a border the copy should be above the original in some places and below it in others.

**Known icosahedral symmetry.** This tool uses the icosahedral reference frame where 2-fold axes are along the x, y and z axes, called **Icosahedral symmetry, xyz 2-fold axes (VIPER)** in the [Multiscale Models](#) dialog. To use a different reference frame requires a small change to the Python code. Specifically, you would change the '222' in file `share/FlattenIcosahedron/__init__.py` to '222r', or '2n5', or '2n3'. Refer to file `share/Icosahedron/__init__.py` for the definitions of these alternate reference frames.



## Hide Dust

**Hide Dust** hides smaller *blobs* (disconnected parts) of a surface. One use is to simplify the display of noisy [volume data](#). See also: [Volume Filter](#), [Measure and Color Blobs](#)

There are [several ways to start Hide Dust](#), a tool in the **Volume Data** category (including from the [Volume Viewer Tools](#) menu). It is also implemented as the command [sop hideDust](#).

The surface of interest should be displayed and then chosen from the **Surface** menu. Blobs can be measured in various ways; the cutoff value and slider refer to the method specified in the [options](#).

Nothing will happen until **Hide** is clicked. Clicking **Hide** limits the display of the chosen surface using the current settings and initiates automatic updates of that surface as the cutoff or measurement method is changed. The cutoff can be changed by typing a new value or dragging the slider. Changes affect only the currently chosen surface. Choosing another surface from the menu suspends updating of the previously chosen surface, but its cutoff and measurement method will be retained until it is unhidden. It is necessary to click **Hide** again to initiate automatic updating of the newly chosen surface (unless previously initiated). Clicking **Unhide** stops hiding blobs of the currently chosen surface and halts its automatic updates.

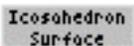
Clicking **Options** reveals the measurement setting (clicking the close button  on the right hides it again):

- **Hide small blobs based on**
  - **size** (default) - bounding box dimension (largest of X, Y, Z)
  - **area** - surface area
  - **volume** - enclosed volume
  - **size rank** - rank by largest **size** (a value of *N* indicates showing the *N* largest blobs by the **size** criterion)
  - **area rank** - rank by largest **area**
  - **volume rank** - rank by largest **volume**

Size measurements will include any blob parts that have been undisplayed or hidden by [zoning](#) or [clipping](#), and blobs at or above the cutoff size will be displayed completely (although possibly still clipped) even if they had been hidden beforehand.

**Close** dismisses the **Hide Dust** dialog; **Help** opens this manual page in a browser window.

## Icosahedron Surface

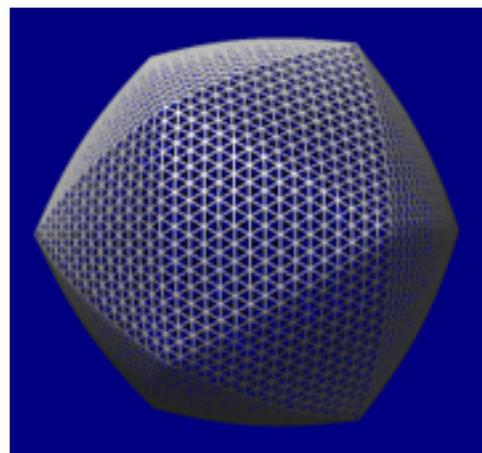


**Icosahedron Surface** creates a surface representing a linear interpolation between an icosahedron and a sphere. The surface is for comparison to virus particles with icosahedral symmetry; it can be colored by density data for such structures with [Surface Color](#).

**Icosahedron Surface** information is saved in [sessions](#). See also: [shape icosahedron](#), [hk cage](#), [meshmol](#), [Cage Builder](#)

There are [several ways to start Icosahedron Surface](#), a tool in the **Higher-Order Structure** category.

Clicking **Show** creates the surface according to the settings:



- **Radius** - radius of the sphere component and distance from the center to a vertex of the icosahedron component (the default is **100** display units, generally Å); the value can be changed by entering a different number or by moving the slider.
- **Sphere factor** - weight to use for the sphere component, where **0** (default) yields an icosahedron and **1** yields a sphere; the value can be changed by entering a different number or by moving the slider. Values outside the range 0-1 can be used, but negative values such that surface points pass through the origin will cause subsequent **Sphere factor** settings to produce the wrong surface.
- **Orientation** - orientation of the icosahedron
  - **xyz 2-fold axes (222)** (default) - with two-fold symmetry axes of the icosahedron along the x, y and z axes. In the [untransformed](#) icosahedron, a *vertically oriented edge* points at the user. (Objects are untransformed when no rotations/translations have been performed, or when the view has been [reset](#) to the default.)
  - **xyz 2-fold axes, alt (222r)** - as above, except rotated 90° about the z axis. In the [untransformed](#) icosahedron, a *horizontally oriented edge* points at the user.
  - **x 2-fold, z 3-fold (2n3)** - with a two-fold symmetry axis of the icosahedron along the x axis and a three-fold symmetry axis along the z axis. The y axis does not coincide with a symmetry axis. A *face* of the [untransformed](#) icosahedron points at the user.
  - **x 2-fold, z 3-fold, alt (2n3r)** - as above, except rotated 180° about the y axis.
  - **x 2-fold, z 5-fold (2n5)** - with a two-fold symmetry axis of the icosahedron along the x axis and a five-fold symmetry axis along the z axis. The y axis does not coincide with a symmetry axis. A *vertex* of the [untransformed](#) icosahedron points at the user.
  - **x 2-fold, z 5-fold, alt (2n5r)** - as above, except rotated 180° about the y axis.
  - **y 2-fold, z 5-fold (n25)** - with a two-fold symmetry axis of the icosahedron along the y axis and a five-fold symmetry axis along the z axis. The x axis does not coincide with a symmetry axis. A *vertex* of the [untransformed](#) icosahedron points at the user.
  - **y 2-fold, z 5-fold, alt (n25r)** - as above, except rotated 180° about the x axis.
- **Subdivision factor** - how finely to subdivide the surface. A factor of **1** (default) yields 20 triangles, the 20 faces of the icosahedron. Values represent the number of subdivisions within an edge of the icosahedron, and only non-negative powers of 2 are used (1, 2, 4, 8, ...). Other numbers can be entered, but will be rounded to a power of 2. The **spacing** is the length of a subdivision.
- **Surface style** - whether the surface should be shown as **solid** or **mesh**
- **Color** - a [color well](#) specifying the surface color
- **Line width** - pixel width of lines in the **mesh style**

Only one “icosahedral” surface can be present at a time. Changing any parameter automatically updates the surface. **Remove** closes this surface model, while **Close** dismisses the **Icosahedron Surface** dialog. **Help** opens this manual page in a browser window.

## TECHNICAL NOTES

**Rounded edges.** Even when the [Sphere factor](#) is zero, the edges of the icosahedron may appear to be rounded because of the way the model is lit. This is especially noticeable in the **solid style**. Each surface vertex has a normal vector that determines how light reflects off the surface. The vertices at the edges between faces of the icosahedron have a normal vector midway between the normals of the faces. This causes the edges to appear rounded instead of sharp. A higher [Subdivision factor](#) (e.g., 32 or 64) will make the edges look sharper.

**Coloring.** The [Surface Color](#) tool assigns each surface vertex a color; between vertices, the colors are interpolated. More detailed color variations will be apparent with a higher [Subdivision factor](#).

---

UCSF Computer Graphics Laboratory / July 2014


[Tools Index](#)
[MD Movie](#)
[Startup and Input](#)
[Viewing Frames](#)
[Recording a Movie](#)
[Per-Frame Scripts](#)
[Plotting](#)
[Occupancy Analysis](#)
[RMSD Analysis](#)
[Clustering](#)
[Structure Averaging](#)
[MD Movie Menu](#)
[Adding a Format](#)


## MD Movie

**MD Movie** is a tool for viewing and analysis of trajectories and other ensembles. For examples of use, see the [Trajectory and Ensemble Analysis tutorial](#). See also: [Morph Conformations](#), [coordset](#)

There are [several ways to start MD Movie](#), a tool in the **MD/Ensemble Analysis** category.

Several formats are supported:

Format	Required Inputs
<a href="#">Amber</a>	<ul style="list-style-type: none"> <li>prmtop (parameter/topology) file, either the <a href="#">older</a> or the newer format (see Appendix E of the <a href="#">Amber 7 Manual</a>)</li> <li>trajectory (coordinates), either <a href="#">formatted</a> or <a href="#">Amber NetCDF</a>, which is binary. Multiple trajectory files can be specified, to be concatenated in the order given. Additional trajectory files can be <a href="#">loaded</a> at a later stage.</li> </ul> <p>Amber residue names are remapped to standard PDB residue names where possible. Partial charges are assigned as the atom <a href="#">attribute</a> named <b>charge</b>. Amber NetCDF support is courtesy of Mingfeng Yang.</p>
<a href="#">CHARMM</a> , <a href="#">NAMD</a> (PSF/DCD), or <a href="#">X-PLOR</a>	<ul style="list-style-type: none"> <li>PSF (protein structure file; X-PLOR-style topology)</li> <li>DCD (binary trajectory). Multiple DCD files can be specified, to be concatenated in the order given. Additional DCD files can be <a href="#">loaded</a> at a later stage.</li> </ul> <p>Partial charges are assigned as the atom <a href="#">attribute</a> named <b>charge</b>. PSF/DCD support is courtesy of <a href="#">MDTools</a>.</p>

<a href="#">NAMD</a> (prmtop/DCD)	<ul style="list-style-type: none"> <li>• prmtop (AMBER-style parameter/topology file)</li> <li>• DCD (binary trajectory). Multiple DCD files can be specified, to be concatenated in the order given. Additional DCD files can be <a href="#">loaded</a> at a later stage.</li> </ul> <p>Partial charges are assigned as the atom <a href="#">attribute</a> named <b>charge</b>. DCD support is courtesy of <a href="#">MDTools</a>.</p>
<a href="#">GROMACS</a>	<ul style="list-style-type: none"> <li>• portable binary <a href="#">run input file</a> (.tpr)</li> <li>• portable <a href="#">trajectory file</a>, binary (.trr) or compressed (.xtc)</li> </ul> <p>XTC/TRR support is courtesy of <a href="#">XTC Library</a>.</p>
<a href="#">GROMOS</a>	<ul style="list-style-type: none"> <li>• topology</li> <li>• coordinates (trajectory)</li> <li>• PROMD (PROMD input file used to generate the trajectory)</li> <li>• a scale factor to convert the coordinates to angstroms (usually 10.0)</li> </ul>
<a href="#">MMTK</a>	<ul style="list-style-type: none"> <li>• <a href="#">NetCDF</a> trajectory</li> </ul>
<a href="#">particle</a>	<ul style="list-style-type: none"> <li>• particle trajectory based on <a href="#">Amber NetCDF</a> (<a href="#">details</a>)</li> </ul>
<b>PDB, single file</b> (coordinates for each frame bracketed by MODEL and ENDMDL records)	The name of the file must be supplied. The MODEL numbers are interpreted as frame numbers and should start with 1 or 0 and increment by 1.
<b>PDB, multiple files</b> (one file per frame)	Starting and ending file names must be supplied. The file names must include frame numbers, and the frames will be ordered by increasing number (not necessarily consecutive; missing numbers will be ignored). The files must not contain END records.
<a href="#">XYZ</a> (multiple files, one per frame)	Starting and ending file names must be supplied. The file names must include frame numbers, and the frames will be ordered by increasing number (not necessarily consecutive; missing numbers will be ignored).

These input files can be gzipped.

The PDB options generally require coordinates for the same set of atoms to be supplied for each frame. However, if the coordinates supplied for a frame represent only a subset of the atoms in the preceding frame, it will be inferred that the remaining atoms are present but have the same coordinates as in the preceding frame. Changes are cumulated in the forward direction and based on all frames, even when frames are skipped during playback ([step size](#) > 1).

All or a contiguous subsegment of a trajectory can be loaded. If **pipe** is entered as the ending frame number, the input trajectory will be read until it runs out, as long as no other input specifies a smaller ending frame number. For example, for a GROMOS trajectory, the NSTLIM

variable in the PROMD file should also be increased to at least the total number of frames expected. When a pipe has been specified, there will be an attempt to load the entire trajectory up front.

A further option is a **metafile**, simply a text file that specifies the input files/parameters. The first line designates trajectory type (where case is unimportant but any spaces should be stripped, for example, **namd(prmtop/dcd)** or **gromos**), and optionally, starting and ending frame numbers of the range to be loaded. A pipe can be specified as described [above](#). If no frame numbers are supplied, the entire trajectory will be loaded; however, if "?" is given for both the starting and ending frame numbers, a dialog for entering this information will appear. The remaining lines specify input files and parameters in the same order as in the dialog. For example:

```
amber 500 1000
dna.prmtop
dna.mdcrd
```

or:

```
pdb
single
108d.pdb
```

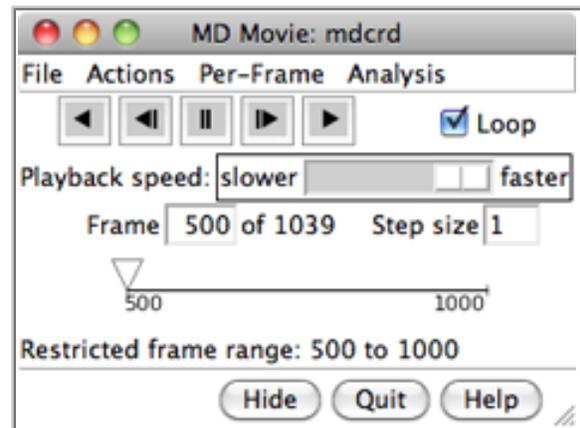
If input files are not in the same directory as the metafile, their pathnames relative to the location of the metafile should be supplied. A metafile can be opened directly from the [Command Line](#) (or the system command line upon [startup](#)) using the prefix **md:** or **movie:**. This will start **MD Movie** and open the trajectory data.

## VIEWING FRAMES

After input files and parameters have been specified, the first set of coordinates will be displayed and the **MD Movie** controller will appear:

If a subsegment was specified, a message about the restricted frame range will be displayed temporarily. The total number of frames in the trajectory (whether loaded or not) will continue to be reported. For all [input formats](#) except PDB and XYZ, the coordinates for a given frame are not read in until that frame is viewed or used for analysis.

From left to right, the buttons mean: play backward continuously; go back one [step](#); stop; go forward one [step](#); and play forward continuously. When the **Loop** option is on, forward play can wrap from the end to the beginning of the loaded trajectory and reverse play can wrap from the beginning to the end. The rate of continuous play can be adjusted with the **Playback speed** slider; up to a 1-second delay can be added between frame advances. When continuous play is not in use, the display can also be controlled by moving the pointer (inverted triangle) on the timeline or by entering a new **Frame** number. The **Step size** controls the level of sampling for continuous playback. For example, a step size of 3 indicates that only every third frame will be shown; however, when forward play loops from end to beginning, the movie will start at the first frame loaded, and when reverse play loops from beginning to end, the movie will start at the last frame loaded. Frame number and step size changes take effect when return (**Enter**) is pressed.



Frames can also be navigated by moving the black vertical line on a [distance or angle plot](#) or by using the command [coordset](#). Pausing any continuous playback with the dialog is recommended before using these other methods.

\*\* Protein secondary structure assignments are not recomputed automatically over the course of a trajectory. If displaying protein as a [ribbon](#), see the [note](#) below. \*\*

The current frame or all frames that have been viewed can be saved as a PDB file with [File... Save PDB](#). The view can be [held steady](#) on [selected](#) atoms. One can define [scripts](#) to be executed at each frame update. A complete menu listing with short descriptions is included [below](#).

**Hide** closes the interface without exiting from **MD Movie**; the interface can be reopened using the **Tools** menu entry for the [instance](#) of **MD Movie**. **Quit** exits from **MD Movie** and removes the structure from the Chimera window. **Help** opens this manual page in a browser window.

## RECORDING A MOVIE

Images can be captured during trajectory playback and automatically assembled into a movie file. Manipulations in Chimera can be performed and [per-frame scripts](#) executed during the playback/image-saving process. See also: [coordset](#), [making movies](#)

### Important notes:

- Protein secondary structure assignments are not recomputed automatically over the course of a trajectory. If a protein or peptide is displayed as a [ribbon](#) with secondary-structure-specific scaling (width and height) and significant conformational changes are occurring, users may want to reassign secondary structure at each frame. This can be done with a [per-frame script](#) that includes the Chimera command [ksdssp](#).
- For a one-to-one relationship between trajectory frames and recorded image frames, the [Playback speed](#) slider should be moved to the far right. Otherwise, identical image frames will be inserted between the unique data frames to slow playback. The frame number parameters described below refer only to the unique data frames. See also: [frame rate and movie speed](#)
- On certain platforms, other windows should not overlap the Chimera graphics window during recording ([details...](#)). This is not an issue when [raytracing](#) is used.

[File... Record movie](#) opens an interface for specifying image capture and movie assembly parameters:

- **File name** - name for the resulting movie file
- **File type** (movie format) choices:
  - H.264 [.mp4]
  - VP8/WebM [.webm]
  - Theora [.ogv]
  - Quicktime [.mov]
  - AVI MSMPEG-4v2 [.avi]
  - MPEG-4 [.mp4]
  - MPEG-2 [.mpg]
  - MPEG-1 [.mpg]
  - WMV2 [.wmv]
- **Starting frame** (first frame loaded, by default) - number of the first frame at which to save an image

- **Step size [n]** - level of sampling within the indicated range (every  $n^{\text{th}}$  frame will be used)
- **Ending frame** (last frame loaded, by default) - number of the last frame to possibly include (if not skipped by sampling); must be within the range originally loaded
- **Encode forward then backward ("roundtrip")** (true/false) - whether to include the frames in reverse order as the second half of the movie
- **Rendering:**
  - **Chimera** (default) - Chimera rendering, normally offscreen ([details...](#)). Images can be supersampled, that is, initially generated at a higher resolution and then sampled down to the final size.
    - **Supersample** (1x1/2x2/3x3/4x4) - how many pixels to sample in the X and Y dimensions for each pixel in the final saved image; thus, 1x1 corresponds to no supersampling. Higher values increase the smoothness of edges in saved images and increase calculation time with little effect on file size. 3x3 is generally recommended when supersampling is done.
  - **POV-Ray** - [raytrace with POV-Ray](#). This rendering option is the slowest but includes fancier effects such as high-quality shadows. The **POV-Ray Options** button opens the corresponding [preferences](#).

#### Advanced Options:

- **Quality** (highest/higher/high/**good**/medium/fair/low) - higher quality corresponds to higher (variable) playback bit rates and a larger movie file, assuming the same window size and movie frame rate
- **Image format** - image file format (however, PNG will be used regardless of this setting if [raytracing](#) is done):
  - JPEG [.jpeg]
  - PNG [.png]
  - PPM [.ppm] (default)
- **Additional recording options** - options are the same as for the command [movie record](#); if this field is left blank, image frames will be saved with default names in a default location (but normally deleted after movie encoding, depending on the encoding options)
- **Additional encoding options** - options are the same as for the command [movie encode](#); if this field is left blank, the movie will be encoded to play at 25 frames per second, and image frames will be deleted after the movie has been encoded

Clicking **Record** plays the trajectory and saves image frames according to the specified starting and ending points and step values. The series of images is encoded as a movie file. One duplicate image is included at the end to avoid a "motion blur" at the end of the movie. Stopping playback with the [MD Movie controller](#) in the middle of the playback/image-saving process will abort recording without generating a movie file.

**Close** closes the dialog without initiating recording. **Image Tips** shows the [tips on preparing images](#), and **Help** opens this manual page in a browser window.

#### PER-FRAME SCRIPTS

[Per-Frame... Define script](#) allows specification of a script to be executed at each trajectory frame. Scripts can be written in [Chimera commands](#) or Python code, and can incorporate trajectory frame numbers. Python scripts can also access the molecule model instance.

Examples:

- the following Chimera command script ([eachFrame.com](#)) identifies hydrogen bonds at each frame and shows only the residues involved:

```
chain @ca
~disp solvent
hb line 2 color yellow reveal true
```

Commands to be executed only at a specific frame can be indicated with a leading *#frame\_num*: (for example, #50:) that will be stripped before the command is executed.

- the following Python script ([specificFrames.py](#)) initiates events at specific frames:

```
from chimera import runCommand
frame = mdInfo['frame']
if frame == mdInfo['startFrame']:
    runCommand("roll y 3 40")
elif frame == 10:
    runCommand("color green")
elif frame == 50:
    runCommand("color orange")
    runCommand("rl :ala; color dodger blue :ala; rep stick :ala")
elif frame == 90:
    runCommand("~rl; rep wire :ala")
else:
    runCommand("color byhet")
```

For Python scripts, the chimera module is automatically imported.

**Note:** Frame arguments in commands such as [roll](#) refer to *image frames* rather than to unique data (trajectory) frames. For a one-to-one correspondence between image and trajectory frames, the [Playback speed](#) slider must be positioned all the way to the right and the viewing step size set to 1. Whereas an **MD Movie** per-frame script executes at each trajectory frame, commands to be executed at each image frame (independent of any trajectory) can be specified with the [perframe](#) command. The rate at which Chimera draws image frames can be controlled with [set maxFrameRate](#).

**Insert text file** allows browsing to a text file and placing its contents in the script area. **Save to file** saves the current contents of the script area to a text file.

**OK** and **Apply** execute the script with and without closing the dialog, respectively. If the movie is playing, the script will continue to be executed for each trajectory frame until [Stop running script](#) is chosen; if the movie is halted on a single frame, the script will be executed for that frame and will not be executed again until a different frame is shown. **Clear** deletes the contents of the script area. **Close** closes the dialog without executing the script, and **Help** opens this manual page in a browser window.

## PLOTTING

[Analysis... Plot](#) allows plotting values of structure measurements versus frame number:

- **Distances**
- **Angles** (each defined by three atoms, not necessarily bonded)
- **Dihedrals** (each defined by four atoms, not necessarily bonded)

- **RMSD** (root-mean-square deviation of specified atoms relative to a reference frame)

For distances and angles, the appropriate number of atoms (two, three, or four) should be [selected](#).

RMSD calculations require specifying the atoms of interest and the reference frame number. As in [all-by-all RMSD analysis](#), least-squares-fit RMSD values are calculated without applying any transformation. The atoms can be specified by [selection](#) (where lack of selection indicates all atoms) in combination with the options:

- **Ignore solvent and non-metal ions (true/false)** - whether to ignore [solvent](#) residues and monatomic anions (Cl<sup>-</sup>, Br<sup>-</sup>, etc.)
- **Ignore hydrogens (true/false)**
- **Ignore metal ions (alkali/true/false)** - whether to ignore alkali metal ions (Li<sup>+</sup>, Na<sup>+</sup>, K<sup>+</sup>, etc.), or all metal ions (if **true**), or none (if **false**, thus including any metal ions)

Clicking **Plot** plots the measurement value versus trajectory frame number.

If frames have not yet been viewed/loaded, a dialog will appear with the choices: load all frames (in the range specified in the initial input), load every  $n^{\text{th}}$  frame, or simply wait until a frame is viewed to load it. Loading via this dialog is faster than viewing/playing the whole trajectory. If not all frames are loaded at this point, the plot will be filled in with additional values as the trajectory is viewed.

All measurements of the same type (for example, distances) are placed in a single plot. The measurements are listed in a table to the right of the plot, with columns:

- line color (a [color well](#))
- whether **Shown** on the plot
- measurement value at the current frame
- for distances/angles: names of the atoms defining the measurement
- for RMSDs: reference frame and number of atoms (per frame) used in the calculation

One or more rows (measurements) can be chosen in the table by clicking and dragging; the corresponding atoms will be [selected](#). **Ctrl**-click toggles the status of a single row. Clicking **Delete** removes the chosen measurements from the table and plot.

Plots can be hidden individually using the  button above the table or collectively by closing the dialog window. In either case, the plots have not been deleted and can be shown again using the **MD Movie** [Analysis... Plot](#) menu.

The current frame is indicated with a vertical black line on the plot. Clicking elsewhere on the plot will reposition the line and jump to the corresponding frame, and playback can be controlled directly by dragging the line. Pausing any [continuous playback](#) is recommended to facilitate such manual control over frame viewing.

Below each plot are standard [plot navigation icons](#) provided by [matplotlib](#). Images saved via

the icon  will not include the vertical black line. The **Dump Values** button at the bottom of the dialog allows saving measurements to a text file, using the number of decimal places specified in [Structure Measurements](#).

## OCCUPANCY ANALYSIS

It may be interesting to see which regions of space are highly populated by certain atoms relative to others in the trajectory or ensemble. For example, cations or water hydrogens may tend to occupy space around a negatively charged solute group. Occupancies can be represented as a three-dimensional grid of values, or *volume data*. [Analysis... Calculate occupancy](#) can be used to generate such data and display it with [Volume Viewer](#). The occupancy map can be [saved to a file](#) and later reopened in Chimera.

The occupancy values are simply counts of how many times an atom in the specified set falls within a grid cell, except that the [weighted-sphere option](#) may award increments of >1 per atom. Both [sphere options](#) may increment multiple grid cells per atom.

Usually, one should first define a reference set of atoms to hold steady (the trajectory frames will be transformed to keep these atoms in the same place and orientation, as much as possible). This is accomplished by [selecting](#) the desired reference atoms and then choosing [Actions... Hold selection steady](#). When the [selection](#) is later changed, the "hold steady" atoms will not change unless [Actions... Hold selection steady](#) is used again.

Next, one should [select](#) the atoms for which occupancy data will be collected. The atoms in the [selection](#) can be combined into a single set of occupancy data, or [segregated](#) by [atom type](#).

- **Starting frame** (first frame loaded, by default) - number of the first frame to include in occupancy calculations; must be within the range originally loaded
- **Step size [n]** - level of sampling within the indicated range (every  $n^{\text{th}}$  frame will be used)
- **Ending frame** (last frame loaded, by default) - number of the last frame to possibly include (if not skipped by sampling); must be within the range originally loaded
- **Limit data collection to within cutoff distance of any "held steady" atoms (true/false)** - whether data should be collected only within the [cutoff](#) distance of any atom in the prior [hold steady](#) specification. This option is ignored if there was no [hold steady](#) specification.
- **Cutoff distance (10.0 by default)** - Å distance used to [limit the volume data region](#) to a zone around atoms in a prior [hold steady](#) specification
- **Volume grid spacing (default 1.0 Å)** - resolution of the resulting volume data grid
- **Volume data name** - name to use for the resulting volume data set(s); the name(s) will be listed in the [Volume Viewer](#) interface. For [separate data sets](#) collected for different [atom types](#), the atom type symbol is enclosed in brackets and appended to the name.
- **Collect data separately for each atom type in selection (true/false)** - whether to make a separate set of volume data for each [atom type](#) in the [selection](#)
- **Treat atoms as** - whether to consider atomic [VDW radii](#) in occupancy calculations (the larger the [grid spacing](#) relative to the radii, the smaller the differences in results among the options)
  - **points** (default) - increment the count for each grid cell containing an atom center
  - **uniform VDW spheres** - increment the count for each grid cell with any part inside the [VDW radius](#) of an atom
  - **center-weighted VDW spheres** - increment the count for each grid cell with any part inside the [VDW radius](#) of an atom, but award cells in the outermost layer an increment of 1, the next layer inward an increment of 2, the next layer 3, etc. A grid cell is in the outmost layer if it overlaps the atom sphere but any adjacent grid cell (including those diagonally adjacent) does not.

**OK** collects occupancy data for the [selected](#) atoms according to the specified starting and

ending points and step values, with trajectory frames transformed according to a prior [hold steady](#) specification, if any. The resulting data grid(s) are displayed with [Volume Viewer](#). The dimensions of an occupancy grid will be the smallest needed to enclose any nonzero values (which could be smaller than the [region of data collection](#)). The occupancy map can be saved to a file using [File... Save map as](#) in the [Volume Viewer](#) menu.

**Close** closes the dialog without initiating the calculation. **Help** opens this manual page in a browser window.

## RMSD ANALYSIS

[Analysis... RMSD map](#) can be used to generate a map of all-by-all pairwise root-mean-square deviations (RMSDs) among specified frames. See also: [plotting RMSD](#) (vs. a single reference frame), [clustering](#) a trajectory, [Ensemble Match](#), [match](#)

- **Starting frame** (first frame loaded, by default) - number of the first frame to include in RMSD calculations; must be within the range originally loaded
- **Step size [n]** - level of sampling within the indicated range (every  $n^{\text{th}}$  frame will be used)
- **Ending frame** (last frame loaded, by default) - number of the last frame to possibly include (if not skipped by sampling); must be within the range originally loaded
- **Lower RMSD threshold (white)** (0.5 by default) - initial lower RMSD threshold for coloring, in Å; white will be used for this value and lower. If [auto-recoloring](#) is turned on, the threshold may change after all of the values have been computed.
- **Upper RMSD threshold (black)** (3.0 by default) - initial upper RMSD threshold for coloring, in Å; black will be used for this value and higher. RMSD values between the thresholds will be mapped to grayscale. If [auto-recoloring](#) is turned on, the threshold may change after all of the values have been computed.
- **Restrict map to current selection, if any (true/false)** - whether to use only the [selected](#) atoms, when a [selection](#) exists
- **Ignore solvent and non-metal ions (true/false)** - whether to ignore [solvent](#) residues and monatomic anions (Cl<sup>-</sup>, Br<sup>-</sup>, etc.)
- **Ignore hydrogens (true/false)**
- **Ignore metal ions (alkali/true/false)** - whether to ignore alkali metal ions (Li<sup>+</sup>, Na<sup>+</sup>, K<sup>+</sup>, etc.), or all metal ions (if **true**), or none (if **false**, thus including any metal ions)
- **Auto-recolor for contrast (true/false)** - whether the RMSD map should be recolored automatically after all values have been computed, using [thresholds](#) that enclose the middle third of values (*i.e.*, the lowest third of values will be white, the middle third in grayscale, the highest third black)

**OK** and **Apply** initiate the calculations with and without closing the dialog, respectively, while **Close** simply dismisses the dialog. **Help** opens this manual page in a browser window.

For each frame-to-frame comparison, the least-squares-fit RMSD between the indicated sets of atoms will be calculated, without applying any transformation. The values will be shown as grayscale squares within an *RMSD map*. The calculations may require additional frames to be read (frames within the loaded range are not actually read until viewed or otherwise used) and may take several minutes, depending on the size of the system and the number of frames. The

**Abort** button at the bottom of the RMSD map dialog allows termination of a calculation in progress. To decrease computational time, use a sparser sampling of frames (larger step size) and/or fewer atoms in the calculation.

Multiple RMSD maps can be open at the same time. A given map can be recolored (without recalculation of RMSD values) using the map's menu option **RMSD... Change thresholds** to adjust the white/black [threshold settings](#). When the cursor is placed over a map, the corresponding frame numbers and RMSD value are given below the map. Clicking on a map places the corresponding frame numbers in the **Frame** fields; clicking **Go** will show the corresponding frame of the trajectory in the graphics window.

Below the plot are standard [navigation icons](#) provided by [matplotlib](#).

Other buttons on the RMSD map dialog:

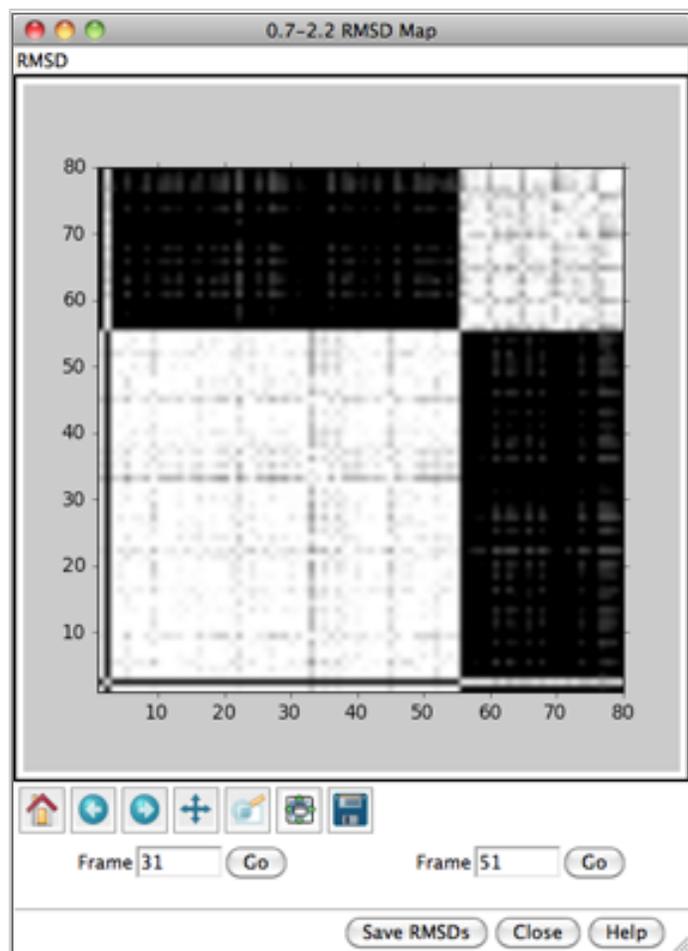
- **Save RMSDs** - save the matrix of RMSD values to a text file
- **Close** - dismiss/delete the RMSD map
- **Help** - show this manual page in a browser window

Alternatively, the [Ensemble Match](#) tool can be used to calculate all-by-all pairwise RMSD values for an ensemble read from a single PDB file. The file would need to be [opened in a standard way](#) rather than with MD Movie. Unlike MD Movie, [Ensemble Match](#) can perform the corresponding pairwise superpositions (*i.e.* match one structure to another), but its use is impractical for ensembles with very many members.

## CLUSTERING

[Analysis... Cluster](#) can be used to cluster the trajectory based on pairwise best-fit root-mean-square deviations (RMSDs). A representative frame will be identified for each cluster. For more about the method, see [Ensemble Cluster](#). See also: [RMSD analysis](#)

- **Starting frame** (first frame loaded, by default) - number of the first frame to include in clustering calculations; must be within the range originally loaded
- **Step size [n]** - level of sampling within the indicated range (every  $n^{\text{th}}$  frame will be used)
- **Ending frame** (last frame loaded, by default) - number of the last frame to possibly include (if not skipped by sampling); must be within the range originally loaded
- **Cluster based on current selection, if any (true/false)** - whether to limit calculations to only the [selected](#) atoms, when a [selection](#) exists
- **Ignore solvent and non-metal ions (true/false)** - whether to ignore [solvent](#) residues and monatomic anions (Cl<sup>-</sup>, Br<sup>-</sup>, etc.)



- **Ignore hydrogens** (true/false)
- **Ignore metal ions** (alkali/true/false) - whether to ignore alkali metal ions (Li<sup>+</sup>, Na<sup>+</sup>, K<sup>+</sup>, etc.), or all metal ions (if **true**), or none (if **false**, thus including any metal ions)

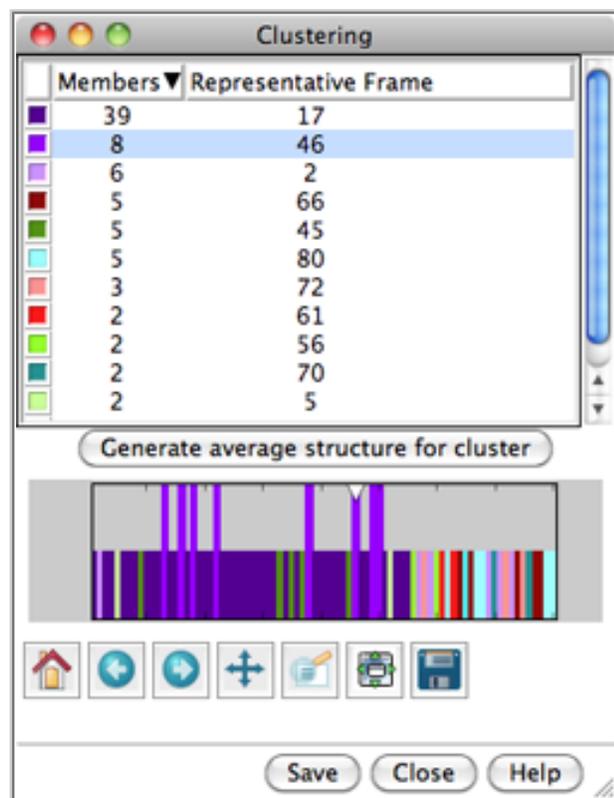
**OK** and **Apply** initiate the calculations with and without closing the dialog, respectively, whereas **Close** simply dismisses the dialog. **Help** opens this manual page in a browser window.

For each frame-to-frame comparison, the least-squares-fit RMSD between the indicated sets of atoms will be calculated, without applying any transformation. The calculations may require additional frames to be read (frames within the loaded range are not actually read until viewed or otherwise used) and may take several minutes, depending on the size of the system and the number of frames. The **Abort** button on the progress dialog allows terminating the calculation. To decrease computational time, use a sparser sampling of frames and/or fewer atoms in the calculation.

### Trajectory Cluster Dialog

Clustering results are shown in a dialog. The top part of the dialog lists the clusters, and for each, the number of **Members** (how many of the input frames belong to the cluster) and the frame number of the best representative. Different colors are used to show membership in the different clusters within the timeline plot in the bottom part of the dialog. The color used for a cluster can be changed by clicking its [color well](#) and using the [Color Editor](#).

Clicking the line for a cluster in the top part of the dialog displays its representative frame in the main window and makes the bars for all of the members of that cluster taller on the timeline plot. The current frame is indicated with an inverted triangle on the plot, and the trajectory can be navigated by moving the triangle, just as in the [MD Movie controller dialog](#). Although more than one cluster can be chosen with the mouse in the top part of the dialog, only one frame can be displayed at a time.



An [average structure](#) for a cluster can be calculated by choosing the cluster in the top part of the dialog and clicking **Generate average structure for cluster**.

Below the plot are standard [navigation icons](#) provided by [matplotlib](#).

**Save** allows writing the clustering information (membership and representatives) to a text file. **Close** dismisses the cluster dialog. **Help** opens this manual page in a browser window.

### STRUCTURE AVERAGING

An average structure can be computed for the whole trajectory or user-specified frame ranges with [Analysis... Average structure](#) in the **MD Movie** menu, or for a chosen cluster by clicking the **Generate average structure...** button in the [cluster dialog](#).

A simple average of the Cartesian (x,y,z) coordinates of the biopolymeric components of the trajectory (proteins/peptides and nucleic acids) is calculated and the result opened as a new model. The average structure may be distorted, especially in conformationally variable areas.

Choosing [Analysis... Average structure](#) opens a dialog with the following options:

- **Starting frame** (first frame loaded, by default) - number of the first frame to include in averaging
- **Step size [n]** - level of sampling within the indicated range (every  $n^{\text{th}}$  frame will be used)
- **Ending frame** (last frame loaded, by default) - number of the last frame to possibly include (if not skipped by sampling); must be within the range originally loaded

The following also apply to cluster averaging:

- **Align structures based on current selection, if any (true/false)** - align the frames to be averaged by matching the [selected](#) atoms
- **Omit hydrogens (true/false)** - whether to omit hydrogens from the average structure
- **Include metal ions (true/non-alkali/false)** - which metal ions to include in the average structure: all, all non-alkali (default, omitting  $\text{Li}^+$ ,  $\text{Na}^+$ ,  $\text{K}^+$ , *etc.*), or none
- **Name average structure (default average)** - name for the new model

Clicking **OK** calculates the average structure and opens it as a new model.

## MD MOVIE MENU

### File

- **Save PDB...** open a [dialog](#) to save the current frame or all frames that have been viewed as a PDB file
- **Record movie...** set parameters for saving image frames and encoding them as a movie file
- **Load Additional Frames...** (only available for Amber trajectories and DCD files, see [formats](#)) read an additional trajectory file and append its frames to the end of the current trajectory

### Actions

- **Hold selection steady** - transform subsequent trajectory frames to keep the [selected](#) atoms in the same place and orientation, as much as possible (since the atoms may move relative to one another). For best results, the selection should include at least three atoms, not linearly arranged, to define a frame of reference. If the selection is changed, **Hold selection steady** must be chosen again to start using the new selection.
- **Stop holding steady** - return to the default behavior of only transforming the coordinates according to user manipulations and commands

### Per-Frame

- **Define script...** open a dialog for entering a [script](#) (written in [Chimera commands](#) or **Python** code) to be executed after each trajectory frame update. **OK** and **Apply** execute the script with and without closing the dialog, respectively. If the movie is playing, the script will continue to be executed for each trajectory frame until **Stop**

running script is chosen.

- **Stop running script** - do not run the script after subsequent trajectory frame updates

## Analysis

- **Calculate occupancy...** collect and display (with [Volume Viewer](#)) occupancy data for the [selected](#) atoms over a specified set of frames ([details](#))
- **Cluster...** cluster the trajectory based on all-by-all RMSD comparisons among specified frames ([details](#))
- **RMSD map...** calculate a two-dimensional RMSD map containing all-by-all comparisons among specified frames ([details](#))
- **Average structure...** calculate a Cartesian average structure over specified frames ([details](#))

## ADDING A FORMAT

MD Movie uses the Trajectory module to read the various formats (*chimera/share/Trajectory*, where *chimera* is the Chimera installation location). **Trajectory** contains a subdirectory, **formats**, which in turn contains subdirectories that each correspond to the Python module for a supported format. By convention, the module (and directory) name for each format is the name of the format with the first letter of each word capitalized and all other letters lowercase. For example, the MMTK module's name is **Mmtk**.

A format's module is typically structured so that the code that interfaces with Trajectory's generic format handling is in `__init__.py`, and the code specific to reading a particular format is in [another python file](#), usually named after the format itself (for example, **Gromos.py**).

`__init__.py` must support the following:

- If the format name displayed to the user should be different from the module name (usually it should, due to capitalization), then there must be a global variable named *formatName* that is initialized to the display name of the format.
- A class named *ParamGUI* must be defined to handle presenting the file-loading interface to the user. It must have two methods:
  - `__init__`, which receives a *Tkinter.Frame* instance argument. The `__init__` method should populate the frame with widgets for gathering the input information for the format from the user.
  - `loadEnsemble`, which takes as arguments a starting frame number, ending frame number, and callback function. `loadEnsemble` needs to compose a list of the arguments that were provided by the user to the widgets defined in `__init__`, and then call this module's global `loadEnsemble` function (see below) with that list as the first argument and the start/end frame number and callback as the remaining three arguments.
- A global `loadEnsemble` function that generates an *ensemble* instance (discussed later). This function is not only called by the *ParamGUI.loadEnsemble* method, but also when a [metafile](#) is used to specify the input parameters. This function takes four arguments: a format-specific list of input parameters, a starting frame number, an ending frame number, and a callback function to start the **MD Movie** interface. The global `loadEnsemble` function should call the interface with the generated ensemble as an argument, and should remember the provided format-specific values as preferred defaults for future uses of the format.

`__init__.py` files are very similar from format to format. The simplest way to generate a new `__init__.py` file is to copy and modify another format's. Gromos format provides a good example, as it involves multiple input files and a non-file parameter.

The **format-specific .py file** defines an *ensemble* class that gets instantiated from `__init__`.  
`py`'s `loadEnsemble` function. The *ensemble* class must support the following methods:

- An `__init__` method that takes the format's input parameters and start/end frames as arguments. The `__init__` method may read input files or do whatever is necessary to support the other instance methods (such as call into a C/C++ module to read the files, as is done for Amber format).
- A `GetDict` method that takes a string argument. The string specifies what data should be returned. The possible string values are:
  - *atomnames* - return a list of the atom names; a residue's atoms must be consecutive
  - *elements* - return a list of the atom elements. These should be instances of `chimera.Element` (which can be initialized with a string such as "Fe" or a number). Trajectory's `determineElementFromMass` function may be useful if the format does not specify the atomic number directly or it cannot be determined easily from the atom name.
  - *resnames* - return a list of the residue names
  - *bonds* - return a list of bonds (2-tuples of indices into the *atomnames* list)
  - *ipres* - a list of the first atom of each residue (indices into *atomnames*, but unlike previous indices these are 1-based, so the first element of *ipres* will always be 1)
- A `__getitem__` method taking a frame-number argument (starting with 1): return a list of 3-tuples corresponding to the xyz coordinates of the atoms in that frame (in the same order as *atomnames*). The coordinates should be in Å.
- A `__len__` method that returns the total number of frames in the trajectory (not just the number of frames between the user-specified start/end frames).

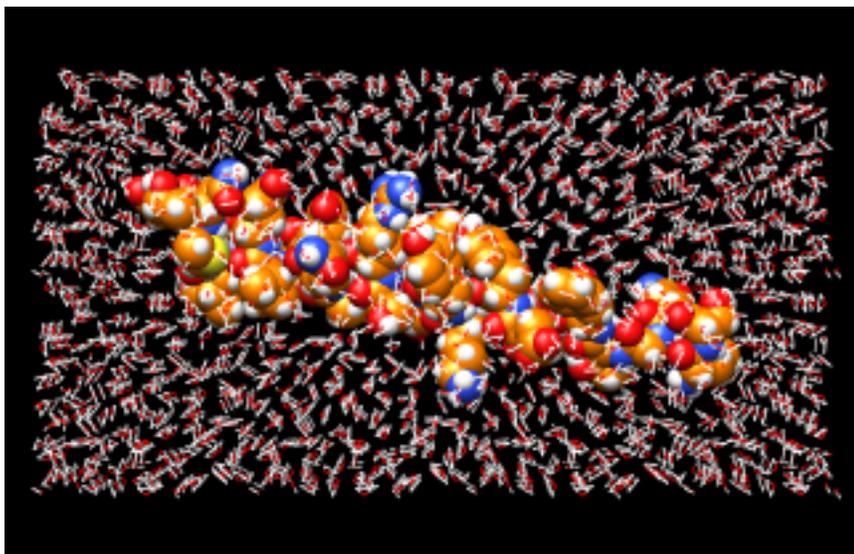
---

UCSF Computer Graphics Laboratory / March 2014

## Solvate Solvate

**Solvate** adds solvent around molecule models using [AmberTools](#). Thanks to Wei Zhang (The University of Texas Health Science Center at Houston) for implementing this tool.

**Solvate** is mainly useful for adding solvent to a system before [writing a prmtop file](#) for [Amber](#). Although [Minimize Structure](#) will use the charges from different water models added with **Solvate**, it will still use the TIP3P model for the remaining water parameters. See also: [Add Ions](#), [Write Prmtop](#)



There are [several ways to start Solvate](#), a tool in the **Structure Editing** and **Amber** categories. It is also implemented as the command [solvate](#).

The model of interest should be chosen from the list. Multiple models can be chosen, but they will be considered individually rather than as a combined system.

Solvent addition requires explicit hydrogens. If the molecule model does not include hydrogens, a dialog for running [AddH](#) beforehand will appear.

### Solvate method options:

- **Box** - rectangular box with edges no closer than **Box size** (Å) to any atom in the solute
- **Cap** - spherical cap with the specified **Cap center** and **Cap Radius** (Å). The center can be:
  - a residue identified by a number which is the residue ID
  - an atom identified by a string like **ddd.xxx**, where **ddd** is the residue ID and **xxx** is the name of the atom
- **Oct** - truncated octahedron with edges no closer than **Oct size** (Å) to any atom in the solute
- **Shell** - a layer of solvent extending **Shell extent** (Å) from the solute. The shell will be irregular in shape since it reflects the contours of the solute.

The **Solvent Model** can be any of the following:

- **CHCL3BOX** - chloroform
- **MEOHBOX** - methanol
- **NMABOX** - N-methylacetamide
- **POL3BOX** - POL3 water model
- **QSPCFWBOX** - qSPC/Fw water model
- **SPCBOX** - SPC/E water model
- **SPCFWBOX** - SPC/Fw water model
- **TIP3PBOX** - TIP3P water model

- **TIP3PFBOX** - TIP3P/F water model
- **TIP4PBOX** - TIP4P water model
- **TIP4PEWBOX** - TIP4P/Ew water model

One can **Remove existing ions/solvent** (recommended) before solvating the system. The affected atoms are those [automatically classified](#) by Chimera as **ions** and **solvent**. Partial charges corresponding to the chosen solvent model will be assigned as the [attribute](#) named **charge** to atoms in the existing (if not removed) and newly added solvent residues. The process of solvation may move (translate) the structure.

**OK** initiates adding solvent and dismisses the panel, while **Apply** adds solvent without dismissing the panel. Addition may take several seconds; progress is reported in the [status line](#). **Close** dismisses the panel without adding any solvent. **Help** brings up this manual page in a browser window.

See the [AmberTools](#) documentation for further details.

## MS and DMS Files

MS and DMS files contain dot molecular surfaces in the format output by the programs *ms* (written by Michael Connolly) and [dms](#) (a reimplementation [available](#) from the UCSF Computer Graphics Lab). This [format](#) is referred to as "DMS" below. Such files can also be created with the Chimera tool [Write DMS](#).

DMS files can be used as input to [sphgen](#), part of the [DOCK](#) suite of programs. See also: [Sphgen sphere files](#)

In Chimera, there are [several ways](#) to open DMS files. File type is indicated with a suffix (`.ms` or `.dms`, part of the filename) or a prefix (`ms:` or `dms:`, not part of the filename).

Although DMS files contain solvent-excluded molecular surfaces, they are read into Chimera as VRML models that will not update to reflect changes in any corresponding atoms. Only a single color, set when the file is opened, is used. However, a DMS surface can be associated with a particular molecule model so that the two cannot be moved or [activated/deactivated](#) for motion independently.

Surface-related Chimera menu items and the command [surface](#) only apply to [MSMS surfaces](#), not VRML surfaces.



Write DMS

**Write DMS** saves part or all of a [molecular surface](#) in Chimera as a [DMS file](#), such as used by the [sphgen](#) module of [DOCK](#). See also: [Sphgen sphere files](#)

Since Chimera's surface calculation method differs from that in the program [dms](#), the dots in DMS files produced in the two different ways will not coincide exactly, but parameters can be set to make the results as similar as possible. For example, the [dms](#) default density setting of 1.0 gives  $\sim 5.0$  dots/Å<sup>2</sup> whereas the Chimera default is 2.0 dots/Å<sup>2</sup>. The vertex density in Chimera can be specified in the [surface](#) command or changed in a pre-existing surface with [setattr](#), for example:

*Command:* [setattr](#) s density 5

Additional ways to change the **vertex density** and related parameters are given [below](#).

There are [several ways to start Write DMS](#), a tool in the **Structure Editing** category. The [MSMS surface](#) of interest should be chosen from the pulldown list. Options:

- **Save normals** - include normal vectors (required by [sphgen](#))
- **Limit output to displayed surface sections** - only write surface vertices that are displayed, which could include vertices that are invisible due to [clipping](#) or [zoning](#), or that simply fall outside the current view. Display of per-atom surface patches can be controlled with the [surface](#) command or

with the [Actions... Surface](#) menu.

**Save** writes a DMS file using the specified name and location and dismisses the dialog, while **Close** simply dismisses the dialog. **Help** opens this manual page in a browser window.

Other possibly relevant surface parameters:

- **probe radius** - the [dms](#) default and Chimera default are both 1.4 Å
- **show disjoint surfaces** - whether to calculate disconnected blobs of surface such as inside bubbles; omitting them simplifies the surface and may be a good idea, unless the binding site of interest is completely enclosed

These parameters can be changed with commands, for example:

*Command:* [setattr](#) s probeRadius 1.5

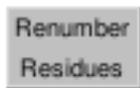
*Command:* [setattr](#) s allComponents false

Along with **vertex density**, these settings can also be specified in the [surface](#) command or changed in a pre-existing surface by using the [MSMS attributes panel](#) or by [selecting](#) the surface and using [Selection Inspector](#). The values to use for subsequently created surfaces can be set in the [New Surfaces preferences](#).

Finally, the [Chimera default radii](#) differ from the [dms default radii](#). There may not be a reason to prefer one set over the other, but if desired, the radii in Chimera can be changed to the *dms* default values by using [Define Attribute](#) to read the assignment file [dmsrad.txt](#).

## Limitation

A limitation of the [DMS files](#) from [Write DMS](#) (not relevant to [sphgen](#) use, however) is that all points are labeled C for contact surface, whereas S should be used for saddle surface and R for reentrant surface. This limitation comes from the [MSMS package](#) that Chimera uses to calculate molecular surfaces, which reports all points as type 3 (contact) although types 1 (toric reentrant) and 2 (spheric reentrant) are also documented.



## Renumber Residues

There are [several ways to start Renumber Residues](#), a tool in the **Structure Editing** category. It is also implemented as the command [resrenumber](#). See also: [Build Structure](#), [Change Chain IDs](#), [modifying and saving data](#)

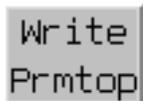
The residues to renumber can be indicated as:

- **selected residues** - all residues containing any [selected](#) atoms
- **chains** - all residues in chain(s) chosen from the list

The indicated residues from *each* chain will be renumbered **starting from** the entered integer. In other words, numbering will start over for each chain rather than incrementing from the previous chain.

Clicking **Apply** or **OK** (which also dismisses the dialog) applies the renumbering, if possible. If the renumbering would produce one or more duplicate residue numbers within a chain, an error message will appear and the numbering will not be applied.

**Close** dismisses the dialog; **Help** opens this manual page in a browser window.



## Write Prmtop

**Write Prmtop** writes [Amber](#) parameter/topology and coordinate files using [AmberTools](#). Thanks to Wei Zhang (The University of Texas Health Science Center at Houston) for implementing this tool. See also: [AddH](#), [Add Charge](#), [Add Ions](#), [Solvate](#)

There are [several ways to start Write Prmtop](#), a tool in the **Amber** category. A [file-saving dialog](#) will appear.

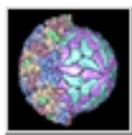
The **Save model** should be chosen from the pulldown list of molecule models.

The **force field type** can be any of the following ([details](#)):

- **AMBER ff99SB**
- **AMBER ff99bsc0**
- **AMBER ff02pol.r1**
- **AMBER ff03ua**
- **AMBER ff03.r1**

[AddH](#) will be called if the model to save lacks hydrogens. [Add Charge](#) will be called as needed to assign charges consistent with the chosen force field type. Two files will be saved: [Amber](#) parameter/topology (.prmtop) and coordinate (.inpcrd).

See the [AmberTools saveAmberParm](#) documentation for further details.


[Tools Index](#)
**Multiscale Models**
[Introduction](#)
[Creating Multiscale Models](#)
[Low-Res Surfaces](#)
[Structural Hierarchy](#)
[Selecting Components](#)
[Display Options](#)
[Loading Atomic Coordinates](#)
**Multiscale Models**


**Multiscale Models** facilitates exploration of macromolecular assemblies, especially large, complicated structures such as viral capsids and ribosomes, by:

- creating multimers from coordinates and matrices in PDB and mmCIF files
- generating low-resolution surfaces for efficient viewing
- allowing navigation within user-defined structural hierarchies

The state of **Multiscale Models** is included in saved [sessions](#). See also: [Unit Cell](#), [Flatten Icosahedron](#), [RMF Viewer](#), [sym](#), [play](#), [measure inertia](#), [msc](#), the [biological unit](#) function in the [Model Panel](#), [fetching](#) PDB-biounit and PQS files, and the following reference:

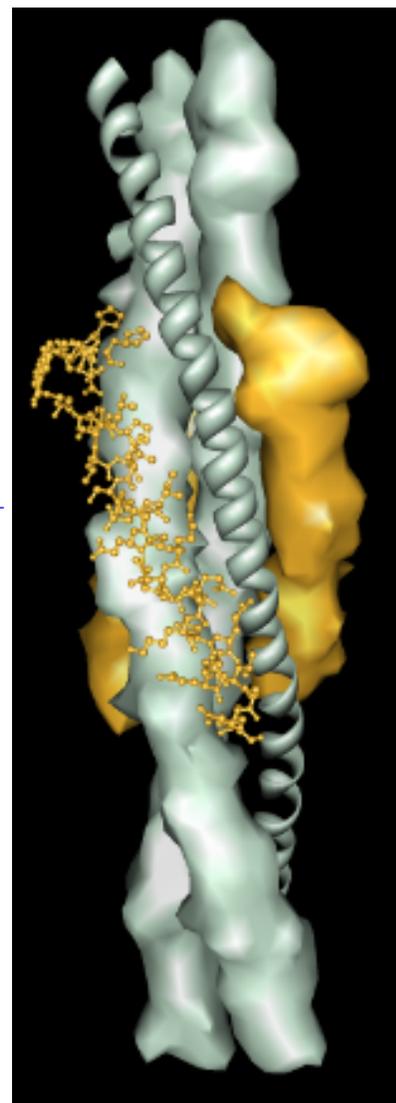
[Software extensions to UCSF chimera for interactive visualization of large molecular assemblies](#). Goddard TD, Huang CC, Ferrin TE. *Structure*. 2005 Mar;13(3):473-82.

There are [several ways to start Multiscale Models](#), a tool in the **Higher-Order Structure** category. When a [VIPER file](#) is read, **Multiscale Models** is automatically started and used to construct the virus capsid.

The dialog has three sections:

- [Select chains](#)
- [Act on selected chains](#)
- [Models from molecules and matrices](#)

Contents are described below in an order reflecting the general work flow.

**CREATING MULTISCALE MODELS**


It is first necessary to [open](#) a file (PDB or mmCIF) containing either the coordinates for an entire multimeric complex (such as a ribosomal subunit) or the coordinates for a lower-order structure and the matrix information needed to generate a multimer. The types of matrices in a file, if any, can be determined by viewing the file directly in a text editor, or once a PDB file has been opened, using the **PDB Headers...** button near the bottom of the [molecule model attributes panel](#). See [Unit Cell](#) for more about symmetry matrices.

Under **Models from molecules and matrices**, the **Multimer** setting should be adjusted based on the types of information present in the file:

- **Biological unit** - uses PDB [BIOMT matrices](#) or the corresponding information in an mmCIF file
- **Crystal unit cell**
- **Crystal unit cell packed** - with chains packed so that their centers fall within one unit cell box
- **3x3x3 crystal unit cells** - 3 by 3 by 3 block of crystal unit cells
- **3x3x3 crystal unit cells packed** - 3 by 3 by 3 block of crystal unit cells, each packed as described above
- **Crystal symmetry in unit cell**
- **Non-crystal symmetry in unit cell**
- **Icosahedral symmetry, xyz 2-fold axes (VIPER)** - use 60 matrices to produce icosahedral symmetry appropriate for structures in the [Virus Particle Explorer](#) coordinate system
- **Icosahedral symmetry, xyz 2-fold axes, alt** - same as the VIPER symmetry, except rotated 90 degrees about the z axis
- **None** - do not make additional copies; the input already contains the entire structure of interest

Clicking **Make models** generates a multiscale model for each molecule model with any part [selected](#), or if nothing is selected, every molecule model. Additional copies are shown as [surfaces](#), while the display of the original chains is kept the same. However, the original chains (those with loaded atoms) can be [selected](#) and shown in the [surface style](#) too.

The [surfaces](#) of all chains in the multimer are opened as a single model. The set of atomic coordinates first opened remains present, although the atoms may be hidden depending on the [display style](#). Additional copies of the atomic coordinates will be [loaded](#) as needed for various [display styles](#). The surface model and all copies of the model comprise the *multiscale model*.

**Delete selected models** deletes each [multiscale model](#) with at least one chain [selected](#), except for the single model that was first opened. The original model may be undisplayed, but it can still be used to [remake](#) the [multiscale model](#).

## LOW-RESOLUTION SURFACES

Surfaces generated by **Multiscale Models** are usually low resolution, although higher resolutions can be specified. Low-resolution surfaces are efficient for display because they are graphically simple and require much less memory than the corresponding atomic coordinates.

Initial surface colors are assigned automatically, and the surfaces of chains with the same sequence (and all of their symmetry copies) are colored alike. Surface color, resolution and other parameters can be adjusted in the [Act on selected chains](#) section. Surface display style, color, and transparency can also be adjusted with the main Chimera [Actions menu](#).

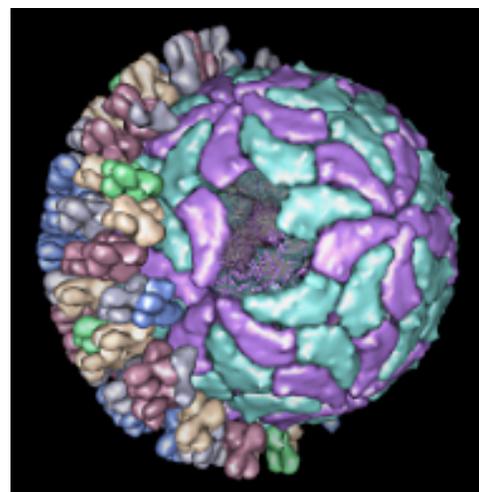
The **Multiscale Models** dialog does not act on standard [molecular surfaces](#), and several Chimera commands that act on molecular surfaces do not affect surfaces from **Multiscale Models**. Although a surface created by **Multiscale Models** with [resolution](#) set to 0 will have the same geometry as a molecular surface, it is not implemented in the same way.

Another way to show a low-resolution surface for atomic coordinates is to calculate a density map from the coordinates with [molmap](#) and display an isosurface of that density.

## STRUCTURAL HIERARCHY

By default, the structural hierarchy contains three levels: PDB chain, PDB molecule, and multimer. To define a different set of levels, it is necessary to write a Python script to create the [multiscale model](#) instead of using the **Make models** button. Plans are to develop a user interface for defining the hierarchy.

The default levels sometimes correspond poorly to biologically significant levels of structure. For example, the bluetongue virus capsid (2btv, see the figure) is a two-layered structure, with an outer shell made up of trimers in five symmetry classes and an inner shell composed of dimers. The PDB file contains 17 chains comprising one asymmetric unit of the full virus shell, which is a 60-mer. Levels such as shell, dimer, trimer, and trimer class would be useful in this case. The Python script [bluetongue.py](#) loads 2btv and creates the [multiscale model](#) with these levels of structure.



## SELECTING COMPONENTS

A chain can be selected by picking its [surface](#) in the same way that atoms and bonds are [picked](#) (by default, with **Ctrl**-left mouse button click). If the **Shift** key is held down at the same time, the selection is added to (or subtracted from, if already selected) the previous selection rather than replacing it. Even though a single chain may appear as disjoint segments, the entire chain is selected, as indicated by highlighting of the surface.

When a chain has been selected by picking its [surface](#), its constituent atoms and bonds are not selected. However, if the surface or any atoms or bonds within the chain are selected, **Multiscale Models** considers the chain selected. Even though the chain selection highlighting is only visible on the surface, the chain remains selected when shown in other [display styles](#).

The **Select chains** section provides several ways to adjust the selection:

- **All** - select all chains in all [multiscale models](#); useful for showing hidden models
- **With loaded atoms** - select the chains for which atomic coordinates are already [loaded](#) (to avoid making extra copies for higher-resolution [display styles](#))
- **Clear** - return to a state with nothing selected

An existing chain selection can be extended:

- **Up** - promote the selection to the next higher level in the hierarchy (by default, PDB chain -> PDB molecule -> all copies of the molecule in the multimer)
- **Copies** - promote the selection to all copies of the selected chains
- **Atoms** - also [select](#) the constituent atoms and bonds, [loading](#) additional copies of the coordinates as required
- **Loaded atoms** - also [select](#) the constituent atoms and bonds with coordinates already [loaded](#)

The following options depend on which of the atoms within a chain are selected:

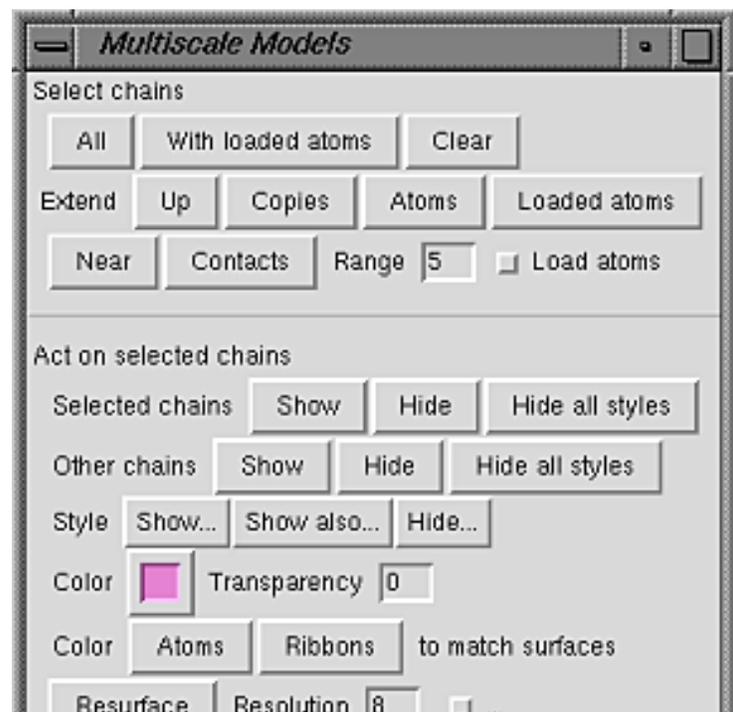
- **Near** *expands* the selection to include additional atoms within **Range** Å of the original selection
- **Contacts** *changes* the selection to include the atoms in the initially unselected set within **Range** Å of the original selection, and the atoms in the original selection *within Range Å of the initially unselected set*

The resulting selection also includes all bonds between selected atoms. If the original selection contains multiscale model [chain selections](#) without any selection of specific constituent atoms, all of the atoms in the selected chains are considered part of the original selection. The coordinates of all atoms within [multiscale model](#) chains are included in the distance calculations, whether [loaded](#) or not. If the **Load atoms** option is on, any virtual (not-yet-loaded) copies with atoms within **Range** will be [loaded](#) and the specific atoms within **Range** will be selected.

## DISPLAY OPTIONS

Buttons in the **Act on selected chains** section control how chains are displayed. Display styles, color, and transparency can also be adjusted with the main Chimera [Actions menu](#).

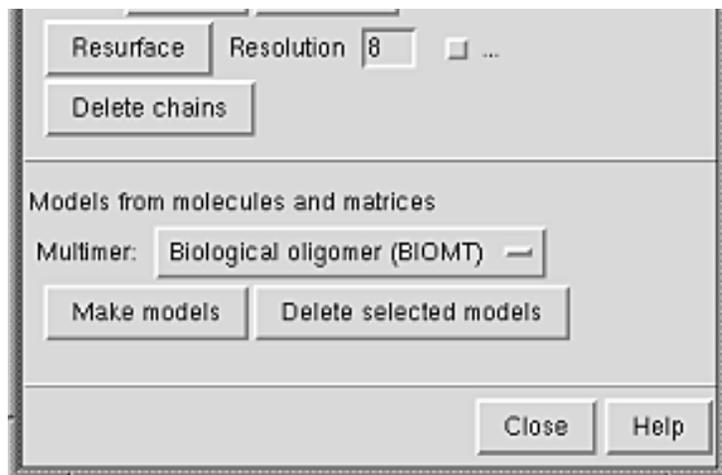
**Show** and **Hide** display and undisplay [surfaces](#), respectively; **Hide all styles** undisplays all [display styles](#). This can be done for the [Selected chains](#) or for **Other chains** (the unselected chains within each [multiscale model](#) containing at least one selected chain). The rest of the



section affects only the [selected](#) chains.

The **Style** for display can be

- [Surface](#)
- [Ribbon](#)
- [Wire](#)
- [Stick](#)
- [Ball & Stick](#)
- [Sphere](#)



All styles except **Surface** require atomic coordinates to be [loaded](#). **Show..** displays the chains in the chosen style only. **Show also...** adds the chosen style to what is already shown, except that a given set of atoms and bonds can only be shown with one [representation](#) at a time (wire, stick, ball-and-stick, or sphere). **Hide...** can be used to undisplay **Surface**, **Ribbon**, or **Atoms and Bonds**.

**Color** and **Transparency** apply to [surfaces](#) only (although there are buttons to color **Atoms** and **Ribbons** to match). Clicking the [color well](#) allows the color to be adjusted with the [Color Editor](#). Transparency can also be adjusted using the [Color Editor](#) (opacity =  $A = 1 - \text{transparency}$ ) or by typing a value in the **Transparency** field and pressing return (Enter). Transparency can range from 0 to 1; values below or above the range are treated as 0 or 1, respectively.

The **Resolution** (**8** by default) controls the level of detail in a [surface](#); a higher number gives a lower resolution. A low-resolution surface is made by counting the atoms in each cell of a three-dimensional grid and then making an isosurface of this occupancy map. The **Resolution** is the grid spacing in display units (usually Å). The isosurface is smoothed to reduce artifacts associated with using an arbitrarily aligned grid.

Setting the **Resolution** to 0, however, triggers calculation of a solvent-excluded molecular surface instead of an isosurface, based on the current [VDW radii](#) and [New Surfaces preferences](#) for probe radius, vertex density, and disjoint surfaces. This **Multiscale Models** molecular surface is not affected by the [threshold and smoothing parameters](#) and will not be updated to reflect any subsequent changes in the underlying atoms or their [VDW radii](#).

It is necessary to press return (Enter) or click **Resurface** to recalculate the surface after changes in **Resolution** or the numerical parameters described below.

The checkbox marked "..." reveals additional adjustable parameters:

- **Threshold atom density [D] CA only [d]** ( $D=0.02$  and  $d=0.002$  by default) - the isosurface threshold in atoms per unit volume (usually Å<sup>3</sup>);  $d$  is used for chains with CA atoms only
- **Smoothing factor** (**0.3** by default) - how much to smooth the isosurface during each step. In one smoothing step, each surface vertex is moved some fraction of the way toward the average position of the neighboring vertices. This fraction is the **Smoothing factor**.

- **Smoothing iterations** (2 by default) - how many smoothing steps to perform

**Delete chains** undisplay the selected chains and removes them from the [multiscale model](#). Deleted chains cannot be restored except by [remaking](#) the [multiscale model](#). If all of the chains in the [multiscale model](#) are deleted, the surface model is closed. Although undisplayed, atomic coordinates for the selected chains are not deleted if part of the original set (the PDB model first opened) or a copy with any chain that has not been deleted.

## LOADING ATOMIC COORDINATES

Atomic coordinates are loaded for the PDB model first opened, but additional copies of the coordinates are loaded only as needed, when [display styles](#) other than [surfaces](#) are shown.

When possible, copies are made from the coordinates already in memory rather than by opening the original input file multiple times. If all copies of the atomic coordinates have been deleted, however, Chimera will try to open the original input PDB file again. If the file no longer exists in the same location, an error message will appear. There is currently no way of indicating that the file is available in a different location. Solutions are to restore the file to its former location or, in the context of a saved [session](#), to change the location information within the [session](#) file before [restarting](#).

Sometimes an excessive amount of coordinate copying is inadvertently set in motion. A dialog enabling the process to be halted will appear after 5 seconds of coordinate copying. Clicking **Stop** halts copying after the copy in progress is complete.

---

UCSF Computer Graphics Laboratory / June 2014

## Unit Cell

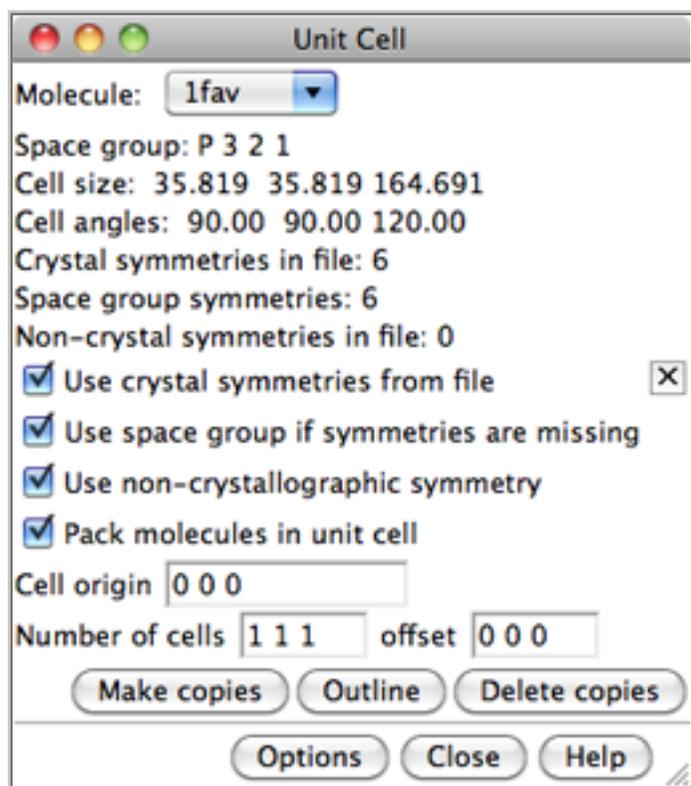
**Unit Cell** builds crystallographic unit cells using symmetry information from the input coordinate file (PDB or mmCIF). A crystallographic unit cell consists of a unique set of coordinates, duplicated and transformed according to the crystallographic and (if present) noncrystallographic symmetries in the crystal. **Unit Cell** can be used to regenerate the full unit cell, or only those parts defined by crystallographic symmetry or noncrystallographic symmetry. See also: [Crystal Contacts](#), [Multiscale Models](#), [sym](#), the [biological unit](#) function in the [Model Panel](#), [fetching](#) PDB-biounit and PQS files

There are [several ways to start Unit Cell](#), a tool in the **Higher-Order Structure** category.

After a PDB or mmCIF file has been [opened](#) in Chimera, **Unit Cell** can generate symmetry-related copies if the file contains sufficient information. In the dialog, **Molecule** can be set to any open molecule model. The space group, unit cell parameters, and numbers of transformation matrices for the current **Molecule** are shown, if available.

- **Make copies** loads copies of the coordinates from the same source as the originally opened model (a local file or a file fetched from the Web) and transforms them as specified by the [options](#).
- **Outline** toggles whether a white outline of the unit cell is shown.
- **Delete copies** removes all copies of the molecule except the original.

The copies can be [saved](#). **Close** simply dismisses the dialog, while **Help** opens this page in a browser window.



Clicking **Options** reveals additional settings (clicking the close button  on the right hides them again). The descriptions below refer to PDB files, but the corresponding mmCIF information can also be used:

- **Use crystal symmetries from file** - apply *crystallographic* symmetry described by SMTRY1, SMTRY2, and SMTRY3 matrices in REMARK 290 lines of a PDB file
- **Use space group if symmetries are missing** - for files without crystallographic symmetry matrices, use the space group name in the CRYST1 record of a PDB file to look up the crystallographic symmetry. Space group names are the full Hermann-Mauguin symbols, *e.g.*, P 21 1 rather than P 21 (see column **xHM\_SpgrpSymbol** in the [CCP4 summary listing](#)).
- **Use non-crystallographic symmetry** - apply *noncrystallographic* symmetry described by MTRIX1, MTRIX2, and MTRIX3 matrices in a PDB file. Only a small fraction of PDB entries have MTRIX records (~10% in 2003). Most of these describe how to transform the coordinates of one chain to match

closely the coordinates of another chain already present in the PDB file. In a few cases (~0.25% of PDB entries), the MTRIX records describe a transformation that produces a new copy whose coordinates are not already in the file. These two cases are distinguished by the contents of column 60 in the MTRIX records. A "1" means that the transformation yields coordinates that are already given (these MTRIX records are ignored by **Unit Cell**), whereas a blank space means that the transformation will produce a new copy. MTRIX column 60 may be set incorrectly in some files. Further, MTRIX records are sometimes missing. For example, 1cd3 has no MTRIX records, but the remarks describe how to produce them from the BIOMT records. Of course, this cannot be handled by **Unit Cell**.

MTRIX records do not describe crystallographic symmetries, but additional symmetries of the asymmetric unit of the crystal. Because the copies of the molecule occupy nonequivalent positions in the crystal, they usually have small structural differences due to differing crystal contacts. Thus, an independent set of coordinates is usually included for these copies, and MTRIX records are not needed. MTRIX records are often used for icosahedral virus particles, where the PDB file will include the coordinates of one molecule in the shell and MTRIX records describing how to place copies that are not crystallographically equivalent. The size of the virus particle precludes independent refinement of coordinates for each copy of a molecule in the shell.

- **Pack molecules in unit cell** - whether to pack the chains so that their centers fall within one unit cell box
- **Cell origin** coordinates (default **0 0 0**) are in unit cell lengths and describe translations of the unit cell along its axes. Values of a coordinate that differ by integer amounts are equivalent; *e.g.*, (0.3 0 0) is equivalent to (-0.7 0 0) and (4.3 0 0). Changing the origin may rearrange the copies and shift the outline (if present). The shift will occur in the direction that maintains the center of the original copy within the box.
- **Number of cells** (default **1 1 1**, a single cell) **offset** (default **0 0 0**) allows generating a block of multiple unit cells. The first three values specify dimensions in number of cells along the unit cell axes; the second three values specify the placement of the block relative to the cell containing the original structure. For example, number of cells **3 1 1** and offset **-1 0 0** would give a 3x1x1 block of cells with the original structure in the middle cell along the first axis.

Changes in cell origin or number can be applied by pressing Enter (return) or clicking the **Make copies** button.

## Limitations

Many problems are due to information that is missing from (or incorrect within) the input file. One way to validate symmetry information is to identify steric clashes with the [Crystal Contacts](#) tool.

**Unit Cell** does not generate multimers defined by [BIOMT matrices](#), but this can be done with [Multiscale Models](#) or the command [sym](#).

Only the following CIF unit cell fields are used:

```
symmetry_space_group_name_h-m
symmetry_equiv_pos
symmetry_equiv_pos_site_id
```

## Unit Cell

```
symmetry_equiv_pos_as_xyz  
space_group_symop_operation_xyz  
cell_length_a, cell_length_b, cell_length_c  
cell_angle_alpha, cell_angle_beta, cell_angle_gamma  
cell_formula_units_z
```

---

UCSF Computer Graphics Laboratory / June 2014

## Small-Angle X-Ray Profile

### Small-Angle X-Ray Profile

calculates a theoretical small-angle X-ray scattering (SAXS) profile from a set of atoms. The result can be displayed along with an experimental profile provided by the user. Calculations are performed with the program [FoXS](#),

[FoXS: a web server for rapid computation and fitting of SAXS profiles](#). Schneidman-Duhovny D, Hammel M, Sali A. *Nucleic Acids Res.* 2010 Jul 1;38 Suppl:W540-4.

either [locally](#) or via a web service hosted by the [UCSF RBVI](#). See also: [fetching](#) PDB-biounit and PQS files, [sym](#)

There are [several ways to start Small-Angle X-Ray Profile](#), a tool in the **Higher-Order Structure** category.

The **Molecule** setting controls which atoms will be included in the calculation:

- any one of the open molecule models
- **all molecules** - all molecule models combined
- **selected atoms** - the currently [selected](#) atoms

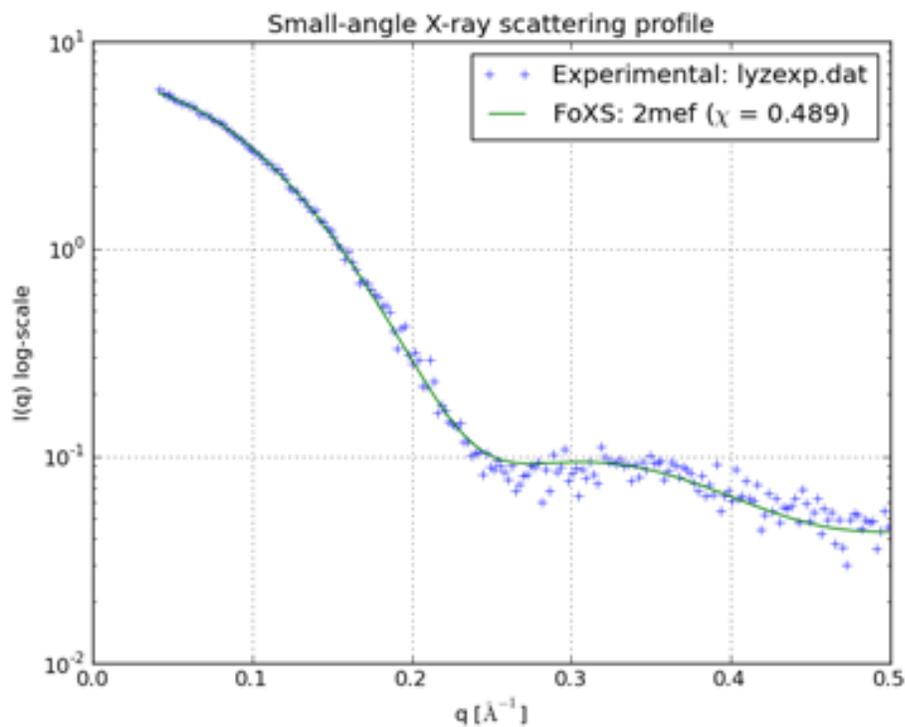
Water molecules are automatically ignored.

Optionally, an **Experimental profile** can be read from a file. If an experimental profile is provided, the theoretical profile will be scaled to fit the data. The file should contain an initial comment line followed by three whitespace-preceded columns: q value, measured scattering, and measured scattering error (see an [example file](#) for lysozyme). The error values are used for fitting but are not plotted.

For comparison to experimental data, the structure should be as similar as possible to the experimental sample in terms of which atoms are present (not missing domains, loops, or parts of residues). [Dock Prep](#) can be used to repair truncated side chains and/or add [explicit hydrogens](#).

Clicking **Options** reveals additional settings (clicking the close button  on the right hides them again):

- **Excluded volume adjustment** (on by default) - whether to adjust the excluded volume parameter (c1) to



$$q = (4\pi\sin\theta) / \lambda$$

$2\theta$  is the scattering angle,  $\lambda$  the incident wavelength

improve the fit of the theoretical profile to the [experimental profile](#)

- **Hydration (water) layer (on by default)** - whether to include a hydration layer parameter (c2) to improve the fit of the theoretical profile to the [experimental profile](#)
- **Experimental background adjustment (off by default)** - whether to apply a background adjustment to the [experimental profile](#); this tends to reduce the intensity at high values of q
- **Maximal q value (default 0.5)** - up to what value of q the profile should be computed
- **Profile size (default 500)** - number of points in the computed profile
- **Implicit hydrogens (on by default)** - whether to ignore any explicit hydrogen atoms and consider hydrogen atoms for proteins and nucleic acids implicitly by adding their form factors; for proper recognition of these residues, the structure should have standard PDB atom and residue names. If the structure includes other groups (lipids, sugars, *etc.*), this option should be turned off and explicit hydrogens added instead.
- **Fast coarse-grained profile (off by default)**
- **Use new plot window (off by default)** - whether to show the profile in a new window instead of adding it to the most recent pre-existing plot
- **Local executable file (optional)** - a local copy of the **FoXS** executable; if none is specified, the RBVI web service will be used
- **Output file location (optional)** - local directory in which to place results, stderr, and stdout files

For further details, consult the [FoXS help](#).

Clicking **Calculate Profile** computes the theoretical profile and displays it as a solid line. Data points in the [experimental profile](#) (if any) are shown as plus signs, and a measure of the quality of fit between the experimental and theoretical profiles,  $\chi$ , is given in the legend. A lower value of  $\chi$  indicates a better fit.

**Close** dismisses the **Small-Angle X-Ray Profile** dialog. **Help** opens this manual page in a browser window.

## Plot Navigation

Plots are generated with [matplotlib](#). Icons for manipulating and saving the view are provided below the graph. The mode icons act as toggles: clicking the icon will turn the mode on or off, depending on the current state.

icon	meaning
	default (initial) view
	previous view
	next view
	“pan/zoom” mode (toggle)
	“zoom to rectangle” mode (toggle)
	sliders for graph scaling



save as image file  
(it may be necessary to include the proper  
filename suffix to indicate format: \*.png  
[Portable Network Graphics], \*.ps [PostScript],  
\*.eps [Encapsulated PostScript], \*.pdf  
[Portable Document Format], \*.svg [Scalable  
Vector Graphics])



## Scale Bar

The **Scale Bar** tool draws a scale bar and associated label. Scale bars are often included in images for presentation and publication. Chimera has no inherent spatial units, so the scale is determined by the coordinates of the input data. While atomic coordinates are usually in angstroms, many other units (nanometers, microns, *etc.*) may be encountered, depending on the type of data. The state of **Scale Bar** is included in saved [sessions](#). See also: [Color Key](#)

There are [several ways to start Scale Bar](#), a tool in the **Higher-Order Structure** category. Starting **Scale Bar** brings up a dialog for creating the scale bar.

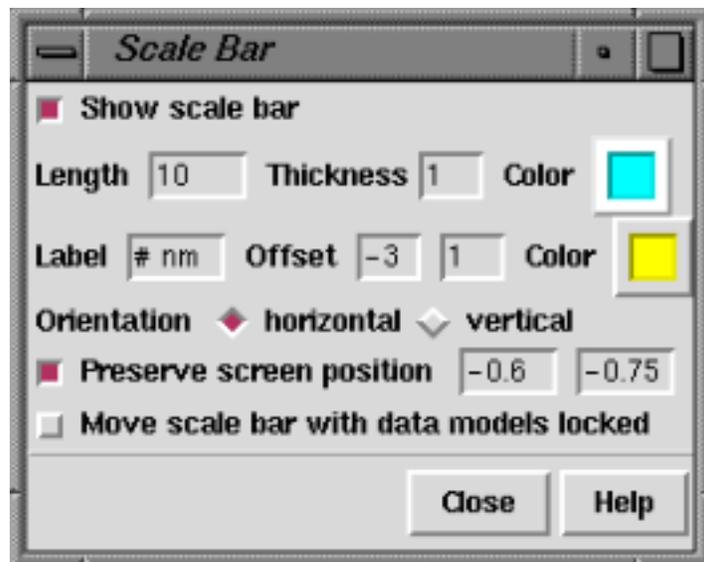
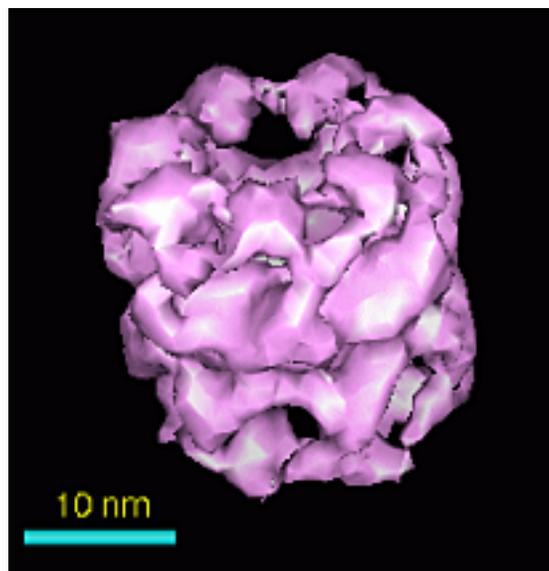
Turning **Show scale bar** on creates and displays a bar according to the existing settings; turning it off deletes the bar. Deleting and re-creating the scale bar frequently resolves problems such as disappearance of the bar or label. The bar is a cylinder that [exists in three dimensions](#).

Typed-in settings will not be applied until the Enter (return) key has been pressed. **Length** and **Thickness** are expressed in the current display units. The bar and label **Color** settings can be changed by clicking the respective [color wells](#). The **Label** text is as shown in the adjacent field except that the **Length** value is substituted for every occurrence of the pound sign # (zero, one, or multiple times). The label **Offset** is the position of the bottom left corner of the label relative to the middle of the bar edge, in the current display units. Bar **Orientation** may be **horizontal** or **vertical**. When the bar is **horizontal**, the label **Offset** is relative to the top of the bar; increases in the two values move the label rightward and upward, respectively. When the bar is **vertical**, the label **Offset** is relative to the right side of the bar; increases in the two values move the label downward and rightward, respectively.

The **Preserve screen position** fields describe the horizontal and vertical positions, respectively, of the center of the bar within the graphics window. The horizontal value ranges from -1 to 1 going from left to right and the vertical value ranges from -1 to 1 going from bottom to top. **Preserve screen position** keeps the bar center at the indicated position when the view is scaled (of course, the bar is still scaled). When **Preserve screen position** is off, the bar can drop out of view when the view is scaled up.

The **Preserve screen position** fields describe the horizontal and vertical positions, respectively, of the center of the bar within the graphics window. The horizontal value ranges from -1 to 1 going from left to right and the vertical value ranges from -1 to 1 going from bottom to top. **Preserve screen position** keeps the bar center at the indicated position when the view is scaled (of course, the bar is still scaled). When **Preserve screen position** is off, the bar can drop out of view when the view is scaled up.

## MOVING THE BAR IN 3 DIMENSIONS



When the bar is created with **Show scale bar**, or when one of the **Orientation** settings is chosen, the bar is made parallel to the XY plane (the plane of the screen). The ends of the bar have the same Z coordinate, where Z is the dimension perpendicular to the screen. When created with **Show scale bar**, the bar is placed halfway between the near and far clipping planes (these can be viewed and moved in the [Side View](#)).

When **Move scale bar...** is off, the data models are [activated for motion](#) and the scale bar is deactivated. Turning **Move scale bar...** on reverses these settings, so that the bar can be [translated and rotated with the mouse](#) while the other models remain fixed. After a scale bar has been moved around, its ends can have different Z coordinates (one end of the bar can be closer than the other), which is usually not desired. Clicking one of the **Orientation** settings is an easy way to bring the bar back parallel to the plane of the screen.

Two additional issues relate to the existence of the scale bar in three dimensions:

- **Depth cueing.** By default, the display is brightest at the front clipping plane and dimmest at the rear clipping plane. If the scale bar is too dim, there are two general ways to make it brighter:
  - the position of the scale bar relative to the clipping planes can be adjusted, by [moving the scale bar](#) or one or both clipping planes; the clipping planes can be moved using the [Side View](#)
  - depth cueing can be turned off or adjusted, using the [Effects panel](#)
- **Perspective.** By default, the display has perspective, such that nearby items appear larger than items that are distant. There are two general ways to ensure that scale is accurately depicted:
  - the scale bar can be [moved](#) to the same depth as the data (or the approximate average depth, since the data is generally three-dimensional)
  - the perspective can be turned off so that items appear the same size regardless of how far away they are, using the [Camera panel](#) (set **projection** to **orthographic**)

Of course, changing the depth cueing, moving the clipping planes, or turning off perspective will also affect the appearance of the displayed data, whereas moving only the scale bar will not. When **Move scale bar...** is on, the scale bar can be Z-translated (moved closer or farther away) [with the mouse](#) or with the command [move](#).

## THE SCALE BAR IS A MOLECULE

The scale bar is implemented as a molecule model containing one residue, three atoms (two ends of the bar and one for label positioning), and one bond (connecting the end atoms). **Thus, any actions or commands that affect the display of atoms and bonds can also alter the appearance of the scale bar.** When the data models contain atomic coordinates (that is, when they are also molecule models), it may be best to adjust their appearance as desired for the final image before creating the scale bar.

Turning **Show scale bar** off and then back on will cure many ills. This actually deletes the scale bar (closes the scale bar model) and creates a new one in accordance with the settings in the **Scale Bar** dialog. When the scale bar is created, it is opened as the lowest available model number. The end atoms are undisplayed and the bond between them is [represented](#) as a stick (cylinder). Only one scale bar can exist at a time.

## CURRENT LIMITATIONS

**Label font and size cannot be controlled separately from those of atom labels.**

The font type and size of scale bar labels can only be changed along with other atom/residue labels in the [Labels preferences](#). However, labels of independently controllable size can be created using [2D Labels](#) instead.

**Label text can turn out smaller in printed and saved images.**

Label text is restricted to a certain height in pixels. When an image is saved or printed at a different resolution than the current screen display, the size of the label relative to the rest of the image will be different than it is in the display. Higher resolution of the saved or printed image corresponds to a smaller relative label size.

**The label atom is sometimes visible.**

As mentioned above, the scale bar is implemented as a molecule, with one of its atoms defining the location of the label. The label atom must remain displayed for the label to be displayed. When the scale bar is created, the label atom is colored to match the current background. If the [background color](#) is subsequently changed, the label atom may be visible against the new background. Deleting and re-creating the scale bar with **Show scale bar** solves this problem by making the label atom match the new background color.

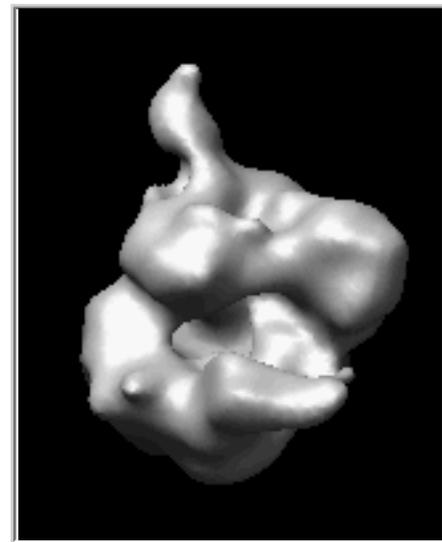


## Morph Map

**Morph Map** morphs between two related sets of [volume data](#) (*maps*). The data sets should have the same grids: dimensions, spacing, and numbers of points. Note the command [vop resample](#) can be used to make a copy of one map that has the same grid as another. By default, morphing intermediates are generated by linear interpolation. A smooth progression from one map to the other can be displayed and recorded as a movie. See also: [Morph Conformations](#), [Volume Series](#), [making movies](#), the [ParM filament tutorial](#) at the Chimera web site

There are [several ways to start Morph Map](#), a tool in the **Volume Data** category (including from the [Volume Viewer Tools](#) menu). It is also implemented as the command [vop morph](#).

The **First map** and **Second map** should be chosen from the adjacent lists of open volume data sets. If the maps of interest are not already open, the **Browse...** buttons can be used to locate and open them.



Moving the **Fraction** slider generates the corresponding interpolated state. The morph is opened as new volume data set, and its display settings ([threshold level/color](#), [surface smoothing](#), *etc.*) can be adjusted in [Volume Viewer](#). At a fraction of 0.0, the interpolated map is the same as the **First map**, and at a fraction of 1.0, it is the same as the **Second map**.

Clicking **Options** reveals additional settings that can be hidden again by clicking the small button  on the right.

- **Movie start** [ $f_s$ ] **end** [ $f_e$ ] **step** [*increment*] - playback/recording setup:  $f_s$  (default **0.0**) is the smallest fraction to be visited during playback or the movie,  $f_e$  (default **1.0**) is the largest, and *increment* (default **0.1**) is the change in fraction per frame
- **Undisplay original maps** - undisplay the endpoint maps whenever the fraction is changed
- **Interpolate colors** - interpolate map colors; only applies when the maps have the same number of [coloring thresholds](#) (contour levels for surface/mesh display, coloring control nodes for solid display)
- **Multiplier for second map** [*factor*] - scale the values in the second map by *factor* before calculating intermediate values
- **Adjust threshold for constant volume** - automatically adjust the [threshold](#) (contour level) to keep the enclosed volume constant
- **Add to first map instead of interpolating** - instead of linearly interpolating between data values D1 (from the first map) and D2 (from the second):

$$D1(1-\text{fraction}) + D2(\text{fraction})$$

calculate intermediate values with:

**D1 + D2(fraction)**

This treats the second map as a delta to be applied to the first map. For example, the second map could represent normal mode fluctuations of the first map.

**Play** oscillates playback continuously until **Stop** is clicked. **Record** opens a dialog for [recording a movie](#).

**Close** closes the **Morph Map** interface, and **Help** opens this manual page in a browser window.

## Recording a Movie

Clicking the **Record** button in **Morph Map** opens a dialog with settings for recording a movie:

- movie **File name** and location
- **File type**, with choices:
  - H.264 [.mp4]
  - VP8/WebM [.webm]
  - Theora [.ogv]
  - Quicktime [.mov]
  - AVI MSMPEG-4v2 [.avi]
  - MPEG-4 [.mp4]
  - MPEG-2 [.mpg]
  - MPEG-1 [.mpg]
  - WMV2 [.wmv]
- **Encode forward then backward ("roundtrip") (true/false)** - whether to include the frames in reverse order as the second half of the movie
- **Rendering:**
  - **Chimera** (default) - Chimera rendering, normally offscreen ([details...](#)). Images can be supersampled, that is, initially generated at a higher resolution and then sampled down to the final size.
    - **Supersample** (1x1/2x2/3x3/4x4) - how many pixels to sample in the X and Y dimensions for each pixel in the final saved image; thus, 1x1 corresponds to no supersampling. Higher values increase the smoothness of edges in saved images and increase calculation time with little effect on file size. 3x3 is generally recommended when supersampling is done.
  - **POV-Ray** - [raytrace with POV-Ray](#). This rendering option is the slowest but includes fancier effects such as high-quality shadows. The **POV-Ray Options** button opens the corresponding [preferences](#).

## Advanced Options:

- **Quality** (highest/higher/high/**good**/medium/fair/low) - higher quality corresponds to higher (variable) playback bit rates and a larger movie file, assuming the same window size and movie frame rate
- **Image format** - image file format (however, PNG will be used regardless of this setting if [raytracing](#) is done):
  - JPEG [.jpeg]

- PNG [.jpeg]
- PPM [.jpeg] (default)
- **Additional recording options** - options are the same as for the command [movie record](#); if this field is left blank, image frames will be saved with default names in a default location (but normally deleted after movie encoding, depending on the encoding options)
- **Additional encoding options** - options are the same as for the command [movie encode](#); if this field is left blank, the movie will be encoded to play at 25 frames per second, and image frames will be deleted after the movie has been encoded

Clicking **Record** plays the morph from start to end, saving an image at each step, and encodes the resulting images into a movie file.

**Close** dismisses the dialog without initiating recording. **Image Tips** shows the [tips on preparing images](#), and **Help** opens this manual page in a browser window.



# Chimera User's Guide

CHIMERA

Tutorials

Basic Functions

Tools

## [Tools Index](#)

## Volume Tracer

### [Introduction](#)

### [Menus](#)

### [Features](#)

### [Marker Sets](#)

### [Markers](#)

- [Creation](#)
- [Movement](#)
- [Annotation](#)

### [Links](#)

### [Appearance](#)

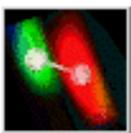
- [Color, Radius](#)
- [Smooth Paths](#)

### [Surface Creation](#)

### [Marker Files](#)

### [Limitations](#)

### [Possible Developments](#)



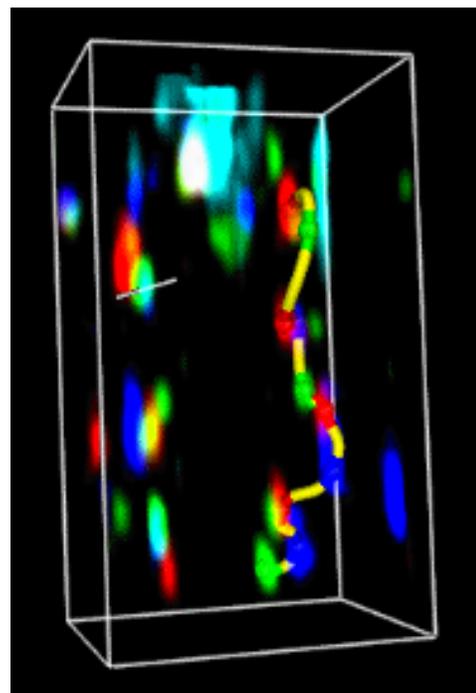
## Volume Tracer

**Volume Tracer** allows interactive placement of markers within [volume data](#) displayed by [Volume Viewer](#). The markers can be labeled and connected by smooth, interpolated paths, and paths can be stacked to form a surface. Markers can also be placed, moved, and connected in arbitrary ways to build conceptual models independent of any volume data. Markers and related parameters are included in saved [sessions](#). See also: [Volume Series](#), [vop ridges](#), [shape tube](#), [mask](#), [measure](#), [meshmol](#)

This tool was initially designed to trace fluorescently labeled chromosomes in light microscope data and protein backbones in cryoelectron microscope density maps.

### Definitions:

- A *marker* is a sphere with an associated position, radius, and color, implemented in Chimera as an atom.
- A *marker note* is a text annotation associated with a marker, implemented as an atom label.
- A *link* is a connection between markers, implemented as a bond and displayed as a cylinder with user-specified radius and color.
- A *marker set* is a named collection of markers and links, implemented in Chimera as a molecule model.
- Multiple marker sets can be present, but only one at a time can be the *active set*. The [Marker set menu](#) shows the name of the active set. Various [menu](#) operations apply to the active set, and newly created markers and links are added to this set.
- A *marker file* includes marker set name(s), marker positions, marker annotations, links, colors, and radii.



There are [several ways to start Volume Tracer](#), a tool in the **Volume Data** category (including from the [Volume Viewer](#) Tools menu).

## VOLUME TRACER MENUS

- **File** - read/save [markers](#) and [links](#)
- **Actions** - hide/show/delete [markers](#) and [links](#)
- **Mouse** - control how [markers](#) and [links](#) are created and manipulated with the mouse
- **Features** - control which features are shown in the **Volume Tracer** dialog

#### ← File:

- **Open marker set...** bring up a [dialog](#) for browsing to and opening a [marker file](#)
- **New marker set** - start a new [marker set](#) and make it the [active set](#)
- **Save current marker set** - save the [active set](#) to the same [marker file](#) it was opened from, or if none, to a specified filename
- **Save current marker set as...** save the [active set](#) to a [marker file](#)
- **Save all marker sets as...** save all [marker sets](#) to a single [marker file](#)
- **Save selected marker sets as...** save all sets chosen in the [marker set list](#) (or if none, the [active set](#)) to a single [marker file](#)
- **Close marker set** - delete all sets chosen in the [marker set list](#) (or if none, the [active set](#))

#### ← Actions:

- **Show markers** - show the [selected](#) markers, or if none, all markers in the [active set](#)
- **Hide markers** - hide the [selected](#) markers, or if none, all markers in the [active set](#)
- **Delete markers** - delete the [selected](#) markers
- **Show links** - show the [selected](#) links, or if none, all links in the [active set](#)
- **Hide links** - hide the [selected](#) links, or if none, all links in the [active set](#)
- **Delete links** - delete the [selected](#) links
- **Show marker notes** - show annotations for the [selected](#) markers, or if no markers are selected, for all markers in the [active set](#)
- **Hide marker notes** - hide annotations for the [selected](#) markers, or if no markers are selected, for all markers in the [active set](#)
- **Delete marker notes** - delete annotations for the [selected](#) markers
- **Transfer selected markers to current set** - transfer the [selected](#) markers to the [active set](#); this allows markers to be transferred from one set to another

#### ← Mouse (the same settings are also in the [Mouse mode options](#)):

- **Place markers on high density** (on by default) - when the [assigned button](#) is clicked over any volume data display region, add a marker at the nearest local maximum above the threshold along the line of sight ([details...](#))
- **Place markers on data planes** (on by default) - when the [assigned button](#) is clicked over a [single-plane display](#), add a marker in that plane
- **Place markers on surfaces** - when the [assigned button](#) is clicked over a [surface](#),

- add a marker at the nearest surface point along the line of sight
- **Place markers outside data** - when the [assigned button](#) is clicked, add a marker halfway between the near and far [global clipping planes](#)
- **Place markers while dragging** - create a stream of markers as the cursor is moved with the [assigned button](#) held down
- **Move and resize markers** - allow dragging a marker with the [assigned button](#), including changing its depth when the Shift key is also held down and changing its size when caps lock is on
- **Marker color matches volume color** - color new markers [dropped on data spots](#) to match the spots; color other new markers the specified [Marker color](#)
- **Link new marker to selected marker** - as a new marker is [created](#), link it to the previously [selected](#) marker
- **Link consecutively selected markers** - when an existing marker is [selected](#), link it to the previously [selected](#) marker

## FEATURES

The **Features** menu lists potential sections in the **Volume Tracer** dialog, with checkboxes controlling which sections are shown. A section can be closed by unchecking its entry in the **Features** menu, unchecking its [feature button](#), or clicking its close button () if present.

- [Feature buttons](#)
- [Marker color and radius](#) (shown by default)
- [Marker notes](#)
- [Marker set list](#)
- [Marker set menu](#) (shown by default)
- [Mouse button menu](#) (shown by default)
- [Mouse mode options](#)
- [Rename marker set](#)
- [Slice display](#)
- [Smooth paths](#)
- [Surfaces](#)
- **Show only default panels** - restore the previously [saved](#) state of the **Features** menu (and thus which sections are present in the **Volume Tracer** dialog)
- **Save default panels** - save the state of the **Features** menu (and thus which sections are present in the **Volume Tracer** dialog) in the Chimera [preferences file](#)
- **Save default dialog settings** - save many current **Volume Tracer** settings other than the state of the **Features** menu in the [preferences file](#)
- **Use factory default settings** - reset the **Features** menu and other **Volume Tracer** settings to the factory defaults without changing the [preferences file](#)

### ← Feature buttons

This section contains a row of small, unlabeled buttons representing the *other* possible sections of the **Volume Tracer** dialog. Checking a feature button has the same effect as checking a box in the [Features menu](#). Although arranged horizontally, the buttons are

in the same order as the menu entries. Placing the cursor over a feature button shows the name of the feature.

## MARKER SETS

A *marker set* is a named collection of [markers](#) and [links](#), implemented in Chimera as a molecule model.

Multiple marker sets can be present, but only one at a time can be the *active set*. The [Marker set menu](#) shows the name of the active set. Various [menu](#) operations apply to the active set, and newly created markers and links are added to this set.

At first, the [active](#) (and only) marker set is **marker set 1**. [File... New marker set](#) starts a new marker set and makes it the [active set](#). Marker sets are initially named **marker set 1**, **marker set 2**, *etc.*, but can be [renamed](#). Markers can be [transferred](#) from one set to another. [File... Close marker set](#) removes the [active set](#). A marker set cannot be recovered after it has been removed.

### ← Marker set menu

The name of the [active set](#) is shown next to the words **Marker set**. Clicking on the name reveals a menu of all open marker sets, from which a different set can be designated as the [active set](#).

### ← Marker set list

This section is a list of all open **Marker Sets**. A set can be designated as the [active set](#) by clicking its entry in the list. Although only one set at a time can be the [active set](#), multiple sets can be chosen from the list by dragging with the mouse or Ctrl-clicking on individual entries. The chosen sets can then be saved to a single [marker file](#) with [File... Save selected marker sets as...](#) or deleted with [File... Close marker set](#).

### ← Rename marker set

In this section, the [active set](#) can be renamed by entering a new name and pressing the return key (Enter).

## MARKERS

A *marker* is a sphere with an associated position, radius, and color, implemented in Chimera as an atom. Each marker belongs to a [marker set](#) (molecule model). As markers are created, they are assigned successive residue numbers for convenient [command-line specification](#) (for example, **#0:10-18** specifies markers 10-18 in model 0). Just like other atoms in Chimera, markers can be [selected](#) and their attributes can be [inspected and modified](#).

Markers that are [selected](#) can be hidden, shown, deleted, or [transferred](#) to the [active set](#) using the **Volume Tracer [Actions menu](#)**. When no markers are selected, the hide and

show (but not delete) [Actions](#) apply to all markers in the [active set](#). A marker cannot be recovered after it has been deleted.

Markers that are [selected](#) can also be used to define a zone of volume data display (with [Zone](#) in [Volume Viewer](#)), other surface display (with [Surface Zone](#)), or surface coloring (with [Color Zone](#)).

A newly created marker is always added to the [active set](#). When a marker is created, it is [selected](#) and any other markers are deselected. The [Mouse mode options](#) and [Mouse menu](#) control how markers are created, while the [Mouse button menu](#) controls which mouse button is used. When more than one mode of marker creation could apply to a single click of the [assigned button](#), the one that gives the frontmost marker will “win.”

#### ← Mouse button menu

**Use [button] to place/move markers** controls which mouse button will be used for marker operations. This *assigned button* will cease to perform any previously assigned function while assigned to one or more marker operations in the [Mouse mode options](#) or [Mouse menu](#).

#### ← Mouse mode options

This section controls how markers and [links](#) are created. The checkbox options are also available in the [Mouse menu](#).

- **Place markers on high density** (on by default) - when the [assigned button](#) is clicked over any volume data display region, add a marker at the nearest local maximum above the threshold along the line of sight. The marker will be placed at the screen (X,Y) position of the cursor at the time of the mouse click, at the depth (Z-coordinate) of the *closest local maximum* above the [detection threshold](#). If no value along the line of sight exceeds the threshold, or if only a [single plane](#) of data is displayed, this option will not generate a marker. A [slice display](#) shows the data values along the line of sight and can be used to adjust the threshold. A marker cannot be placed on a data *minimum* automatically; it must be created first and then manually [moved](#) to a minimum.
- **Place markers on data planes** (on by default) - when the [assigned button](#) is clicked over a [single-plane display](#), add a marker in that plane
- **Place markers on surfaces** - when the [assigned button](#) is clicked over a [surface](#), add a marker at the nearest surface point along the line of sight
- **Place markers outside data** - when the [assigned button](#) is clicked, add a marker halfway between the near and far [global clipping planes](#)
- **Place markers while dragging** - create a stream of markers as the cursor is moved with the [assigned button](#) held down. This is primarily useful along with [Place markers on data planes](#) for generating strings or loops of markers for subsequent use in [surface creation](#).
- **Dragging marker spacing** - how far apart to drop markers as they are created

with [Place markers while dragging](#). The spacing is in grid units if volume data is present, otherwise screen pixels. A loop will be closed automatically if its start and end markers are within this distance.

- **Move and resize markers** - enable moving and resizing markers by dragging with the [assigned button](#). In the following, the X axis is horizontal in the plane of the screen, Y is vertical in the plane of the screen, and Z is orthogonal to the screen.
  - A marker can be dragged from one (X,Y) position to another with the [assigned button](#). Dragging with both the [assigned button](#) and the Shift key held down changes the Z-coordinate of the marker; dragging down pulls the marker closer (out of the screen) while dragging up pushes the marker further away (into the screen). The Shift key can be pressed and released repeatedly during a single dragging motion to switch between (X, Y) and Z-coordinate movement. It is difficult to see marker depth without using [stereo](#), although the [Side View](#) may be helpful. Another technique is to watch for intersection of the marker with a [volume isosurface](#) as marker depth is adjusted.
  - A marker can be resized by dragging with the [assigned button](#) when caps lock is on; dragging down decreases the [radius](#) while dragging up increases it.

If any [marker creation](#) mode is also active, the result of clicking the [assigned button](#) depends on the location of the click. Clicking over an existing marker grabs it for movement/resizing, while clicking elsewhere creates a new marker.

Only one marker can be moved or resized at a time. A marker cannot be grabbed if there is another selectable item (such as an atom, bond, or another marker) in front of it. Volume displays created by [Volume Viewer](#) are not selectable, so they will not interfere with marker grabbing.

A newly created marker can also be moved in what was the Z-dimension at the time of its creation by dragging the vertical line in the [slice graph](#).

- **Marker color matches volume color** - color new markers [placed on data spots](#) to match the spots; color other new markers the specified [Marker color](#)
- **Link new marker to selected marker** - as a new marker is [created](#), link it to the previously [selected](#) marker. One (and only one) marker must have been selected beforehand.
- **Link consecutively selected markers** - when an existing marker is [selected](#), link it to the previously [selected](#) marker. Only one marker should be selected at a time. This mode allows markers [created](#) previously to be linked in any desired pattern.

#### ← Slice display

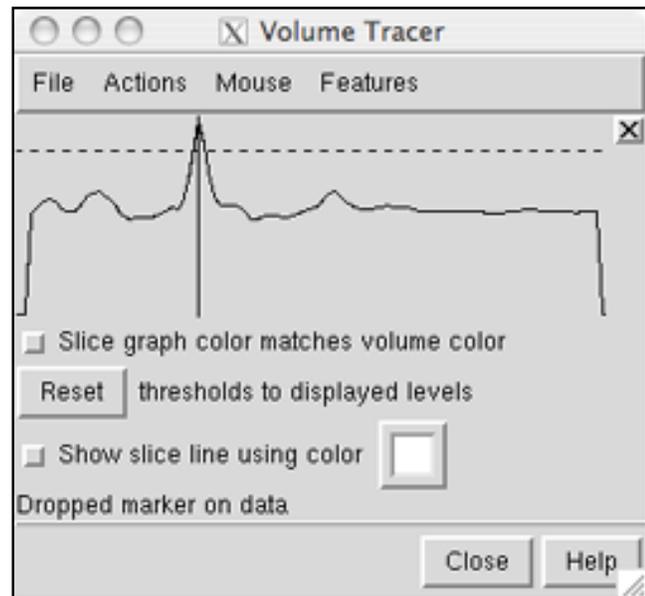
A slice graph shows the data profile along the line of sight (the Z dimension) under the cursor when the [assigned](#)

#### Slice display

[button](#) is clicked and marker [creation](#) and/or [movement](#) is enabled. Near to far is shown from left to right.

In the slice graph, each displayed volume data set under the mouseclick is shown with a solid line and a dotted line. **Slice graph color matches volume color** indicates that the lines should be shown in the same color as the volume display. The solid line represents data values; the dotted line is the [detection threshold](#) for placing markers. When a marker is [placed on a data spot](#), its Z-position is indicated with a vertical black line. The vertical line can be dragged to move the marker to a different depth.

This is useful when a marker has been placed on a faint foreground spot rather than its intended destination, a brighter background spot.



The *detection threshold* is initially the same as the lowest [display threshold](#) for that data set in [Volume Viewer](#). As needed, the detection threshold can be dragged higher or lower in the slice graph with the mouse. This does not change the display threshold. Once a detection threshold has been moved, it is shown with longer dashes. **Reset** sets the detection threshold(s) back to the existing display threshold(s).

When **Show slice line using color** is on, creating a marker within a volume display region generates a line along the current Z axis that intersects the marker. The color of the line is specified in the adjacent [color well](#). Changing the color will not affect an existing slice line, only those subsequently drawn. If there is no volume data, no slice line will be generated. Because the line follows the line of sight at the time of marker creation, it will not be visible until there has been some rotation. Upon rotation, it should be evident which data spots the line intercepts and whether the marker has been placed on the intended spot. The slice line will disappear when a newer marker is created, and if the newer marker is within a volume display region, a slice line will be drawn for the newer marker. Slice lines are implemented as VRML models.

#### ← Marker notes

A *marker note* is a text annotation associated with a [marker](#), implemented as an atom label in Chimera. Notes on markers that are [selected](#) can be hidden, shown, or deleted using the **Volume Tracer Actions menu**. When no markers are selected, the hide and show (but not delete) [Actions](#) apply to all marker notes in the [active set](#). A marker note cannot be recovered after it has been deleted.

Entering text in the **Marker note** field and pressing the return key labels each [selected](#) marker with that text. Notes cannot occupy multiple lines. The adjacent [color well](#) specifies the color of the text. Editing the color changes the note color of any [selected](#) markers.

The font size of notes (along with all atom labels) can be adjusted collectively in the [Labels preferences](#). For publication/presentation, [2D Labels](#) may be a useful alternative;

they can be in multiple sizes and colors, placed anywhere in the graphics window. They do not move when models are rotated or translated.

## LINKS

A *link* is a connection between [markers](#), implemented in Chimera as a bond and displayed as a cylinder with user-specified radius and color. Only one link can exist for a given pair of markers. Just like other bonds in Chimera, links can be [selected](#) and their attributes can be [inspected and modified](#).

Links that are [selected](#) can be hidden, shown, or deleted using the **Volume Tracer Actions menu**. When no links are selected, the hide and show (but not delete) **Actions** apply to all links in the [active set](#). A link will also be hidden when the markers it connects are hidden. A link cannot be recovered after it has been deleted.

If the command [bondzone](#) has been issued, links that are [selected](#) can also be used to define a zone of volume data display (with [Zone](#) in **Volume Viewer**), other surface display (with [Surface Zone](#)), or surface coloring (with [Color Zone](#)).

A newly created link is always added to the [active set](#). When a link is created, it is [selected](#) and any other links are deselected. The [Mouse mode options](#) and [Mouse menu](#) control how links are created.

## APPEARANCE

### ← Marker color and radius

This section specifies the colors and radii of [markers](#) and [links](#).

Editing the **Marker color** (using the associated [color well](#)) changes the color of any [selected](#) markers. New markers will also receive the specified **Marker color** unless [Marker color matches volume color](#) is turned on. In that case, new markers [placed on data spots](#) will be colored to match the spots, but other new markers will receive the specified **Marker color**.

Conversely, [Color Zone](#) can be used to color volume data surfaces to match nearby [selected](#) markers.

When marker **radius** is not specified, a default value is determined as follows:

- When the marker is within a volume display region and the displayed data is not a [subsample](#), marker radius will be set to the X, Y, or Z grid spacing of the data, whichever is smallest.
- When the marker is within a volume display region and the displayed data is a [subsample](#), marker radius will be set to the X, Y, or Z grid spacing of the subsampled data, whichever is smallest.
- When the first marker in a set is created outside any volume display region or in the absence of any volume data, its radius will be set to 1/100 of the distance between the near and far clipping planes. The clipping plane positions can be

viewed and adjusted in the [Side View](#).

- When a subsequent marker (not the first in a set) is created outside any volume display region or in the absence of any volume data, its radius will be set to equal the radius of the first marker in the set, regardless of how that value was determined.

Entering a marker **radius** value (with or without pressing the return key) sets the radii of subsequently created markers. When the return key is pressed, the radii of any [selected](#) markers will be changed accordingly. Radii are expressed in the physical display units, not grid index units. Markers can also be [resized with the mouse](#).

The number of subdivisions used for drawing the markers (and thus the smoothness of the spheres) can be increased by raising **subdivision** in the [Effects tool](#). Higher values increase smoothness, but may slow performance.

Editing the **Link color** (using the associated [color well](#)) changes the color of any currently [selected](#) links and subsequently created links.

Links can be shown as lines (**radius** = 0) or as cylinders (**radius** > 0). When link **radius** is not specified, it is set to half of the [default marker radius](#). Entering a link **radius** value (with or without pressing the return key) sets the radii of subsequently created links. When the return key is pressed, the radii of any [selected](#) links will be changed accordingly.

#### ← Smooth paths

With this section of the dialog, interpolated paths can be generated for [linked](#) chains of [markers](#). Clicking **Show** generates a curve for each chain of markers in the [active set](#); if multiple links emanate from a marker, so will multiple curve segments. ([Links](#) can be hidden with [Actions... Hide links](#).) Each curve is an Overhauser (also called Catmul-Rom) spline, a cubic spline where the tangent at each marker is parallel to the vector between the preceding and following markers. Curves for a marker set are implemented as [pieces](#) of a [surface model](#). **Unshow** removes any curves drawn for the [active set](#).

After a parameter is changed, it is necessary to click **Show** or press return within any **Smooth paths** parameter field to apply the change. Paths can be shown as lines (**Curve radius** = 0) or as tubes constructed from multiple cylinders (**Curve radius** > 0). The number of line segments or cylinders forming the curve between a linked pair of markers equals **Segment subdivisions** + 1.

A path is colored based on [marker and link colors](#). **Band length** specifies the length of curve centered on each marker that should be colored the same as the marker. Parts of the curve further away from a marker receive the color of the corresponding link. Generally, color bands are not exactly the specified length because:

- Curves consist of a finite number of [subdivisions](#). A change from one color to another along the curve occurs linearly over a single line segment when **Curve radius** = 0 and abruptly between two cylinders when **Curve radius** > 0.
- When **Curve radius** > 0, a path consists of cylinders joined by spheres. Even when **Band length** = 0, there is still one joining sphere at the position of the marker that is the same color as the marker. How much of this sphere is visible

depends on the local degree of bending.

## SURFACE CREATION

### ← Surfaces

A series of connected paths (either closed loops or open curves) can be combined to form a surface. Typically, a path is generated for each of several [volume data planes](#) using the settings [Place markers on data planes](#) and [Place markers while dragging](#). The stack of paths can then be joined in the orthogonal direction to create a surface. Surfaces created by **Volume Tracer** are saved in [sessions](#) and can be used to segment volume data with the command [mask](#).

Clicking **Create** uses all [selected](#) markers, or if none are selected, all markers in the [active set](#), to generate a surface. The markers must be linked into paths, and the paths used to make a given surface must be either all closed loops or all open-ended. Branched paths are not allowed, and the surface cannot form forked tubes. **Cap surface end loops** makes the surface cover the ends of tubes formed by closed-loop paths.

Each surface created by **Volume Tracer** is a single [surface piece](#) in the [surface model](#) named **Traced surfaces**. There is only one **Traced surfaces** model, but it can contain multiple [surface pieces](#) created from different collections of markers. Clicking **Create** repeatedly for the same markers will generate multiple identical [surface pieces](#). **Delete** closes (removes) any [surface pieces](#) of the **Traced surfaces** model that are [selected](#). It is necessary to delete and recreate a surface to update it after [moving a marker](#).

## MARKER FILES

A *marker file* includes marker set name(s), marker positions, marker notes, links, colors, and radii (but not [smooth paths](#) or [surfaces](#)).

Marker files can be read and written using the **Volume Tracer File menu**. In addition, they can be opened with Chimera's general file-reading mechanisms, just like other [registered file types](#). The [Chimera marker file type](#) can be indicated with the suffix **.cmm** (part of the filename) or the prefix **markers:** (not part of the filename).

The format is actually XML, a text file that can be viewed and edited with an editor and can be read or created by other programs. Libraries for parsing XML are available in many programming languages. The following marker file describes two markers connected by a link:

```
<marker_set name="marker set 1">
<marker id="1" x="-6.1267" y="17.44" z="-3.1338" radius="0.35217"/>
<marker id="2" x="1.5395" y="16.277" z="-3.0339" r="0" g="1" b="1"
radius="0.5" note="An example note"/>
<link id1="2" id2="1" r="1" g="1" b="0" radius="0.17609"/>
</marker_set>
```

The **r**, **g**, and **b** attributes are red, green, and blue color components, respectively, with possible values ranging from 0 to 1.

Extra attributes can be present within the marker, link, and marker\_set tags. While these are not used by **Volume Tracer**, they provide a mechanism for retaining information from or passing information to other programs. For example,

```
<marker id="3" x="3.23" y="4.34" z="5.45" r="0" g="0" b="1" radius=".4"
  note="105" somethingextra="hoola hoop"/>
<link id1="1" id2="2" r="0" g="1" b="0" radius=".2" leprechaun="gold"/>
```

shows the extra attributes somethingextra and leprechaun. Extra attributes included in an input file are retained and included in files written back out.

## LIMITATIONS

**May place markers on data hidden with Surface Zone.** Markers may be placed in regions of volume data that have been hidden with [Surface Zone](#). Markers land at the same locations they would if no surface zone were in effect, possibly in volume regions outside of the displayed zone.

**Transparency not saved.** Although markers and links can be transparent (when transparent [marker and link colors](#) are used), transparency is not saved in [marker files](#).

## POSSIBLE DEVELOPMENTS

- enable automatic marker creation/placement on minima
- add option to center markers on local maxima or minima in 3D
- add option to determine link colors from marker color (atom halfbond)
- optimize slice graph display so that marker placement is not slow
- allow definition of different groups of markers within a single marker set
- supply Python code for reading and writing the XML files containing saved marker sets
- make tubes using triangle lists for better resolution control and efficiency
- set curve radius automatically based on marker and/or link radius
- redraw curves automatically when curve parameters are changed
- allow curves to have a uniform color even when not all markers/links are the same color

---

UCSF Computer Graphics Laboratory / June 2014



## Volume Filter

**Volume Filter** includes several options for smoothing or transforming [volume data](#). See also: [Hide Dust](#), [vop](#), [mask](#), [segment](#)

There are [several ways to start Volume Filter](#), a tool in the **Volume Data** category (including from the [Volume Viewer Tools](#) menu).

### Filter type:

- **Gaussian** - perform [Gaussian filtering](#)
  - **Width** - one standard deviation in physical units (such as Å) of the 3D isotropic Gaussian function (the command [vop gaussian](#) allows using different widths along X,Y,Z)
  - **Value type** - whether to produce a map with the same [value type](#) as the input, 32-bit float, or 64-bit float
- **Median 3x3x3** - smooth the data by setting each value to the median of the 27 values in the enclosing 3x3x3 box of grid cells (the command [vop median](#) allows using different box sizes)
  - **Iterations** - how many cycles of value reassignment to perform
- **Bin** - reduce data size by averaging the values in NxNxN cells of the original data, where N is the **Bin size**. Different bin sizes along the different axes can be specified by entering three numbers separated by spaces.
- **Laplacian** - perform [Laplacian filtering](#)
- **Fourier Transform** - apply [Fourier transform](#)
- **Scale** - change data values (not grid point locations) and/or types:
  - **Shift** - add a constant to each value; can be negative
  - **Scale** - multiply values by a factor
  - **Value type** - cast values to a different [value type](#): int (8-, 16-, or 32-bit), uint (8-, 16-, or 32-bit), or float (32- or 64-bit)

Pressing return (Enter) in the **Shift** or **Scale** value field is equivalent to clicking **Filter**. Shifting and scaling are applied in that order. Shifting and scaling use 32-bit float to avoid truncation problems.

- **Flatten** - scale data values by factor  $(1 + a*i + b*j + c*k)$  where i,j,k are the grid indices and a,b,c are calculated to zero out the first moments of the resulting map (make its mass balance at the center of the grid)

Clicking **Filter** processes the [current set](#) of data in [Volume Viewer](#). If the current set is the result of a previous application of the same type of filtering, it will simply be updated to reflect the new settings (the original data are re-filtered with the new settings). Otherwise, a new volume data set will be created and made the current set. The new data set can be [saved to a file](#) with [Volume Viewer](#).

Clicking **Options** reveals additional settings (clicking the close button  on the right hides them again):

- **Displayed subregion only** - whether to limit the procedure to the [current display region](#) (which could be a [subregion](#) of the current set)

- **Displayed subsampling only** - whether to use only the displayed subsample (when [step](#) > 1) instead of the full resolution of the current set
- **Adjust threshold for constant volume** - adjust the contour level on the result to enclose the same volume as the contour surface on the current set
- **Immediate update** - update the result as soon as a slider parameter (Gaussian width or median iterations) is changed; only applies when the current set is the result of a previous application of the same type of filtering

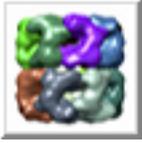
**Close** dismisses the **Volume Filter** dialog; **Help** opens this manual page in a browser window.

## TECHNICAL NOTES

**Gaussian filtering.** Convoluting the data with a Gaussian function improves the ratio of signal to noise but reduces resolution. It is fastest for data sizes that are powers of 2, and can be very slow when insufficient memory is available. It produces a map with 32-bit floating point values and uses negligible additional memory. It may be helpful to limit the input to just a subsample or subregion of the original data. Although it uses a fast Fourier transform calculation method, it does not use map periodicity. Values outside the map boundaries are treated as zero.

**Laplacian filtering.** The Laplacian operation is a sum of second derivatives. Laplacian filtering is useful for edge detection but amplifies noise, so it may be necessary to perform smoothing such as Gaussian filtering beforehand. Finite differences  $v(i-1) - 2*v(i) + v(i+1)$  along each axis are used, and voxels at the edge of the box are set to zero.

**Fourier transform.** Only the magnitudes of the complex Fourier components are included in the new data set; the phases are discarded and the constant component is set to zero. The box containing the Fourier transform (with axes in units of reciprocal space) is centered on the original data and scaled to have the same total volume. Some properties of the original data are evident from the Fourier transform. High-frequency components are near the edges of the box, low-frequency components near the center. Volume data is typically oversampled (voxel size two to three times smaller than the actual data resolution) and this causes the Fourier transform to have nonzero values only in the middle half or third of its bounding box. The missing wedge in electron microscope tomograms can also be seen. Spikes radiating along the principal axes in the Fourier transform are caused by nonperiodicity of the original data.



## Segment Map

**Segment Map** partitions [volume data](#) to create a [surface model](#) with one or more *segmentation regions* (specialized [surface pieces](#)) shown in different colors. Along with the [Fit to Segments](#) tool, **Segment Map** is part of the [Segger package](#) described in:

[Quantitative analysis of cryo-EM density map segmentation by watershed and scale-space filtering, and fitting of structures by alignment to regions.](#) Pintilie GD, Zhang J, Goddard TD, Chiu W, Gossard DC. *J Struct Biol.* 2010 Jun;170(3):427-38.

Segmentation regions can be measured with [Measure and Color Blobs](#) and colored or selected by size using [Render/Select by Attribute](#). The [Measure and Color Blobs](#) tool can also be used on the [planar caps](#) where regions are [clipped](#).

See also: [Segger documentation](#) at NCMI, [Segger how-to](#) at the Chimera website, [Volume Viewer](#), [Volume Filter](#), [Color Zone](#), [segment](#), [mask](#), [measure](#)

## Methodology

**Segment Map** performs *watershed segmentation*: a density map is partitioned so that each local maximum has its own region, and the boundaries between regions lie at the valleys between the local maxima. The goal is usually one region per protein or domain, but the initial segmentation often gives more numerous and smaller regions than desired, especially if the data are noisy. Segger provides various ways to combine regions after the initial segmentation:

- [automatic grouping](#) by *scale-space filtering* (smoothing the data), done by default
- interactive grouping [by manual selection](#)
- interactive grouping [by connectivity](#)
- interactive grouping using fitted molecules

However, it may also be necessary to smooth the data *before* segmentation, as discussed [below](#).

## Data Size and Preprocessing

For working with large density maps (> 500 Mbytes), a 64-bit version of Chimera and a machine with more than 4 Gbytes of memory are recommended.

It may be necessary to process a density map beforehand to allow segmentation to finish in a reasonable amount of time. Maps of size  $256^3$  with grid spacing about 1/3 of the map resolution are manageable on desktop computers circa 2010, while larger maps may require smoothing and binning. Calculation of 100,000 watershed regions is reasonable, but calculation of a million is unusably slow. A map of size  $256^3$  can easily have a million local maxima if it is noisy. Gaussian filtering and binning to reduce the number of

local maxima can be performed with [Volume Filter](#) or the command [vop](#). Gaussian filtering reduces resolution to about six times the Gaussian standard deviation. For visualization and analysis, it is desirable for a density map to be oversampled by three times in each dimension, so binning should aim for a grid spacing of approximately twice the Gaussian standard deviation.

If interactive rotation speed is poor after segmentation, see [response time issues](#).

## Dialog and Basic Usage

There are [several ways to start Segment Map](#), a tool in the **Volume Data** category (including from the [Volume Viewer Tools](#) menu).

The map of interest should be chosen from the pulldown menu of open volume data sets next to **Segment map**. The threshold (contour level) of the map in [Volume Viewer](#) should be adjusted before segmentation, because the calculation will use only the voxels with density values above that level.

Clicking **Segment** partitions the chosen map by the watershed method plus smoothing and grouping as specified in the [options](#). The segmentation result is shown as a [surface model](#) with one or more *segmentation regions* (specialized [surface pieces](#)) of different colors. The resulting number of regions is reported in the dialog and the [Reply Log](#). The segmentation is given the same name as the corresponding map except with `.seg` appended.

The **Current segmentation** is indicated in the dialog and can be changed by choosing a different segmentation name from the pulldown menu. Designating a segmentation as the current segmentation shows it and hides any others. Grouping and ungrouping operations modify the current segmentation model rather than creating a new one. However, multiple segmentation models can be open at the same time, and may have been generated from different density maps or opened from previously [saved](#) segmentation files.

Grouping and ungrouping can be specified interactively by [selection](#). With default [Mouse preferences](#), Ctrl-click selects a region and Shift-Ctrl-click selects additional regions without erasing the pre-existing selection.

- If no segmentation regions are selected, clicking **Ungroup** backtracks to the previous stage of grouping of the [current segmentation](#). By default, **Segment** automatically applies three rounds of smoothing and grouping; after that, the three intermediate stages could be accessed by clicking **Ungroup** three times. If regions are selected, **Ungroup** will backtrack to the previous stage of grouping for the selected region(s).
- If no segmentation regions are selected, **Group** traverses the stages in the opposite direction as **Ungroup**, or performs further grouping. If regions are selected, **Group** will combine the selected regions into a single group.

Grouping, ungrouping, and other manipulations can also be performed with the **Segment Map Regions menu**. Segmentation results are not included in saved Chimera [sessions](#), but can be saved using the **Segment Map File menu**. **Close** dismisses the **Segment Map** dialog. **Help** shows documentation in a browser window.

Clicking **Options** reveals additional settings (clicking the close button  on the right hides them again):

- **Smoothing steps [M] step size [M] voxels** - how many rounds of automatic smoothing and grouping (default  $N=3$ ) should be performed right after watershed segmentation when the **Segment** button is clicked, and how much smoothing to perform at each round (default  $M=1$ ). If  $L$  is the minimum of the grid spacing along the three axes, the standard deviation of the Gaussian used for smoothing will be  $MxL$  in the first round and increase by  $MxL$  in each subsequent round. A single round consists of smoothing the data, then letting each local maximum point from the previous round move by steepest ascent to the location of a new local maximum. If two or more previous maxima converge on a new maximum, their regions from the previous round are coalesced into a single region. In general, the number of rounds  $N$  is the dominant parameter to adjust when fewer, larger regions are desired. A larger step size  $M$  may be beneficial for noisier maps. However, a smaller step size gives more gradual changes per round, thus more rounds to achieve a similar end result, and more intermediate stages that can be traversed with the **Group** and **Ungroup** buttons. After the automatic rounds have completed, clicking **Group** executes the minimum number of additional rounds needed to change the grouping further.
- **Keep only regions having at least [minsize] voxels** - throw out regions with fewer than *minsize* voxels (default 1, retaining all regions). This applies only to the initial segmentation, not after smoothing.
- **Stop grouping if number of regions  $\leq$  [minreg]** - backtrack to the previous stage of automatic grouping if the latest round would give fewer than *minreg* regions (default 1). This is useful for generating a segmentation with no fewer regions than the known or suspected number of components in a complex.
- **Display at most [maxreg] regions** - to avoid degrading performance, limit the number of displayed regions to *maxreg* (default 2000). If a greater number of regions is generated, however, those which are not displayed will still be included in subsequent smoothing and grouping calculations. If [response time](#) is poor when a high number (approaching *maxreg*) of regions is shown, decreasing *maxreg* may help.
- **Surface granularity [trianglesize] voxels** - surfaces consist of triangles; smaller triangles give a smoother appearance, but more triangles take longer to render. The default *trianglesize* of 1 gives triangles comparable in size to the map grid spacing. A value of 2 will make the triangles twice as large (quadruple the area) and reduce their number by about a factor of 4. Increasing *trianglesize* is recommended if [response time](#) is poor.
- **Group with mouse [button]** allows grouping watershed regions connected at some density level, controlled interactively by mouse drag. Activating this option reassigns the indicated mouse button (which can be changed using the pulldown menu) from its normal function in the [Mouse preferences](#) to grouping by connectivity. If the density map has not been segmented already, the initial click on the map with the assigned button performs watershed segmentation using only the voxels with density values above the current threshold setting in [Volume Viewer](#), without subsequent smoothing/grouping. If grouping has already been performed, some ungrouping may be needed before using the connectivity method, as it will not group watershed regions together if they are already in different groups. Watershed regions continuous with the clicked density are grouped and shown in the same color. The level of density used to determine connectivity can be adjusted by mouse drag. Clicking a watershed region with the assigned button and dragging up (down) shows what regions would be continuous with the first at more (less) permissive density cutoffs. The coloring updates automatically: regions that would be connected are shown in the same color as the first, and unconnected but adjacent regions are shown in their own group colors, or if not grouped, in gray. The group being adjusted with the mouse will not extend into a pre-

existing group. Releasing the mouse button groups the matching-color regions. Clicking with the assigned button on the background displays all of the ungrouped watershed regions in the [current segmentation](#), and clicking again displays the grouped ones as well. Any previously created group can be adjusted by clicking one of its constituent watershed regions with the assigned button and dragging as described above.

## Segmentation Region Attributes

Segmenting a map generates [segmentation regions](#) and automatically assigns them several [attributes](#). These attributes can be inspected and new attributes assigned using the **Region Attributes** dialog, opened by choosing **Regions... Attributes table** from the [Segment Map menu](#). Segmentation regions can be colored or [selected](#) based on their attribute values using the [Render/Select by Attribute](#) tool. See also: [Define Attribute](#), [measure spine](#)

In the **Region Attributes** dialog, each row is a region, and each column is an attribute that can be shown or hidden using the **Columns** menu. The table can be sorted by the values in any displayed column by clicking the column header. Clicking the header once sorts the entries in order of increasing value and places an up arrowhead (triangle) in the header. Clicking again sorts the entries in decreasing order and places a down arrowhead (inverted triangle) in the header.

One or more regions (rows) in the table can be chosen with the left mouse button. A contiguous block of rows can be chosen by dragging, or by clicking on the first (or last) line and then **Shift**-clicking on the last (or first). **Ctrl**-click toggles the status of a single line. Choosing a region in the table [selects](#) it, and [selecting](#) a region in the graphics window will center and highlight the corresponding row in the table.

The automatically created attributes are read-only:

- **region** - region ID number
- **grid points** - number of density map grid points within the region
- **grouped** - number of regions grouped to form the region (next level in hierarchy, not necessarily the number of watershed regions)
- **has surface** - 1 if a surface has been made (not necessarily shown) for the region
- **contacts** - number of contacting regions
- **edge distance** - number of voxels to the nearest edge of the density map
- **bounds** (hidden by default) - grid indices bounding the region:  $i_{min}$ ,  $j_{min}$ ,  $k_{min}$ ,  $i_{max}$ ,  $j_{max}$ ,  $k_{max}$

New attributes for the chosen region(s) can be created or their values changed using:

- **Set attribute** *[attr\_name]* to value *[attr\_value]* or **[snapshot]**

Region attribute values can be integers, floating point numbers, text strings, or images, and a given attribute (column) can have multiple types of values, although only one per region (row). An *attr\_value* can be entered, or the **snapshot** button clicked to set the value to an image of the graphics window. An image can be shown at full size by clicking its miniature version in the table, and saved as PNG, JPEG, or TIFF using the **Save** button on the full-size view.

The table can filtered to show only the rows meeting some attribute criterion:

- **Filter list** [*criterion*]

The *criterion* is specified in Python syntax. Examples:

```
grid_points > 1500
note
contacts == 0 and note and "good" in note
```

where the second would list only regions with an attribute named **note** and the third would list only regions that are not in contact with other regions and that have an attribute named **note** containing the word **good**.

Clicking **Update** refreshes the **Region Attributes** dialog to reflect any changes in the [current segmentation](#) or its groupings. **Close** dismisses the dialog, and **Help** shows this manual page in a browser window.

The contents (excluding images) of the rows and columns currently shown in the filtered table can be saved in a comma-separated format using **File... Export** in the **Region Attributes** dialog menu. All of the attributes (including images) can be saved along with other segmentation information using **File... Save Segmentation** in the [Segment Map menu](#).

## Segment Map Menu

Menu entries apply only to the [current segmentation](#) unless stated otherwise.

### File

- **Open segmentation...** open a previously saved Segger segmentation file
- **Save segmentation** - save the segmentation to a Segger file, name/location previously specified
- **Save segmentation as...** save the segmentation to a Segger file, specify name/location
- **Save selected regions to .mrc file...** save density map masked by the [selected](#) segmentation regions to an MRC file (map dimensions set to the minimal box containing the regions)
- **Save all regions to .mrc file...** save density map masked by all segmentation regions collectively as an MRC file (map dimensions set to the minimal box containing the regions)
- **Save each region to .mrc file...** save density map masked by each [selected](#) segmentation region as a separate MRC file (map dimensions set to the minimal box containing the corresponding region); if no regions are selected, all are used
- **Close segmentation** - close the current segmentation model (a [surface model](#))
- **Close all segmentations except displayed** - close all hidden segmentation models
- **Close all segmentations** - close all segmentation models

### Regions

- **Show all** - show all segmentation regions ([surface pieces](#)) at the current level of grouping
- **Show only selected** - show the [selected](#) segmentation regions, hide all others

- **Show adjacent** - show regions in contact with the [selected](#) regions
- **Show grouping** - show the watershed regions within groups rather than the larger group surfaces; affects the [selected](#) regions, or if none are selected, all regions; does not change the groupings, only how they are shown
- **Unshow grouping** - show the larger group surfaces rather than the watershed regions; affects the [selected](#) regions, or if none are selected, all regions
- **Hide** - undisplay the [selected](#) regions, or if none are selected, all regions
- **Make transparent** - make 55% transparent the [selected](#) regions, or if none are selected, all regions
- **Make opaque** - make opaque (0% transparent) the [selected](#) regions, or if none are selected, all regions
- **Color density map** - color the density map display to match all regions (subsequently changing the threshold will erase the coloring for [volume displays](#) in the **surface** or **mesh** but not **solid** style)
- **Select groups** - select all regions composed of grouped watershed regions
- **Select boundary regions** - select regions that contain at least one grid point within three voxels of the edge of the density map
- **Invert selection** - deselect the [selected](#) regions and *vice versa*
- **Group selected** - group the [selected](#) regions
- **Ungroup selected** - ungroup the [selected](#) regions
- **Smooth and group** - execute the minimum number of rounds of [smoothing and grouping](#) needed to produce further grouping (equivalent to clicking the **Group** button with nothing selected)
- **Delete selected** - delete the [selected](#) regions (cannot be undone)
- **Enclosed volume** - report the approximate total volume enclosed by the [selected](#) regions (number of grid points and  $\text{\AA}^3$  value obtained by multiplying the number of grid points by the voxel volume); more precise measurements can be obtained with [Measure and Color Blobs](#)
- **Mean and SD** - report density map mean and standard deviation within each [selected](#) region of the [current segmentation](#), which could be based on a binned version of the [chosen map](#)
- **Mask map with selected** - mask the density map with the [selected](#) regions to create a new map (copy the density values inside the regions, set values elsewhere to zero, but maintain the original map dimensions)
- **Show axes for selected** - show principal axes of inertia as arrows for each [selected](#) region
- **Hide all axes** - remove all principal axes arrows
- **Attributes table...** show table of [region attributes](#)

## Response Time Issues

**High number of surface triangles.** Surfaces are drawn as many small triangles. The number of surface triangles can be reduced by increasing the [granularity](#) (triangle size). Doubling the value will reduce the number of triangles by approximately a factor of 4 and may improve rendering speed by a similar factor. Also, the display can be limited to a specified [maximum number of regions](#).

**Selected surfaces.** Chimera shows [selections](#) with a green outline (assuming default [Selection preferences](#)). Selection outlines can increase the time to draw a single frame as much as fivefold. If rotation is slow, avoid having objects selected when the selection is not needed.

**Mac graphics speed.** Rendering many small segmentation surfaces on Mac OS 10.6 is about three times slower than on other platforms (Linux, Windows). As mentioned above, the display can be limited to a

specified [maximum number of regions](#). Also, increasing the [surface granularity](#) will improve rendering speed. The poor performance on the Mac is because Chimera has disabled use of the fast rendering technique of OpenGL vertex buffer objects for small surfaces, as it causes minute-long freezes on the Mac when over 5000 segmentation surfaces are shown.

## MultiFit

**MultiFit** performs simultaneous rigid fitting of multiple atomic-resolution structures into [density maps](#) at resolutions as low as 25 Å. Multiple copies of a structure can be fit [assuming cyclic symmetry](#), or multiple different structures can be fit [without symmetry constraints](#). The method is described in:

[Determining macromolecular assembly structures by molecular docking and fitting into an electron density map](#). Lasker K, Sali A, Wolfson HJ. *Proteins*. 2010 Nov 15;78(15):3205-11.

[Inferential optimization for simultaneous fitting of multiple components into a CryoEM map of their assembly](#). Lasker K, Topf M, Sali A, Wolfson HJ. *J Mol Biol*. 2009 Apr 24;388(1):180-94.

The program **MultiFit** optimizes a score based on the quality of fit of the structures in the map, protrusions of the structures from the map envelope, and shape complementarity between pairs of structures. Chimera provides a graphical interface to running this program via a web service hosted by the [UCSF RBVI](#).

See also: [Volume Viewer](#), [Fit in Map](#), [Fit to Segments](#), [fitmap](#), [measure symmetry](#)

There are [several ways to start MultiFit](#), a tool in the **Volume Data** category. The atomic structure(s) and density map of interest should be open in Chimera and designated as described below.

- If using **Cyclic symmetry**:

**Fit [*N*] copies of [*molecule-model / selected atoms*] in map [*map-model*]**

The stoichiometry is specified directly as *N* copies of the set of input atoms, each copy to be treated as a single rigid body. For example, if *N* is 7 and the input set of atoms has stoichiometry  $\alpha_2\beta$ , the result will contain 7 copies of the  $\alpha_2\beta$  module, without changes in the  $\alpha$ - $\alpha$  or  $\alpha$ - $\beta$  interactions within the module.

- If using **No symmetry**:

The **models to fit** should be chosen (highlighted) in the table with the mouse. A model should be opened multiple times if the assembly is thought to contain more than one copy of that structure. A block of models can be chosen by dragging, or by clicking on the first (or last) line in the desired block and then **Shift**-clicking on its last (or first) line. **Ctrl**-click toggles the state (chosen or not) of a single line. The table includes:

- model **ID** number
- model color (a [color well](#))
- whether **Active** ([movable](#))
- whether **Shown** (display-enabled)
- **Fit globally** indicates whether to search translations all over the density map and all

orientations instead of only shifts of up to  $\pm 15$  Å in each direction and rotations of up to  $\pm 30^\circ$  from a model's current position. The translational (shift) search increment is the map voxel size, whereas the rotational increment is set in the **MultiFit** [options](#). Global fitting may take significantly longer than local fitting.

- o **Model** name

Clicking **Calculate anchor graph** performs an initial assessment (also via the web service) of which regions in the density map could accommodate models. One sphere per chosen model is placed in the map to guide placement by the user prior to local fitting. The anchor graph is created as a [marker model](#). The anchor graph will be replaced if the calculation is rerun.

Results also depend on:

- the contour level ([surface display threshold](#)) of the map in [Volume Viewer](#), which sets the size and shape of the map envelope
- certain **MultiFit** [options](#)

Clicking **Fit** runs the calculation as a background task. Clicking the information icon  in the Chimera [status line](#) will bring up the [Task Panel](#), in which the job can be canceled if desired. **Reset** deletes any existing results; otherwise, new results will be appended to those already in the dialog (rather than replacing them automatically). **Close** dismisses the dialog. **Help** opens this manual page in a browser window.

Clicking the **Options** button reveals additional settings (clicking the close button  on the right hides them again):

- **Number of output assemblies** (default **10**) - how many solutions to the fitting problem to return to the user, after ranking by score
- **Resolution of the density map (Å)** (default **10.0**) - should be set to the resolution of the chosen experimental map; the same value is used to simulate density from the atomic coordinates, to evaluate their quality of fit to the experimental map
- **Sampling angle (15 to 270°)** (default **30**) - rotational increment for sampling model orientations
- **Output file location (optional)** - local directory in which to place files used or produced by the calculation, including the most recent set of results, stderr, and stdout files

## MultiFit Results

When the calculation finishes, the results are listed in a table in the **MultiFit** dialog. Each row in the table represents a solution to the fitting problem; the top 101 solutions are returned. A single solution consists of multiple models (the  $N$  copies of the input set of atoms) in their fitted positions. One or more rows can be chosen with the left mouse button to display the corresponding solution(s) in the main Chimera window. **Ctrl-click** toggles the state (chosen or not) of a single row. Columns in the table:

- **Index** - a numerical identifier, simply assigned starting with zero for the best solution found
- **c** - structure color (a [color well](#), which also allows changing the color)
- **Correlation** - cross-correlation value representing the quality of fit of the structures in the map
- **Name** - auto-generated solution name; models within a solution are named by appending an

underscore and additional integer to the solution name

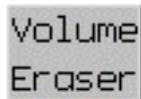
The solution list can be sorted by the values in any displayed column by clicking the column header. Clicking the header once sorts the entries in order of increasing value and places an up arrowhead (triangle) in the header. Clicking again sorts the entries in decreasing order and places a down arrowhead (inverted triangle) in the header.

Currently, **MultiFit** results are not saved in [sessions](#), other than the models displayed at the time of saving. To save a solution:

1. click its row in the **MultiFit** dialog to display its models
2. use [File... Save PDB](#) in the Chimera menu to save the models as one or more PDB files **relative to** the density map model

---

UCSF Computer Graphics Laboratory / August 2011

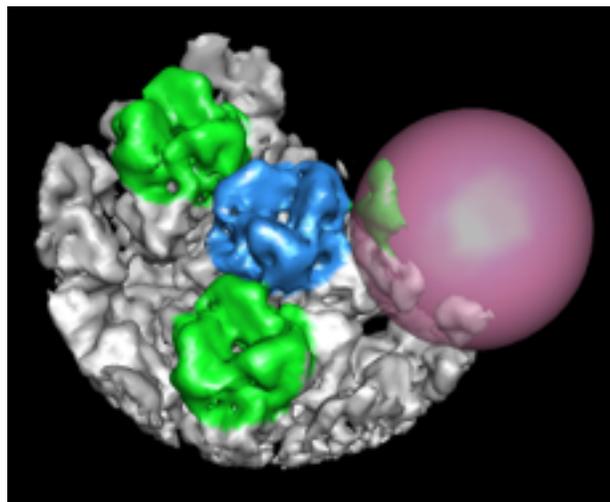


## Volume Eraser

**Volume Eraser** allows parts of [volume data](#) maps to be zeroed out interactively. After the **Volume Eraser** sphere has been moved to a desired location, data values for all points inside the sphere can be set to zero. The modified data can be [saved to a file](#) with [Volume Viewer](#). See also: [Keyboard Shortcuts](#), [mask](#), [vop](#)

There are [several ways to start Volume Eraser](#), a tool in the **Volume Data** category (including from the [Volume Viewer Tools](#) menu).

**Show volume-erasing sphere** controls whether the sphere is shown and can be moved with the mouse. This option is automatically activated when **Volume Eraser** is started and deactivated when the tool is closed or iconified. Upon activation, the sphere will appear in the center of the screen at the frontmost face of the [current display region](#) of data in [Volume Viewer](#). The position of the sphere is forgotten upon deactivation.



The sphere's **Color** can be adjusted by clicking the [color well](#) and using the [Color Editor](#). The sphere's **Radius** can be adjusted by entering a value or moving the slider. The radius is initially set to one-tenth of the largest dimension of the [current display region](#) of volume data.

**Use mouse [button] to move sphere** controls which mouse button is used to move the sphere. The indicated mouse button, which can be changed using the pulldown menu, is reassigned from its existing function to sphere movement. Combining the indicated button with Shift moves the sphere in the Z dimension (perpendicular to the screen); dragging down/left brings the sphere forward and dragging up/right moves it away.

Clicking **Erase** zeroes the sphere-enclosed portion of the volume data. There is no undo capability. However, a copy of the [current display region](#) is automatically generated in the first cycle of erasure, and erasures apply only to that copy. The copy is named after the original data; if the original data is named **data123**, the copy will be named **Writable copy of data123**. The original data is not affected, allowing the process to be restarted if a mistake is made.

After one or more cycles of erasure, the modified data can be [saved to a file](#) with [Volume Viewer](#).

**Close** closes the **Volume Eraser** interface. **Help** opens this manual page in a browser window.

## Keyboard Shortcuts

Several [accelerators](#) (keyboard shortcuts) are available for [editing volume data](#), including:

- **es** - erase inside the sphere
- **eo** - erase outside the sphere
- **wv** - create a writable copy of the [current display region](#) in [Volume Viewer](#) (rarely needed because a copy is created automatically in the first erasure cycle)

Accelerators are disabled by default. One way to enable their use is with the command [ac](#).

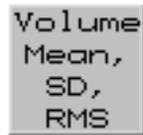
## Technical Notes

**The sphere is implemented as a marker.** The **Volume Eraser** sphere is implemented as a [marker](#), like those created by [Volume Tracer](#). Moving the sphere is equivalent to [moving a marker](#).

**Hidden data is also erased.** Data can be erased even if it is hidden by clipping planes, zoning (with [Surface Zone](#) or [zoning](#) in [Volume Viewer](#)), or [subregion selection](#) in [Volume Viewer](#).

---

UCSF Computer Graphics Laboratory / September 2008

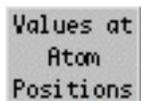


## Volume Mean, SD, RMS

**Volume Mean, SD, RMS** calculates statistics for the [current set](#) of data in [Volume Viewer](#).

There are [several ways to call Volume Mean, SD, RMS](#), a tool in the **Volume Data** category (including from the [Volume Viewer Tools](#) menu). It is also implemented as the command [measure mapStats](#). For normalizing the data, see [vop scale](#).

Calling the tool calculates the mean, the standard deviation (SD) from the mean, and the root-mean-square (RMS) deviation from zero for the [current set](#) of data and reports the results in the [Reply Log](#). Only data for the current [subregion](#) and [step size](#) are used. SD and RMS are both root-mean-square deviations, but the SD is calculated relative to the average value (mean) whereas the RMS is calculated relative to zero.



## Values at Atom Positions

**Values at Atom Positions** maps [volume data](#) to atom positions and assigns the values as an atom [attribute](#). See also: [Fit in Map](#), [Rotamers](#), [Surface Color](#), [measure mapValues](#)

There are [several ways to start Values at Atom Positions](#), a tool in the **Volume Data** category (including from the [Volume Viewer Tools](#) menu). The **Molecule** (containing the atoms onto which data values will be mapped) should be chosen from the list of open molecule models. The **Volume data** should be chosen from the list of open volume data sets. **Browse...** can be used to locate and open a volume data file.

The [attribute](#) will be named `value_name`, where *name* is the name of the volume data set (except with any characters other than numbers and letters changed to underscores). For example, an attribute based on the volume data file `default.phi` will be named `value_default_phi`.

Values of the attribute are assigned to all atoms in the chosen molecule model. The transformed coordinates of the molecule model and the volume data (but see [below](#)) are used to map data values to atoms. For each atom within the bounds of the volume data, the value is found by trilinear interpolation from the eight corners of the enclosing volume grid cell. Atoms outside the bounds of the volume data are assigned a value of zero.

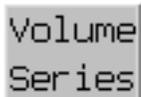
Clicking **Histogram** assigns the values and opens [Render/Select by Attribute](#), which includes a histogram of the values.

**Close** dismisses the dialog, and **Help** opens this manual page in a browser window.

### LIMITATIONS

**Values not saved in sessions.** The new attribute and associated values are not saved in [session](#) files.

**Volume data must be displayed before it can be transformed separately.** There is no volume model until the data is displayed with [Volume Viewer](#). When there is no volume model, the transformation of the volume data is assumed to be the same as that of the chosen molecule model. To use a different transformation for data mapping, the volume data must be displayed and the volume model moved while the molecule model is held fixed (or *vice versa*).



## Volume Series

**Volume Series** displays an ordered sequence of [volume data](#) sets. A sequence could represent a time series or a comparison of states not necessarily ordered in time. See also: [Morph Map](#), [making movies](#)

There are [several ways to start Volume Series](#), a tool in the **Volume Data** category (including from the [Volume Viewer Tools](#) menu). It is also implemented as the command [vseries](#).

Clicking **Open...** raises a file browser for opening a volume series as either of the following:

- multiple [volume data](#) files sorted by name
- a single [volume series](#) file in Priism time series format

A Priism time series file can also be opened in [other ways](#), which will automatically start the **Volume Series** tool. Opening a volume series of any type also starts [Volume Viewer](#). Each member of the series is treated as a separate set of data in in [Volume Viewer](#).

Multiple volume series may co-exist in Chimera. A volume series is named after the single file it was read from, or the first file if read from multiple files. Choosing the name of a volume series from the **Data** listing switches the display to that series, and it can be closed with **Close Series**. Similarly, choosing **All** allows simultaneous display or removal of all of the volume series in Chimera.

The members of a series are indexed 0, 1, 2, ... and these indices are referred to as the **Time**. Any numbers in the individual filenames are not used, except to sort the names.

### APPEARANCE AND PLAYBACK

The **Time** slider can be moved manually to show a particular set of data and make it the [current set](#) of data in [Volume Viewer](#). Settings in [Volume Viewer](#) ([display style](#), [step size](#), [threshold level/color](#), [surface smoothing](#), *etc.*) control its appearance and will be applied automatically to other series members as they are displayed. For efficient playback, however, the [Volume Viewer](#) dialog will not update to show current settings until the [Time](#) value has not changed for 1 second.

**Normalize threshold levels** adjusts the [thresholds](#) (contour levels) to keep the enclosed volume constant throughout the series. This is useful when the signal level in the data changes over time or between states.

Clicking **Play** cycles through the series in the specified manner:

- **forward** - in increasing order, then back to the first (01230123...)
- **backward** - in decreasing order, then back to the last (32103210...)
- **oscillate** - alternating increasing and decreasing order (01232101...)

Clicking **Stop** halts playback. By default, playback is as fast as possible, which can be fairly slow for large data. If playback is too fast, it can be slowed with the **Maximum playback speed** option. If playback is too slow, it may be helpful to:

- reduce the size of the data by increasing its [step size](#) and/or limiting the displayed [region](#) in [Volume Viewer](#)
- adjust caching parameters:
  - **Data cache size (Mb)** [*size*] (default as specified in the [data display options](#) of [Volume Viewer](#)) - how much memory to allot for volume data. Increasing the value to retain more or all of the volume data can speed playback because it decreases how much time is spent reading the data from disk. However, it is important not to increase the value to exceed or come close to the computer's physical memory, as that will result in swapping of data to disk and poor performance. Clicking **Current use** shows how much of the data cache is being used in total and by each data set.
  - **Cache [N] renderings** - store volume rendering (surface triangle or solid voxel) information for the *N* most recent displays. This can speed playback because less time is spent recalculating display information. This information is stored separately from the data cache, but the data cache is used to store the *N* associated sets of volume data (even if that exceeds the specified [data cache size](#)). There is no hard limit to the memory used to cache rendering information. Surface renderings use memory proportional to the number of triangles composing the surface. Solid renderings use memory proportional to the number of data voxels displayed, and it is generally only feasible to cache solid display information for small data sets.

## MARKERS AND PATHS

**Show markers** indicates whether to display markers previously created with [Volume Tracer](#) to trace spatial and temporal paths. Each marker is labeled with the [time](#) of the data set on which it was placed. The labels are not shown. They are used by **Volume Series** to display only the markers for the currently displayed [time](#). Markers for specified numbers of earlier and later time points can also be shown.

**Volume Series** controls the display of markers in the [active set](#) (current set) of markers in [Volume Tracer](#). Using [Volume Tracer](#),

- multiple marker sets can be created and any one chosen for display
- the current marker set can be [saved](#) to a [file](#) for later use

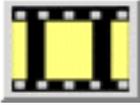
The **Color zone around markers** option colors volume contour surfaces to match markers within a specified **range**. All of the markers associated with the current [time](#) are used to color the current surface. The coloring does not apply to solid displays.

## NOTES

**No session support.** The state of **Volume Series** is not included in [session](#) files. A saved and restored session will include the volume data sets in [Volume Viewer](#), but the data sets will not be associated with **Volume Series**, and there is no way to specify that they should be. However, [markers and paths](#) can be saved for later use.

**Slow playback when caching renderings.** The second time a surface is displayed, a graphics optimization (building an OpenGL display list) is done to speed up future drawing of that same surface by putting it in graphics card memory. That optimization is time-consuming. The third time the surface is displayed will take much less time if the graphics card has sufficient memory.

**Threshold normalization brightness glitches in solid mode.** Sometimes a data set will appear dimmer than the preceding or following sets when solid rendering and [threshold normalization](#) are used. This can happen when only two nodes are used to specify the transfer function on the [Volume Viewer histogram](#), with one of the nodes at the highest data value. If a particular data set has a maximum data value much higher than the adjacent sets, that will substantially lower the brightness at typical data values. This can be remedied by adding a third node between the high and low nodes by Ctrl-clicking on the [Volume Viewer histogram](#).



## Movie Recorder

**Movie Recorder** captures image frames from Chimera and assembles them into a movie file. Movies of trajectories can be recorded with [MD Movie](#). See also: [Animation](#), [making movies](#), [tips on preparing images](#)

There are [several ways to start Movie Recorder](#), a tool in the **Utilities** category. It is also implemented as the command [movie](#).

Any desired adjustments in the window size or [Frame Options](#) should be made before recording is started. The window size is reported as **Resolution** in the status area of the dialog. The window can be resized manually or with the command [windowsize](#). On certain platforms, other windows should not overlap the Chimera graphics window during recording ([details...](#)). This is not an issue when [raytracing](#) is used.

On the **Movie Recorder** dialog, clicking **Record** initiates frame capture and changes the button to **Stop**. Clicking **Stop** halts frame capture. Frame capture can be restarted by clicking **Record** again. This can be repeated until the desired segments have been saved. **Reset** resets the frame counter to zero and (unless [Save images on Reset](#) has been set to **true**) deletes the frames.

Clicking **Make movie** initiates encoding the saved frames into a movie file. During encoding, the button changes to **Cancel movie**, which can be clicked to halt the process. When the movie is fully assembled, a [reset](#) will occur unless **Reset after encode** has been turned off. Output settings:

- **Output format** - output movie file format
  - H.264 [.mp4]
  - VP8/WebM [.webm]
  - Theora [.ogv]
  - Quicktime [.mov]
  - AVI MSMPEG-4v2 [.avi]
  - MPEG-4 [.mp4]
  - MPEG-2 [.mpg]
  - MPEG-1 [.mpg]
  - WMV2 [.wmv]
- **Output file** - output movie file name and location

### Frame Options:

- **Directory** - image file location
- **Filename pattern** - string for naming image files
- **Format** - image file format (however, PNG will be used regardless of this setting if [raytracing](#) is done):
  - JPEG [.jpeg]
  - PNG [.png]
  - PPM [.ppm] (default)

- **Rendering:**
  - **screen grab** (default) - capture the Chimera window contents; render onscreen. This rendering option is the fastest, but image quality is no better than the display. The graphics window must be kept completely visible: not overlapped by other windows or menus, moved offscreen, or blanked out by a screensaver.
  - **Chimera** - Chimera rendering, normally offscreen ([details...](#)). Images can be supersampled, that is, initially generated at a higher resolution and then sampled down to the final size.
    - **Supersample** (1x1/2x2/**3x3**/4x4) - how many pixels to sample in the X and Y dimensions for each pixel in the final saved image; thus, 1x1 corresponds to no supersampling. Higher values increase the smoothness of edges in saved images and increase calculation time with little effect on file size. 3x3 is generally recommended when supersampling is done.
  - **POV-Ray** - [raytrace with POV-Ray](#). This rendering option is the slowest but includes fancier effects such as high-quality shadows. The **POV-Ray Options** button opens the corresponding [preferences](#).
- **Save images on Reset** (true/false) - whether to keep individual image files when a [reset](#) occurs

### Movie Options:

- **Bit rate controls quality and file size** - a higher bit rate corresponds to higher quality (better appearance) but also a larger movie file, assuming the same window size and [frame rate](#). A movie can be encoded to play back at a constant or variable bit rate.
  - **Variable bit rate** (default **on**); **Quality:** (highest/higher/high/**good**/medium/fair/low)
  - **Constant bit rate (Kbits/s)** (default **off** but set to **2000**)
- **Frames per second** (default **25**) - movie playback rate in terms of image frames
- **Play forward then backward** (off by default) - whether to include the frames in reverse order as the second half of the movie

**Close** dismisses the **Movie Recorder** dialog. **Image Tips** shows the [tips on preparing images](#), and **Help** opens this manual page in a browser window.



## Notepad

**Notepad** allows creating *session notes*, descriptive text that is saved along with a [session](#).

There are [several ways to start Notepad](#), a tool in the **Utilities** category. Text and [symbols](#) can be entered directly or pasted into the window. **Clear** erases the window contents, whereas **Close** dismisses **Notepad** without altering its contents. **Help** opens this manual page in a browser window.

Session notes can also be created and/or edited within the [File... Save Session As...](#) dialog, which further allows saving a thumbnail image for the session. The thumbnail image and notes for a session can be viewed later (without actually restoring the session) by simply clicking the session file name in the [File... Restore Session...](#) dialog.

See also: [model notes](#) in the [Model Panel](#), [alignment annotations](#) in [Multalign Viewer](#)

---

UCSF Computer Graphics Laboratory / February 2013



## Palette Editor

A *palette* is an ordered set of colors and associated method of interpolation.

The **Palette Editor** allows creating custom palettes and choosing a palette from numerous preset options and any previously created custom palettes. See also: the [Color Editor](#), [background](#)

There are [several ways to start](#) the **Palette Editor**, a tool in the **Utilities** category. In addition, clicking a *palette well* (analogous to a [color well](#) but rectangular rather than square, such as used for a [gradient](#) in the [Background preferences](#)) opens the **Palette Editor**.

**Interpolation options:**

- **HLS** (default) - in hue-lightness-saturation (colorwheel) color space
- **RGB** - in red-green-blue color space
- **discrete** - no interpolation, sharp transitions from one color to another

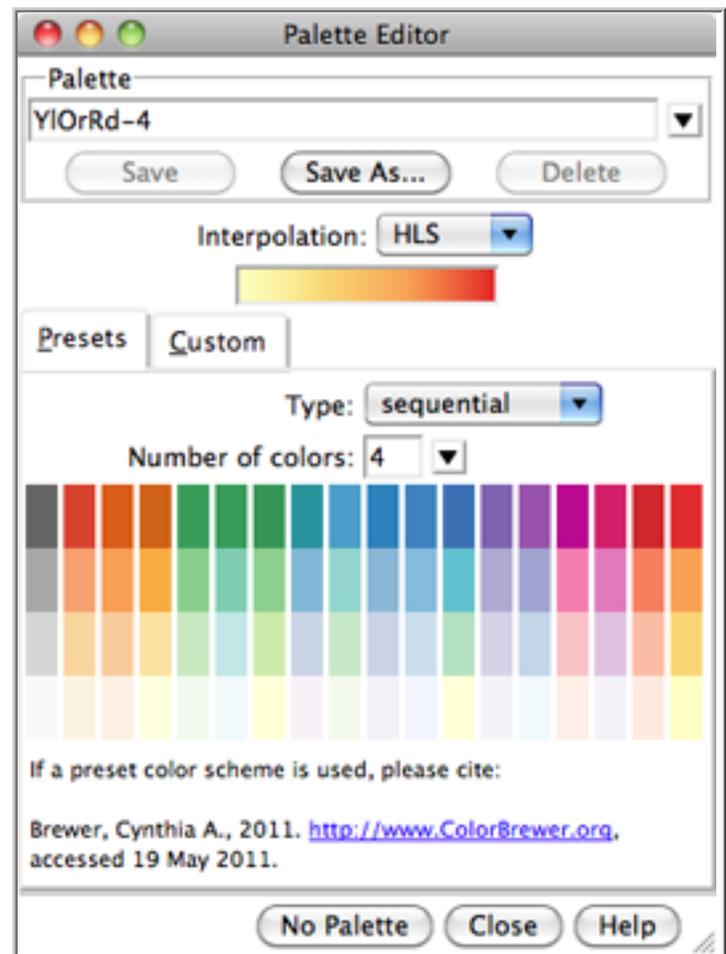
The **Presets** section provides access to many built-in color schemes. The schemes were copied from [ColorBrewer](#) on May 19, 2011, and should be credited as follows:

Brewer, Cynthia A., 2011. <http://www.ColorBrewer.org>, accessed 19 May 2011.

The preset schemes are categorized by **Type** based on the data they were designed to represent:

- **sequential** - unsigned ordered data (magnitude comparisons)
- **diverging** - signed ordered data or other situations where the two extremes are to be emphasized equally
- **qualitative** - categories or groupings without magnitude comparisons

The **Custom** section provides a user-specified number of [color wells](#) (up to 12) for defining a new palette. Clicking a color well brings up the [Color Editor](#) for interactive color definition. The colors together with the method of [interpolation](#) comprise a palette. Palettes can be named, saved, and later retrieved from the pulldown list indicated by the black inverted triangle near the top of the dialog. A built-in palette (**Chimera default** or any of the presets) cannot be overwritten or deleted, but it can be saved to a new name. Custom palettes are saved in the Chimera [preferences file](#), and are only updated with any changes



when **Save**, **Save As**, or **Delete** is used.

The **Help** button brings up this manual page in a browser window. **Close** dismisses the **Palette Editor**.

## ReadStdin ReadStdin

**ReadStdin** allows entry of [Chimera commands](#) through standard input (*stdin*). Messages in response are sent to standard output (*stdout*). Note [RESTServer](#) is designed as a replacement for **ReadStdin**. See also: [listen](#), [list](#)

Independent of **ReadStdin**, if Chimera is started in [nogui mode](#) without command/script file input, a prompt will be supplied in the system shell for entering [Chimera commands](#) on standard input.

There are [several ways to start ReadStdin](#), a tool in the **Utilities** category. After **ReadStdin** has been started, text from *stdin* will be interpreted as Chimera [commands](#). Messages resulting from the execution of those commands will be sent to *stdout* instead of the [Reply Log](#) (although certain messages that require an answer will still raise a dialog).

Each command string entered via *stdin* will be echoed to *stdout* with "CMD " appended at the front. Messages resulting from execution of the command will be sent to *stdout*, followed by "END" to denote the end of output for that command. End-of-output signaling is useful for synchronizing an outside program with Chimera. **ReadStdin** can be halted with an end-of-file character (Ctrl-d on Unix).

Messages resulting from operations performed with the menu, graphical interfaces, or commands entered directly in the [Command Line](#) will still be sent to the [Reply Log](#), not *stdout*.



## Reply Log

The **Reply Log** contains informational, warning, and error messages from Chimera. Error messages are in red and warnings are in blue italics. When **IDLE** is open, however, messages are sent there instead of to the **Reply Log**. See also the [Messages preferences](#).

There are [several ways to start](#) the **Reply Log**, a tool in the **Utilities** category.

**Clear** erases the log. **Copy** copies just the selected portion so that it can be pasted into another application window, while **Save** brings up a dialog for saving the entire contents to a file. The log can be searched for a string of interest, and the **Forward** and **Back** buttons can be used to navigate between multiple occurrences of the string.

**Close** dismisses the **Reply Log**, and **Help** opens this manual page in a browser window.

## RESTServer

**RESTServer** allows execution of [Chimera commands](#) through a REST (REpresentational State Transfer) interface. See also: [listen](#), [list](#)

There are [several ways to start RESTServer](#), a tool in the **Utilities** category. When **RESTServer** is started, it prints a message of the form

```
REST server on host 127.0.0.1 port port_number
```

If **RESTServer** is launched by specifying `--start` on the Chimera command line, the output is sent to *stdout*; this is particularly useful for programs that want to launch and control Chimera programatically. Requests may then be sent to Chimera via URLs that start with `http://localhost:port_number/`. Chimera supports two types of requests: predefined content supplied by Chimera and command execution.

Predefined content may be accessed by appending the path in the table below to the URL `http://localhost:port_number/static/`:

Path	Content
<code>cmdline.html</code>	An interface for executing commands in Chimera. Typically, this page is accessed from a browser; the user can type in commands and see the replies in the web page.

To execute commands, requests should be sent to the URL `http://localhost:port_number/run` with the Chimera [commands](#) encoded in one or more `command` parameters. Chimera can process either the GET or POST method in the HTTP request. Messages resulting from the sequential execution of these commands will be sent as **text/plain** reply to the request instead of the [Reply Log](#) (although certain messages that require an answer will still raise a dialog).

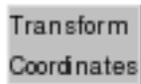
**RESTServer** is designed as a replacement for [ReadStdin](#). **ReadStdin** is frequently used in conjunction with the [listen](#) command. With both **ReadStdin** and **listen** generating output to *stdout*, a program parsing Chimera responses needs to demultiplex the output into separate sources. **RESTServer** uses its network connections strictly for command execution and returning command output. The [listen](#) command has been augmented to take a `rest` option to send notification messages to a REST server instead of the standard channels. By using separate communication channels for notification and command execution, a program that controls Chimera remotely need not handle possibly mixed output from different sources.



**Undo Move** allows backtracking through the most recent model positions. This is useful when a structural alignment or docking of models relative to one another is accidentally destroyed by moving one or more models when the others are fixed. **Redo Move** allows previously undone movements to be reapplied. There are [several ways to call](#) **Undo Move** and **Redo Move**, tools in the **Movement** category.

As soon Chimera is started, the positions of all open models are stored whenever motion stops for at least one second, a new model is opened, or **Undo Move** or **Redo Move** is used. The many intermediate positions that occur during motions are not recorded. Only rotations and translations are recorded, not scale, clipping plane positions, or other [Camera](#) settings. Only the 100 most recent sets of positions are saved.

When **Undo Move** is used to regenerate positions saved before a model was opened, that model is not moved.



## Transform Coordinates

**Transform Coordinates** applies a specified rotation and translation to a model.

There are [several ways to start Transform Coordinates](#), a tool in the **Movement** category.

The **Model** should be chosen from the list of open models. Its transformation is specified with:

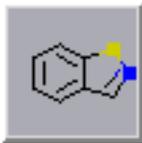
- **Euler angles [ $\varphi$   $\theta$   $\psi$ ]** - rotations defined by the [x-convention](#), where the first rotation is by an angle  $\varphi$  about the Z-axis, the second is by an angle  $\theta$  (potentially ranging from 0 to  $\pi$ ) about the new X-axis, and the third is by an angle  $\psi$  about the new Z-axis.
- **Shift [X Y Z]** - translations along X, Y, and Z

For a molecule model (atomic structure or [marker set](#)):

- **Move atoms instead of coordinate axes** - whether to change the atomic coordinates of the model rather than its transformation matrix

Clicking **Apply** performs the rotation and translation. The cumulative change from the original transformation is reported in the dialog. **Reset** restores the original transformation. **Set** is equivalent to clicking **Reset** and then **Apply**.

Cumulative change information and original coordinates are not retained in saved [sessions](#).



## Structure Diagram

**Structure Diagram** generates 2D images of small organic molecules using a web service provided by the [UCSF Resource for Biocomputing, Visualization, and Informatics \(RBVI\)](#). This web service is implemented in Java using the [Chemistry Development Kit](#).

Use is only recommended for *small* organic molecules, on the order of 50 or fewer nonhydrogen atoms. See also: [ViewDock](#), using [open](#) to [fetch](#) by SMILES string or PubChem CID.

There are [several ways to start Structure Diagram](#), a tool in the **Utilities** category.

Information sent to the server can be a [SMILES](#) string or a 3D structure (**Molecule**) already open in Chimera. Clicking **Apply** sends the information to the server and displays the resulting 2D diagram (image in PNG format). Hydrogen atoms are inferred by the server. Image dimensions are set by the size of the **Structure Diagram** dialog when **Apply** is clicked, and the PNG file can be saved to disk with **Save**. **Close** dismisses the dialog, while **Help** opens this manual page in a browser window.

---

UCSF Computer Graphics Laboratory / July 2010

**Usage:****2dlabels** *action arguments***Usage:****2dlabels write** *filename***Usage:****2dlabels read** *filename*

**2dlabels** is the command-line implementation of [2D Labels](#), which allows adding text, [symbols](#), and arrows to the display. This is useful for annotating [images](#) and [movies](#). The labels exist in the X,Y plane (the plane of the screen) and do not have a “depth” (Z-coordinate); they appear in front of any graphics displays and will not move when models are moved. 2D labels can be of multiple colors and sizes, and they can be faded in or out. See the [video mini-example](#). See also: [label](#), [colorkey](#), [labelopt](#), [rlabel](#), [windowsize](#), [movie-related commands](#)

Text and symbols are laid out horizontally, whereas the arrows can be oriented in any direction in the plane. (Although symbols also include some arrows in specific orientations, they cannot be oriented arbitrarily.) For 3-dimensional arrows, see [BILD format](#).

2D label information is saved in [sessions](#). 2D labels can also be saved (collectively) to a separate [label file](#) with the **write** keyword or imported from such a file with **read**. These keywords cannot be truncated.

2D labels created with the [2D Labels](#) graphical interface are assigned names **label2d\_id\_0**, **label2d\_id\_1**, **label2d\_id\_2**, *etc.* in order of creation, allowing their subsequent modification with the **2dlabels** command.

The *action* can be one of the following:

- [create](#) - create a 2D text/symbol label
- [change](#) - modify a 2D text/symbol label
- [delete](#) - delete a 2D text/symbol label
- [acreate](#) (synonym **arrowcreate**) - create a 2D arrow
- [achange](#) (synonym **arrowchange**) - modify a 2D arrow
- [adelete](#) (synonym **arrowdelete**) - delete a 2D arrow

The action keyword cannot be truncated. Argument keywords (given below) can be truncated, and their case is not important. Default values are indicated with **bold**. A vertical bar “|” designates mutually exclusive values.

- **2dlabels create** *name* [ **text** *label-contents* ] [ **color** *label-color* ] [ **size** *font-size* ] [ **style** *font-style* ] [ **typeface** *font-typeface* ] [ **xpos** *x-position* ] [ **ypos** *y-position* ] [ **bgColor** *rectangle-color* ] [ **margin** *rectangle-padding* ] [ **outline** *rectangle-outline-width* ] [ **visibility** *hide* | **show** ]

The *name* is required and serves as an identifier for subsequently modifying or deleting the 2D text/symbol label. The *label-contents* (default nothing) can include letters, numbers, symbols from the keyboard; additional symbols can be copied (for example, from [here](#)) and pasted into the command line, or indicated with [unicode](#) character codes or character names. Unicode examples:

- text 'the \u03b1 subunit' gives the  $\alpha$  subunit
- text 'reactants \N{RIGHTWARDS ARROW} products' gives reactants  $\rightarrow$  products

Spaces and quote marks can be included by enclosing the entire *label-contents* in single or double quote marks. Carriage returns are indicated with `\n`. The *label-color* can be any [color name](#) that specifies a single color (default **white**). Any transparency in the color is ignored. The *font-size* must be an integer greater than zero (default **24**). The *font-style* can be **normal** (default), **bold**, **italic**, or **bold italic**, where the latter should be enclosed in single or double quotation marks. The *font-typeface* can be **serif**, **fixed**, or **sans serif** (default), where the latter should be enclosed in single or double quotation marks. The *x-position* and *y-position* are the X and Y coordinates of the lower lefthand corner of the label (defaults are **0.5** and **0.5**). The X axis is horizontal and the Y axis is vertical, and both range from 0 to 1 in the visible portion of the window. It is possible to place a label partially or completely outside the visible area, however. A label background (colored rectangle under the label text) can be shown with the **bgColor** option by setting *rectangle-color* to any [color name](#) that specifies a single color (default **None**, no label background). The colored rectangle will extend *rectangle-padding* pixels (default **9**) beyond the text in each direction (left, right, top, bottom), and an additional outline in **white** for dark rectangle colors or **black** for light rectangle colors will be drawn around the rectangle if *rectangle-outline-width* > 0 is specified (default **0**, no outline). The **visibility** option specifies whether the label should be displayed. An invisible label can be faded in or a visible label faded out using [change](#).

- **2dlabels change** *name* [ text *label-contents* ] [ color *label-color* ] [ size *font-size* ] [ style *font-style* ] [ typeface *font-typeface* ] [ xpos *x-position* ] [ ypos *y-position* ] [ bgColor *rectangle-color* ] [ margin *rectangle-padding* ] [ outline *rectangle-outline-width* ] [ visibility hide | show [ frames *N* ] ]

The 2D label to modify is specified by the *name* it was assigned with [create](#). All 2D labels collectively can be specified with \* (asterisk) instead of *name*. The label can be faded in or out gradually using the **visibility** option with **frames** specified. All other options are as described above for [create](#), except the defaults are to leave the current label unchanged.

- **2dlabels delete** *name*

The 2D label to delete is specified by the *name* it was assigned with [create](#). All 2D labels collectively can be specified with \* (asterisk) instead of *name*.

- **2dlabels acreate** *name* start *x1,y1* end *x2,y2* [ color *arrow-color* ] [ weight *weight* ] [ head blocky | solid | pointy | pointer ] [ visibility hide | show ]

The *name* is required and serves as an identifier for subsequently modifying or deleting the 2D arrow. The keywords **acreate** and **arrowcreate** are interchangeable. The **start** and **end** arguments are required and specify the the start and end (tip) locations of the arrow in the X,Y plane, where the X axis is horizontal and the Y axis is vertical, and both range from 0 to 1 in the visible portion of the window. The X and Y coordinates should be separated by a comma only (no spaces). The *arrow-color* can be any [color name](#) that specifies a single color (default **white**). The *weight* (default **1.0**) is a scale factor for the overall thickness of the arrow. The **head** option specifies arrowhead style. The **visibility** option specifies whether the arrow should be displayed. An invisible arrow can be faded in or a visible arrow faded out using [achange](#).

- **2dlabels achange** *name* [ **start** *x1,y1* ] [ **end** *x2,y2* ] [ **color** *arrow-color* ] [ **weight** *weight* ] [ **head** *blocky* | *solid* | *pointy* | *pointer* ] [ **visibility** *hide* | *show* [ **frames** *N* ]]

The keywords **achange** and **arrowchange** are interchangeable. The 2D arrow to modify is specified by the *name* it was assigned with [acreate](#). All 2D arrows collectively can be specified with \* (asterisk) instead of *name*. The **start**, **end**, **color**, **weight**, and **head** options are as described above for [acreate](#), except the defaults are to leave the current arrow unchanged. The arrow can be faded in or out gradually using the **visibility** option with **frames** specified.

- **2dlabels adelete** *name*

The keywords **adelete** and **arrowdelete** are interchangeable. The 2D arrow to delete is specified by the *name* it was assigned with [acreate](#). All 2D arrows collectively can be specified with \* (asterisk) instead of *name*.

**Usage:****ac** [*shortcut-list*]

Accelerators are simple [keyboard shortcuts](#) for many operations in Chimera.

- Without arguments, the command **ac** simply enables use of [accelerators](#). Accelerators are disabled by default.
- Given a *shortcut-list* of one or more [accelerators](#) separated by spaces, **ac** executes them in the order given without changing the pre-existing mode (without changing whether or not accelerators are enabled). This effectively converts the accelerators to commands, allowing them to be easily interspersed with other commands and included in [command files](#).

See the [Keyboard Shortcuts](#) page for an explanation of [how keystrokes are interpreted](#) when accelerators are enabled at the same time the [Command Line](#) is shown.

Accelerators can be disabled with the accelerator **af** or the accelerator **cl** (which also shows the [Command Line](#)).

See also: [general keyboard shortcuts](#)

**Usage:**

**addaa** *res\_type,res\_number[,conformation]* *adjacent\_res*

**Addaa** adds an amino acid of type *res\_type* with sequence number *res\_number* to *adjacent\_res*, which must be the first or last residue of a chain. The *res\_type* is a three-letter residue name for one of the 20 standard amino acids (case is unimportant), and *res\_number* can include an insertion code. The *adjacent\_res* [specification](#) may contain just a part of that residue (such as one atom or bond) but not more than one residue. If *adjacent\_res* is a single amino acid not part of a longer chain, the *res\_number* is used to determine whether the new residue will be added N- or C-terminally.

The *conformation* can be specified with a keyword:

keyword	description	$\varphi$ (°)	$\psi$ (°)
<b>ext</b>	extended (default)	180	180
<b>alpha</b>	alpha helix	-57	-47
<b>abeta</b>	antiparallel beta sheet	-139	135
<b>pbeta</b>	parallel beta sheet	-119	113

or with a comma-separated pair of values:

*phi,psi*

(the desired  $\varphi$  and  $\psi$  angles in degrees, respectively). If the new residue is added at the C-terminus,  $\varphi$  and  $\psi$  apply to the new residue,  $\psi$  measured as if the residue's OXT were the N of a further C-terminal residue. If the new residue is added at the N-terminus,  $\varphi$  applies to *adjacent\_res* and  $\psi$  to the new residue (for which  $\varphi$  is undefined, as there is no heavy atom substituent to delineate the torsion angle around the residue's N-CA bond).

An added residue may not have an optimal geometry and may clash with other parts of the structure. Residues added with the **alpha** keyword are assigned the [residue attribute](#) value **isHelix=true**; those added with the **abeta** or **pbeta** keyword are assigned the [residue attribute](#) value **isStrand=true** (**isSheet=true**). However, the backbone dihedral angles are merely those appropriate for the designated conformational state and do not ensure that the added residue will fall within that type of secondary structure (which depends on the conformations of multiple residues). After building is complete, [ksdssp](#) can be used to evaluate and reassign secondary structure.

The temperature factor for the new residue is set to the highest currently found in the model.

Example:

```
addaa tyr,30 :29.a
```

addaa

- add tyrosine in an extended conformation as residue 30, after residue 29 in chain A (the new residue will also be in chain A)

See also: [swapaa](#), [delete](#), [ksdssp](#), [Build Structure](#)

**Usage:**

**addcharge std** [spec [atom-spec](#)] [chargeModel *version*]

**Usage:**

**addcharge nonstd** [atom-spec](#) *net-charge* [method am1 | gas ]

**Usage:**

**addcharge [all** [spec [atom-spec](#)] [chargeModel *version*] [method am1 | gas ]]

**Addcharge** is the command-line implementation of [Add Charge](#). It assigns atomic partial charges and Amber/GAFF atom types as the [attributes charge](#) and [gaffType](#), respectively. Atomic partial charges are included in saved [Mol2 files](#). The charges and Amber/GAFF atom types are used by other Chimera tools such as [Minimize Structure](#).

Charge assignment requires explicit hydrogens, which can be added beforehand with [AddH](#) (or the command [addh](#)). Structures often have problems such as truncated sidechains that can lead to non-integral net charges. The [Dock Prep](#) tool includes options for fixing such problems prior to hydrogen addition and charge assignment.

- Atoms in *standard residues* (water, standard amino acids, standard nucleic acids, and a few common variants and capping groups) are assigned charges and types from [Amber \(details\)](#) for standard residues).

The keyword **std** (or **s**) indicates assignments should only be made to standard residues. If **spec** is used, the charge assignment will be limited to the entire model(s) containing the [specified](#) atoms.

The keyword **chargeModel** indicates Amber force field *version* ([details](#)):

- **ff14SB** or **14SB** (default)
  - **ff12SB** or **12SB**
  - **ff02pol.r1** or **02pol.r1** (however, polarizability is not supported by [minimize](#), see [note](#))
  - **ff03ua** or **03ua** - treats amino acid sidechain aliphatic hydrogens as if collapsed onto the corresponding carbons; these “extra” hydrogens will be removed (however, the united-atom approach is not supported by [minimize](#), see [note](#))
  - **ff03.r1** or **03.r1**
  - **ff99SB** or **99SB** (deprecated)
  - **ff99bsc0** or **99bsc0** (deprecated)
- Except for *monatomic ions*, which are simply assigned their net charges, atoms in *nonstandard residues* are assigned charges and [GAFF types](#) determined by [Antechamber](#), which is included with Chimera ([details](#) for nonstandard residues).

The keyword **nonstd** (or **n**) indicates assignments should only be made to nonstandard residues containing the [specified](#) atoms. All of these residues must be of the same type (with the same residue name and containing the same types of atoms connected in the same way). The *net-charge* for that type of residue must also be supplied; it can be zero, or a positive or negative integer. The charge calculation [method](#) can be **am1-bcc** (default, synonym **am1**) or **gasteiger** (synonym **gas**).

- The keyword **all** (or **a**, or the absence of arguments) indicates charges should be assigned to both standard and nonstandard residues. If atoms are [specified](#), the charge assignment will be limited to the entire model(s) containing the specified atoms. If nonstandard residues are found, a dialog for specifying their net charges will appear. The **chargeModel** for standard residues and charge calculation **method** for nonstandard residues can be specified as described above.

Explicit hydrogens are required for adding charges. [AddH](#) will be called as needed to add hydrogens.

If a nucleic acid chain has a 5' terminal phosphate, the user will be asked whether this group should be deleted; otherwise, its atoms will be assigned charges of zero (the Amber charge set lacks parameters for 5' phosphates).

Unrecognized atom names and non-integral net charges are reported in the [Reply Log](#); such problems should be examined and [resolved](#). Note that chain-terminal nucleotide residues will normally have non-integral charges, but the 5' and 3' charges sum to an integer.

See also: [addh](#), [defattr](#), [setattr](#), [minimize](#), [pdb2pqr](#), [Dock Prep](#), [reading charges from files](#)

## Usage:

### addh [options](#)

**Addh** adds hydrogens to molecules, as well as OXT atoms where missing from peptide C-termini. It is the command-line implementation of [AddH](#). Chimera uses atom and residue names, or if these are not "standard," atomic coordinates, to determine connectivity and [atom types](#); the atom types are then used to determine the number of hydrogens to be added and their positions. The positions of pre-existing atoms are not changed, but any lone pairs and unidentifiable-element atoms are deleted. If any atoms cannot be assigned a [type](#), a dialog will appear. It is necessary to click on the line for each unassigned atom and then indicate its proper substituent geometry and number of substituents.

Added hydrogens will be colored the [element color](#) (default white) if the attached atom is colored by element, otherwise to match the attached atom.

By default, if amino acids have standard residue names, each histidine sidechain will be protonated based on its local environment, whereas the sidechains of other residue types will be assigned [protonation states](#) reasonable at physiological pH, regardless of the local environment: negative glutamic acid and aspartic acid, positive lysine and arginine, and neutral cysteine and tyrosine. Alternative protonation states of histidine, glutamic acid, aspartic acid, lysine, and cysteine can be specified with special residue names in the input coordinate file (see the [options](#)). To specify the protonation states of individual residues interactively, use the [AddH](#) graphical interface instead of the **addh** command.

See the [AddH](#) manual page for details including [bond lengths](#) and [effects on radii](#). See also: [delete](#), [hbonds](#), [addcharge](#), [pdb2pqr](#), [vina](#), [Dock Prep](#)

## Options

Option keywords for **addh** can be truncated to unique strings and their case does not matter. A vertical bar "|" designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

### **spec** [atom-spec](#)

Whether to restrict hydrogen addition to a subset of the open molecule models. Hydrogens will be added to the entire model(s) containing the specified atoms. If [atom-spec](#) includes any spaces, it must be enclosed in single or double quote marks.

### **inlsolation** true | false

Whether hydrogen placement should be affected by atoms within the same model only. Otherwise, other models in the vicinity (except submodels of the same model) may affect hydrogen placement, regardless of whether they were designated for hydrogen addition.

### **hbond** true | false

Hydrogens are placed based on the [types](#) of the existing atoms and to avoid clashes; this option controls whether H-bond formation should also be considered. Considering H-bonds increases calculation time, and the method is still under development. Even when

**hbond** is **true**, hydrogen positions are not energy-minimized, and a globally optimal network in terms of the number of H-bonds or total H-bonding energy is not necessarily found. Both intra- and inter-model hydrogen bonds are considered, including interactions with models to which hydrogens are not being added (as in [FindHBond](#) when the [scope](#) of the calculation is set to **both**).

**useHisName true** | false

Whether to use residue names for histidine to determine which sidechain nitrogens should be protonated: the  $\delta$ -nitrogen in residues named HID, the  $\epsilon$ -nitrogen in HIE, and both nitrogens in HIP. Residues named HIS will be treated as unspecified, and may end up with either or both sidechain nitrogens protonated depending on the local environment and whether H-bonds are being considered (whether **hbond** is **true**). If **useHisName** is **false**, all histidine residues will be treated as unspecified regardless of which of the above residue names are used.

**useGluName true** | false

Whether to use residue names for glutamic acid to determine sidechain charge state: GLU negatively charged and GLH neutral (OE2-protonated). If **false**, all glutamic acid sidechains will be made negatively charged regardless of which of the above residue names are used.

**useAspName true** | false

Whether to use residue names for aspartic acid to determine sidechain charge state: ASP negatively charged and ASH neutral (OD2-protonated). If **false**, all aspartic acid sidechains will be made negatively charged regardless of which of the above residue names are used.

**useLysName true** | false

Whether to use residue names for lysine to determine sidechain charge state: LYS positively charged and LYN neutral. If **false**, all lysine sidechains will be made positively charged regardless of which of the above residue names are used.

**useCysName true** | false

Whether to use residue names for cysteine to determine sidechain charge state: CYS neutral and CYM negatively charged. If **false**, all cysteine sidechains will be made neutral regardless of which of the above residue names are used.

**Usage:**

**adjust** ( **angle** | **bangle** | **bondangle** ) *new-angle* [atom-spec](#)

**Usage:**

**adjust** ( **length** | **blength** | **bondlength** ) *new-length* [ **movingSide** **smaller** | **larger** ]  
[atom-spec](#)

The **adjust** command can be used to change a bond angle (valence angle) or bond length. See also: [angle](#), [bond](#), [rotation](#), [Adjust Bonds](#), [Adjust Bond Angles](#)

Keywords can be truncated to unique strings and their case does not matter. An [atom-spec](#) indicates all bonds with both endpoints included.

Only one bond angle (keyword **angle**, synonyms **bangle** and **bondangle**) can be changed at a time, so exactly two bonds with one atom in common must be specified. The *new-angle* is given in degrees. The first bond specified defines which set of atoms should move as the valence angle is changed.

Any number of bond lengths (keyword **length**, synonyms **blength** and **bondlength**) can be changed in a single command to the same *new-length* in Å. The **movingSide** option indicates which set of atoms should move as the length of a bond is changed (default **smaller**).

**Usage:**

**alias** [ [^]*name* [ *wordlist...* ] ]

**Usage:**

**~alias** [ [^]*name* ]

**Alias** assigns to *name* the specified *wordlist*. Subsequent occurrences of *name* (space- and/or semicolon-delimited) in commands will be replaced with *wordlist*. Using *^name* indicates that only occurrences at the beginning of a line should be replaced. This is useful for aliasing a long command to a short name without having to worry about that same name appearing (and being expanded) in the middle of some other command. Conversely, aliases defined without *^* are useful for [atom specification](#) strings that are tedious to type. See also: [namesel](#), [preset](#)

Aliases are saved in [sessions](#). Aliases defined with *^name* and that do not require arguments (see [below](#)) are also saved in the [preferences file](#) and listed in a top-level [Aliases menu](#). Choosing an entry in the **Aliases** menu executes the alias. Aliases can be set up automatically by placing **alias** commands in a startup [command file](#) (see the [Command Line preferences](#)).

The **alias** command without any arguments shows the definitions of all current aliases in the [Reply Log](#). The definition of a specific alias can be shown with **alias name**.

The command **~alias** without arguments deletes all aliases, whereas a specific alias can be deleted with **~alias name** (*^name* should be used instead of *name* if the alias was defined that way).

Within *wordlist*, **\$1**, **\$2**, **\$3** ... may be used to indicate the first, second, third ... arguments of *name*. To have a string with spaces in it handled as a single argument, replace each space with an underscore. Any underscores will be replaced with spaces before the command is executed. For example,

```
alias colorsrf color $1,s #0$2.a
alias helix5 :200-228
colorsrf light_sea_green helix5
```

is equivalent to

```
color light sea green,s #0:200-228.a
```

which colors the molecular surface of the specified residues light sea green.

Note that *wordlist* may contain multiple commands separated by semicolons, for example:

```
alias inspect ~color; ~rl; focus $1; color byatom $1; rl $1
inspect :26
```

Avoid embedding semicolons in an alias by accident. For example, the following will not color anything red because the entire line is interpreted as merely creating an alias:

```
alias others ligand | ions | solvent; color red others
```

In this case, the aliasing and coloring commands should be on separate lines.

**Usage:****align** [atom-spec](#)**align** [atom-spec1](#) [atom-spec2](#)**Usage:****align** *object*

**Align** rotates and translates the view to place two atoms, the centroids of two sets of atoms, or the axis of a geometrical object along the line of sight at the center of the Chimera window.

When two atoms are to be aligned, [atom-spec](#) must specify only those two atoms. The order of specification is important, as the first will be placed in front and the second behind. If the two atoms are [selected](#), they can be specified collectively with the word **selected**, **sel**, or **picked**, and the order in which they were selected will be used. When order information is lacking (for example, the atoms were selected simultaneously), there is no control over which atom will be in front or back, but the view can be flipped afterwards as described below.

If two atom-specification strings separated by a space are supplied ([atom-spec1](#) and [atom-spec2](#)), the centroids of the two sets of atoms will be aligned, with the first in front and the second behind.

Alternatively, instead of atoms, an axis or plane object created with [define](#) or [Axes/Planes/Centroids](#) can be specified by ID (for example, **a4** or **p1**). An axis object will be aligned along the line of sight at the center of the Chimera window, as will the normal axis of a plane object (the plane object will be in the plane of the screen). There is no control over which end of the axis will be in front, but the view can be flipped 180° with the command:

**[turn](#) y 180 center view**

It may still be necessary to adjust the scale and/or clipping plane positions to obtain the desired view, for example with [focus](#) or [window](#).

See also: [center](#), [tile](#), [define](#)

**Usage:**

**angle** *atom1 atom2 atom3* [ *atom4* [ **dihedral true | false** ]]

**Usage:**

**angle** *object1 object2*

The **angle** command measures angles:

- formed by three atoms
- formed by four atoms (dihedral angle, or if **dihedral false** is specified, the angle between the *atom1-atom2* and *atom3-atom4* vectors). The **dihedral** keyword can be truncated, and the option is default **true**. Synonyms for true: True, T, t, 1. Synonyms for false: False, F, f, 0.
- between two geometric objects (two axes, two planes, or one axis and one plane) created with [define](#) or [Axes/Planes/Centroids](#)

Any [atom specification](#) that gives the proper number of atoms can be used. The order of specification is important, but atoms need not be connected. Objects are specified by ID as shown in the [Axes/Planes/Centroids table](#). For angle measurements, axes and planes are treated as infinite. The angle value in degrees is reported in the [status line](#) and [Reply Log](#).

Three or four atoms can be specified collectively using the word **selected**, **sel**, or **picked** if the current [selection](#) contains only those atoms; the order in which the atoms were [selected](#) (most likely [picked](#)) is used. When order information is lacking (for example, the atoms were selected simultaneously), the angle actually measured may not be the one intended.

The **angle** command provides a static measurement; changes are not reported automatically. Angles formed by atoms can be monitored continuously using the [Angles/Torsions panel](#).

See also: [adjust](#), [rotation](#), [ramachandran](#), [distance](#), [define](#)

**Usage:**

**aniso** [ **spec** [atom-spec](#) ] [ **scale factor** ] [ **smoothing level** ] [other-options](#)

**Usage:**

**~aniso** [ **spec** [atom-spec](#) ]

**Aniso** is the command-line implementation of [Thermal Ellipsoids](#), which shows atomic anisotropic B-factors as ellipsoids, their principal axes, and/or their principal ellipses. These special depictions will be created only for atoms with anisotropic B-factor information. Anisotropic B-factors are read from the input coordinate file (from ANISOU records in a PDB file or the analogous in CIF/mmCIF) and are included with only certain high-resolution structures. Ellipsoid axes and radii correspond to the eigenvectors and eigenvalues of the atomic mean-square displacement matrix. The radii are proportional to the root-mean-square displacements (RMSDs), the square roots of the eigenvalues.

Unless restricted to certain atoms with the **spec** option, the command **aniso** removes any pre-existing anisotropic B-factor depictions and generates new ones for just the currently displayed atoms. The command **~aniso** removes the depictions.

Some [predefined combinations](#) of settings are available from the **Presets** menu in the [Thermal Ellipsoids](#) graphical interface.

See also: [define](#), [measure inertia](#), [Render by Attribute](#), [geometric objects](#)

**Tip:** Since the ellipsoids may be obscured when atoms are shown as balls or spheres, using a wire or thin stick [representation](#) is recommended (see [represent](#), [linewidth](#)). Sticks can be made thinner by decreasing the molecule stick scale attribute, for example with the command:

```
setattr m stickScale 0.5
```

**Options**

Option keywords for **aniso** are case-independent and can be truncated to unique strings. A vertical bar "|" designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

**spec** [atom-spec](#)

Create/remove anisotropic B-factor depictions for just the [specified](#) atoms. Depictions will be created only for atoms that are also displayed.

**scale factor**

Adjust the size of the depictions by a constant *factor* (default **1.0**, no scaling). The unscaled ellipsoid radii equal the atomic RMSDs along the ellipsoid axes. The scaling factor for 50% probability is 1.53818. The [Thermal Ellipsoids](#) interface can be used to calculate the scaling factor for any probability  $\geq 0\%$  and  $< 100\%$ .

**smoothing** *level*

The *level* (default **3**) controls how many planar facets are used to draw ellipsoids and ellipses. Higher values give smaller facets and a smoother appearance.

**showEllipsoid** *true|false*

Whether to display ellipsoids.

**color** *colorname*

Color to use for ellipsoids, where the *colorname* can be **none** (default, meaning to match the corresponding atoms) or any [color name](#) that specifies a single color.

**transparency** *percent*

Ellipsoid transparency expressed as a percentage, where 0% is completely opaque and 100% is completely transparent. If not specified, the transparency (if any) of the [ellipsoid color\(s\)](#) will be used.

**axisFactor** *factor*

Length of principal axis depictions, specified as a multiplicative *factor* relative to ellipsoid size. If **axisFactor** is not specified, principal axes will not be depicted.

**axisThickness** *d*

Principal axes are depicted as rods with square cross-sections, where *d* is the side length of the square (default **0.01 Å**).

**axisColor** *colorname*

Color to use for principal axis depictions, where the *colorname* can be **none** (default, meaning to match the corresponding atoms) or any [color name](#) that specifies a single color.

**ellipseFactor** *factor*

Size of principal ellipse depictions, specified as a multiplicative *factor* relative to ellipsoid size. If **ellipseFactor** is not specified, principal ellipses will not be depicted.

**ellipseThickness** *d*

Thickness of principal ellipse depictions (default **0.02 Å**).

**ellipseColor** *colorname*

Color to use for principal ellipse depictions, where the *colorname* can be **none** (default, meaning to match the corresponding atoms) or any [color name](#) that specifies a single color.

## Usage:

**apbs** [ **molecule** [atom-spec](#) ] [options](#)

Like the Chimera [APBS](#) tool, the **apbs** command runs [APBS](#) (Adaptive Poisson-Boltzmann Solver) electrostatics calculations. The process can use either a web service provided by the [National Biomedical Computation Resource \(NBCR\)](#) or a locally installed copy of the program. Users should cite:

[Electrostatics of nanosystems: application to microtubules and the ribosome.](#) Baker NA, Sept D, Joseph S, Holst MJ, McCammon JA. *Proc Natl Acad Sci USA*. 2001 Aug 28;98 (18):10037-41.

A structure should be prepared for APBS calculations by reconstructing missing heavy atoms, adding hydrogens, and assigning atomic charges and radii. These tasks can be done with [pdb2pqr](#) alone or in combination with parts of [Dock Prep](#). Atomic charges can be assigned with [addcharge](#) or [pdb2pqr](#), although the latter may be preferred because it includes force fields developed specifically for Poisson-Boltzmann calculations. By [default](#), any explicit [solvent](#) (typically water) will be omitted from the APBS calculation.

If more than one molecule model is present, the [molecule](#) option should be used to specify which to act upon. \*\* If charges were assigned with [pdb2pqr](#), the model output by that step should be used, not the original model.\*\*

Focusing will be performed automatically; that is, there will be an initial electrostatics calculation on a larger grid with relatively coarse divisions, followed by another calculation on a smaller grid with finer divisions, for which the boundary conditions are determined from the first run [keyword [mg-auto](#), [details](#) at the APBS site]. Default grid sizes (see [dime](#), [cglen](#), and [fglen](#) below) are based on the dimensions of the input structure.

The resulting [electrostatic potential map](#) will be opened as a new model in Chimera and the [Electrostatic Surface Coloring](#) tool for coloring [molecular surfaces](#) by potential will appear. Alternatively, the map can be shown as isopotential surfaces; these are not displayed automatically, but can be shown by starting [Volume Viewer](#) and clicking the eye icon or by using the [volume](#) command.

See also: [coulombic](#), [scolor](#), [DelPhiController](#)

## Options

Option keywords for **apbs** can be truncated to unique strings and their case does not matter. A vertical bar “|” designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

**molecule** [atom-spec](#)

Limit the calculation to the specified model (the molecule model containing the specified atoms). Only one model should be specified. If [atom-spec](#) includes any spaces, it must be

enclosed in single or double quote marks. \*\* If charges were assigned with [pdb2pqr](#), the model output by that step should be used, not the original model.\*\*

**solvent** true | false

Whether to include any explicit [solvent](#) (typically water molecules) present in the [input model](#).

**output** *file*

Pathname (name and location) of the output [electrostatic potential map](#) (\*.dx type). If not specified, a temporary name and location will be used.

**dime** *nx,ny,nz*

Grid points per processor; dimensions in integer grid units along the molecule X, Y, and Z axes; commonly used values are 65, 97, 129, and 161 [[details](#) at the APBS site].

**cglen** *xlen,ylen,zlen*

Dimensions in Å of the coarse grid along the molecule X, Y, and Z axes; the coarse grid should completely enclose the biomolecule [[details](#) at the APBS site].

**cgcent** true | false

Whether to center the coarse grid on the molecule [[details](#) at the APBS site].

**\_cgcentcoord** *xcent,ycent,zcent*

If not centering on the molecule, coordinates of the center of the coarse grid in the molecule coordinate system.

**fglen** *xlen,ylen,zlen*

Dimensions in Å of the fine grid along the molecule X, Y, and Z axes; the fine grid should enclose the region of interest in the molecule [[details](#) at the APBS site].

**fgcent** true | false

Whether to center the fine grid on the molecule [[details](#) at the APBS site].

**\_fgcentcoord** *xcent,ycent,zcent*

If not centering on the molecule, coordinates of the center of the fine grid in the molecule coordinate system.

**bcfl** *condition*

How to initialize potential at the boundary of the coarse grid [[details](#) at the APBS site], where *condition* can be:

- **zero** - boundary potential set to zero; generally not recommended
- **sdh** (default) - single Debye-Hückel; potential set to values prescribed by a Debye-Hückel model for a single sphere with a point charge, dipole, and quadrupole; the boundary should be sufficiently far from the molecule
- **mdh** - multiple Debye-Hückel; potential set to values prescribed by a Debye-Hückel model for multiple noninteracting spheres with point charges; works better than the

single approximation when the boundary is near the molecule, but can be very slow for large molecules

Results from the coarse run are then used to initialize potential at the boundary of the fine grid.

**pdie** *dielectric1*

Solute dielectric constant (default **2.0**) [[details](#) at the APBS site].

**sdie** *dielectric2*

Solvent dielectric constant (default **78.54**) [[details](#) at the APBS site].

**chgm** *method*

How to map atomic partial charges onto grid points [[details](#) at the APBS site], where *method* can be:

- **sp10** - trilinear interpolation (linear splines), charges mapped onto nearest-neighbor grid points; resulting potentials are very sensitive to the grid setup
- **sp12** (default) - cubic B-spline discretization, charges spread out to two layers of grid points (nearest- and next-nearest neighbors); intermediate sensitivity to the grid setup
- **sp14** - quintic B-spline discretization, charges spread out to three layers of grid points; lowest sensitivity to the grid setup

**ion** true | false

Whether to include mobile ions in the calculation [[details](#) at the APBS site].

**\_posion** *charge,conc,radius*

If including mobile ions, the positive ion charge in electron units (the value should be positive), molar concentration, and radius in Å. The total system of mobile ions must be electroneutral; for example, if the positive ion has twice the charge magnitude of the negative ion, its concentration should be half as high.

**\_negion** *charge,conc,radius*

If including mobile ions, the negative ion charge in electron units (the value should be negative), molar concentration, and radius in Å.

**\_equation** **lpbe** | npbe | smpbe

Which form of the Poisson-Boltzmann equation to use:

- **lpbe** (default) - linearized
- **npbe** - nonlinear (full); more computationally expensive to solve than the linearized equation, but more accurate for highly charged systems such as nucleic acids or phospholipid membranes
- **smpbe** - size-modified [[details](#) at the APBS site]

**srfm** *model*

How to map dielectric values and ion accessibility [[details](#) at the APBS site], where *model*

can be:

- **mol** - partition space into regions of solute and solvent dielectric by the molecular surface (solvent-excluded surface calculated with the specified [solvent radius](#) and [density of points](#)), and into regions of ion inaccessibility or accessibility by the VDW surface inflated by the [ion radius](#)
- **smol** (default) - as above, except smooth the dielectric and accessibility values to reduce sensitivity to the grid setup
- **spl2** - use a cubic-spline surface to partition regions of solute dielectric and ion inaccessibility from regions of solvent dielectric and ion accessibility; the spline window width is set to 0.3 Å [[swin details](#) at the APBS site]
- **spl4** - use a seventh-order polynomial to partition regions of solute dielectric and ion inaccessibility from regions of solvent dielectric and ion accessibility

**sdens** *density*

Density of points used to calculate a molecular surface for [mapping values](#) (default 10.0 points/Å<sup>2</sup>) [[details](#) at the APBS site].

**srad** *radius*

Solvent (probe) radius used to calculate a molecular surface for [mapping values](#) (default 1.4 Å) [[details](#) at the APBS site].

**temp** *T*

Temperature to use in the Poisson-Boltzmann equation (default 298.15 K) [[details](#) at the APBS site].

**backend** *opal* | *local*

Whether to use an [Opal](#) web service (default) or a locally installed executable.

**location** *opal-URL* | *local-path*

Depending on the [backend](#) setting, the URL of the web service (default is the URL for the service provided by the [NBCR](#)) or the pathname of the local executable.

**wait** *true* | *false*

Whether to wait for the calculation to finish before starting to execute any subsequent commands.

**[Usage:](#)**

**aromatic** [ **circle** | **disk** | **off** ] [atom-spec](#)

**[Usage:](#)**

**~aromatic** [atom-spec](#)

**Aromatic** controls the display of ring aromaticity:

- **circle** - circle outline
- **disk** - filled circle (default)
- **off** - no aromaticity display (same as using **~aromatic**)

Whether and how aromaticity is shown are molecule model [attributes](#). If the [atom-spec](#) is blank, all molecule models will be affected; otherwise, the entire model(s) containing [atom-spec](#) will be affected.

Aromaticity display can also be controlled using the [molecule model attributes panel](#), the [Selection Inspector](#), or the command [setattr](#). These other methods also allow adjusting the aromaticity display color and circle line style.

Aromaticity display overrides [ring fill](#); aromatic rings can only be shown as filled when their aromaticity display is turned off.

See also: [fillring](#), [nucleotides](#), [represent](#), the [Actions menu](#)

**Usage:**

**background solid** ( *color\_name* | **current** | **none** )

**Usage:**

**background gradient** ( *color1,color2,...* | *palette\_name* | **current** | **none** )  
[ **discrete** | **RGB** | **HLS** ] [ **opacity** *alpha* ]

**Usage:**

**background image** ( *image\_file* | **current** | **none** )  
[ **zoomed** | **stretched** | **centered** | **tiled** ] [ **scale** *factor* ] [ **opacity** *alpha* ]

The **background** command sets the [graphics window](#) background to a single [solid color](#), a [gradient](#) of multiple colors, or an [image](#) from a file. These settings can also be adjusted (and saved for later uses of Chimera) with the [Background preferences](#). Background transparency in Chimera-rendered PNG or TIFF images can be specified in the [Save Image dialog](#) or at startup with the [--bgopacity option](#) (e.g., if images will be saved with the [copy](#) command instead of the dialog). For background transparency in [raytraced](#) images, see the [POV-Ray Options preferences](#). See also: [color](#), [colordef](#), [set](#), [Effects](#), the [Color Editor](#), the [Palette Editor](#), [coloring](#), [movie-related commands](#)

Except for the words **current** and **none**, the keywords can be truncated and their case does not matter. Optional keywords can be given in any order. Examples:

```
background solid light blue
back grad coral,deep sky blue,navy blue rgb
back image ~/Desktop/scenery.png sc 0.5 center
back im current op 0.25
```

The command **background solid** is equivalent to [set bgColor](#) and sets the background to a single solid color. The color can be specified as:

- *color\_name* - any [color name](#) that specifies a single color. Any transparency in the color will be ignored.
- **current** - the current solid color
- **none** (default background) - no color, in this case equivalent to black

By default, the color used for [depth cueing](#) will match the solid background color, but it can be changed independently (for example, with the command [set dcColor](#)).

**Background gradient** sets the background to a vertical gradient of multiple colors, given as one of the following:

- a comma-separated list of at least two color names in bottom-to-top order. Each can be any [color name](#) that specifies a single color, except that R,G,B,[A] tuples cannot be used.

- *palette\_name* - the name of a [palette](#), as defined and listed in the [Palette Editor](#)
- **current** - the current gradient color series and [interpolation method](#). The default gradient is white at the bottom, blue at the top, HLS-interpolated.
- **none** - no gradient; set the background to the current [solid color](#)

The gradient colors can be shown as **discrete** bands or interpolated in the **RGB** (red-green-blue) or **HLS** (hue-lightness-saturation) color space, where the latter is the default. If an interpolation method is specified along with a palette name, the specification will override the method included in the [palette](#). The *alpha* value can range from 0.0, meaning complete transparency, to 1.0, meaning complete opacity (default). If the gradient is transparent, the current [solid color](#) will show through. If a transparent gradient is used together with a [transparent background](#), the background of an image saved from Chimera as PNG or TIFF will not only show the blending of the gradient over the solid color, but also have the same overall opacity as was specified for the gradient.

**Background image** specifies using an image from a file. Accepted formats include PNG, TIFF, and JPEG. The word **current** can be used to indicate a previously specified image. The word **none** indicates resetting the current image to nothing and changing the background to the current [solid color](#). If **image** is the last word of the command (in this case, it can be placed after the options), a [dialog](#) for browsing the filesystem will appear. The image is fit into the graphics window according to one of the following modes:

- **zoomed** (default) - automatically scale the image to fill the window, cropping the top/bottom or sides of the image as needed (no image distortion)
- **stretched** - automatically scale the image to fill the window, stretching or squashing the image as needed
- **centered** - center the image; any surrounding border will be shown in the current [solid color](#)
- **tiled** - tile the image

The **scale** is expressed as a *factor* relative to the actual image size and only affects the **centered** and **tiled** modes. The **opacity** value *alpha* can range from 0.0, meaning complete transparency, to 1.0, meaning complete opacity (default). If the image is transparent, the current [solid color](#) will show through. If a transparent background image is used together with a [transparent background](#), the background of an image saved from Chimera as PNG or TIFF will not only show the blending of the background image over the solid color, but also have the same overall opacity as was specified for the background image.

**Usage:**

**bond** [atom-spec](#)

**Usage:**

**~bond** [atom-spec](#)

**Bond** generates a bond between two specified atoms. The [atom-spec](#) must contain exactly two atoms, and the atoms must belong to the same model. (Connections between atoms in different models can be drawn with [PseudoBond Reader](#).)

**~Bond** deletes all bonds among atoms in [atom-spec](#) (which may contain any number of atoms). Only bonds with both flanking atoms in [atom-spec](#) are deleted.

See also: [adjust](#), [bonddisplay](#), [combine](#), [delete](#), [longbond](#), [Build Structure](#), [PseudoBond Reader](#)

**Usage:**

( **bondcolor** | **bondcolour** ) *color\_name* [atom-spec](#)

**Usage:**

( **~bondcolor** | **~bondcolour** ) [atom-spec](#)

**Bondcolor** sets bond color independent of the color(s) of the flanking atoms. It is equivalent to [color](#) with the **,b** specifier. Individual bond color (see [coloring hierarchy](#)) is set and [halfbond mode](#) is turned **off** for each specified bond.

The *color\_name* can be **none** or any [color name](#) that specifies a single color.

**~Bondcolor** turns [halfbond mode](#) **on** for the specified bonds. This masks individual bond color settings, as each halfbond will be colored to match the flanking atom.

Only bonds with both flanking atoms in [atom-spec](#) are affected.

See also: [bondrepr](#), [bonddisplay](#), [bondzone](#)

**Usage:**

**bonddisplay** *mode* [atom-spec](#)

The **bonddisplay** command controls how bond display depends on the display of flanking atoms. The *mode* can be:

- **always** or **on** (displayed regardless of whether the flanking atoms are displayed)
- **never** or **off** (not displayed regardless of whether the flanking atoms are displayed)
- **smart** or **default** (displayed only when both flanking atoms are displayed)

Only bonds with both flanking atoms in [atom-spec](#) are affected.

See also: [display](#), [longbond](#), [bond](#), [bondrepr](#)

bondrepr

[Chimera Commands Index](#)

### **Usage:**

**bondrepr** *style* [atom-spec](#)

**Bondrepr** changes the [draw modes](#) of bonds according to *style*. *Style* can be:

- **wire** (as in the wire [representation](#))
- **stick** (as in the stick [representation](#))

Only bonds with both flanking atoms in [atom-spec](#) are affected.

See also: [color](#), [represent](#), [bonddisplay](#)

**Usage:**

**bondzone** [ *spacing* ]

**Usage:**

~**bondzone**

**Bondzone** specifies that points along any [selected](#) bonds (and [Volume Tracer](#) links) should be used in addition to any [selected](#) atoms (and [Volume Tracer](#) markers) to define zones with the following tools:

- [Color Zone](#)
- [Surface Zone](#)
- [Volume Viewer](#) (see the [Zone feature](#))

The points along the bonds will be placed  $spacing \times (\text{bond radius})$  apart, with default  $spacing=1$ . [Link radii](#) in [Volume Tracer](#) are equivalent to bond radii, except when a link radius is 0.0, the corresponding bond radius is 1.0.

~**Bondzone** restores the default behavior of using only the [selected](#) atoms to define zones with the tools listed above. **Bondzone** status is stored in the [preferences file](#).

**Usage:****cd** *path\_name*

**Cd** changes the current working directory to *path\_name*. If you are not familiar with the concept of directories and path names, see "UNIX for Beginners" (Kernighan) and the description of the **cd** command in section 1 of the "UNIX User's Manual." Note that subsequent commands with filename arguments (for example, [open](#) and [write](#)) are executed in the new working directory.

Example:

```
cd ../crodna
```

center

[Chimera Commands Index](#)

**Usage:**

( center | centre ) [atom-spec](#)

Center places the collective center of the atoms and/or [surface pieces](#) in [atom-spec](#) at the center of the screen. Surface models and their pieces can be specified by model number or as a [selection](#) ([details...](#)).

See also: [align](#), [cofr](#), [focus](#), [tile](#), [window](#), [Mouse preferences](#)

chain

[Chimera Commands Index](#)

**Usage:**

**chain** [atom-spec](#)

The **chain** command draws [pseudobonds](#) between the [specified](#) atoms (unless they are directly connected by real bonds) and undisplay all other atoms. This is particularly useful for displaying only the backbone of a protein.

Example:

```
chain @ca
```

- chain all alpha carbons

See also: [display](#), [show](#)

**Usage:**

**changechains** *from-chains to-chains* [atom-spec](#)

**Changechains** reassigns chain identifiers, where *from-chains* is an existing chain identifier or comma-separated list of identifiers and *to-chains* is the desired chain identifier or comma-separated list of identifiers in the corresponding order. The scope can be limited to specific models with an [atom-spec](#), where *from-chains* will be reassigned to *to-chains* in only the models with at least one specified atom. For chain identifiers, case is important.

For example:

```
changechains A,B B,A
```

- reassign all chain A as chain B and *vice versa*

It is not possible to change the identifier of only part of a chain. The [Change Chain IDs](#) graphical interface can be used to handle multiple disconnected chains with the same ID.

If the reassignment would give one or more duplicate residue numbers within a chain, an error message will appear and the reassignment will not occur.

See also: [resrenumber](#), [write](#)

**Usage:**

**chirality** *chiral-centers*

**Chirality** reports R/S configurations in the [Reply Log](#) for one or more [specified](#) *chiral-centers* (atoms). The current implementation includes only the simpler Cahn-Ingold-Prelog rules and will fail in complicated cases such as when one center's assignment depends on the chiralities of other centers in the molecule.

See also: [invert](#), [getcrd](#)

**Usage:**

**clip** ( **hither** | **yon** ) *distance* [ *frames* ]

**Usage:**

**~clip** ( **hither** | **yon** )

**Usage:**

**clip** ( **on** | **off** )

**Clip** can be used to move the **hither** (front) or **yon** (back) [global clipping plane](#) a specified *distance* from its current position. The *distance* is in physical units of length, usually Å; a positive number moves the plane towards the user, while a negative number moves it away from the user. The movement is repeated for the specified number of image update *frames* (default **1**). See the [video mini-example](#). See also: [section](#), [thickness](#), [mclip](#), [sop cap](#), the [Side View](#), [movie-related commands](#)

**~Clip** halts an ongoing clip plane motion. Commands continue to be processed while the requested motion is in progress. To pause command processing until the motion is finished, use the [wait](#) command.

A second use of **clip** is to turn [global clipping](#) on and off. Turning it on does not restore previous clipping plane positions.

**Usage:**

**open** [ **noprefs** ] [ **nowildcard** ] [ *model-ID* ] [ *filename1 filename2 ...* ]

**Usage:**

( **close** | **~open** ) ( *model-spec* | **all** | **session** )

The command **open** reads [local files](#) and fetches [data from the Web](#). Chimera [input files](#) may contain structures to be displayed, commands or code to be executed, or other data. Any resulting models in Chimera are assigned the specified *model-ID* number, or if none is specified, the lowest number(s) available. When multiple models are opened with a single command, *model-ID* should not be specified unless there is some reason to assign all of the models the same number.

Various attributes of newly opened molecules can be customized in the [New Molecules preferences](#) (see also [set autoColor](#)). The **noprefs** keyword indicates that a user's [New Molecules preferences](#) should not be applied. This prevents inconsistent behavior of [command files](#) and [demos](#) potentially caused by the different preferences settings of different users.

If a model is closed and another model opened with the same model number, none of the transformations applied to the previous model are applied to the newly opened model. The transformation matrix of one model can be applied to another model using [matrixcopy](#).

The command **close** or **~open** removes structures and other data: model(s) specified by number, or **all** models (everything listed in the [Model Panel](#)), or the **session** (all models, plus data without associated model numbers, such as [2D labels](#) and [sequence alignments](#)). Closing a session shows the [Rapid Access](#) interface.

See also: [read](#), [runscript](#), [split](#), [start](#), [Rapid Access](#), [Fetch by ID](#), [Chimera input file types](#)

## Local Files

A filename is generally a pathname to a local file, either absolute or relative to the current working directory. The working directory can be changed with [cd](#). A pathname may start with "~" to indicate the user's home directory. Multiple files can be specified by including \* (wild card), ? (single-character wild card), and/or square-bracketed character ranges in a filename. Such filename expansion or *globbing* is on by default, but can be turned off with the **nowildcard** keyword. If no filename is supplied, a [dialog](#) for browsing the filesystem will appear.

Any of the [registered file types](#) can be opened.

A file type can be specified by a [suffix](#) that is part of the filename or by [prefix:filename](#), where [prefix](#) is not part of the filename. If a prefix and a suffix are both given, the prefix overrides the suffix. Filenames, prefixes, and suffixes are case-sensitive. Unrecognized prefixes are assumed to be part of the filename. For many of the [registered file types](#), files that are gzipped (as indicated by **.gz** following the regular filename) are recognized and opened. Similarly, compressed files (as indicated by **.Z**) can be recognized

and opened for many input types if *gzip* is on the user's execution path (can be run by entering *gzip* at the system command line). [PDB format](#) is the default input type.

## Files from the Web

Other than a local file, a filename can specify data to be retrieved over the Web. Fetched files can be cached locally and reused as needed depending on the [Fetch preferences](#) and [PDB preferences](#).

- many [registered types](#) can be opened by specifying an URL of the form **http://[...]** as the filename. Prefix and suffix rules are as described above for [local files](#), but filename globbing and reading compressed or gzipped files are not supported.
- the prefix **pdbID**: indicates a PDB ID (4-character Protein Data Bank identifier) to be used to fetch a PDB-format file from the Protein Data Bank:
  - Chimera will first attempt to find the file within a local installation of the Protein Data Bank. The default places to look for a local installation are `/usr/mol/pdb/` and then `/mol/pdb/` — these can be changed by editing the Python file `share/chimera/pdbDir` within a Chimera installation.
  - Next, Chimera will look for the file in any personal PDB directories designated in the [PDB preferences](#).
  - Next, the PDB subdirectory of the [fetch download directory](#) will be checked (unless the use of previously fetched files has been disallowed in the [Fetch preferences](#)).
  - Finally, if not found locally, the file will be fetched from the [Protein Data Bank web site](#) (unless fetching has been disallowed in the [PDB preferences](#)).
- any input specified as PDB format (which is the default) will be fetched in the same way if the filename resembles a PDB ID and is not found in the current working directory or any [personal PDB directories](#)
- the prefix **cifID**: indicates a PDB ID to be used to fetch an mmCIF-format file from the [Protein Data Bank](#)
- the prefix **ndb**: indicates an [NDB](#) ID to be translated into a PDB ID and used to fetch the corresponding structure from the [Protein Data Bank](#)
- the prefix **biounitID**: indicates a PDB ID to be used to fetch [biological assemblies](#) from the [Protein Data Bank](#); there may be multiple assemblies for a given entry, and a specific assembly can be designated by appending “.N” to the PDB ID. For example, **1dok** specifies all biological assemblies of entry [1dok](#), **1dok.2** specifies biological assembly 2, and **1dok.1.2** specifies biological assemblies 1 and 2. Each assembly will be opened as a separate main model number, possibly containing [submodels](#).
- the prefix **pqsID**: indicates a PDB ID to be used to fetch the predicted biological unit from the [Protein Quaternary Structure server](#) (predictions are not available for all PDB entries; some entries have multiple predictions, and for those, multiple files will be retrieved)
- the prefix **viperID**: indicates a PDB ID to be used to fetch an entry from the [Virus Particle Explorer database](#) of icosahedral virus capsid structures (the capsid will be constructed automatically with [Multiscale Models](#))
- the prefix **cath**: indicates a [CATH domain ID](#) to be used to fetch a domain structure from [CATH](#)
- the prefix **scop**: indicates a [SCOP domain ID](#) to be used to fetch a domain structure from the [ASTRAL](#) database
- the prefix **castp**: or **CASTp**: indicates a PDB ID (with chain ID optionally appended, for example **2gsh.A**) to be used to fetch a structure and its precomputed pocket measurements from the

[Computed Atlas of Surface Topography of proteins](#) (does not include results for all PDB entries; measurements will be displayed in a [pocket list](#))

- the prefix **modbase:** indicates a SwissProt, TrEMBL, GenPept or PIR accession code for which to fetch PDB-format comparative models from [ModBase](#) (associated information will be shown in a [list](#)). If multiple modeled structures are available for the sequence, they will be opened as [submodels](#) of a single model number.
- the prefix **smiles:** or **SMILES:** indicates a [SMILES](#) string to be converted to a 3D structure using the [smi23d web service](#) provided by the [Chemical Informatics and Cyberinfrastructure Collaboratory](#) at Indiana University, or if that does not produce a structure, the [SMILES translator](#) provided by the [National Cancer Institute CADD group](#). The [smi23d service](#) deploys the same procedure used to populate the **Pub3D** database (below).
- the prefix **pubchem:** or **PubChem:** indicates a [PubChem](#) compound identifier (CID) for which to fetch a modeled 3D structure from the **Pub3D** database using a [web service](#) provided by the [Chemical Informatics and Cyberinfrastructure Collaboratory](#) at Indiana University. **Pub3D** is described in [Willighagen et al., BMC Bioinformatics 8:487 \(2007\)](#). About 99% of the compounds in [PubChem](#) are available; the structure generation pipeline generally handles organic compounds, but not inorganic, metallo-, or highly unstable species such as radicals.
- the prefix **edsID:** indicates a PDB ID to be used to fetch an electron density map from the [Electron Density Server](#) (not all PDB entries have maps available; [Volume Viewer](#) will be used to show the map)
- the prefix **edsdiffID:** indicates a PDB ID to be used to fetch an electron density difference map (fo-  
fc) from the [Electron Density Server](#) (not all PDB entries have maps available; [Volume Viewer](#) will be used to show the map)
- the prefix **emdbID:** indicates a numerical identifier to be used to fetch an electron density map from the [Electron Microscopy Data Bank](#)
- the prefix **emdbfitID:** indicates a numerical identifier to be used to fetch an electron density map from the [Electron Microscopy Data Bank](#), along with any corresponding PDB entries (which may or may not be in the fit positions relative to the map)
- the prefix **uniprot:** indicates a [UniProt](#) accession code or identifier to be used to fetch a protein sequence and its annotations and show them in [Multalign Viewer](#) (much as described for [PDB/UniProt Info](#), except independent of structure). This type of fetch is not saved locally even if a [download directory](#) has been specified. UniProt's [ID mapping service](#) can be used to obtain UniProt identifiers from other sequence database identifiers.

**Usage:**

**cofr** [ [view](#) | [models](#) | [front](#) | [independent](#) | [fixed](#) ]

**Usage:**

**cofr** [ [atom-spec](#) | *x,y,z* [ [coordinateSystem](#) *N* ]]

**Usage:**

**~cofr**

If no arguments are supplied, **cofr** reports the coordinates of the current center of rotation in the laboratory frame of reference. These values are also shown in the [Rotation tool](#).

The keyword options specify [center of rotation method](#):

- **view** - continually adjusting to the current center of view
- **models** - the **center of models** method: center of the bounding sphere of the displayed parts of [active](#) models
- **front** - the **front center** method (also specified with **~cofr**): when the view is zoomed out, behaves like **center of models**; when the view is zoomed in on parts of objects, behaves like **center of view**, except the center of rotation depth (Z-coordinate) is set to that of the frontmost displayed unclipped atom whose VDW sphere intercepts a line perpendicular to the screen in the window center. The center is not updated when only rotations are performed.
- **independent** - each model rotating about its own center rather than a single collective center; can also be specified with the command [set independent](#)
- **fixed** - at a fixed point relative to the model(s) (changes only the method, retains the current center)

The remaining options specify a particular **fixed** center of rotation. An [atom-spec](#) can include any combination of [atoms](#) and [surface pieces](#); the center of rotation will be set to the center of the bounding sphere of the specified items. Alternatively, the center can be entered as coordinates *x,y,z* in a specified coordinate system. The **coordinateSystem** can be specified by reference model number *N*, optionally preceded by #; otherwise, the laboratory frame of reference will be used.

The center of rotation can also be controlled with the [Rotation tool](#).

See also: [center](#), [focus](#), [set independent](#), [Mouse preferences](#), [3D manipulation](#), [movie-related commands](#), [Set Pivot](#) and [Focus](#) in the [Actions menu](#)

**Usage:**

( **color** | **colour** ) *color\_name*[,a][,f][,r][,s][,v][,l][,la][,lr][,lb][,b] [atom-spec](#)

**Usage:**

( **~color** | **~colour** )[,a][,f][,r][,s][,v][,l][,la][,lr][,lb][,b] [atom-spec](#)

The **color** command sets colors of atoms, bonds, [ring fill](#), [ribbons](#), labels (3D), [molecular surfaces](#), [VDW surfaces](#), and nonmolecular [surface models](#). The command **~color** sets the color to none (no color assigned); assignments at other levels in the [coloring hierarchy](#) may become visible. See also: [colordef](#), [ribinsidecolor](#), [scolor](#), [modelcolor](#), [rainbow](#), [rangecolor](#), [transparency](#), [background](#), [set](#), [setattr](#), [msc](#), [coloring](#)

The *color\_name* can be:

- any of [169 built-in color names](#)
- a name defined previously with [colordef](#)
- a [Tk color code](#) such as #7fffd4 or #b0b030306060 ([more examples](#))
- an R,G,B or R,G,B,A tuple, where R is the red component, G is the green component, B is the blue component, and A is the opacity (1 - transparency), each in the range 0-1. The values should be separated by commas only, not spaces. Example: 1,0.25,0.8,0.3
- **colorpanel**, **fromeditor**, or **editor** - the color currently defined in the [Color Editor](#)

The **color** command also accepts:

- **none** - no color assigned (like using **~color**); assignments at other levels in the [coloring hierarchy](#) may become visible
- **byatom** or **byelement** - a [color-by-element scheme](#); does not apply to ribbons or residue labels
- **byhet** or **byhetero** - same as **byatom**, except leaving carbon atoms unchanged

The assignment can be restricted with one or more specifiers:

- **a** - atoms (and [normally](#), bond halves are colored to match the flanking atoms; see **b** for bonds-only coloring)
- **f** - [ring fill](#)
- **r** - [ribbons](#); equivalent to using [ribcolor](#)
- **s** - [molecular surfaces](#) and nonmolecular [surface models](#)
- **v** - [VDW surfaces](#)
- **l** - atom and residue labels
- **la** (or **al**) - atom labels
- **lr** (or **rl**) - residue labels
- **lb** (or **bl**) - bond labels
- **b** - bonds only (not the flanking atoms), turning off [halfbond mode](#); **~color** with the **,b** specifier turns [halfbond mode](#) back on; equivalent to using [bondcolor](#)

In the absence of a specifier, all of the above are affected except bonds-only, and [halfbond mode](#) is turned on so that bond halves are colored to match the flanking atoms. However, VRML models, [2D Labels](#), and [pseudobonds](#) and their labels are not affected. A pseudobond or its label can be colored by [selecting](#) the pseudobond and using the [Selection Inspector](#) to change its color or label color.

The [atom-spec](#) indicates which atoms (and/or associated bonds, ribbon segments, surface patches, *etc.*) should be affected, and a [similar specification](#) can be used for nonmolecular [surface models](#). If no specification is given, all applicable items are affected.

In Chimera, visible color is determined by a [hierarchy](#). **Color** sets atom, atom label, and surface colors at the atom level, and ribbon and residue label colors at the residue level. [Modelcolor](#) sets colors at the model level. [Surfcolor](#) sets which level is used as the source for visible surface colors; this must be the atom level for **color** to control the visible surface color.

Transparency can be set independent of color with the [transparency](#) command.

Examples:

**color green #2:his**

- color all histidine residues in model 2 green, as well as any associated labels, ribbon segments, surface patches, *etc.*

**color #ff00ff,r,s #3**

- color ribbons and surface of model 3 #ff00ff (magenta)

**~color,a,l #0:4**

- remove the color assignments of atoms and their labels in residue 4 of model 0 (the color will default to the model-level color)

**color cyan,r helix; color magenta,r strand; color gray,r coil**

- color protein ribbons by secondary structure

**Usage:**

( **colordef** | **colourdef** ) *color\_name* *red green blue* [ *alpha* ]

**Usage:**

( **colordef** | **colourdef** ) *color\_name* *existing\_color\_name*

**Usage:**

( **colordef** | **colourdef** ) *existing\_text\_color\_name*

**Usage:**

( **colordef** | **colourdef** ) **list**

**Colordef** can be used to define a new color or to change the definition of an existing color. See also: [color](#), [ribinsidecolor](#), [modelcolor](#), [scolor](#), [transparency](#), [background](#), the [Color Editor](#), [coloring](#)

The *color\_name* can be:

- an arbitrary name not yet associated with any color (can contain letters, digits, underscores)
- a name defined previously with **colordef**
- a [built-in color name](#), but only those without spaces. However, redefining built-in colors is generally not recommended because of the potential for confusion. For example, "red" could be redefined to mean some other color. A more reasonable use is to change the [color-by-element scheme](#), e.g., to assign green instead of gray to carbons (see the last [example](#))

color name	red	green	blue
red	1.0	0.0	0.0
green	0.0	1.0	0.0
blue	0.0	0.0	1.0
cyan	0.0	1.0	1.0
magenta	1.0	0.0	1.0
yellow	1.0	1.0	0.0
white	1.0	1.0	1.0
black	0.0	0.0	0.0

The *red*, *green*, *blue*, and *alpha* arguments must be floating point numbers between zero and one, inclusive. The *alpha* value denotes opacity (1 - transparency), where 0.0 means completely transparent and 1.0 means completely opaque (default).

The *existing\_color\_name* can be any [color name](#) that specifies a single color. The name **colorpanel**, **fromeditor**, or **editor** can be used to save the current [Color Editor](#) color to a name before further changes are made.

Examples:

```
colordef lilac 0.5 0.5 1.0
colordef transparent_red 1 0 0 .4
colordef same_transparent_red 1,0,0,.4
colordef puce #551129
colordef raisin puce
colordef myblue colorpanel
colordef C green
```

If only an *existing\_text\_color\_name* is supplied (any of the [169 built-in color names](#) or a name defined previously with [colordef](#)), the color's red, green, blue, and alpha component values will be reported in the [status line](#) and [Reply Log](#).

The **list** keyword indicates listing all of the user-defined color names in the [Reply Log](#).

A convenient way to define colors automatically is to place **colordef** commands in a startup [command file](#) (see the [Command Line preferences](#)). Color definitions are also saved in [sessions](#).

**Usage:**

**colorkey** *x1,y1 x2,y2* [options](#) *value1 color1 value2 color2 [... valueN colorN]*

**Usage:**

**~colorkey**

**Colorkey** is the command-line implementation of [Color Key](#), which shows the relationship between a coloring scheme and the values of a qualitative or quantitative property such as an [attribute](#). See also: [2dlabels](#), [rangecolor](#), [coulombic](#), [scolor](#)

Whereas making a color key with [Color Key](#) or commands [rangecolor](#) and [coulombic](#) requires manual placement by click and drag, the **colorkey** command specifies position directly: *x1,y1* and *x2,y2* are the coordinates of opposite corners of the rectangular color bar in the plane of the screen. Each coordinate can range from 0 to 1, with 0 corresponding to bottom/left and 1 to top/right. The value-color pairs must be given at the end of the command, following any [options](#). The values are also the labels and need not be numeric. Each color can be specified with any [color name](#) that denotes a single color.

The command **~colorkey** removes the color key.

**Options**

Option keywords can be truncated to unique strings and their case does not matter. A vertical bar “|” designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

**colorTreatment** **blended** | distinct

Whether the colors along the bar should be blended (default) or discontinuous.

**labelSide** left/top | **right/bottom**

Whether the labels should be positioned to the left of a vertical key and above a horizontal key, or to the right of a vertical key and below a horizontal key (default).

**labelColor** *color*

Single color for all label text (default **white**). The *color* can be any [color name](#) that specifies a single color, or **None** to make the labels match the individual colors in the key.

**justification** left | **decimal-point** | right

How to align the labels in a vertical key; labels in horizontal keys are always center-justified.

**numLabelSpacing** **proportional-to-value** | equal

How how to space numeric labels along the key; regardless of this setting, equal spacing will be used if any label (any value) is non-numeric or the values are not given in numerical order.

**labelOffset** *offset*

Additional space between the labels and the color bar (default **0** pixels).

**fontSize** *size*

Label font size (default **24**).

**fontStyle** **normal** | italic | bold | bold-italic

Label font style.

**fontTypeface** **sans-serif** | serif | fixed

Label font typeface.

**borderColor** *color*

Color of the outline around the color bar (default **None**, no outline shown). The *color* can be any [color name](#) that specifies a single color.

**borderWidth** *width*

Width of the outline around the color bar (default **3** pixels).

**tickMarks** **true** | **false**

Whether to display short lines extending from the color bar toward the labels; if a border is shown, the border color is also used for the tick marks.

**tickLength** *length*

Length of tick marks (default **10** pixels).

**tickThickness** *thickness*

Width of tick marks (default **4** pixels); to make flush with the border (if any), use a value at least twice the border width.

**Usage:****combine** *atom-spec options*

The **combine** command combines multiple molecule models into a single model without repositioning the models or adding bonds. It can also be used to make a copy of a single molecule model. By contrast, the [Join Models](#) section of [Build Structure](#) merges two models while repositioning one model to form a bond with the other. **Combine** is the command-line implementation of the [copy/combine](#) function in the [Model Panel](#).

Although a narrower *atom-spec* may be supplied, the entire model(s) containing the specified atoms will be included in a new model. The colors, labels, and display settings of the input model(s) will be copied to the new model. To copy settings from one existing molecule model to another without creating a new model, use the command [mcopy](#). See also: [split](#)

**Options**

Option keywords for **combine** can be truncated to unique strings and their case does not matter. A vertical bar "|" designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

**refSpec** *ID*

Which model's coordinate system to use for the new model (default is the lowest-numbered model involved in the combination). The reference model's untransformed coordinates (if part of the combination) and its current transformation will be carried over into the new model. The coordinates of any other model in the combination will be transformed relative to the reference model. *ID* is the number of the reference model preceded by #.

**newchainids** true | false

Whether to allow changes in chain identifiers to make each residue in the new model uniquely specifiable; otherwise, residues may be renumbered to make them uniquely specifiable. Even when **newchainids** is true, residues in [chains water and het](#) may be renumbered.

**name** *new-model-name*

Name for the new model (default **combination**). Model names are shown in the [Model Panel](#) and other model lists.

**modelId** *N*

Open the new structure as model number *N* (an integer, optionally preceded by #). The default is the lowest unused number.

**close** true | **false**

Whether to close the original model(s).

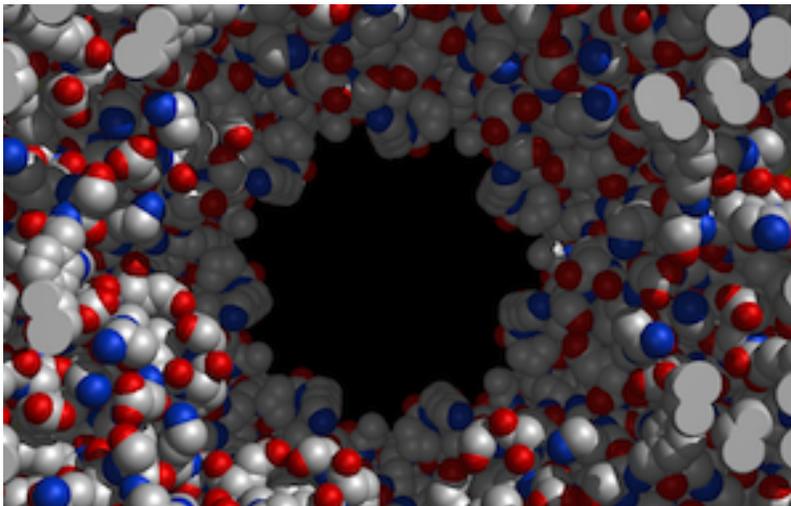
**log** true | false

Whether to report changes in chain identifiers or residue numbering in the [Reply Log](#).

**Usage:****conic options**

The **conic** command produces a space-filling rendering of the displayed molecule(s). The current orientation and coloring are retained. Atoms are depicted as spheres with realistic highlighting and shadowing effects (the [VDW radii](#) are used). This image is not interactive and can take anywhere from a few seconds to a few minutes to produce. Clicking the left button returns to interactive graphics.

The **conic** command is actually an alias that uses [pdbrun](#) to send data to the outside program [conic](#). There are many options, which are detailed in the [conic manual page](#).



\*On **Mac** systems, the **conic -o** flag or configuration file [output](#) option must be used to generate any output at all. If display is desired in addition to an output file, the [-s](#) flag should also be used.

See also: [pdbrun](#), [neon](#), [represent](#), [raytracing with POV-Ray](#)

**Usage:**

```
coordset model-spec frame [ holdSteady atom-spec ]
```

**Usage:**

```
coordset model-spec [start],[end],[step] [ holdSteady atom-spec ] [ loop N ]
```

**Coordset** plays through frames of a trajectory previously loaded into Chimera. See the [video mini-example](#). See also: [morph](#), [play](#), [MD Movie](#), [movie-related commands](#)

**\*\* Protein secondary structure assignments are not recomputed automatically over the course of a trajectory. If a protein is displayed as [ribbon](#), users may want to reassign secondary structure at each frame, as described [below](#). \*\***

*Model-spec* is the model number of the trajectory, preceded by #. A comma-separated list or range of model numbers can be used to specify multiple trajectories at once. Atoms to hold as steady as possible upon frame updates can be [specified](#) using the **holdSteady** keyword.

Supplying only one *frame* number without commas indicates simply going to that frame. A positive number is taken literally, whereas a negative number *N* is interpreted as the *N*th-to-last frame.

The **coordset** command and [MD Movie](#) graphical interface are integrated, in that frame updates with the command will be reflected in the graphical interface, and hold-steady and per-frame operations specified in the graphical interface will be applied during playback with the command. Examples:

```
coordset #0 10
  (go to frame 10 of trajectory model #0)
coordset #0 -10
  (go to the 10th-to-last frame)
```

Supplying comma-separated frame numbers or even commas without preceding numbers indicates playing from *start* (default is the current frame) to *end* (default is the last frame) in increments of *step*. The default *step* is 1 if *start* < *end*, -1 if *start* > *end*, determined after any negative frame numbers are converted to the corresponding actual frame numbers. The **loop** option indicates repeating playback *N* times, wrapping directly from *end* to *start*. Examples:

```
coordset #0 1,21 holdSteady @ca
  (play from frame 1 to 21 of trajectory model #0, holding CA atoms steady)
coordset #0 1,
  (play from frame 1 to the end)
coordset #0 ,15
  (play from the current frame to 15)
coordset #0 ,,5
  (play from the current frame to the end, every 5th frame)
coordset #0 21,1
  (play from frame 21 to 1, backwards)
coordset #0 -1,1,-10
```

coordset

(play from the last frame to the first, every 10th frame, backwards)

## Recomputing Secondary Structure

If a protein or peptide is displayed as a [ribbon](#) with secondary-structure-specific scaling (width and height) and significant conformational changes are occurring, users may want to reassign secondary structure at each frame. This can be done with the command [perframe](#) in combination with [ksdssp](#), for example:

```
perframe ksdssp; coordset #2 1,21; wait 21; ~perframe
```

## Usage:

**copy** [ **file** *filename* ] [ *format* ] [ **printer** *printername* ] [ **width** *x* ] [ **height** *y* ] [ **units** *type* ] [ **dpi** *pixels-per-inch* ] [ **supersample** *n* ] [ **raytrace** [ *rtwait* ] [ *rtclean* ] ]

**Copy** saves the contents of the graphics window as an image, either rendered directly by Chimera or raytraced. Background transparency in Chimera-rendered PNG or TIFF images can be specified at startup with the [--bgopacity option](#) if images will be saved with the **copy** command (the [Save Image dialog](#) also contains the option). For background transparency in [raytraced](#) images, see the [POV-Ray Options preferences](#). See also: [window size](#), [preset](#), [export](#), [Save Image](#), [tips on preparing images](#)

To copy a molecule model to create a new model, use the command [combine](#) instead. To copy settings from one molecule model to another, use the command [mcopy](#).

If the **file** keyword is given, the image is saved in *filename*. If "-" is supplied instead of a filename, a [dialog](#) for [saving](#) a file will appear. The **printer** option does not work on **Windows**. On other platforms, the **printer** option sends the image in PostScript form to the printer named *printername*. If "-" is supplied instead of a printer name, the image is sent to the default system printer. If neither **file** nor **printer** is specified, a [dialog](#) for [saving](#) a file will appear.

If **width** and **height** are not specified, the image will have the same pixel dimensions as the graphics window. If only one dimension is specified, the other will be set according to the graphics window aspect ratio; if both are specified but the image aspect ratio is not the same as the graphics window aspect ratio, the image contents will differ from what is shown in the window.

Width and height values are assumed to be in pixels unless **units** are specified as one of the following types:

- **inches**
- **millimeters**
- **centimeters**
- **points** (72 points = 1 inch)

If units are specified as one of the above, a **dpi** value (*pixels-per-inch*) can also be specified, or the current **Print resolution** setting in the [Save Image dialog](#) will be used. If width and height are in pixels, **dpi** is ignored.

If the image is rendered directly by Chimera (the **raytrace** keyword is not supplied):

- The image will be rendered offscreen as permitted by the system. Where offscreen rendering is not supported (for example, X11 on Mac OS), the graphics window will be used to draw the image, and should not be obscured while that is occurring ([details...](#)).
- The level of [supersampling](#) shown in the [Save Image dialog](#) will be used unless a

different level is specified with the **supersample** keyword. Possible values range from 1 (no supersampling) to 4 (4x4 supersampling). In addition, a value of 0 can be used to indicate onscreen rather than offscreen rendering, without supersampling.

- The *format* can be any of the following:
  - **ps** (default) - PostScript, suffix \*.ps
  - **eps** - encapsulated PostScript, \*.eps
  - **png** - \*.png
  - **ppm** - \*.ppm
  - **tiff** - LZW-compressed TIFF, \*.tif, \*.tiff
  - **tiff-fast** - uncompressed TIFF
  - **jpeg** - \*.jpg, \*.jpeg
  - **stereo png** - \*.pns
  - **stereo jpeg** - \*.jps

If *format* is not specified separately, it is guessed from the suffix of *filename*. The **stereo png** and **stereo jpeg** options will create a *cross-eye stereo* pair in a single image twice as wide as the specified size. More commonly, however, stereo images would be generated simply by switching the Chimera window to stereo (for example, with the [stereo](#) command) and using a standard, non-stereo-specific format.

If the image is [raytraced with POV-Ray](#) (the **raytrace** keyword is supplied):

- The *format* can only be **png**, specified directly or by using a *filename* with the \*.png suffix.
- The **rtwait** keyword indicates further commands should not be executed until POV-Ray has finished generating the image (useful within [scripts](#)).
- In addition to an image, raytracing generates files for POV-Ray containing the scene (\*.pov) and the raytracing options (\*.ini); the **rtclean** keyword indicates these two files should be removed after the image has been created.
- Several other parameters can be adjusted in the [POV-Ray Options preferences](#).
- Raytracing (but not the preceding export to POV-Ray files) is a background task that can be monitored or canceled using the [Task Panel](#).

**Usage:**

**coulombic** *options* *value1 color1 value2 color2 [... valueN colorN]* *surface-spec*

**Coulombic** is the command-line implementation of [Coulombic Surface Coloring](#), which colors surfaces by electrostatic potential calculated with Coulomb's law:

$$\varphi = \sum [q_i / (\epsilon d_i)]$$

$\varphi$  is the potential (which varies in space),  $q$  are the atomic partial charges,  $d$  are the distances from the atoms, and  $\epsilon$  is the dielectric, representing screening by the medium or solvent. A distance-dependent dielectric is sometimes used to approximate screening by implicit solvent. By default, **coulombic** uses a distance-dependent dielectric ( $\epsilon = 4d$ ). The calculation can handle structures with or without explicit [hydrogens](#), and a [grid](#) of the values can be generated (see [why this might be useful](#)).

Whereas [Coulombic Surface Coloring](#) can only color [molecular surfaces](#) based on the charges of the residues they enclose, the **coulombic** command allows specifying [which atoms to use](#) independently of which surface to color. This allows coloring the surface of one molecule by the potential from another, for example, or coloring nonmolecular surfaces such as density isosurfaces.

The value/color pairs define how surface electrostatic potential is mapped to colors. At least two value/color pairs must be supplied. Electrostatic potential values (*value1*, *value2*, etc.) are in units of kcal/(mol·e) at 298 K. Each color name (*color1*, *color2*, etc.) can be any [color name](#) that specifies a single color. The command does not have default color settings, but the defaults in the [Coulombic Surface Coloring](#) tool correspond to **-10 red 0 white 10 blue**.

The surface of interest should already be displayed. The *surface-spec* can be:

- a model number preceded by #
- the word **selected**, **sel**, or **picked**, meaning the current [selection](#)
- an [atom-spec](#), meaning the corresponding [molecular surface](#)

Entire surface models will be affected, even if only partly specified. A blank *surface-spec* will be taken to mean all applicable surface models: all [molecular surfaces](#), or if [atoms](#) are given, all [surface models](#). The [atoms](#) option is mandatory for coloring nonmolecular surfaces.

The calculation requires charge assignments, which in turn require hydrogens. An existing structure lacking hydrogens is not changed, but a copy is created in memory, [protonated](#), and [assigned charges](#) ([details](#)), which are then transferred to the existing structure. Selenomethionine (MSE) residues are treated as methionines (MET) for purposes of charge assignment. Where hydrogens are missing from the existing structure, their charges are collapsed onto the adjacent heavy atom: such hydrogens are *implicit*.

Alternatively, a structure may already have *explicit* hydrogens, or they can be [added](#) beforehand in Chimera. A structure may also have pre-existing charge assignments, such as from [addcharge](#) or a previous use of **coulombic**. If all of the atoms corresponding to the chosen surface already have charges,

those values are used rather than assigned anew the first time Coulombic coloring is applied to that surface. In subsequent applications, the existing charges will be used unless the [hisScheme](#) is changed, which forces the charges to be assigned anew. Another way to force reassignment is to remove the charges with the command [~setattr a charge](#).

See [Coulombic Surface Coloring](#) for more details, including discussions of [implicit vs. explicit hydrogens](#) and [limitations](#) of the method. See also: [apbs](#), [scolor](#), [rangecolor](#), [addh](#), [addcharge](#), [pdb2pqr](#)

## Options

Option keywords for **coulombic** can be truncated to unique strings and keyword case does not matter. A vertical bar “|” designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

### atoms [atom-spec](#)

Use only the charges of the atoms in [atom-spec](#), plus their implicit hydrogens, if any, to compute the potential. Otherwise, the charges of all residues enclosed in the [molecular surface](#) to be colored (all atoms in the corresponding [surface category](#), plus their implicit hydrogens, if any) will be used. The **atoms** option allows coloring the surface of one molecule by the potential from another, for example, or coloring nonmolecular [surface models](#) (types other than MSMSModel). **\*\*When using this option, one should consider carefully whether to change the offset with [surfDist](#); a zero or negative value may be appropriate.\*\***

### distDep true | false

Whether the dielectric should be distance-dependent; whether  $\epsilon$  should vary in proportion to the distance from each charge.

### dielectric *C*

Set the dielectric constant to *C* (default **4.0**), where  $\epsilon = Cd$  if [distDep](#) is **true**,  $\epsilon = C$  if [distDep](#) is **false**.

### surfDist *offset*

How far out from each surface vertex, along its normal, to evaluate the electrostatic potential (default **1.4 Å**). The rationale for looking outward is that the values at the centers of any interacting atoms are more relevant than those at their surfaces. A [molecular surface](#) is *solvent-excluded*; it shows where the surface of a spherical probe (typically of radius 1.4 Å) can lie. Thus, 1.4 Å out from the molecular surface is about as close as the probe center can get, the *solvent-accessible* surface. However, for coloring the surface of one molecule by the potential from another molecule using the [atoms](#) option, a zero or negative offset may be appropriate.

### hisScheme HID | HIE | HIP | none

How to determine the charge states of histidines in structures without hydrogens:

- HID - neutral sidechain, implicitly protonated at  $\delta$ -nitrogen
- HIE - neutral sidechain, implicitly protonated at  $\epsilon$ -nitrogen
- HIP - positive sidechain, implicitly protonated at both sidechain nitrogens

- **none** (default) - protonation state chosen based on the local H-bonding environment

These settings apply only to residues named HIS. Histidines that already have the special names (HID, HIE, HIP) will be protonated accordingly. Changing the **hisScheme** from what was used previously (via command or graphical interface) indicates that the charges on the should be erased and assigned anew.

### key

Bring up the [Color Key](#) tool, filled in with the appropriate colors and values, and set to **Use mouse for key placement** for creating/positioning the color key in the graphics window. The mouse setting can be toggled to allow moving models with the mouse. See also the [colorkey](#) command.

Using any of the following three options generates a grid of the Coulombic potential values and starts the [Electrostatic Surface Coloring](#) and [Volume Viewer](#) tools (see [why this might be useful](#)):

### gspacing *r*

Grid point spacing in each dimension (default  $r = 1.0 \text{ \AA}$ ).

### gpadding *d*

Distance to extend the grid in each dimension beyond the atoms enclosed in the surface (default  $d = 5.0 \text{ \AA}$ ).

### gname *data-name*

Dataset name (default **Coulombic ESP**).

**Usage:**

**crystalcontacts** *model\_number distance options*

The **crystalcontacts** command checks for clashes between copies of a structure when they are positioned according to symmetry information in the input coordinate file (PDB or mmCIF). It is the command-line implementation of [Crystal Contacts](#).

The structure must have been opened from a file containing symmetry information; otherwise, there is no reason to use this command. The *model\_number* (preceded by #) specifies the structure, and *distance* specifies what will be considered a close contact. Atoms to exclude from the calculation (solvent, *etc.*) should be [deleted](#) beforehand.

The results are shown schematically. Each copy of the original structure is shown as a ball, and close contacts between copies are shown as cylinders connecting the balls:

- **green balls** = the original structure and copies related by noncrystallographic symmetry
- **blue balls** = copies related by crystallographic symmetry, together with the green balls representing one unit cell
- **yellow balls** = copies from neighboring unit cells
- **red cylinders** = close contacts

See [Crystal Contacts](#) for details. See also: [sym](#), [distance](#), [findclash](#), [Unit Cell](#)

**Options**

Option keywords can be truncated to unique strings, and their case does not matter. A vertical bar "|" designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

**copies** true|false

Whether to load the full atomic coordinates of any contacting copies.

**residueInfo** true|false

Whether to write a table of residue contact information to the [Reply Log](#).

**buriedAreas** true|false

Whether to calculate per-residue buried solvent-accessible surface area (SASA) values and report them in the [Reply Log](#). To calculate the buried area for a given residue, only the residues in contact with that residue are used; this procedure is computationally much more tractable but may give slightly different results than a [buried-area calculation](#) using all atoms in multiple copies of the structure. Thus, the *distance* used to define contacts should be large enough to include all atoms with buried SASA, at least twice the [probe radius](#) plus twice the maximum [VDW radius](#) (with default radii, about 6.6 Å). Each contacting residue will be assigned an [attribute](#) named **maxCrystalBuriedArea** that gives the maximum buried SASA of any of its copies within the crystal.

**probeRadius** *r*

The probe radius *r* (default 1.4 Å) is the size of the sphere rolled over the surface for [buried area](#) calculations.

**intraBioUnit** true|false

Whether contacts between subunits in the same biological unit should be included in the results. The biological unit is defined by [BIOMT matrices](#) in the input PDB file.

**replace** true|false

Whether to replace any pre-existing schematic representation ([marker set](#)) from **crystalcontacts** with the new one rather than generating an additional model.

## Usage:

**defattr** *attr-file* [options](#)

**Defattr** is the command-line implementation of [Define Attribute](#), a tool for assigning [attribute](#) values to atoms, residues, and models. It is possible to define new attributes or to (re)assign values for existing attributes. Attribute values can be rendered visually ([rangecolor](#), [Render by Attribute](#)) and used in [selection](#) and command-line [atom specification](#).

The attribute values and their targets are specified in an input [attribute assignment file](#). The input file name/pathname (*attr-file*) must be specified before any [options](#). If *attr-file* includes spaces, it must be enclosed in single or double quote marks. If *attr-file* is **browse** or **browser**, a [dialog](#) for opening the file will appear.

## Options

Option keywords for **defattr** can be truncated to unique strings and their case does not matter. A vertical bar "|" designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

**log** true|false

Whether to send details about the attribute assignments to the [Reply Log](#).

**raiseTool** true|false

Whether to open the [Render/Select by Attribute](#) tool after assigning attributes. Even when this option is **false**, the attribute(s) will be available within the [Render/Select by Attribute](#) tool if it is opened later in the session.

**spec** [atom-spec](#)

Whether to restrict the attribute assignment to a subset of the open models. The entire model(s) containing the specified atoms will be considered during the attribute assignment; any other models will be ignored. Whether attribute values are actually assigned to a model or to its constituent atoms or residues also depends on the input [attribute assignment file](#).

If [atom-spec](#) includes any spaces, it must be enclosed in single or double quote marks.

See also: [setattr](#), [rangecolor](#), [Define Attribute](#), [Render by Attribute](#)

**Usage:**

**define** *object options* [atom-spec](#)

**Usage:**

**~define** [ *ID-list* | **axes** | **planes** | **centroids** ]

The **define** command calculates and displays geometric objects based on atomic coordinates. It is the command-line implementation of [Axes/Planes/Centroids](#). The *object* can be:

- [axis](#)
- [plane](#)
- [centroid](#)

Axes can also be calculated from centroids or some combination of atoms and centroids. Each object is created as a [surface model](#) (hidden from the [Model Panel](#)) in the coordinate system of the lowest-numbered model involved in its calculation. The objects are listed in the [Axes/Planes/Centroids table](#) and can be used in distance and angle measurements. See also: [distance](#), [angle](#), [align](#), [aniso](#), [measure center](#), [measure inertia](#), [shape](#), [PipesAndPlanks](#), [geometric objects](#)

The **~define** command removes objects, either those specified as a space-separated list of IDs (for example, **p1 a3**), or all **axes**, all **planes**, all **centroids**, or all of the preceding if **~define** is given without arguments.

Some options of **define** are specific to a given type of object, while others are [general](#). Keywords and their sub-keywords can be truncated to unique strings, and their case does not matter. Synonyms for true: True, 1. Synonyms for false: False, 0. A vertical bar “|” designates mutually exclusive options, and default settings are indicated with **bold**.

• **define axis** [ **perHelix** true | false ] [ **helicalCorrection** true | false ] [ **massWeighting** true | false ]  
[general-options](#) [atom-spec](#)

Define an axis for the [specified](#) atoms or for each protein helix in the model(s) containing those atoms (if **perHelix** is true). If **perHelix** is false, pre-existing centroid objects can also be included in the calculation; a set of atoms and/or centroids can be specified by [selection](#) and then using **sel** or one of its equivalents in the command line, or by including space-separated object IDs (for example, **c1 c2**) with the [atom specification](#) string. Eigenvectors/values are calculated from the coordinates of each set of atoms/centroids after subtracting the position of their collective centroid. Each axis is anchored at the corresponding collective centroid and aligned with the first eigenvector of the corresponding coordinates (prior to any helical correction). Helical correction (by default, on if **perHelix** is true, off otherwise) adjusts the orientation to reduce the spread of atom-axis distances. Mass-weighting (off by default) cannot be combined with helical correction or the **perHelix** option. For mass-weighting, the radius of a centroid (Å) is used as its “mass” (daltons). Axes are displayed as rods. Axis radius is reported in the [Reply Log](#).

Setting **perHelix** to **true** indicates defining an axis for each peptide/protein helix in the entire molecule model(s) containing the [specified](#) atoms, even if only parts were specified. Peptide/protein helix assignments are taken from the input structure file or generated with [ksdssp](#). Only the backbone atoms N, CA, C are used to define the axes. Sometimes two helices are adjacent in sequence (not separated by any other residues), and the integer-valued residue attribute **ssld** is used to distinguish them. Although such cases are detected automatically, **ssld** can also be set manually with [setattr](#).

- **define plane** [ **thickness** *d* ] [ **padding** *offset* ] [general-options](#) [atom-spec](#)

Define a plane for the [specified](#) atoms. Eigenvectors/values are calculated from the atomic coordinates after subtracting the position of their non-mass-weighted centroid. The plane is anchored at the centroid and aligned with the first two eigenvectors (the third eigenvector is normal to the plane). Planes are displayed as disks. The default **thickness** is 0.1 Å. The disk center is the non-mass-weighted centroid of the atoms; disk radius can be set to a fixed value as described [below](#) or set automatically to enclose the projections of the atoms (default). An extra margin (**padding**, default 0.0 Å) can be added to the automatically determined radius, but is ignored if a fixed radius is supplied. Plane radius is reported in the [Reply Log](#).

- **define centroid** [ **massWeighting** true | false ] [general-options](#) [atom-spec](#)

Define a centroid for the [specified](#) atoms. Centroids are displayed as spheres. The default **radius** is 2.0 Å. Centroids can be purely geometric or calculated with atomic mass-weighting. Centroid coordinates are reported in the [Reply Log](#).

## General Options

### **color** *colorname*

Set the object color to *colorname*, which can be any [color name](#) that specifies a single color, or **none** (default), meaning to match the structure as much as possible.

### **radius** *r*

Set object radius to a fixed value *r*. If this option is not used, radius will be set automatically to:

- for axes, the average axis-atom/centroid distance, or for axes based on only two atoms and/or centroids, the constant radius last used in the [axis definition](#) dialog
- for planes, to enclose the projections of the atoms (an additional margin can be specified with the [padding](#) option)
- for centroids, to the default value of 2.0 Å

### **name** *name*

Assign the object a name (default **axis** for an axis, **plane** for a plane, **centroid** for a centroid).

define

**number  $N$**

Assign the object the integer part of its ID. The ID will be  $aN$  for an axis,  $pN$  for a plane,  $cN$  for a centroid. The default is the next available number for that type of object. If the ID is already in use, the new object will replace the pre-existing object.

**raiseTool true|false**

Whether to raise the [Axes/Planes/Centroids table of objects](#).

delete

[Chimera Commands Index](#)

**Usage:**

**delete** *[atom-spec](#)*

**Delete** removes the specified atoms and their bonds, like **delete** in the [Atoms/Bonds](#) section of the [Actions menu](#). The removed items cannot be recovered except by reopening the original input.

See also: [bond](#), [longbond](#), [display](#), [show](#), [close](#)

display

[Chimera Commands Index](#)

**Usage:**

**display** [atom-spec](#)

**Usage:**

**~display** [atom-spec](#)

**Display** selectively displays the [specified](#) atoms. **~Display** can be used to remove unwanted atoms from an existing display.

See also: [chain](#), [show](#), [surface](#), [bonddisplay](#), [modeldisplay](#), [objdisplay](#), the [Model Panel](#), the [Actions menu](#)

distance

[Chimera Commands Index](#)

**Usage:**

**distance** [ **signed** ] *item1 item2*

**Usage:**

**~distance** [ *atom1 atom2* ]

**Usage:**

**~distance all**

The **distance** command measures the distance between two items:

- two atoms
- two geometric objects created with [define](#) or [Axes/Planes/Centroids](#)
- one atom and one geometric object

Any [specification](#) that gives the proper number of items can be used, including the word **selected**, **sel**, or **picked** to indicate the current [selection](#) of atoms and/or objects. Otherwise, objects can be specified by ID as shown in the [Axes/Planes/Centroids table](#). The distance is shown temporarily in the [status line](#) and recorded in the [Reply Log](#).

Atom-atom distance measurements are *monitors*, meaning that a [pseudobond](#) is drawn between the two atoms and the pseudobond's label will automatically update to reflect any changes in the distance. Updates are not sent to the [Reply Log](#), however. The [Distances](#) tool lists the current distance monitors and allows setting default distance [formatting](#) and [depiction](#) options. These settings are stored in a user's [preferences file](#).

An individual distance monitor can be removed by using **~distance** with [specification](#) of the two atoms involved. Using **~distance** without any argument or with the keyword **all** removes all distance monitors.

Other types of distance measurements (atom-object and object-object) are simply reported in the [status line](#) in addition to the [Reply Log](#). For distance measurements, axes are treated as finite line segments, while planes are treated as infinite. For atom-plane and centroid-plane distances only, the **signed** keyword can be supplied to indicate that distances on opposite sides of the plane should be reported with opposite signs.

Examples:

```
distance :bgc@c1 a4
```

- measure the distance between atom C1 in residue BGC and axis a4

```
distance c1 c2
```

distance

- measure the distance between centroid c1 and centroid c2

**distance signed p2 :313.a@o**

- measure the signed distance between plane p2 and atom O in residue 313 of chain A

**~distance #1:12@CA #0:47@CA**

- stop monitoring the distance between atom CA of residue 12 in model 1 and atom CA of residue 47 in model 0

See also: [findclash](#), [hbonds](#), [crystalcontacts](#), [angle](#), [define](#), [measure distance](#)

**Usage:**

**echo** *text*

**Echo** places the *text* argument in the [status line](#) and [Reply Log](#). **Echo** may be used in scripts to send messages back to the user.

**Usage:**

**export** [ *format format* ] [ *filename* ]

**Usage:**

**export list**

The command **export** saves the graphical scene as a file named *filename* in the specified format (see [limitations](#)):

- **X3D** [.x3d] - [X3D](#) (default)
- **COLLADA** [.dae] - [COLLADA](#) XML-based format; can be viewed with Mac Preview and incorporated into iBooks
- **OBJ** [.obj] - [Wavefront OBJ](#) (surfaces only); colors are written to a separate material definition file [.mtl]
- **POV-Ray** [.pov] - input for the [Persistence of Vision Raytracer](#) (alternatively, POV-Ray images can be produced [directly](#))
- **RenderMan** [.rib] - [RenderMan](#)<sup>®</sup> Interface Bytestream
- **STL** [.stl] - triangle-based format (binary) native to stereolithography CAD software from [3D Systems](#)<sup>®</sup> (see the [Chimera 3D Model Gallery](#))
- **VRML** [.wrl, .vrm] - [Virtual Reality Markup Language](#)
- **VTK** [.vtk] - format used by [Visualization Toolkit](#)
- **WebGL** [.html] - HTML5 format with embedded [WebGL](#), to show a manipulable rendering of the scene in a web page (experimental feature)

The format can be specified with the *filename* suffix or separately with the keyword **format**. The keyword specification is case-independent and will override any specification by suffix, and the appropriate suffix will be added if *filename* does not already include it. If *filename* is omitted, a [dialog](#) for specifying the file name and location will appear.

The command **export list** reports the available *format* settings in the [status line](#) and [Reply Log](#). Chimera extensions that enable further export types can add to the *format* list.

POV-Ray, RenderMan, STL, and VRML files are generated by postprocessing an initial X3D file. The conversion programs *x3d2pov*, *x3d2RM*, *x3d2stl*, and *x3d2vrml* are included with Chimera (in the bin directory) but can be used separately.

See also: [struts](#), [save](#), [write](#), [copy](#), [pdbrun](#), [Write DMS](#), [exporting a scene](#), [raytracing with POV-Ray](#)

**Usage:**

**fillring** [ **thin** | **thick** | **unfilled** | **off** ] [atom-spec](#)

**Usage:**

**~fillring** [atom-spec](#)

**Fillring** controls whether and how rings (up to 6 members only) are shown as filled:

- **thin** - “paper” thin regardless of [atom/bond representation](#)
- **thick** (default) - as thick as the surrounding bonds
- **unfilled** or **off** - no ring fill (same as using **~fillring**)

Ring fill display and style are residue [attributes](#), so the entire residues(s) containing [atom-spec](#) will be affected. These settings can also be adjusted using the [Selection Inspector](#) or the command [setattr](#).

Aromatic rings can only be shown as filled when their [aromaticity display](#) is turned off.

See also: [aromatic](#), [nucleotides](#), [represent](#), the [Actions menu](#)

**Usage:****findclash** [atom-spec](#) [options](#)**Usage:****~findclash**

**Findclash** identifies interatomic clashes and contacts based on [VDW radii](#) and user-specified criteria. It is the command-line implementation of [Find Clashes/Contacts](#). Terminology:

- *clashes* - unfavorable interactions where atoms are too close together; close contacts
- *contacts* - all kinds of direct interactions: polar and nonpolar, favorable and unfavorable (including clashes)

During [continuous monitoring](#), such interactions can be shown with [selection](#), coloring, and [pseudobonds](#). Discontinuous monitoring additionally allows writing out results and assigning the largest [overlap](#) per atom as an [attribute](#) named **overlap**. The command **~findclash** removes any [pseudobonds](#) that have been drawn to show the interactions and halts any continuous monitoring.

The *overlap* between two atoms is defined as the sum of their [VDW radii](#) minus the distance between them and minus an [allowance](#) for potentially hydrogen-bonded pairs:

$$overlap_{ij} = r_{VDW_i} + r_{VDW_j} - d_{ij} - allowance_{ij}$$

See also: [distance](#), [hbonds](#), [crystalcontacts](#), [intersurf](#), [minimize](#), the [PseudoBond Panel](#)

**Options**

Option keywords for **findclash** can be truncated to unique strings and their case does not matter. A vertical bar "|" designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

**test** [atom-spec](#) | **other** | model | self

Which interactions to find: between the specified atoms and a second set of atoms (indicated by [atom-spec](#)), between the specified atoms and all other atoms (**other** or **others**, default), intra-model interactions between the specified atoms and all other atoms (**model**), or among the specified atoms (**self**). If [atom-spec](#) includes any spaces, it must be enclosed in single or double quote marks.

**interSubmodel** true | false

Whether to look for interactions between [submodels](#) of the same model.

**overlapCutoff** *cutoff*

Pairs of atoms with [overlap](#)  $\geq$  *cutoff* (default **0.6** Å) will be identified. A larger positive *cutoff* restricts the results to more severe clashes, whereas a negative *cutoff* can also identify favorable contacts.

- For detecting [clashes](#), a *cutoff* of 0.4-1.0 Å and a hydrogen bond [allowance](#) of 0.2-0.6 Å are recommended. The **findclash** command defaults are the same as the default clash criteria in the [Find Clashes/Contacts](#) graphical interface: 0.6 and 0.4 Å, respectively.
- For detecting [contacts](#), negative *cutoff* values of 0.0-(-1.0) Å are recommended along with a hydrogen bond [allowance](#) of 0.0 Å. The default contact criteria in the [Find Clashes/Contacts](#) graphical interface are -0.4 and 0.0 Å, respectively.

#### **hbondAllowance** *allowance*

An *allowance*  $> 0$  (default **0.4** Å) reflects the observation that atoms sharing a hydrogen bond can favorably approach each other more closely than would be expected from their [VDW radii](#). See [above](#) for suggested values. In the calculation of [overlap](#), the *allowance* is only subtracted for pairs comprised of a donor (or donor-borne hydrogen) and an acceptor. This is equivalent to using smaller radii to characterize hydrogen-bonding interactions (for example, see [Li and Nussinov](#), Proteins 32:111 (1998)). As in [FindHBond](#), possible donor groups are hydrogen-bearing nitrogen, oxygen, and sulfur atoms, and possible acceptor groups are nitrogen, oxygen, and sulfur atoms with a lone pair.

#### **bondSeparation** *N*

Interactions between atoms that are  $\leq N$  bonds apart will be ignored ( $N=4$  by default).

#### **intraRes** true | false

Whether to include intraresidue interactions.

#### **intraMol** true | false

Whether to include intramolecular interactions, where a molecule is defined as a covalently bonded set of atoms.

#### **continuous** true | false

Whether to initiate continuous checking, which will continue until the specified atoms are removed, or  $\sim$ **findclash** is used, or **findclash** is used again without setting this option to **true** (since **false** is the default). Only moving a model relative to another will trigger a new check, whereas continuous checking with the [Find Clashes/Contacts](#) graphical interface will also detect changes from bond rotation or trajectory playback, regardless of relative model motion.

#### **selectClashes** true | false

Whether to [select](#) the identified clash/contact atoms (and deselect all other atoms).

#### **colorClashes** true | false

Whether to color the identified clash/contact atoms the [clashColor](#) and the remaining atoms the [nonclashColor](#).

**clashColor** *colorname*

Color to use for the identified clash/contact atoms when [colorClashes](#) is **true** (default red). *Colorname* can be **none** or any [color name](#) that specifies a single color.

**nonclashColor** *colorname*

Color to use for atoms other than the identified clash/contact atoms when [colorClashes](#) is **true** (default **none**, which removes per-atom color assignments and reveals the [model-level colors](#)). *Colorname* can be **none** or any [color name](#) that specifies a single color.

**makePseudobonds** true | false

Whether to create [pseudobonds](#) to show the identified interactions. These pseudobonds can be removed with the command `~findclash`. The [PseudoBond Panel](#) can also be used to close the pseudobond group (named **contacts**) or alter its display.

**pbColor** *colorname*

Color to use for the [pseudobonds](#) when [makePseudobonds](#) is **true** (default **yellow**). *Colorname* can be **none** or any [color name](#) that specifies a single color.

**lineWidth** *width*

Line width to use for the [pseudobonds](#) when [makePseudobonds](#) is **true** (default 2.0 pixels).

**reveal** true | false

Whether to turn on the display of residues containing a contact atom if that atom is not displayed initially. (By default, if the atom on either end of a [pseudobond](#) representing a contact is not shown, the pseudobond itself is not shown, although it still exists; displaying the atom allows the contact pseudobond to be shown.)

**setAttrs** true | false

Whether to assign the largest [overlap](#) per atom as an [attribute](#) named **overlap**. Disallowed with [continuous monitoring](#).

**saveFile** *file*

Whether to write contact/clash information to a file (this option is **off** unless specified). The information includes atom specifications (see [namingStyle](#)), [overlaps](#), and interatomic distances. *File* is the output file name/pathname; if it includes spaces, it must be enclosed in single or double quote marks. If *file* is **browse** or **browser**, a [dialog](#) for saving the file will appear (unless the word is enclosed in quotes, which would generate an output file by that name). Disallowed with [continuous monitoring](#).

**log** true | false

Whether to write contact/clash information (as described above for [saveFile](#)) to the [Reply Log](#). Disallowed with [continuous monitoring](#).

**namingStyle** simple | command | serial

How to describe atoms in contact/clash information written to a [file](#) or the [log](#):

- **simple** - residue name, residue specifier, and atom name (for example, HIS 16.A ND1)
- **command** or **command-line** - [command-line specifier](#) (for example, :16.A@ND1)
- **serial** or **serialnumber** - atom serial number (for example, 126)

Model number will be included when multiple models are present. If **namingStyle** is not supplied, the **Atomspec display style** specified in the [Labels preferences](#) will be used.

**summary** true | false

Whether to write the total number of interactions found to the [status line](#) and [Reply Log](#).

**Usage:****( hbonds | findhbond ) [options](#)****Usage:****( ~hbonds | ~findhbond )**

**Hbonds** (or **findhbond**) is the command-line implementation of [FindHBond](#), which uses [atom types](#) and [geometric criteria](#) to identify possible hydrogen bonds (H-bonds). See the [video mini-example](#). See also: [addh](#), [findclash](#), [distance](#), [PseudoBond Reader](#)

It is not necessary for hydrogen atoms to be present. All potential H-bonding interactions fulfilling the criteria are shown. For example, even if it is not possible for a particular hydroxyl group to donate a hydrogen bond to two particular acceptors *simultaneously*, both possibilities will be displayed if the hydroxyl lacks an explicit hydrogen atom. If the hydroxyl group has an explicit hydrogen atom, however, only H-bonds compatible with the position of the hydrogen will be found. Consult the [FindHBond](#) manual page for more information on the [method and criteria](#).

By [default](#), hydrogen bonds are shown as [pseudobonds](#). \*\*If the endpoint atoms are subsequently moved relative to each other, the lines will “stretch” rather than disappearing, even if the atoms no longer meet the criteria for hydrogen bonding.\*\* After such changes, the H-bonds should be updated by rerunning the calculation, or removed, *e.g.* with the command **~hbonds** (or **~findhbond**).

**Options**

Option keywords for **hbond** (**findhbond**) can be truncated to unique strings and their case does not matter. A vertical bar “|” designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

**makePseudobonds true | false**Whether to show the H-bonds as lines (a group of [pseudobonds](#) named **hydrogen bonds**).

The pseudobonds will be drawn between the donor and acceptor atoms, or if hydrogens are present, between the hydrogens and acceptor atoms.

**lineWidth width**Line width to use for [pseudobonds](#) depicting H-bonds (when in the **wire** draw mode); *width* is **1.0** pixels by default.**color colorname**Color to use for [pseudobonds](#) depicting H-bonds (but see also [twoColors](#) and [slopColor](#)).The default color is similar to cyan. *Colorname* can be **none** or any [color name](#) that specifies a single color.**showDist true | false**Whether to label the [pseudobonds](#) with the corresponding distances. The number of

decimal points and whether the Å symbol is displayed can be set in the [Distances](#) tool (other settings in that tool will not affect **hydrogen bonds** pseudobonds, however).

**selRestrict** [atom-spec](#) | cross | both | any

Whether to limit H-bond detection to one of the following:

- those between atoms in [atom-spec](#) and the current [selection](#); the specified atoms and the selection may overlap, but [atom-spec](#) will not overrule other restrictions (for example, if the specified atoms are in a different model than the selected atoms but [intermodel](#) is **false**, no H-bonds will be found). If [atom-spec](#) includes any spaces, it must be enclosed in single or double quote marks.
- **cross** - H-bonds with exactly one atom [selected](#)
- **both** - H-bonds with both atoms [selected](#)
- **any** - H-bonds with at least one atom [selected](#)

This option is **off** unless specified.

**spec** [atom-spec](#)

Whether to restrict H-bond detection to a subset of the open molecule models. The *entire model(s)* containing the specified atoms will be considered. If [atom-spec](#) includes any spaces, it must be enclosed in single or double quote marks.

**interSubmodel** true | false

Whether to look for H-bonds between [submodels](#) of the same model.

**intermodel** true | false

Whether to look for H-bonds between models.

**intramodel** true | false

Whether to look for H-bonds within models.

**intraMol** true | false

Whether to include intramolecular H-bonds, where a molecule is defined as a covalently bonded set of atoms.

**intraRes** true | false

Whether to include intraresidue H-bonds.

**cacheDA** true | false

Whether to cache and reuse donor and acceptor information rather than regenerating it each time the command is used; useful for calculations on different conformations of the same structure. The default is **true** for a trajectory open in [MD Movie](#) (although a trajectory may not be recognized until more than one set of coordinates has been read), otherwise **false**.

**relax** true | false

Whether to relax the [precise criteria](#) for hydrogen bonding.

**distSlop** *tolerance*

How much to relax the distance criteria if [relax](#) is **true**; *tolerance* is **0.4 Å** by default.

**angleSlop** *tolerance*

How much to relax the angle criteria if [relax](#) is **true**; *tolerance* is **20.0** degrees by default.

**twoColors** true | false

Whether to use different colors for H-bonds meeting and not meeting the precise criteria (*i. e.*, whether a different color should be used for H-bonds outside the precise criteria but within the tolerance values).

**slopColor** *colorname*

If [twoColors](#) is **true**, the color to use for depicting H-bonds not meeting the precise criteria but within the tolerance values. The default color is similar to orange. *Colorname* can be **none** or any [color name](#) that specifies a single color.

**reveal** true | false

Whether to turn on the display of residues containing an H-bonding atom if that atom is not displayed initially. (By default, if the atom on either end of a [pseudobond](#) representing an H-bond is not shown, the pseudobond itself is not shown, although it still exists; displaying the atom allows the H-bond to be shown.)

**retainCurrent** true | false

Whether to keep the [pseudobonds](#) depicting a previously determined set of H-bonds through a subsequent round of detection (as opposed to replacing the old with the new).

**saveFile** *file*

Whether to save H-bond information to a file (this option is **off** unless specified). The information includes donor and acceptor atom specifications (see [namingStyle](#)), donor-acceptor distances, and when hydrogens are present, hydrogen atom specifications and hydrogen-acceptor distances. *File* is the output file name/pathname; if it includes spaces, it must be enclosed in single or double quote marks. If *file* is **browse** or **browser**, a [dialog](#) for saving the file will appear (unless the word is enclosed in quotes, which would generate an output file by that name).

**log** true | false

Whether to write H-bond information (as described above for [saveFile](#)) to the [Reply Log](#).

**namingStyle** simple | command | serial

How to describe atoms in H-bond information written to a [file](#) or the [log](#):

- **simple** - residue name, residue specifier, and atom name (for example, HIS 16.A ND1)
- **command** or **command-line** - [command-line specifier](#) (for example, :16.A@ND1)
- **serial** or **serialnumber** - atom serial number (for example, 126)

Model number will be included when multiple models are present. If **namingStyle** is not supplied, the **Atomspec display style** specified in the [Labels preferences](#) will be used.

**batch** true | false

Whether to suppress certain errors; useful for no-GUI batch processing. Currently, the only error suppressed when **batch** is **true** is that which arises if no atoms are [selected](#) when [selRestrict](#) is specified.

**Usage:**

**fitmap** *fit-structure ref-map* [options](#) [global-search-options](#)

The command **fitmap** fits atomic coordinates into a density map or one density map into another. It is the command-line implementation of [Fit in Map](#), but it also has options that are not available in that tool:

- [global search](#) with random initial placement
- [sequential fitting](#) of multiple different structures
- [symmetric fitting](#) of copies of the same structure

The maps usually represent electron density, but other types of [volume data](#) can also be used.

The *fit-structure*, either [specified atoms](#) or a map model, will be fit to *ref-map* (a map model). Map models are specified by model number preceded by #. If atoms are specified, only those atoms will be used in the fitting calculation, but the entire model(s) containing them will be repositioned unless [moveWholeMolecules](#) is set to false.

Prior to [local optimization](#), the fit model should be placed in a trial position relative to the reference map before fitting. This usually involves [interactive manipulation](#) and toggling models between [active](#) and immovable states. However, the **fitmap** command also has [global search options](#) that are not available in the [Fit in Map](#) graphical interface.

The calculation will stop and the fit structure will be repositioned after [convergence](#) or a [maximum number of steps](#) (default 2000), whichever comes first. Reissuing the **fitmap** command may further improve results, especially if convergence was not reached. The [Fit in Map](#) graphical interface can be used to undo the fit. After fitting, atomic coordinates can be [saved](#) relative to the reference map.

Information such as the number of optimization steps, shift, and rotation is sent to the [Reply Log](#). If the entire fit model is repositioned, its transformation relative to the reference map is given as a [transformation matrix](#) and as an axis of rotation (a unit vector), point on the axis, degrees of rotation, and shift parallel to the axis.

See also: [volume](#), [measure](#), [molmap](#), [sym](#), [Fit to Segments](#), [Values at Atom Positions](#), [MultiFit](#), [saving maps after fitting](#)

**Options**

Option keywords for **fitmap** can be truncated to unique strings, and their case does not matter. Synonyms for true: True, 1. Synonyms for false: False, 0. A vertical bar "|" designates mutually exclusive options, and default settings are indicated with **bold**.

**resolution** *r*

Generate a density map from the coordinates of the specified atoms and perform [map-in-](#)

[map](#) fitting instead of [atoms-in-map](#) fitting. Both types of fit values will still be reported. The map is generated by describing each atom as a Gaussian distribution of width proportional to  $r$  and amplitude proportional to the atomic number; other map generation parameters are set to [molmap](#) defaults. If atoms are specified but this option is not given, *atoms-in-map* fitting will be performed:

The average map value at fit atom positions is maximized. For each atom within the bounds of the reference map, the map value is found by trilinear interpolation from the eight corners of the enclosing data grid cell. Atoms outside the bounds of the map are not used for computing averages.

**metric overlap** | correlation | cam

Which metric to use for *map-in-map* fitting:

The **overlap** (default except during [symmetric fitting](#)) is the sum over fit map grid points of the product of the fit map value and the reference map value at that point, determined by trilinear interpolation. It can be expressed as the inner product of vectors  $\mathbf{u}$  and  $\mathbf{v}$  containing the fit map values and the corresponding interpolated reference map values:

$$\text{overlap} = \langle \mathbf{u}, \mathbf{v} \rangle$$

The other possibilities are **correlation** about zero (default during [symmetric fitting](#)) and **cam** (correlation about the mean):

$$\text{correlation} = \frac{\langle \mathbf{u}, \mathbf{v} \rangle}{|\mathbf{u}| |\mathbf{v}|}$$

$$\text{cam} = \frac{\langle \mathbf{u} - \mathbf{u}_{\text{ave}}, \mathbf{v} - \mathbf{v}_{\text{ave}} \rangle}{|\mathbf{u} - \mathbf{u}_{\text{ave}}| |\mathbf{v} - \mathbf{v}_{\text{ave}}|}$$

where  $\mathbf{u}_{\text{ave}}$  is a vector with all components equal to the average of the components of  $\mathbf{u}$  and  $\mathbf{v}_{\text{ave}}$  is defined analogously. The correlation equals the cosine of the angle between the vectors (after subtraction of averages) and can range from -1 to 1, whereas the range of overlap values depends on the scaling of the maps.

**envelope true** | false

For [map-in-map](#) fitting, whether to use only the grid points in the fit map with values above the map's lowest contour level in [Volume Viewer](#). Otherwise, all nonzero-valued fit map grid points will be included. (Exception: for [symmetric fitting](#), this option controls which reference map points, rather than fit map points, are used.)

**shift** true | false

Whether to allow translation of the fit structure during [local optimization](#).

**rotate** true | false

Whether to allow rotation of the fit structure during [local optimization](#).

**moveWholeMolecules** true | false

Whether to reposition the entire model(s) containing the specified atoms. If false, only the specified atoms will be moved. Regardless of this setting, only the specified atoms will be used to calculate the fit. This option is ignored (always **true**) when [global searching](#) is performed.

**gridStepMax** *max*

Initial step size, default **0.5** grid unit, where a grid unit is the spacing between reference map grid points. See [local optimization algorithm](#).

**maxSteps** *N*

Maximum number of optimization steps per use of the **fitmap** command (default **2000**). See [local optimization algorithm](#).

**gridStepMin** *min*

Criterion for *convergence*, when step size falls below *min* grid units (default **0.01**). See [local optimization algorithm](#).

**eachModel** true | false

When multiple fit models are specified, whether to fit each model independently of the others (cannot be combined with [sequential fitting](#)). The [listFits](#) option can be used to show the results in the [Fit List](#) dialog.

**listFits** true | false

Whether to show results in the [Fit List](#) dialog (default **true** when [global searching](#) is performed, otherwise default **false**).

**sequence** *M*

When multiple fit models are specified, whether to fit each model in turn after subtracting the density corresponding to the other models (cannot be combined with the [eachModel](#) option). Only applies to [map-in-map](#) fitting; if atomic models were specified, the [resolution](#) option must be used to generate maps from those models. *M* is the number of individual structure fits to perform, each time first subtracting (temporarily) from the reference map the density corresponding to the other specified fit models in their current positions (default *M* = **0**, no sequential fitting). Thus, the fit models should be placed in trial positions beforehand by [interactive manipulation](#) and/or prior fitting runs. If *M* is greater than the number of fit models, the calculation will continue to cycle through those models in the order listed. In tests, good convergence was attained by cycling through all of the models five times. Currently sequential fitting cannot be done in the same command as [symmetric fitting](#) or [global search](#). See also: [sequential fitting video tutorial](#) (Web)

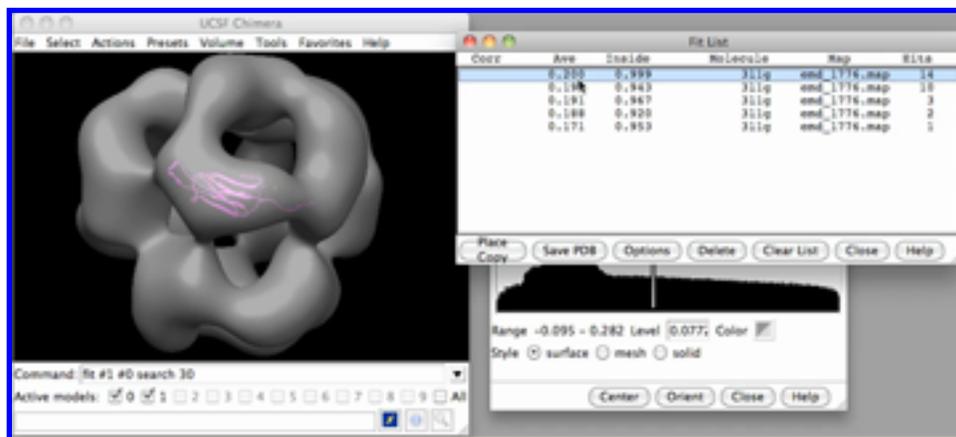
**symmetric** true | false

Whether to use the symmetry of the reference map while fitting. Only applies when the reference map has a [symmetry assignment](#), and to [map-in-map](#) fitting; if atoms were specified, the [resolution](#) option must be used to generate a map from those atoms. During symmetric fitting, the fit map and its symmetry-related virtual copies are fit into the reference map using the [metric](#) of **correlation** (default) or **cam**. Overlaps between fit map copies additively raise the fit density and tend to lower the correlation. For computational efficiency, only one asymmetric unit of the reference map is considered explicitly (reference map grid points closer to the center of the original fit map than to the centers of its copies). The [envelope](#) setting determines whether all nonzero-valued reference map grid points in the asymmetric unit or only those above the contour level (default) are used. Currently symmetric fitting cannot be done in the same command as [sequential fitting](#) or [global search](#). Whereas symmetric fitting uses virtual copies of the fit map, symmetry-related actual copies of the corresponding atomic model can be created with the command [sym](#). See also: [symmetric fitting video tutorial](#) (Web)

## Global Search Options

Click for [video tutorial](#) on [fitmap](#) global search (Web)...

A [search](#) value  $N > 0$  indicates some degree of global searching with the [fitmap](#) command. In global search,  $N$  initial placements of the fit model within the reference map are generated randomly, then subjected to [local optimization](#). The whole model will be moved regardless of the [moveWholeMolecules](#) setting. The resulting unique fits are listed in a separate dialog, the [Fit List](#), where uniqueness depends on [rotational differences](#), [translational differences](#), and lack of [equivalence by symmetry](#).



In addition, the user can require [some fraction](#) of the fit atoms or fit map grid points to be inside the reference map contour surface for the fit to be retained. Only the first fit in a uniqueness cluster is listed, along with the number of cluster members (hits).

**search**  $N$

Number of initial placements (prior to [local optimization](#)) of the fit model within the reference map (default 0, no global search). The [placement](#) option can be used to constrain initial placements to only rotations or shifts from the current position.

**placement** s | r | sr

In global search, how to generate initial placements of the fit model:

- **s** - random shifts (translations) starting from the current position, keeping the geometric center of the fit atoms or fit map grid points within the bounding box of the displayed part of the reference map. The search radius can be restricted further with the [radius](#) option. The [envelope](#) setting determines whether all nonzero-valued fit map grid points or only those above the contour level (default) are used to calculate the center.
- **r** - random rotations starting from the current position

- **sr** (default) - both shifts and rotations

This option does not affect what movements are allowed during local optimization, which are instead set by [shift](#) and [rotate](#).

#### **radius** *maxdist*

Limit the global search to initial placements within *maxdist* of the current position.

#### **clusterAngle** *angle*

The *angle* (default 6°) is the rotational difference required for a fit to be considered unique. Only unique fits are included in the list of results.

#### **clusterShift** *shift*

The *shift* (default 3 Å) is the translational difference required for a fit to be considered unique. Only unique fits are included in the list of results.

#### **asymmetricUnit** **true** | false

If the reference map has symmetry information (such as from [measure symmetry](#)), whether to list only the fits from one asymmetric unit. In other words, whether to exclude symmetry-equivalent fits from being considered unique. Of a symmetry-equivalent set of fits, the one that places the fit structure geometric center closest to volume box fractional coordinates (0.75,0.55,0.55) in the reference map is included in the list of results.

#### **inside** *fraction*

The *fraction* is what proportion of fit atoms or fit map grid points must lie inside the reference map contour surface for the fit to be retained (default 0.1). The [envelope](#) setting determines whether all nonzero-valued fit map grid points or only those above the contour level (default) are considered.

## Fit List

Unique fits from running the [fitmap](#) command with [global search](#) are enumerated in a **Fit List** dialog. The [listFits](#) option can be used to show the results of other [fitmap](#) runs as well (for example, fitting multiple models independently with the [eachModel](#) option).

Clicking a row *chooses* the corresponding fit and moves the fit model to regenerate it. The **Fit List** is included in saved [sessions](#). Columns:

- **Corr** - [map-in-map](#) fitting metric: correlation about zero
- **Ave** - [atoms-in-map](#) fitting metric: average map value at fit atom positions
- **Inside** - what proportion of fit atoms ([atoms-in-map](#) fitting) or fit map grid points ([map-in-map](#) fitting) are inside the reference map contour surface. The [envelope](#) setting determines whether all nonzero-valued fit map grid points or only those above the contour level (default) are considered.
- **Clash** - overlap of the fit model contour surface with its symmetry-related copies, see [Fit List options](#) (applies only to [map-in-map](#) fitting when the reference map has symmetry information)
- **Molecule** - fit model
- **Map** - reference model

- **Hits** - number of fits in the uniqueness cluster

Clicking **Options** reveals additional settings (clicking the close button  on the right hides them again):

- **Smooth motion between fits [ M ] steps** - number of frames over which to move the fit model from its current position to the fit position when a row is clicked
- **Show clash volume fraction between symmetric copies** - whether to show a column of **Clash** values, the amount of overlap between the fit model contour surface and its symmetry-related copies (applies only to [map-in-map](#) fitting when the reference map has symmetry information). The values depend on the fit map contour level. The clash fraction equals the number of grid points from symmetry-related copies that fall inside the original copy, divided by the total number of grid points within the original.

Other buttons:

- **Place Copy** - if atoms were specified as the fit structure, open a copy for each [chosen fit](#) as a new model
- **Save PDB** - if atoms were specified as the fit structure, save each [chosen fit](#) as a PDB file
- **Delete** - remove each [chosen fit](#) from the list
- **Clear List** - remove all fits
- **Close** - dismiss the **Fit List**
- **Help** - show documentation in a browser window

**Usage:**

**fly** [*master-frames*] *pos1* [[ *frames1\_2*] *pos2*] [[ *frames2\_3*] *pos3*] ... [[ *framesN-1\_M*] *posM*]

**Fly** uses cubic interpolation to smoothly traverse a series of positions previously named and saved with [savepos](#). The command [reset](#) can also interpolate between positions, but it only considers a pair of positions (start and end) at a time, whereas **fly** can consider multiple positions and generate a path that visits and leaves intermediate positions without discontinuities in motion. See the [video mini-example](#). See also: [play](#), [scene](#), [movie-related commands](#)

A position includes model transformations, scale, and [clipping plane](#) positions (see [savepos](#) for further details). Saved positions are included in saved [Chimera sessions](#).

**Fly** starts abruptly at the first position, *pos1*, then proceeds through the list of subsequent positions. Besides positions previously saved with [savepos](#), the current position can be indicated with the name **start**. (Using the name **start** in the **fly** command also saves the position; it could be restored later with [reset](#).) The *master-frames* argument (default **1**) is how many frames each "leg" of the journey (from one position to the next) should take, unless overridden by a frames argument specific to that leg. Examples:

**fly 20 start p1**

- go from the current position to p1 in 20 frames

**fly p1 p2 p3**

- go from p1 to p2 in 1 frame and then to p3 in 1 frame

**fly 50 p1 p2 p3**

- go from p1 to p2 in 50 frames and then to p3 in 50 frames

**fly p1 10 p2 25 p3**

- go from p1 to p2 in 10 frames and then to p3 in 25 frames

**fly 40 p1 100 p2 p3**

- go from p1 to p2 in 100 frames and then to p3 in 40 frames

**Usage:****focus** [ [atom-spec](#) ]**Usage:****~focus**

**Focus** adjusts the view and center of rotation based on displayed atoms, bonds, ribbon segments, and/or [surfaces](#). Items that are invisible only because they fall outside the window boundaries or are [clipped](#) are still considered displayed. Any combination of [atoms](#) and [surface pieces](#) can be specified, but only those which are also displayed will be considered.

**Focus** with an [atom-spec](#):

1. turns on [global clipping](#) and adjusts the clipping plane positions and scale to include the specified items that are also displayed
2. sets the [center of rotation method](#) to the **center of view** (like [cofr view](#))

**Focus** with nothing specified (equivalent to **~focus**):

1. turns off [global clipping](#) and adjusts the view to include all displayed items (like [window](#) sans arguments)
2. sets the [center of rotation method](#) to **front center** (like [~cofr](#))

An exception to the above is that the [center of rotation method](#) is not changed if it is **independent**.

The center of rotation can also be controlled with the [Rotation tool](#).

See also: [center](#), [cofr](#), [window](#), the [Actions menu](#)

freeze

[Chimera Commands Index](#)

**[Usage:](#)**

**freeze**

Freeze stops all motion on the screen.

See also: [move](#), [rock](#), [roll](#), [section](#), [select](#), [thickness](#), [turn](#), [movie-related commands](#)

**Usage:**

**getcrd** [ **xformed** ] [atom-spec](#)

The **getcrd** command reports the coordinates of the [specified](#) atom(s) in the [Reply Log](#). The **xformed** keyword indicates reporting the transformed coordinates (laboratory frame of reference) instead of the untransformed coordinates (model frame of reference, default). The keyword can be truncated.

See also: [chirality](#), [write](#)

help

[Chimera Commands Index](#)

**Usage:**

**help** [ *command* | *filename* | **commands** ]

**Help** displays information on the selected *command*, or if a full pathname to a file (beginning with a "/") is given instead, the contents of that file are displayed. Giving the keyword **commands** shows the entire [Commands Index](#). Using **help** without any arguments calls up the [documentation search](#) dialog.

See also: the [Help menu](#)

**Usage:****hkcage** *h k* [options](#)

The **hkcage** command creates a mesh of hexagons and pentagons to represent the arrangement of proteins in an icosahedral virus capsid. The related tool [Icosahedron Surface](#) generates a mesh of triangles rather than hexagons and pentagons. See also: [shape icosahedron](#), [meshmol](#), [Cage Builder](#)

The arrangement of subunits in an icosahedral capsid can be described as a sheet of hexagons in which curvature is introduced by replacing certain hexagons with pentagons, as in a geodesic dome. The pentagons occupy the points of the icosahedron, while the indices *h* and *k* refer to the number and arrangement of hexagons in each face. A virus T number ( $T = h^2 + hk + k^2$ ) is proportional to the number of subunits in the capsid. [VIPERdb](#) includes viruses with  $T = 1, 3, 4, 7, 13$ , and in a few cases, even higher. For more details, see the [VIPERdb description](#) and the following reference:

[Quasi-equivalent viruses: a paradigm for protein assemblies](#). Johnson JE, Speir JA. *J Mol Biol*. 1997 Jun 27;269(5):665-75.

The required arguments *h* and *k* can each be zero (but not both zero) or a positive integer. The calculation can be slow if large values are used.

The mesh is created as a new [surface model](#) centered at (0,0,0). Hexagons are bent where they cross from one triangular face of the icosahedron to another, so that certain "hexagon" edges are formed by two straight line segments rather than one.

**Options**

Option keywords for **hkcage** can be truncated to unique strings, and their case does not matter. Synonyms for true: True, 1. Synonyms for false: False, 0. A vertical bar "|" designates mutually exclusive options, and default settings are indicated with **bold**.

**radius** *r*

The radius *r* is the distance from the center of the icosahedron to a 5-fold vertex (default **100.0**).

**orientation** *type*

The *type* of icosahedral orientation can be:

- **222** (default) - with two-fold symmetry axes along the X, Y, and Z axes
- **2n5** - with two-fold symmetry along X and 5-fold along Z
- **n25** - with two-fold symmetry along Y and 5-fold along Z
- **2n3** - with two-fold symmetry along X and 3-fold along Z
- **222r** - same as 222 except rotated 90° about Z
- **2n5r** - same as 2n5 except rotated 180° about Y
- **n25r** - same as n25 except rotated 180° about X
- **2n3r** - same as 2n3 except rotated 180° about Y

**color** *mesh-color*

The *mesh-color* can be any [color name](#) that specifies a single color (default **white**).

**linewidth** *width*

The *width* is the pixel linewidth of the mesh display (default **1.0**).

**sphere** *f*

The sphere factor *f* is the weight of the sphere component in an interpolation between an icosahedron and a sphere of equal radius. It ranges from the default of **0.0** (icosahedron) to 1.0 (sphere). The interpolation only involves vertex positions and will not generate curved mesh lines.

**replace** *true|false*

Whether to replace an existing **hk cage** mesh with the new one rather than opening an additional model.

## Usage:

**intersurf** [atom-spec1](#) [atom-spec2](#) [options](#)

**Intersurf** is the command-line implementation of the [Intersurf](#) tool, which creates and displays an [interface surface](#) between two [specified](#) sets of atoms.

The surface is generated as a [surface model](#) and assigned the same model number and transformation as the corresponding molecule model (or, if atoms from more than one model were used, the lowest-numbered of those models). Models can be hidden or closed using the [Model Panel](#).

The surface is colored to show the separation across the interface, by default red for smaller distances and blue for larger. Currently this distance-to-color mapping can only be adjusted in the [Intersurf](#) graphical interface. However, alternative color mappings not based on the separation across the interface can be applied with [scolor](#), or the surface can be colored to match nearby atoms using [Color Zone](#). After an interface surface has been generated, the [Intersurf](#) graphical interface will contain a histogram populated with the corresponding atom-atom distances. A new [color mapping](#) can be defined in the histogram, then applied by reissuing the same **intersurf** command.

Examples:

```
intersurf #0:.a #1:.b pair chain prune 15 bias .2
```

- generate an interface surface between chain A in model 0 and chain B in model 1, biasing the surface toward the former; disregard residues in each chain whose centroids are not within 15.0 Å of any residue centroid in the other chain

```
intersurf main ligand pair atom solid false
```

- generate a mesh interface surface between the sets of atoms [classified](#) as main and ligand

## Options

The option keywords apply only to interface surface generation, not coloring. Option keywords for **intersurf** can be truncated to unique strings and their case does not matter. A vertical bar "|" designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

```
pairing model|chain|atom
```

- **model** - calculate an interface surface between two models; default when [atom-spec1](#) and [atom-spec2](#) specify atoms in two different models
- **chain** - calculate an interface surface between two chains; default when [atom-spec1](#) and [atom-spec2](#) specify atoms in the same model (of course, they should specify two different chains)

- **atom** - calculate an interface surface between two sets of atoms that are not whole chains or models (cannot be done with the [Intersurf](#) graphical interface); [atom-spec1](#) and [atom-spec2](#) should specify mutually exclusive sets of atoms

It is necessary to use **pairing chain** to specify an interface between chains in two different models and **pairing atom** to specify an interface between sets of atoms (other than whole chains or whole models) in two different models.

**solid** true|false

Whether to generate a solid surface rather than the default mesh.

**track** true|false

Whether to overwrite any existing interface surface (otherwise, an additional [surface model](#) will be generated).

**select** true|false

Whether to [select](#) the atoms that [define](#) the interface.

**prune** *distance*

Ignore residues in one set of atoms whose centroids are not within *distance* (default 30.0 Å) of any residue centroid in the other set of atoms. If this option is not specified, or if *distance* is set to 0.0 Å, pruning will not be performed.

**bias** *fraction*

How to divide the space between the two sets of atoms. Each vertex in the interface surface lies between two atoms, one from each set. The *fraction* can range from 0 to 1, where 0.5 (default) places each interface surface point equidistant from the [VDW surfaces](#) of the two corresponding atoms. A value of 0.0 places interface surface vertices at the VDW surface of the set specified with [atom-spec1](#) and a value of 1.0 places them at the VDW surface of the set specified with [atom-spec2](#). Where the VDW surfaces of the two sets coincide, changes in the value will have little effect on interface surface vertex placement.

See also: [scolor](#), [measure buriedarea](#), [Color Zone](#), [Surface Zone](#), [Measure Volume](#)

**Usage:****invert** [atom-spec](#)

The **invert** command exchanges the positions of two substituents of an atom, potentially inverting a chiral center. Substituents of atoms that are not chiral centers can also be swapped. The [Invert](#) section of [Build Structure](#) is the corresponding graphical interface. See also: [chirality](#)

The [atom-spec](#) must contain exactly one or two atoms:

- If a single atom is specified, clicking **Swap** exchanges the positions of its two smallest substituents based on number of atoms, or if those are the same, atomic weights of the atoms directly bonded to the selected atom. Implicit hydrogens on the selected atom are considered, but not those on its substituents. These rules are not meant to reproduce the much more complex “priority” calculations used in chirality determination.
- If two atoms directly bonded to the same central atom are specified, clicking **Swap** will exchange the positions of the substituents rooted at those atoms.

Any unintended results can be reversed by reissuing the same command.

**Usage:**

**ksdssp** [ *-c cutoff* ] [ *-h helix\_min* ] [ *-s strand\_min* ] [ *-S file* ] [ *-v* ] [atom-spec](#)

**Ksdssp** is an implementation of the Kabsch and Sander algorithm for defining the secondary structure of proteins, as described in:

W. Kabsch and C. Sander, "Dictionary of Protein Secondary Structure: Pattern Recognition of Hydrogen-Bonded and Geometrical Features" *Biopolymers* **22**:2577 (1983).

Many, but not all, input structure files include protein secondary structure information. In [PDB format](#), secondary structure assignments are described in [HELIX and SHEET records](#). When a peptide or protein structure file that does not include secondary structure assignments is read, **ksdssp** is *automatically* invoked to generate helix and sheet information. **Ksdssp** is also called automatically when [MatchMaker](#) is used with the **Compute secondary structure assignments** option, regardless of whether secondary structure assignments already exist.

In Chimera, helix and sheet assignments are stored as true/false settings (for each amino acid residue) of the [residue attributes](#) **isHelix** and **isStrand** (**isSheet**). The assignments are used for drawing protein [ribbons](#) and may be used by [MatchMaker](#), depending on its settings.

Parameter settings used are reported in the [Reply Log](#). Defaults to be used when **ksdssp** is called automatically can be changed with the [compute SS](#) dialog (opened from the [Model Panel](#)). Defaults for the **ksdssp command**, however, cannot be changed and are given [below](#). Different values can be specified with the command-line [options](#).

**Ksdssp** uses the coordinates of the backbone atoms (N, CA, C, O, and optionally H) of a protein to determine which residues are in helices and beta strands. If an amide hydrogen is missing, it is placed 1.01 Å from N along the bisector of (1) the vector opposite the bisector of C-N-CA, and (2) the vector opposite the C-O vector from the previous amino acid.

The calculation is applied to the models containing [atom-spec](#), and a blank [atom-spec](#) is interpreted as "all." Models that do not appear to contain amino acids (that is, nonpeptide molecules and nonmolecular objects) are ignored.

**Known problem:** reassigning secondary structure sets the [ribbon scaling](#) of some residues to **Chimera default**; any other scalings previously in effect need to be reapplied (for example, with [ribscale](#)).

**Options**

*-c cutoff*

**Ksdssp** estimates the energy of each candidate hydrogen-bonding interaction and classifies it as a hydrogen bond if the energy is no greater than (at least as favorable as) *cutoff*. The default is **-0.5 kcal/mol**, as recommended by Kabsch and Sander, who add that "A good H-bond has about -3 kcal/mol binding energy."

**-h** *helix\_min*

Helices must be at least *helix\_min* residues long; the default is **3**.

**-s** *strand\_min*

Beta strands must be at least *strand\_min* residues long; the default is **3**. Reducing *strand\_min* to 1 is not recommended, as there are bridges in many structures that confuse the algorithm for defining sheets.

**-v**

Verbose mode; send information to the [Reply Log](#), including helix and strand residue ranges and whether interstrand relationships are parallel or antiparallel.

In most cases, the default parameter values are reasonable for computing secondary structure.

See also: [ribbon](#), [ribscale](#), [ribinsidcolor](#), [PipesAndPlanks](#), [Ribbon Style Editor](#), [MatchMaker](#)

**Usage:**

label [ offset default | *x,y,z* ] [atom-spec](#)

**Usage:**

~label [atom-spec](#)

**Label** displays labels for all atoms in [atom-spec](#). **~Label** removes atom labels. The default label information is atom name with any alternate location ID appended, but custom atom labels can be defined using [labelopt](#). Label font, size, and whether to draw labels in front regardless of their Z-offsets (default true) can be set in the [Labels preferences](#). The command [rlabel](#) shows residue labels. See also: [color](#), [2dlabels](#), the [Actions menu](#)

By default, the offset of a label is automatically adjusted based on the [display style](#) of the corresponding atom. The **offset** keyword allows specifying an absolute offset that will not be adjusted, or a return to the **default** behavior. The word **default** can be truncated. An absolute offset is specified by three comma-separated values *x,y,z* representing distances along the X-, Y-, and Z-axes in the laboratory coordinate system:

- the X-axis is horizontal in the plane of the screen, increasing to the right
- the Y-axis is vertical in the plane of the screen, increasing upward
- the Z-axis is perpendicular to the screen, increasing outward (toward the viewer)

Negative distances can be used. Label offsets can also be adjusted [with the mouse](#). The default offset is  $(display\_radius + 0.2),0,0.5$ .

One way to label a bond or [pseudobond](#) is to [select](#) it and use the [Selection Inspector](#).

**Usage:****labelopt ( info | resinfo ) *label-contents***

**Labelopt** defines custom labels containing [attribute](#) values, alone or in combinations with each other and arbitrary text. For labeling with arbitrary text alone (without [attribute](#) values), [setattr](#) should be used instead. Labels already shown are not changed; if descriptor values change, the labels must be displayed again to reflect the changes. See also: [label](#), [rlabel](#), [2dlabels](#), [custom residue labeling](#)

The **info** keyword is used for atom labels, **resinfo** for residue labels. The *label-contents* can be:

- any single *descriptor* described below, primarily the name of an [attribute](#) at the appropriate level (atoms or residues), plus a few special cases
- any combination of arbitrary text and substitution codes of the form *%(descriptor)format*

The latter, more complicated form would be used to combine multiple descriptors with each other and with text, and/or to control specific aspects of the [format](#) such as the number of decimal places.

Examples:

```
labelopt info B = %(bfactor).2f
```

```
label solvent
```

- show labels something like "B = 43.70" containing B-factor values

```
labelopt resinfo %(name)s %(number)s
```

```
rlabel :150-160
```

- show residue labels like "TYR 151"

```
labelopt info %(charge)+.3f
```

```
label :his
```

- show atomic partial charges as labels something like "-0.278" and "+0.332"

For atoms, a *descriptor* can be the name of any atom [attribute](#), whether [built-in](#) or [created](#) during use of Chimera:

- **name** (default) - atom name with any alternate location ID appended
- **residue** - model number (if multiple models are open), residue name, and residue identifier (number.chain), unless in the [Labels preferences](#), **Atomspec display style** is set to **command-line specifier**
- **element**
- **idatmType** - [atom type](#)
- **charge** - partial charge such as assigned by [addcharge](#) (*etc.*)

For residues, a *descriptor* can be the name of any residue [attribute](#), whether [built-in](#) or [created](#) during use of Chimera, as well as the following:

- **name** - residue name
- **1-letter code** - one-letter residue code for standard amino acids, residue name for other residues
- **specifier** - residue number, insertion code, and chain ID
- **number** - residue number
- **insertion code** - residue insertion code
- **chain** - chain ID

For either atom or residue labels, a *format* can be several things, described as components #3-7 of a conversion specifier in the [Python documentation](#), but useful possibilities include:

- **s** - string
- **.Nf** - floating-point number with *N* digits after the decimal point, for example **.2f** to give numbers like 122.57 and -1.66
- **+.Nf** - as above but always signed, for example **+.2f** to give numbers like +122.57 and -1.66

## Usage:

**leap ( start | stop | position | chopsticks | velocity )**

The **leap** command controls the mode of interaction between Chimera and a [Leap Motion Controller](#), which uses two cameras to track finger and hand movements.

See also: [focus](#), [set independent](#), [3D manipulation](#), [Set Pivot](#) and [Focus](#) in the [Actions menu](#)

The **leap** keywords can be truncated to unique strings:

- **start** - initiate using the device with Chimera (in [position](#) mode)
- **stop** - stop using the device with Chimera
- **position** - use position mode, in which the Chimera contents move as the user's hand moves (rotations are multiplied by a factor of 3); see [video](#)
- **chopsticks** - use chopstick mode, in which the Chimera contents move as if grasped between the user's extended index fingers (with rotations multiplied by a factor of 3) and zoom according to the separation between the fingers; thumbs and other fingers should be curled in
- **velocity** - use velocity mode, in which the Chimera contents move based on the user's hand position and orientation relative to a neutral middle position and orientation

## Limitations

**Noise.** Diverse types of noise and jitter interfere with motion control.

**Lack of sensitivity settings.** Currently, the sensitivity of the device (the relative scale of finger/hand motion and model motion in Chimera) cannot be controlled via the **leap** command.

**Velocity mode not very intuitive.** The velocity mode may be difficult to use effectively.

**Usage:****lighting options**

The **lighting** command provides access to settings in [Lighting](#), including [Shininess](#). It also allows changing the colors and specular contributions of the lights. See also: [set](#), [preset](#), [lighting details](#)

Options include:

- [mode](#) - number/type of lights
- [brightness](#) - total illumination
- [contrast](#) - darkness of shading
- [ratio](#) - ratio of primary to secondary lighting
- [key \(fill, back\)](#) - color, direction, specular intensity of the specified light
- [save \(restore, delete\)](#) - save, restore, delete a lighting style
- [sharpness](#) - spread of highlights on the Chimera default material
- [reflectivity](#) - specular intensity of the Chimera default material

Full descriptions are given below. Keywords and their sub-keywords can be truncated to unique strings. A vertical bar "|" designates mutually exclusive options, and default settings are indicated with **bold**. Square brackets [] indicate an argument that can be omitted; if a value is omitted, the current value will be reported in the [status line](#) (transiently) and [Reply Log](#).

- **lighting mode** [ ambient | single | **two-point** | three-point ]

Set number and type of lights according to the lighting mode:

- **ambient** - ambient light only (gives a flat appearance; best combined with [silhouettes](#) and [white background](#))
- **single** - ambient light and one directional light (the **key** light)
- **two-point** (default) - ambient light and two directional lights, **key** and **fill**
- **three-point** - ambient light and three directional lights: **key**, **fill**, and **back**

The different lights and their roles are discussed in the [lighting details](#).

- **lighting brightness** [ *value* ]

Set brightness, the total illumination along the line of sight, disregarding any specular contributions (default **1.16**).

- **lighting contrast** [ *value* ]

Set contrast, or darkness of shading (default **0.83**, range 0.0-1.0); at a given [brightness](#) B, decreasing the contrast C increases the ambient light:  $A = B(1 - C)$ . Contrast does not apply

to the ambient-only [lighting mode](#).

- **lighting ratio** [ *value* ]

Set the ratio of (directional key + ambient) to (directional fill + ambient), disregarding any specular contributions (default **1.25**). The *value* can range from 1.0 to an upper bound that depends on the [contrast](#). Applies only to the two-point and three-point [lighting modes](#).

- **lighting ( key | fill | back ) color** [ *color* ]

Specify key, fill, or back light color (all are **white** by default). The *color* can be any [color name](#) that specifies a single color.

- **lighting ( key | fill | back ) direction** [ *x y z* ]

Specify the direction of the key, fill, or back light as *x y z* components.

- **lighting ( key | fill | back ) specular\_intensity** [ *value* ]

Set the specular intensity of the key, fill, or back light to *value* (default **1.0** for the key light, **0.0** for the others).

- **lighting ( save | restore | delete ) style-name**

The current settings in [Lighting](#), including [Shininess](#), collectively define a style. The **save** keyword indicates saving the current settings as *style-name*. A previously saved style can be applied with **restore** or deleted with **delete**. A lighting style can also be saved, applied, or deleted using the [Lighting](#) interface. Named lighting styles are saved in the Chimera [preferences file](#).

- **lighting sharpness** [ *value* ]

Set the specular sharpness of the Chimera default material (default **30.0**). Lower values yield larger, more diffuse highlights, while higher values yield more pointlike highlights. For more information, see [Shininess](#).

- **lighting reflectivity** [ *value* ]

Set reflectivity, the specular brightness of the Chimera default material (default **1.0**). The components of the specular color are scaled by this value to calculate shiny highlights. For more information, see [Shininess](#).

**Usage:****linewidth** *width* [atom-spec](#)

**Linewidth** changes the thickness of lines in the wire [representation](#). The *width* is in pixels (default **1.0**). Although any positive number can be given, there will be a maximum displayable width that depends on the graphics hardware being used.

If the [atom-spec](#) is blank, all molecule models will be affected; otherwise, the entire model(s) containing [atom-spec](#) will be affected. If an affected model is in a [representation](#) other than wire, the change in linewidth will not be evident until part or all of the model is shown as a wire. Linewidth can also be changed in the [molecule model attributes panel](#).

For large structures, thick lines may be preferable to the stick [representation](#), as lines are simpler to render.

See also: [represent](#), [wire width](#) in the [Actions menu](#), [setattr](#)

[Chimera Commands Index](#)

Separate programs can control Chimera remotely via tools such as [ReadStdin](#) and [REServer](#). The following commands are useful for sending information to such programs:

- [listen](#)
  - [list](#) ( [models](#) | [chains](#) | [physicalchains](#) | [residues](#) | [atoms](#) | [selection](#) | [resattr](#) | [distmat](#) *atom-spec* )
  - [sequence](#) *atom-spec*
- [listen](#) ( [start](#) | [stop](#) ) *what* [ [prefix](#) *prefix* ] [ [rest](#) *URL* ]

The **listen** command tells Chimera to **start** or **stop** generating notifications when certain events take place; *what* may be **models** or **selection**. Example: **listen start models**

If *what* is **models**, then the events of interest are the addition or removal of a model. Each notification message is prefixed with *prefix* (default **ModelChanged**). Typical messages are:

```
ModelChanged: add model #0
ModelChanged: remove model #0
```

If *what* is **selection**, then the event of interest is a change in selected items. Each message is prefixed with *prefix* (default **SelectionChanged**). A typical message is:

```
SelectionChanged: selection changed
```

Normally, notification messages are sent to *stdout*. If *URL* is specified, then the messages are sent to the given URL with the message encoded as parameter **chimeraNotification**.

- [list models](#) [ [spec](#) *atom-spec* ] [ [type](#) *model\_type* ] [ [attribute](#) *attribute\_name* ]

Report the value of an [attribute](#) for models. Information is sent to the [Reply Log](#). By default, the **name** attribute is reported for all models. If **spec** is given, only models containing the specified atoms are reported. If **type** is given, only models with the matching type are reported. Common types include **molecule**, **msmsmodel** **surfacemodel** and **vrmlmodel**. If **attribute** is given, the value for *attribute\_name* is reported instead of the model name.

- [list chains](#) [ [spec](#) *atom-spec* ] [ [attribute](#) *attribute\_name* ]

Report the value of an [attribute](#) of chains. Information is sent to the [Reply Log](#). By default, the **chain** attribute is reported for all chains in all molecule models. If **spec** is given, only chains in models containing the specified atoms are reported. If **attribute** is given, the value for *attribute\_name* is reported instead of **chain**.

- **list physicalchains** [ spec [atom-spec](#) ]

Report the starting and ending sequence numbers for physically connected chains (no missing residues). Information is sent to the [Reply Log](#). By default, all chains in all molecule models are reported. If **spec** is given, only chains in models containing the specified atoms are reported.

- **list residues** [ spec [atom-spec](#) ] [ attribute *attribute\_name* ]

Report the value of an [attribute](#) of residues. Information is sent to the [Reply Log](#). By default, the **type** attribute is reported for all residues in all molecule models. If **spec** is given, only residues containing the specified atoms are reported. If **attribute** is given, the value for *attribute\_name* is reported instead of **type**.

- **list atoms** [ spec [atom-spec](#) ] [ attribute *attribute\_name* ]

Report the value of an [attribute](#) of atoms. Information is sent to the [Reply Log](#). By default, the **idatmType** attribute is reported for all atoms in all molecule models. If **spec** is given, only the specified atoms are reported. If **attribute** is given, the value for *attribute\_name* is reported instead of **idatmType**.

- **list selection** [ level *granularity* ] [ mode *mode* ] [ attribute *attribute\_name* ]

Report the value of an [attribute](#) for items that are currently [selected](#). Information is sent to the [Reply Log](#). The *granularity* determines which items are reported. Supported values are **atom** (default), **residue**, **chain** and **molecule**. The *mode* may be either **any** (default) or **all**. If **any**, items with any selected subitems are reported; if **all**, only items whose subitems are all selected are reported. For example, with **any** a residue of five atoms will be reported if any of the five atoms are selected; with **all**, all five atoms must be selected for the residue to be reported. If **attribute** is given, the value for *attribute\_name* is reported instead of the default, which is **idatmType** for atoms, **type** for residues, **chain** for chains, and **name** for molecules.

- **list resattr**

List the current residue [attributes](#), as would appear in [Render/Select by Attribute](#) (some attributes less likely to be relevant to users are suppressed). Information is sent to the [Reply Log](#).

- **list distmat** [atom-spec](#)

Report all pairwise distances among the specified atoms. The [atom-spec](#) cannot be blank. Information sent to the [Reply Log](#) consists of lines that start with "distmat," followed by the specifiers of the two atoms, followed by the distance in Å. The upper triangle of the distance matrix is given row by row, with atoms in the order (if any) in which they were specified.

- **sequence** [atom-spec](#)

Display the [Sequence](#) dialog for all chains containing the specified atoms. The [atom-spec](#) cannot be blank.

longbond

[Chimera Commands Index](#)

**Usage:**

**longbond**

**Usage:**

**~longbond**

Stretches of residues lacking coordinates are indicated with [pseudobonds](#) named **missing segments**. The command **~longbond** hides these pseudobonds, whereas **longbond** can be used to show them again.

The appearance of the pseudobonds can be adjusted in the [pseudobond attributes panel](#) (opened from the [PseudoBond Panel](#)).

See also: [bond](#), [bonddisplay](#), [delete](#)

## Usage:

**mask** *volume-model surf-models* [options](#)

The **mask** command extracts a [volume data](#) subregion bounded by surfaces ([details...](#)). All kinds of [surface models](#) can be used, but not surfaces in [VRML models](#). A new data set with values of zero in the masked-out areas will be generated, and can be saved to a file using [Volume Viewer](#) or the command [volume](#). See also: [vop](#), [shape](#), [segment](#), [Segment Map](#), [Fit to Segments](#), [Icosahedron Surface](#), [Volume Eraser](#), [Volume Tracer](#)

*Volume-model* can be the model number (preceded by #) of the input volume data set, or the word **ones** to indicate a map with all values set to 1. The [spacing](#) and [border](#) options only apply when the volume data is specified as **ones**.

*Surf-models* specifies the bounding surface(s) and can be one or more comma-separated model numbers (preceded by #) or the word **sel** to indicate the currently [selected](#) surfaces or [surface pieces](#).

## Options

Option keywords for **mask** can be truncated to unique strings, and their case does not matter. Synonyms for true: True, 1. Synonyms for false: False, 0. A vertical bar "|" designates mutually exclusive options, and default settings are indicated with **bold**.

**axis** *x,y,z*

Projection axis vector (default is along the data Y axis: **0,1,0**). This is relevant when the surfaces have holes. The region between surface layers is computed along the specified axis (see [algorithm](#)). Vector coordinates *x,y,z* are relative to the untransformed data. The vector can point in any direction (need not be parallel to the X, Y, or Z axis) and need not be of unit length.

**fullmap** true | false

Make the masked volume data set have the same dimensions as the original volume data set. The full size of the original volume data set is used even if only a subregion is being displayed. Otherwise, the bounds will be set as small as possible to enclose the surfaces, and the masked volume may be smaller. If the input volume is specified as **ones** (which has no defined dimensions), the [border](#) option applies instead.

**pad** *distance*

Before computing the masked volume, move the surface by the specified *distance* along the surface normals. The units of length are the same as for the data (generally Å or nm), and positive or negative values can be used. This allows extracting a larger or smaller region than is enclosed by the original surface. Known problem: if the resulting surface intersects itself, the masked volume will not include the intersection. For larger-region extraction, the problem can be avoided by using [extend](#) instead.

**extend** *N*

Before computing the masked volume, move the surface outward by  $N$  voxels, where  $N$  is a positive integer. In other words, include grid points that are within  $N$  grid units (along the grid X, Y, and Z axes) of the original surface. Unlike [pad](#), this option correctly handles self-intersections of the expanded surface.

**slab** *width* | *d1,d2*

Extract a slab of data around a surface layer. Two additional surfaces, displaced as specified from the existing surface and joined at their edges (if any), are computed but not displayed. Data for voxels between the computed surfaces are retained. If a single value (*width*) is supplied, the two computed surfaces are offset along the normals of the original surface by  $\pm\frac{1}{2}(\textit{width})$ . Alternatively, two values separated by a comma but no spaces can be used to specify the offsets of the two surfaces independently. Positive or negative values can be used, and the units of length are the same as for the data (generally Å or nm). The basic [algorithm](#) applies, except that the original surface is replaced by the pair of computed surfaces.

**sandwich** true | false

Include only volume voxels lying between two surface layers. Otherwise, the volume projected along the axis beyond a single surface layer will also be included (see [algorithm](#)).

**fillOverlap** true | false

When multiple surface pieces are specified, retain the union of the values from masking to each piece separately. For example, if two surface pieces are concentric spheres, **fillOverlap** true will return values for all grid points within the larger sphere, whereas **fillOverlap** false will return values for only those points in the shell between the two surfaces.

**invertMask** true | false

Include the opposite data, the spatial complement of what would otherwise be included.

**spacing** *S* | *Sx,Sy,Sz*

Grid spacing for the output map when the input volume is specified as **ones** (default 1% of the maximum of the X, Y, and Z dimensions of the bounding surfaces); otherwise, the spacing will be the same as for the input map. The spacing is specified in units of length, typically Å. If a single number is supplied, it is used in all three directions; if three numbers are supplied (separated by commas but not spaces), they are used in the X, Y, and Z directions, respectively.

**border** *B*

How far out from the bounding surfaces in all six directions ( $\pm X$ ,  $\pm Y$ ,  $\pm Z$ ) to place the edge of the output map when the input volume is specified as **ones**; otherwise, the bounds will be set as described for the [fullmap](#) option. The border distance  $B$  is specified in units of length, typically Å (default 0.0).

**modelId** *N*

Open the new volume as model number  $N$  (an integer, optionally preceded by #). Submodel specifications # $N.N$  (# required) can also be given. The default is the lowest unused number.

## Mask Algorithm

The masked volume is computed by finding where lines parallel to the [projection axis](#) intersect the surfaces. For a given line, the volume data between the 1st and 2nd, 3rd and 4th, 5th and 6th, ... points of intersection are included in the masked volume, while those between the 2nd and 3rd, 4th and 5th, ... are excluded (unless multiple surface pieces are used and [fillOverlap](#) is set to true). The calculation uses a set of parallel lines that pass through points on a rectangular grid perpendicular to the projection axis. If the projection is along a data axis (data X, Y, or Z), the lines will pass through the grid points of the data; otherwise, lines along the projection axis with spacing equal to the minimum grid plane spacing of the data will be used. For each volume voxel, the intersections of the closest grid line are used to determine inclusion in the masked volume. If there is an odd number of intersections, then points beyond the final one are included in the masked volume unless the [sandwich](#) option is used. In the new data set, values outside the masked region are set to zero and those inside are set to the original volume values. The grid points of the calculated volume align exactly with those of the original volume. By default, the bounds are set to minimally enclose the surfaces (but see the [fullmap](#) and [border](#) options). The full size of the original volume data set is used even if only a subregion is being displayed.

**Usage:****match** [atom-spec1](#) [atom-spec2](#) [options](#)

The **match** command performs least-squares fitting of specified atoms, moving the first set of atoms ([by default](#), the entire models containing them) onto the second. It can also report least-squares-fit root-mean-square deviations (RMSDs) without actually performing the fits, *i.e.*, without moving the atoms. The related command [rmsd](#) calculates RMSDs without any fitting, using the current positions of atoms. See also: [matchmaker](#), [measure rotation](#), [matrixcopy](#), [Ensemble Match](#), [superimposing structures](#)

[Atom-spec1](#) and [atom-spec2](#) must specify equal numbers of atoms. [Options](#) may precede and/or follow the atom specifications.

Atoms are paired in the order specified, *i.e.*, the first atom in the first specification is matched to the first atom in the second specification, the second atom to the second atom, and so on. At least three pairs of atoms are needed to define a unique transformation, although fewer pairs can be used. If atom order is not specified, for example,

```
match #1:fad #0:fad
match #2:246,295 #0:195,221
```

the atoms within a residue are ordered first by name, and where these are identical, by alternate location identifier, and where these are also identical, by serial number.

The two sets of atoms can be specified collectively as a [selection](#) using the word **selected**, **sel**, or **picked**. An ordered selection can be created by [picking](#) atoms one by one, first the atoms of the first set, then those of the second in the corresponding order.

The fit can be [iterated](#). The number of atom pairs in the final fit and their RMSD are reported in the [status line](#) and [Reply Log](#). The [transformation matrix](#) can also be reported.

**Options**

A vertical bar “|” designates mutually exclusive options, of which the default values are indicated with **bold**.

**iterate** *cutoff*

Perform successive rounds of matching to the indicated stringency (default is no iteration). In each cycle of iteration, atom pairs are removed from the match list and the remaining pairs are fitted, until no matched pair is more than *cutoff* Å apart. The atom pairs removed are either the 10% farthest apart of all pairs or the 50% farthest apart of all pairs exceeding the cutoff, whichever is the lesser number of pairs. This excludes conformationally dissimilar regions from the final fit and allows the best-matching regions to be well superimposed.

```
move true | false | molecules | chains | residues | atoms | atom-spec3
```

Whether to move atoms as a result of the fitting calculation, and if so, whether to move the entire molecule models containing the atoms in [atom-spec1](#) (default **true**, equivalent to **molecules**), only the **chains** containing them, only the **residues** containing them, only the **atoms** themselves, or a different set of atoms ([atom-spec3](#)). If **false**, no atoms will be moved, but the reported RMSD and [matrix](#) will still reflect the movement that would have occurred.

\*\* Movements of **chains**, **residues**, **atoms**, or [atom-spec3](#) cannot be undone except by reopening the original structure. \*\*

**showMatrix** true | false

- or -

**show** true | false

Whether to report the transformation matrix in the [Reply Log](#).

### **active**

In addition to what would otherwise [move](#), apply the same transformation to any other models that are [activated for motion](#) (except the reference model(s), *i.e.*, those containing the second set of atoms).

**Usage:**

**( mmaker | matchmaker ) *refstruct matchstruct options***

**Mmaker** (or **matchmaker**) is the command-line implementation of [MatchMaker](#), which superimposes structures by first creating a pairwise sequence alignment, then fitting the aligned residue pairs. Residue types and/or secondary structure information can be used to align the sequences. Fitting uses one point per residue: CA in amino acid residues and C4' in nucleic acid residues. If a nucleic acid residue lacks a C4' atom (some lower-resolution structures are P traces), its P atom will be paired with the P atom of the aligned residue.

Note: if it is already known which residue numbers in one structure should be paired with which residue numbers in the other, another possibility is to use the [match](#) command, which executes more quickly because it does not include a sequence alignment step. See [superimposing structures](#) for a discussion of the different methods available in Chimera.

A reference structure (*refstruct*) and a structure to match (*matchstruct*) must be [specified](#). *Matchstruct* can include multiple models to be matched to *refstruct* independently, but cannot include parts of the same model as *refstruct*. The [pairing mode](#) determines whether chains or models should be specified. If a [specification](#) includes any spaces, it must be enclosed in single or double quote marks. On occasion, it may be useful to [restrict](#) the calculation to certain residues.

Sequence alignment scores, parameter values, and final match RMSDs are reported in the [Reply Log](#).

Examples:

```
mm #0 #1 show true
```

- superimpose model 1 onto model 0 using default settings; [show](#) the sequence alignment. Default settings are to recalculate secondary structure assignments with [ksdssp](#), generate sequence alignments using the Needleman-Wunsch [algorithm](#) with the BLOSUM-62 [residue similarity matrix](#) (weight 0.7) and [secondary structure scoring](#) (weight 0.3), keep the sequence alignment for the [best-scoring pair](#) of chains (one from model 0 and one from model 1), and using that alignment, [iteratively fit](#) the structures with a cutoff of 2.0 Å.

```
mm #0 #1-5 computeSS false
```

- independently superimpose models 1-5 onto model 0 using default settings, except without [recalculating](#) secondary structure assignments.

```
mmaker #0:a #1 pair sb alg sw matrix PAM-150 ss false iter 5.0
```

(example structures: mouse and human phosphoserine phosphatases [1j97](#), [1nnl](#) open as models 0 and 1, respectively)

- align chain A in model 0 with the [highest-scoring chain](#) in model 1 using the Smith-Waterman [algorithm](#)

with the PAM-150 [residue similarity matrix](#) (weight 1.0, no [secondary structure scoring](#)); [iteratively fit](#) the structures using a cutoff of 5.0 Å.

**mm #0:a:b #1:c:d pair ss show true**

(example structures: insulin **1b17**, **1ben** open as models 0 and 1, respectively)

- align the [specific chain pairs](#) A/C and B/D (in models 0/1) using default settings; [show](#) both sequence alignments.

Including specific residues in the *refstruct* and/or *matchstruct* specifications restricts the calculation to only those residues. In general, restriction should only be used in specific cases to suppress results that would otherwise be obtained. For example, two chains that would otherwise align on their N-terminal domains can be forced to align on their C-terminal domains by specifying only the residues in the C-terminal domains. Otherwise, restriction is not recommended, because full-length alignments tend to be of higher quality, and [iteration](#) already serves to exclude poorly superimposed regions from the final fit. Although the unused parts of matched chains will appear in the resulting sequence alignment (if [shown](#)), they have simply been added back in as “filler,” without consideration of how the characters align, after alignment and matching of only the specified residues.

## Options

Option keywords for **mmaker** (**matchmaker**) can be truncated to unique strings and their case does not matter. A vertical bar “|” designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0. Many of the options affect [alignment scoring](#).

### pairing mode

The *mode* controls which chain sequences are used to construct the reference-match alignment:

- **bb** (default) - use the pair of chains, one from the reference model and one from the match model, with the highest alignment score; *refstruct* and *matchstruct* should each specify a model or part of a model
- **sb** - pair a specific chain in the reference model with whichever chain in the match model gives the highest alignment score; *refstruct* should specify a chain, *matchstruct* a model or part of a model
- **ss** - pair specific chain(s) in the reference model with specific chain(s) in the match model; *refstruct* and *matchstruct* should specify equal numbers of chains

### alg alignment-algorithm

The *alignment-algorithm* can be:

- **nw** (or **needle**; default) - Needleman-Wunsch, global
- **sw** (or **smith**) - Smith-Waterman, local

### showAlignment true | false

Whether to show the resulting sequence alignment(s) with [Multalign Viewer](#). When [fit iteration](#) is employed, the pairs used in the final fit will be shown in the alignment as a [region](#) named **matched residues**. The “RMSD: ca” [header](#) is automatically shown above the

sequences, with histogram bar heights representing the [single-point](#) spatial variation among residues associated with a column. In the pairwise case, the value per column is simply the distance between the two associated residues.

\*When the fit has been [restricted](#) to specified residues, the remaining residues of matched chains will still appear in the alignment, but merely as a convenient compact representation; how they are aligned is not meaningful.

\*\*These sequence alignments can be considered a by-product of superposition. Successful superposition only requires the sequence alignment to be partly correct, as incorrect portions tend to be excluded from the fit during [iteration](#). If the sequences are easy to align (highly similar), the sequence alignment is likely to be correct throughout. However, if the sequences are more distantly related, parts of the alignment may be incorrect even when a successful superposition is produced. When **matchmaker** is used simply to superimpose structures, this is not important. However, if one also wants a corresponding sequence alignment, generating a structure-based alignment (after superposition) with [Match -> Align](#) is recommended, especially if the sequences are dissimilar. The structure-based sequence alignment will provide better statistics for describing structural similarity (RMSD, *etc.*) because more columns will be aligned correctly.

**iterate** *cutoff* | false

Whether to iteratively fit the structures, and the cutoff for excluding residue pairs from the fit. An iterative fit will be performed unless this option is set to **false**. The sequence alignment is not changed, but residue pairs in the alignment can be removed from the "match list" used to superimpose the structures. In each cycle, pairs of [atoms](#) are removed from the match list and the remaining pairs are fitted, until no matched pair is more than *cutoff* Å apart (2.0 by default). The atom pairs removed are either the 10% farthest apart of all pairs or the 50% farthest apart of all pairs exceeding the cutoff, whichever is the lesser number of pairs. The result is that the best-matching "core" regions are maximally superimposed; conformationally dissimilar regions such as flexible loops are not included in the final fit, even though they may be aligned in the sequence alignment.

## Alignment Scoring Options

**matrix** *similarity-matrix*

The *similarity-matrix* can be any of those listed in the [MatchMaker](#) graphical interface (case is important): BLOSUM-30, BLOSUM-35, BLOSUM-40, BLOSUM-45, BLOSUM-50, BLOSUM-55, BLOSUM-60, **BLOSUM-62** (default), BLOSUM-65, BLOSUM-70, BLOSUM-75, BLOSUM-80, BLOSUM-85, BLOSUM-90, BLOSUM-100, BLOSUM-N, PAM-40, PAM-120, PAM-150, PAM-250, SDM, HSDM, Nucleic. Matrix files reside in the `share/SmithWaterman/matrices/` directory of a Chimera installation.

If an amino acid matrix (any except **Nucleic**) is chosen, only peptide chains will be aligned; if the **Nucleic** matrix is chosen, only nucleic acid chains will be aligned. An error message will appear if there are no reference-match pairs of the appropriate type.

**gapOpen** *opening-penalty*

When [secondary structure scoring](#) is *not* being used, the *opening-penalty* is subtracted from the score for each gap opened (12 by default). When [secondary structure scoring](#) is

used, secondary-structure-specific gap opening penalties (see [hgap](#), [sgap](#), and [ogap](#)) are used instead.

#### **gapExtend** *extension-penalty*

The *extension-penalty* is subtracted from the score for each increment in gap length (1 by default).

#### **ssFraction** *fraction* | false

Sequence alignment scores can include a residue identity/similarity term, a secondary structure term, and gap penalties. *Fraction* is the relative weight of the secondary structure term and can range from 0 to 1 (default **0.3**). Unless the option is set to **false**, a secondary structure term will be included with a weight of *ssfract* and the residue similarity term will be given a weight of  $(1 - ssfract)$ .

#### **computeSS** true | false

When [secondary structure scoring](#) is used, whether to first identify helices and strands with the [ksdssp](#) algorithm, overwriting any pre-existing secondary structure assignments (except for CA-only structures, which are automatically skipped). This option may improve superposition results by generating consistent assignments, whereas pre-existing assignments may reflect the use of different criteria on different structures. When [secondary structure scoring](#) is *not* being used, this option is ignored and secondary structure assignments are not computed.

#### **hgap** *intrahelix-penalty*

When [secondary structure scoring](#) is used, the *intrahelix-penalty* is subtracted from the score for each gap opened within a helix (**18** by default). When [secondary structure scoring](#) is *not* being used, a generic gap penalty (see [gapOpen](#)) is used instead.

#### **sgap** *intrastrand-penalty*

When [secondary structure scoring](#) is used, the *intrastrand-penalty* is subtracted from the score for each gap opened within a strand (**18** by default). When [secondary structure scoring](#) is *not* being used, a generic gap penalty (see [gapOpen](#)) is used instead.

#### **ogap** *other-penalty*

When [secondary structure scoring](#) is used, the *other-penalty* is subtracted from the score for each gap opened that is not within a helix or strand (**6** by default). When [secondary structure scoring](#) is *not* being used, a generic gap penalty (see [gapOpen](#)) is used instead.

#### **matHH** *helix-helix-score*

When [secondary structure scoring](#) is used, *helix-helix-score* is the value added to the secondary structure term for aligning a residue in a helix with a residue in a helix (default **6**).

#### **matSS** *strand-strand-score*

When [secondary structure scoring](#) is used, *strand-strand-score* is the value added to the secondary structure term for aligning a residue in a strand with a residue in a strand (default **6**).

**matOO** *other-other-score*

When [secondary structure scoring](#) is used, *other-other-score* is the value added to the secondary structure term for aligning a non-helix, non-strand residue with a non-helix, non-strand residue (default **4**).

**matHS** *helix-strand-score*

When [secondary structure scoring](#) is used, *helix-strand-score* is the value added to the secondary structure term for aligning a residue in a helix with a residue in a strand (default **-9**).

**matHO** *helix-other-score*

When [secondary structure scoring](#) is used, *helix-other-score* is the value added to the secondary structure term for aligning a residue in a helix with a non-helix, non-strand residue (default **-6**).

**matSO** *strand-other-score*

When [secondary structure scoring](#) is used, *strand-other-score* is the value added to the secondary structure term for aligning a residue in a strand with a non-helix, non-strand residue (default **-6**).

**verbose** true | false

For each chain-chain pair, send additional information to the [Reply Log](#):

- **Sequences:** followed by the pairwise sequence alignment, *i.e.*, two lines, each containing a sequence name and (gapped) sequence
- **Residues:** followed by two lines, each a comma-separated list of the structure residues associated with the nongap positions of the corresponding sequence; missing structure residues are reported as **None**
- **Residue usage in match (1=used, 0=unused):** followed by two lines, each a comma-separated list of zeroes and ones, indicating which structure residues were used in the final fit

See also: [match](#), [rmsd](#), [measure rotation](#), [matrixcopy](#), [ksdssp](#), [MatchMaker](#), [Match -> Align](#), [superimposing structures](#)

**Usage:**

**matrixcopy** *from\_model to\_model* [ **moving** *move\_models* ]

**Matrixcopy** applies the 4x4 transformation matrix of a model to another model. Models are indicated by integer model number, with or without a preceding # symbol.

If *from\_model* specifies a single model, *to\_model* can specify one or more models to receive the same transformation. If *from\_model* and *to\_model* both specify a single model, the **moving** keyword can be used to transform models other than (or in addition to) *to\_model*: the *move\_models* will move as if rigidly carried along with *to\_model* if the *from\_model* matrix were applied to it, even if *to\_model* is not actually moved. Normally (**moving** not used), only *to\_model* is moved.

If *from\_model* specifies multiple models, then *to\_model* must specify an equal number of models. Pairing is in the order of specification, *i.e.*, the transformation of of the first *from\_model* is applied to the first *to\_model*, second applied to the second, and so on.

Examples:

```
matrixcopy 0 1
matrixcopy 0 1-2
matrixcopy #1 #0,2
matrixcopy #0 #4 moving #1,2,3
```

The latter moves models #1-3 as if rigidly transformed along with #4 if the matrix of #0 were applied to #4.

See also: [match](#), [matchmaker](#), [matrixget](#), [matrixset](#), [measure rotation](#), [sym](#)

**Usage:****matrixget** [ *filename* ]**Usage:****matrixset** [ *filename* ]

**Matrixget** writes the current transformation matrices to a file named *filename*. **Matrixset** reads matrices from the file named *filename* (in the format written by **matrixget**) and applies transformations accordingly. If *filename* is omitted, a [dialog](#) for specifying the name and location will appear. For **matrixget** only, if a single dash is used as the *filename*, the matrix information will be sent to the [Reply Log](#).

Each matrix is of the form

```
Model 0.0
    0.688816 0.672651 -0.270321 -3.40168
    0.327689 0.0437142 0.943774 1.87208
    0.646647 -0.738668 -0.190309 11.6143
```

where the first line indicates *model.submodel* number (with submodel number of zero for models not subdivided into [submodels](#)). The next three lines begin with tabs and contain a 3x3 rotation matrix and (in the fourth column) a translation vector. Scale is not specified.

When multiple models are present, multiple matrices will be written to/read from the same file. An error message will appear if there is no model present for one or more of the matrices read with **matrixset**.

See also: [matrixcopy](#), [measure rotation](#), [reset](#)

**Usage:****mclip** [ *model-spec* [options](#) ]**Usage:****~mclip** [ *model-spec* ]

The **mclip** command controls [per-model clipping](#). See the [video mini-example](#). See also: [surface](#), [sop cap](#), [volume](#), [clip](#), [movie-related commands](#)

The *model-spec* can be a specific model number or range of model numbers (preceded by #), or simply # or the word **all** to indicate all models. **Mclip** without any arguments also clips all of the models. A more complicated *model-spec* can be used to clip a surface model but not the molecule model with which it shares a model number, for example:

```
mclip "#0 & ~ @"
```

The command **~mclip** disables per-model clipping for the specified models, or if none are specified, all models.

**Options**

Option keywords for **mclip** can be truncated to unique strings, and their case does not matter. Synonyms for true: True, 1. Synonyms for false: False, 0. A vertical bar "|" designates mutually exclusive options, and default settings are indicated with **bold**.

Initial defaults apply when per-model clipping is first turned on for a given model. Thereafter, a plane's last specified position and orientation are its defaults (even if the plane was subsequently turned off).

```
coords local | screen
```

Whether [axis](#) and [origin](#) coordinates should be interpreted in the **local** (input model coordinates) or **screen** (viewing) frame of reference. In the **screen** frame of reference, the **x** axis is horizontal in the plane of the screen, the **y** axis is vertical in the plane of the screen, and the **z** axis is normal to the screen. In the **local** frame of reference, the axes rotate along with the model.

```
axis x | y | z | x,y,z
```

Orient the clipping plane perpendicular to the specified axis or vector in the indicated [frame of reference](#). Numerical coordinates *x,y,z* (separated by commas but not spaces) can be used to define a vector of any length pointing in any direction.

```
origin center | x,y,z
```

Place the clipping plane at the specified origin plus any [offset](#) and [stagger](#) along the [axis](#).

The origin can be set to the **center** of the model's bounding box (initial default) or specified with numerical coordinates *x,y,z* in the indicated [frame of reference](#).

**offset** *distance*

Offset the clipping plane along the [axis](#) by *distance* in physical units of length (default **0.0**, can be positive or negative).

**stagger** *spacing*

When clipping multiple models, locate the clipping plane for the lowest-ID model as specified by [origin](#) (or [match](#)) and [offset](#), but additionally offset each successive model's plane by *spacing* in physical units of length (default **0.0**, can be positive or negative). Staggering or otherwise offsetting the clipping planes of multiple surfaces avoids display of an irregular mosaic of colors where two or more [surface caps](#) coincide.

**flip** *true|false*

Whether to flip the plane 180° to hide the opposite part of the model.

**match** *plane*

Position the clipping plane to match *plane* (overrides [axis](#) and [origin](#)):

- *other-model* (a model number preceded by #) - another model's [per-model clipping plane](#)
- **focal** - the [focal plane](#)
- **near** - the near [global clipping plane](#)
- **nearfar** - the near [global clipping plane](#); also turn on [slab](#) mode with [thickness](#) set to coincide with the far [global clipping plane](#)

**slab** *true|false*

Whether to show a slab of the model instead of all parts to one side of the clipping plane (initial default **false**). Slab width can be specified with [thickness](#).

**thickness** *slabwidth*

Set [slab](#) thickness to *slabwidth* in physical units of length (initial default **5.0**).

**Usage:**

**mcopy** *source target* [ **settings** [c][s][v][l][x][y][p] | a ] [ **matchResidues** true | false ]

The **mcopy** command copies display styles and other attributes from a *source* molecule model to one or more *target* molecule models. It does not create new models. To copy an existing molecule model and its settings to create a new model, use the command [combine](#).

The *source* [atom specification](#) may include part or all of a single molecule model, and the *target* [atom specification](#) may include part or all of one or more molecule models other than the source. Model attributes, the attributes of any paired residues, and the attributes of any paired atoms (and the bonds between them) will be copied from the source to each target.

Residue pairing with **matchResidues true** (default) requires identical residue numbers, residue names, and chain identifiers. With **matchResidues false**, residues are simply paired in the order they are given in the *source* and *target* specifications. Either way, pairing atoms within the residues requires identical atom names and alternate location identifiers. Any unmatched atoms and residues (those in *target* but not in *source*) will be ignored, but their existence noted in the [status line](#).

Characters after the **settings** keyword (default **csv**) control which attributes are copied:

- **c** - colors (model-level and atom-level, see [coloring hierarchy](#))
- **s** - atom/bond and ribbon [display styles](#)
- **v** - visibility (model-level, residue-level ribbon, and atom-level display, see [display hierarchy](#))
- **l** - atom and residue labels
- **x** - atomic coordinates, untransformed
- **y** - atomic coordinates, transformed relative to *target*
- **p** - placement (overall model transformation)
- **a** - all of the above, except **y**

The **settings** keyword can be truncated.

Examples:

```
mcopy #0 #1 settings a
mcopy #0 #1-10 set csvl
```

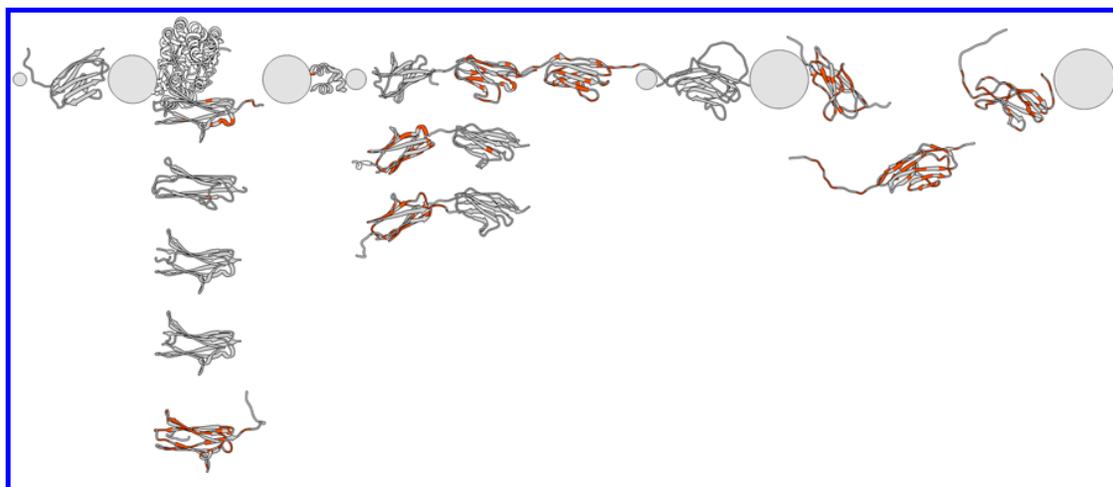
See also: [combine](#), [preset](#), [msc](#)

**Usage:****mda** *input-sequence output-folder* [options](#)

The command **mda** (for **M**ulti**D**omain **A**ssembler) performs several steps toward creating a model of a multidomain protein:

- Obtains the protein sequence specified by *input-sequence*, which can be either of the following:
  - a [UniProt](#) accession number (example: **P02751**); UniProt's [ID mapping service](#) can be used to obtain UniProt accession numbers from other sequence database identifiers
  - a pathname to a local file containing the sequence in [FASTA format](#) (example: `/Users/miller/Desktop/mybpc3.fa`); if the file contains multiple sequences, only the first will be used
- With that sequence as the query, searches the [Protein Data Bank \(PDB\)](#) sequences to find known structures of similar sequence, using a [BLAST](#) web service hosted by the [UCSF Resource for Biocomputing, Visualization, and Informatics \(RBVI\)](#).

- Imports the corresponding PDB structures into Chimera and lays them out from left to right according to sequence matches along the query from N- to C-terminus. Multiple hits to the same or overlapping segments are stacked vertically; gaps in structural coverage are shown as transparent dark gray spheres proportional in volume to the number of missing residues. See the [display options](#).



Performance depends on the number of structures imported into Chimera, which can be very large especially for runs with low filtering (permissive BLAST score and percent ID cutoffs, little or no winnowing) on queries with many known structures. Repeated uses of **mda** will reuse any open models that still meet the criteria and close any others. The command can be aborted during structure import by clicking the red stop icon in the [status line](#) to cancel the foreground task.

The [center of rotation method](#) is set to **independent** so that models rotate about their individual centers rather than a single collective center. The initial layout including scale and model orientations is saved as a [position](#) named **stacked**. Positions can be restored from the [Rapid Access](#) interface or with the command [reset](#) (e.g., `reset stacked`). Another position named **overlay**, better suited to the modeling step, is also saved.

- In the specified *output-folder*, writes output files:
  - the [pseudo-multiple sequence alignment](#) from BLAST in [aligned FASTA](#) format
  - a text file containing information from the run
- Opens the [pseudo-multiple sequence alignment](#) from BLAST in [Multalign Viewer](#); this can be suppressed (along with the [Modeller dialog](#)) using [showAlignment false](#).

6. Opens the interface to [Modeller comparative \(homology\) modeling](#) with the query sequence as the target, all of the imported hit structures designated as templates, and other options set as recommended for use with **mda**, including thorough optimization. The [overlay position](#) should be restored before comparative modeling to decrease the likelihood of generating [models with knots](#). Comparative modeling relies on an accurate sequence alignment. If necessary, the sequence alignment can be [edited manually](#) or [realigned](#) using [Multalign Viewer](#), or the [output](#) FASTA file can be edited outside of Chimera and reopened. Note that the BLAST alignment may omit parts of the hit chains ([more...](#)).

On first use, **mda** also generates three database files (MDA\_seqs.db, MDA\_blast.db and MDA\_uniprot.db) to speed up subsequent uses of the command, but these are not human-readable. These are placed in the user's [Chimera download directory](#), in an MDA subdirectory.

Multidomain assembly is under development; please send any comments and suggestions to [chimera-users@cgl.ucsf.edu](mailto:chimera-users@cgl.ucsf.edu).

See also: [Blast Protein, Fetch by ID](#), the [Chimera-Modeller interface](#)

Examples:

```
mda Q14896 ~/Desktop/MDA percentId 50 showAlignment false coloring blastscore skip 3CX2
mda Q14896 ~/Desktop/MDA limit 1,20; reset overlay
```

A smaller example (as of October 2014, imports five structures):

```
mda p45379 ~/Desktop/MDA percent 60 group false color blast
```

## Options

Option keywords can be truncated to unique strings and their case does not matter. A vertical bar “|” designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

- [Initial search and filtering](#)
- [Display](#)
- [Suppressing dialogs](#)
- [Further limiting the hits](#)

### ← Initial search and filtering

**winnow** *max-per-region*

Use the BLAST option to winnow the results to no more than *max-per-region* hits per region, as described in [Berman et al., J Comput Biol. 7:293 \(2000\)](#). Lower values correspond to more aggressive winnowing and fewer hits returned. All other filters (options below) are applied after winnowing. If the option is not specified or set to zero, no winnowing is done.

**minScore** *score*

Keep only BLAST hits with scores of at least *score* (default **50**). The BLOSUM62 scoring matrix is used. Lowering the cutoff may increase structural coverage of the query, but low-scoring hits may be poorly aligned with the query sequence.

**percentId** *percent*

Keep only BLAST hits with percent sequence identity (%ID) of at least *percent*. %ID is based only on the region of hit-query alignment from BLAST. If the option is not specified, no filtering by %ID is done.

**includeNative** **true** | false

Whether to keep all hits (PDB entries) that are annotated with the same UniProt ID as the query, regardless of **minScore** and **percentIdThreshold** settings.

**suppressDoubles** **true** | false

Whether to keep only the highest-scoring hit to a given PDB entry. An entire PDB structure will be imported if any part is kept as a hit, but redundant chains will be hidden and its position in the **mda** layout will be based on that match to the query. If **suppressDoubles** is **true** (default), a PDB entry will not be imported more than once. If **false**, it is possible for the same PDB entry to be imported more than once and laid out in different places corresponding to different matches along the query. Regardless of this setting, however, only the PDB chain with the most structure residues will be kept as a hit when BLAST finds multiple identical chains from the same PDB entry.

**forceBlast** **true** | false

Whether to re-run BLAST instead of using any [previously cached results](#). Even if this option is **false**, previously cached results will only be used when the query (the *uniprotID*) and [winnowing](#) level are the same as in an earlier use of **mda**. Whenever **mda** runs BLAST, the new results will be added to the cache, and if there were earlier results for the same query and winnowing level, they will be overwritten. Using cached results will speed **mda** execution at the possible cost of missing any newer PDB entries added to the sequence database since the time of caching.

← **Display****group** **true** | false

Whether to group sets of structures that cover similar parts of the query. A structure will be included in such a set if, in the [alignment according to BLAST](#), it overlaps any other member of the set by at least 25 residues, but extends less than 25 residues beyond the alignment region of the member with the longest such region. If **grouping** is **true** (default), the structures in each set will be superimposed, their models will be [grouped](#) in the [Model Panel](#), and only the representative (the member with the longest region of alignment) will be shown. If this option is **false**, the models in a set will still be laid out in the same “column,” but vertically separated from one another.

**hideSubmodels** **true** | false

Whether to hide submodels (except for the first) of multi-model PDB entries, namely NMR ensembles.

**hideAltChain** **true** | false

Whether to hide additional chains of a PDB entry that are identical to (same molecular entity as) the BLAST hit chain.

**hideComplex** **true** | **false**

Whether to hide nearby chains (nonidentical to the BLAST hit) and small molecules.

**deleteHidden** **true** | false

Whether to delete submodels or chains hidden by the three preceding options (improves performance).

**coloring** *scheme*

The *scheme* for coloring the structures can be one of the following:

- **mutations** (default) - residues identical to those in the query sequence **gray**, residues that differ from the query **red**, unaligned parts transparent **gray**, complexed molecules **blue** (see [named colors](#))
- **percentid** - aligned residues a single color (per model) ranging from **gray** for the highest % ID to **red** for the lowest %ID (note the highest and lowest values might not be very far apart), unaligned residues the same color but transparent, complexed molecules **blue**
- **blastscore** - aligned residues a single color (per model) ranging from **gray** for the highest BLAST score to **red** for the lowest BLAST score (note the highest and lowest values might not be very far apart), unaligned residues the same color but transparent, complexed molecules **blue**

In addition, three [attributes](#) are assigned to residues in the hit structures to allow custom coloring. The first two are quantitative, suitable for showing with [Render by Attribute](#) or the [rangecolor](#) command:

- **mda\_blastscore** - BLAST score, with the same value assigned to all residues in the same model as the BLAST hit
- **mda\_percentid** - %ID of BLAST hit, with the same value assigned to all residues in the same model as the hit

Although the third has numerical values, it is a classification rather than a continuously varying quantity:

- **mda\_alignment**, with values:
  - 1 - in the BLAST-aligned region, same amino acid as as the target
  - 2 - in the BLAST-aligned region, different amino acid than the target
  - 3 - in the hit chain but not in the BLAST-aligned region
  - 4 - in a co-complexed chain (a different chain than the hit)

Residues in each class above can be specified directly in commands, e.  
*g.:* `color yellow :/mda_alignment=1`

### ← Suppressing dialogs (helpful during scripted or repetitive use)

**noConfirm** true | false

Whether to simply import the hit structures (no matter how many) rather than asking the user for permission when more than ten are found.

**showAlignment** true | false

Whether to open [Multalign Viewer](#) and the [Modeller interface](#). Not opening these dialogs allows faster execution.

### ← Further limiting the hits (structures to be used as templates)

**limit** *min-per-residue* [, *max-gap-size*]

Limit the number of hits (with priority given to those with higher %ID) such that every residue in the target sequence is covered by the smallest number of hits without going below *min-per-residue* and without consecutive stretches of target residues lacking structural coverage greater than *max-gap-*

*size* (default **20** residues), if possible. (It may not be possible, depending on the initial set of hits and other filtering criteria.) Exclusions by this option can be overridden with [keepPDB](#).

**keepPDB** *pdb1[,pdb2,...]*

Retain hits with the specified PDB IDs even if they would have been removed by the [limit](#) option.

**skipPDB** *pdb1[,pdb2,...]*

Omit hits with the specified PDB IDs.

**excludeSelected** true | false

Whether to omit models with any part currently [selected](#) from the hits in the next **mda** run.

**Usage:****measure** *property arguments*

The **measure** command performs various calculations and sends results to the [Reply Log](#). Possible values of *property*:

- [area](#) - calculate total area of an existing surface
- [buriedArea](#) - calculate surface area buried between two sets of atoms
- [center](#) - calculate center of mass of map and/or atoms
- [contactArea](#) - report the area of one surface within a cutoff distance of another
- [correlation](#) - report correlation of two maps
- [distance](#) - report distances between atoms and surfaces
- [fieldLines](#) - display electric field lines for an [electrostatic potential map](#)
- [inertia](#) - calculate inertia ellipsoid
- [mapStats](#) - calculate map mean, standard deviation from the mean, and RMS deviation from zero
- [mapSum](#) - sum map values above a specified threshold
- [mapValues](#) - interpolate map values at atom positions, assign as an [attribute](#)
- [pathLength](#) - report the total length of specified bonds
- [rotation](#) - report transformation of one model relative to another
- [spine](#) - calculate a [path](#) along the center line of a [segmentation region](#)
- [symmetry](#) - identify map symmetry
- [volume](#) - calculate volume enclosed by an existing surface

Each property keyword has different arguments, described below. Property keywords and their sub-keywords can be truncated to unique strings, and their case does not matter. Models are specified by model number preceded by #. Synonyms for true: True, 1. Synonyms for false: False, 0. A vertical bar “|” designates mutually exclusive options, and default settings are indicated with **bold**.

See also: [distance](#), [angle](#), [vseries measure](#), [Structure Measurements](#)

- **measure area** *surface(s)*

Calculate the total area of each specified [surface piece](#) by summing over its triangles. The surface pieces in a [surface model](#) can be specified collectively by model number, or individually by [selection from the screen](#) and using the word **sel**, **selected**, or **picked**. Like [Measure Volume and Area](#), the calculation uses the full surface even if it is partly hidden by clipping or zoning.

Note that the solvent-excluded and solvent-accessible surface areas of a [molecular surface](#) are reported in the [Reply Log](#) when the surface is first shown, and the values per atom and residue are assigned as [attributes](#) named **areaSES** and **areaSAS**, respectively.

See also: [surface](#), [volume](#)

- **measure buriedArea** [atom-specA](#) [atom-specB](#) [ **probeRadius** *radius* ] [ **vertexDensity** *density* ]

Calculate the surface area buried between two [specified](#) sets of atoms. A surface is calculated for each set of atoms separately (surfA, surfB) and for their combination (surfAB). The surfaces are not created or displayed, but calculated internally. The difference in total area between the separate and combined states is reported for each set, as well as the average over the two sets. Buried areas are reported for both solvent-excluded and solvent-accessible surfaces. Only the averages are sent to the [status line](#), but full results can be viewed in the [Reply Log](#).

The atoms are grouped as specified regardless of [surface category](#), and a set may span multiple models. **Be careful** to specify only the intended atoms, which could mean excluding or [deleting](#) beforehand any solvent, ligands, ions, and/or alternate location atoms. [New Surfaces](#) preference settings are not used; disjoint surfaces are always included, and the default probe radius and vertex density are 1.4 Å and 2.0/Å<sup>2</sup>, respectively.

Atoms are assigned the [attributes](#) **buriedSESArea** (buried solvent-excluded surface area) and **buriedSASArea** (buried solvent-accessible surface area) with their individual contributions to the specified interface. These can be summed over [selected](#) atoms with [Attribute Calculator](#), for example to determine the contribution from carbons only.

To evaluate degree of residue burial in an overall protein structure, as opposed to a specific interface, it may be helpful to calculate [relative exposure](#) (a normalized surface area).

For surfaces without associated atomic coordinates, see [measure contactArea](#). See also: [surface](#), [intersurf](#), [Area/Volume from Web](#)

- **measure center** *spec* [ **level** *contour-level* ] [ **mark** true|false ] [ **radius** *marker-radius* ] [ **color** *marker-color* ] [ **modelId** *model-number* ] [ **name** *model-name* ]

Calculate the center of mass of each density map and/or set of atoms in *spec*. Map centers are reported in grid indices, atomic centers of mass in the atomic coordinate system. The **level** option indicates using only map regions above *contour-level*. If **mark** is **true**, a [marker](#) will be placed at each computed center, with radius *marker-radius* (default based on the contents of *spec*) and color *marker-color* (default **gray**). The *marker-color* can be any [color name](#) that specifies a single color. The marker model is opened as number *model-number* (default next unused number) with name *model-name* (default based on the contents of *spec*).

- **measure contactArea** *surf-model1* *surf-model2* *cutoff* [ **color** *patch-color* ] [ **offset** *d* ] [ **slab** *width* | *d1,d2* ] [ **show** true|false ] [ **smooth** true|false ] [ **optimize** true|false ]

Report the surface area of one [surface model](#) (*surf-model1*) that lies within a *cutoff* distance of another [surface model](#) (*surf-model2*). Unless **show false** or **offset 0** is specified, a new surface model is created to show the corresponding patch of *surf-model1*.

The *patch-color* can be any [color name](#) that specifies a single color (default **red**). The new surface can be **offset** from the original *surf-model1* by a distance *d* specified in physical units, typically Å (default **1.0**). An offset of zero indicates recoloring *surf-model1* to show the patch instead of creating a new surface model. The **slab** option overrides any **offset** and generates a slab of finite thickness instead of a single layer of surface. If a single value is supplied for the slab *width*, its inner and outer layers will be offset from *surf-model1* by  $\pm\frac{1}{2}(\textit{width})$ . Alternatively, two values separated by a comma but no spaces can be used to specify the offsets of the two slab layers independently. Patch or slab offsets can be positive (outward) or negative (inward). *Offsets affect only the display, not the area measurement*, which is taken at the *surf-model1* surface. The **smooth** option smooths the new surface but is generally not recommended. The **optimize** setting speeds up the calculation by disregarding far-apart portions of the surfaces. Currently, each model must contain only a single [surface piece](#) (it may be necessary to turn off surface capping, see [sop cap](#)).

For atomic structures, [measure buriedArea](#) may be more appropriate.

- **measure correlation** *map-model1 map-model2* [ **aboveThreshold** true|false ] [ **rotationAxis** *axis* ] [ **angleRange** *start,end,step* ] [ **plot** true|false ]

Calculate the correlation between two [volume data](#) sets (maps) in two ways:

$$\textit{correlation} = \frac{\langle \mathbf{u}, \mathbf{v} \rangle}{|\mathbf{u}| |\mathbf{v}|}$$

$$\textit{correlation about mean} = \frac{\langle \mathbf{u} - \mathbf{u}_{\text{ave}}, \mathbf{v} - \mathbf{v}_{\text{ave}} \rangle}{|\mathbf{u} - \mathbf{u}_{\text{ave}}| |\mathbf{v} - \mathbf{v}_{\text{ave}}|}$$

where vector **u** contains the values of the first map (*map-model1*) and **u<sub>ave</sub>** is a vector with all components equal to the average of the components of **u**. Vectors **v** and **v<sub>ave</sub>** are defined analogously for the second map (*map-model2*), except that the values are sampled at the grid point locations of the first map using trilinear interpolation.

- If **aboveThreshold** is **true** (default), the calculation will include only the grid points in the first map with values above its lowest contour level in [Volume Viewer](#). Otherwise, all nonzero-valued grid points will be included.
- Specifying **rotationAxis** allows calculating the correlation multiple times for different orientations of the first map about an *axis*, described by an [atom-spec](#) of exactly two atoms (not necessarily bonded or in the same model) or one bond. If two atoms, the order of specification defines a handedness, and right-handed rotations are positive. If a bond, the handedness is not under user control. A bond can only be specified by [selecting](#) it and using the word **selected**, **sel**, or **picked**;

any atoms also selected at the time will be ignored. The calculations are performed internally, without moving the map in the display.

- The **angleRange** arguments control how many correlation calculations should be performed and at what angles of the first map relative to its current position. By default, *start* = 0°, *end* = 360°, and *step* = 2°.
- If **plot** is **true** (default), the correlation values will be graphed in a separate window in addition to being tabulated in the [Reply Log](#). Clicking and dragging in the plot window will show a vertical line and rotate the first map to the indicated angle.

See also: [volume](#), [molmap](#), [fitmap](#), [Fit to Segments](#)

- **measure distance** *atoms-surfs1 atoms-surfs2* [ **multiple** true|false ] [ **show** true|false ] [ **color** *line-color* ]

Report the closest distance between one set of atoms and/or [surface pieces](#) (*atoms-surfs1*) and another set (*atoms-surfs2*). Surface models and their pieces can be specified by model number or as a [selection](#) ([details...](#)). Atoms or surface pieces belonging to both sets are removed from the second set. Distances to surfaces are computed to the vertices of the surface mesh. All surface vertices are considered, even if hidden. Setting **multiple** to **true** gives the closest distance from *each* atom and surface piece in the first set to any in the second set. Lines depicting the distances can be displayed with **show true**, and the *line-color* can be any [color name](#) that specifies a single color (default **cyan**). The lines are generated as a [surface model](#).

See also: [distance](#), [zonesel](#)

- **measure fieldLines** *map-model(s)* [ **lines** *N* ] [ **startAbove** *cutoff* ] [ **step** *s* ] [ **modelId** *model-number* ] [ **color** *line-color* ] [ **lineWidth** *width* ] [ **tubeRadius** *radius* ] [ **circleSubdivisions** *M* ] [ **markers** true|false ]

Calculate electric field lines for one or more [electrostatic potential maps](#). The **lines** option specifies the number *N* of field lines to generate per map (default **1000**). The lines originate at grid points with potential magnitudes greater than **startAbove** *cutoff* value (default the highest contour level displayed for that map, or **10** if no contour levels are displayed) and are traced along the map gradient in steps of *s* (default **0.5**) times the maximum grid spacing of the map ([details...](#)). The field lines are added as a new model with ID number *model-number* (default next unused number), and the *line-color* can be any [color name](#) that specifies a single color (default **0.7,0.7,0.7,1**).

The lines will be shown as wires of **lineWidth** *width* (default **1**) unless a *radius* for tube display is given and/or **markers** is set to **true**. In the case of **markers true**, tubes will be created as a [marker set](#), with default *radius* 0.5 times the maximum grid spacing. In the case of **markers false**, the wires or tubes will be created as a [surface model](#), with *M* facets (default **12**) used to approximate the circular cross-section of tubes. Surface models can subsequently be colored by the potential using [Electrostatic Surface Coloring](#) or the command [scolor](#).

If the command runs without error but no lines (or few) are visible, perhaps many short lines not protruding above the surface were generated. It may be helpful to use a higher **startAbove** value, and secondarily, to increase the total number of **lines** ([details...](#)).

- **measure inertia** [atom-spec](#) [ **perChain** true|false ] [ **showEllipsoid** true|false ] [ **color** *ellipsoid-color* ]

Calculate the inertia ellipsoid for [atom-spec](#), which could include atoms and/or [surface pieces](#). Atoms are mass-weighted; [surface pieces](#) are treated as thin shells with mass proportional to surface area ([details...](#)). If both atoms and surfaces are specified, two separate ellipsoids are calculated (a combined calculation cannot be performed). Principal axes, lengths, moments, and center are reported for each ellipsoid, using the model coordinate system of the first atom or surface piece specified to define it. The vectors v1, v2, and v3 are the principal axes (longest to shortest). The lengths a, b, c are half-diameters along axes v1, v2, and v3, respectively. The moments r1, r2, and r3 are calculated as  $(\text{inertia}/\text{mass})^{1/2}$  about axes v1, v2, and v3, respectively. They can be considered effective radii; placing all of the mass at that distance from the center would reproduce the moment of inertia calculated for the structure around that axis.

The **perChain** option indicates whether to calculate a separate ellipsoid for each chain in [atom-spec](#). If **showEllipsoid** is **true** (default), the ellipsoid(s) will be opened as a [surface model](#). The *ellipsoid-color* can be any [color name](#) that specifies a single color. Otherwise, an ellipsoid will be colored to match the first atom or surface piece in its calculation.

See also: [define axis](#), [aniso](#), [shape ellipsoid](#), [Measure and Color Blobs](#), [geometric objects](#)

- **measure mapStats** *map-model(s)* [ **step** *N* | *Nx,Ny,Nz* ] [ **subregion** *name* | *i1,j1,k1,i2,j2,k2* | **all** ]

Calculate the mean, standard deviation (SD) from the mean, and root-mean-square (RMS) deviation from zero for each specified map. The **step** option indicates whether to use the full resolution of the data (step size **1**, default) or a specified [subsample](#) (step size > 1).

Step sizes must be integers. A step size of *N* indicates using every *N*<sup>th</sup> point. If a single number is supplied, it is used along all three axes; if three numbers are supplied (separated by commas but not spaces), they are used along the X, Y, and Z axes, respectively. The **subregion** option indicates whether to use the full extents of the data (**all**, default) or a specified [subregion](#). A subregion can be specified by:

- *name* previously assigned with [volume](#) (see [nameRegion](#)) or [Volume Viewer](#) (see [Named regions](#))
- grid indices *i1-i2* along the X axis, *j1-j2* along the Y axis, and *k1-k2* along the Z axis. Grid indices must be integers separated by commas but not spaces.

See also: [vop scale](#), [Volume Mean, SD, RMS](#)

- **measure mapSum** *map-model(s)* [ **aboveThreshold** *level* ] [ **step** *N* | *Nx,Ny,Nz* ] [ **subregion** *name* | *i1,j1,k1,i2,j2,k2* | **all** ]

For each specified map, sum values above a threshold *level*, if given (default is no threshold). The **step** and **subregion** options are as described for [measure mapStats](#).

- **measure mapValues** *map-model* [atom-spec](#) [ **name** *attribute-name* ] [ **report** all | *N* ]

Interpolate *map-model* to obtain values at the positions of the atoms in [atom-spec](#). The values will be assigned as an atom [attribute](#), either named as indicated with **name**, or (default) derived by prepending “value\_” to the name of *map-model* and replacing any non-alphanumeric characters with underscores. The **report** option indicates how many values should be reported in the [Reply Log](#), those for the first *N* atoms (default **10**, but 0 can be used) or all.

See also: [Values at Atom Positions](#)

- **measure pathLength** [atom-spec](#) [ **group** connected | models | all ]

Report the total length of the specified bonds, optionally as separate totals for bonds within different **connected** groups or **models**. This measurement applies to [links between markers](#) as well as to standard bonds.

- **measure rotation** *model1 model2* [ **coordinateSystem** *N* ] [ **showAxis** true|false ] [ **showSlabs** true|false ] [ **color** *color* ]

Report the transformation of *model2* relative to *model1* as:

- a matrix in which the first three columns describe a rotation and the fourth describes a translation (performed after the rotation)
- an axis of rotation (a unit vector), point on the axis, rotation angle, and shift parallel to the axis

The transformation is expressed in the coordinate system of *model1* unless a different **coordinateSystem** *N* (model ID number preceded by #) is specified. If **showAxis** is **true** (default), a [marker set](#) showing the axis as a rod will be opened as a separate model. The rod length equals the largest dimension of the bounding box of *model1*, and its diameter is set to 5% of the length. If **showSlabs** is **true** (default **false**), two rectangular slabs showing the rotation axis and angle and the shift will be opened as a [surface model](#). The axis and/or slab *color* can be any [color name](#) that specifies a single color.

See also: [fitmap](#), [superimposing structures](#)

- **measure spine** *region-spec* [ **spacing** *s* ] [ **tipLength** *t* ] [ **color** *color* ]

Calculate and display a [path](#) along the center line of each specified [segmentation region](#). The length of the path (spine) and orthogonal diameters at its midpoint are reported. Regions can be specified as a selection (e.g., **sel**) or with model number(s) (e.g., **#1**). The longest principal axis of inertia of a region is determined using equal weighting of the

enclosed volume grid points. The region is then divided into slices along that axis, with end slices **tipLength** *t* thick and interior slices at least **spacing** *s* thick. Both distances are specified in physical units (typically Å). The default **spacing** *s* is 20 times the minimum segmentation grid plane spacing, and the default **tipLength** *t* = 0.2*s*. The spacing is increased as needed to make the interior slices of equal width. A [marker](#) is placed at the geometric center of the grid points within each slice. The markers are linked, and the markers and links together form a [path](#). The path *color* can be any [color name](#) that specifies a single color. For each pair of consecutive links in the path, the curvature is measured as  $1/r$ , where *r* is the radius of a circle tangent to the midpoints of the links. The minimum, maximum, and average (weighting each value along the path equally) curvatures are assigned as [segmentation region attributes](#) named **curvature minimum**, **curvature maximum**, and **curvature average**, respectively.

See also: [segment sliceimage](#), [Segment Map](#)

• **measure symmetry** *map-model(s)* [ **minimumCorrelation** *mincorr* ] [ **nMax** *n* ] [ **points** *maxpts* ] [ **set** *true|false* ] [ **helix** *rise,angle[,n][,opt]* ]

Check each specified [volume data](#) set (*map*) for cyclic, dihedral, tetrahedral, octahedral, and icosahedral symmetries in standard coordinate systems. Helical symmetry can be considered if approximate parameters are supplied. The symmetry assignment can be used by other commands such as [sym](#) and [fitmap](#), and is included in [Chimera map](#) format. For direct assignment of a specified symmetry, see the command [volume symmetry](#).

If the [correlation](#) of the map with itself after symmetry transformation is at least *mincorr* (default **0.99**), the detected type of symmetry will be reported, and if **set** is true, assigned to the map in Chimera. The correlation calculation uses only map points with values above the displayed contour level; if the number of such points exceeds *maxpts* (default **10,000**), a random sample of *maxpts* is chosen from them and used. Values in the first copy of the map are compared with the superimposed (interpolated) values in the rotated copy of the map.

Center of point symmetry is considered only at the following:

- the grid point nearest the average indices of grid points with values above the displayed contour level. The map's lowest contour level in [Volume Viewer](#) is used.
- one or two grid points based on the overall map dimensions: only the midpoint along axes with odd numbers of points, and along axes with even numbers of points, those on either side of the midpoint. Rather than all possible combinations for axes with even numbers of points, only the two points with all indices lower or all higher are evaluated.

For cyclic and dihedral symmetry, rotation is considered only about the Z axis, and for dihedral symmetry, flipping symmetry only about the X or Y axes. Cyclic (*C<sub>n</sub>*) symmetry is checked for order *n* up to **nMax**, default **8**. If more than one *C<sub>n</sub>* symmetry meets the criterion, those for which a higher multiple is also found are discarded, and of the remaining, the one with the highest correlation is assigned. For example, if *n* = 2, 3, 6, and 7 were to meet the criterion, 6-fold would override 2- and 3-fold, and 6-fold or 7-fold symmetry, whichever gave the highest correlation, would be assigned. Tetrahedral

symmetry is considered in two orientations:

- 2-folds along X, Y, and Z, with a 3-fold along axis (1,1,1)
- 3-fold along Z, with a second 3-fold in the YZ plane such that rotation about the X axis by  $\sim 110^\circ$  is a symmetry operation (EMAN convention)

Icosahedral symmetries are only considered in the eight orientations listed in the [Icosahedron Surface](#) dialog.

The **helix** option specifies looking for helical symmetry with approximate *rise* (in physical units of distance, typically Å) and *angle* (degrees) per asymmetric unit. If this option is used, the other types of symmetry are not considered except for combined helical and cyclic symmetry (for example, [EMD-1757](#), approximately 42 Å rise and  $21^\circ$  twist per subunit). Helical symmetry is infinite, but the number of copies to place when considering that symmetry, *n*, is necessarily finite. If not given, *n* will be determined by dividing the apparent length of the helix in the map by the rise and rounding to the nearest positive integer. The **opt** keyword indicates optimizing the fit of the map copies to itself to identify more accurate helical parameters.

See also: [molmap](#), [volume](#)

- **measure volume** *surface(s)*

Calculate the total volume enclosed by each specified [surface piece](#), not including any interior bubbles. The surface pieces in a [surface model](#) can be specified collectively by model number, or individually by [selection from the screen](#) and using the word **sel**, **selected**, or **picked**. Like [Measure Volume and Area](#), the calculation uses the full surface even if it is partly hidden by clipping or zoning, and holes are treated as if covered by planar caps.

See also: [surface](#), [volume](#)

## TECHNICAL NOTES

### Electric field line placement.

A key issue for [measure fieldLines](#) is which lines to compute, out of an infinite number of possibilities. The basic idea is to make the number of lines originating from a charge proportional to the magnitude of the charge. This is approximated by computing the magnitude of the gradient squared divided by the potential at every grid point and using the grid points with the largest N values for starting N field lines. The rationale is as follows: near a point charge, the potential is  $q/r$ . The gradient squared divided by the potential is  $q/r^{**3}$  ( $= (q/r^{**2})^{**2} / (q/r)$ ). If for a chosen point (gradient squared / potential)  $> C$ , then  $q/r^{**3} > C$  and  $r^{**3} < q/C$ . The number of grid points around a charge is proportional to the volume  $r^{**3}$ , so is linear in charge.

The resulting field lines can vary greatly depending on the charge distribution used to compute the potential. When charges are concentrated on fewer atoms, longer lines are generated, but when charges are distributed onto many atoms, there may be many short lines that do not extend beyond the molecular surface. In that case, it may be helpful to use a higher **startAbove** *cutoff* value and a larger number of

**lines** (total) to get more lines that loop beyond the surface.

### Inertia calculation.

The command [measure inertia](#) computes the moments of inertia of a set of atoms as in classical mechanics:

$$I_{jk} = \sum_i (m_i (\delta_{jk} |x_i|^2 - x_{i,j}x_{i,k}))$$

$I$  is a 3x3 matrix with indices  $j$  and  $k$  ( $j=1,2,3$  and  $k=1,2,3$ ). Each matrix element is a sum over atoms, where  $m_i$  and  $x_i$  are the mass and position of atom  $i$ , respectively, and  $\delta_{jk}$  is 1 for  $j=k$ , otherwise 0. The principal axes are the eigenvectors of the matrix, and the moments about those axes are the eigenvalues. Basically, the moment is a sum of mass times distance squared from the rotation axis. Before this formula is applied, the center of mass position is subtracted from the atom coordinates, so that the measured quantity is the inertia about the center of mass. The approach for surfaces is analogous, where atoms are replaced by vertices of the triangulated surface, and the “mass” of each vertex is  $\frac{1}{3}$  of the area of the attached triangle. This treats the surface as a thin shell. The “inertia ellipsoid” shown by Chimera is not the same as the one defined in physics. Instead, it is the ellipsoid that has the same inertia as the measured object:

- For atoms, we show the surface of a uniform-density solid ellipsoid that has the same principal axes and moments as the atoms.
- For surfaces, we show an ellipsoidal surface that as a thin shell has the same axes and moments as the measured surface.

**Usage:****meshmol** *surf-model stick-radius*

The **meshmol** command creates a molecule model from a surface. This allows mesh display as cylindrical sticks, whereas surface model meshes can only be shown as lines. **Meshmol** works on all kinds of [surface models](#), but not on surfaces in [VRML models](#).

*Surf-model* is the model number of the surface, optionally preceded by #.

A new molecule model is created in which the "bonds" (the mesh lines in the surface model) are shown as sticks of the specified *stick-radius*. It is colored to match the surface coloring, which could be a single color or different colors for different vertices. The molecule model is also a [marker set](#).

*Advantages* of treating a surface as a molecule include the ability to display sticks, as mentioned above, and to perform other operations available for atoms and bonds, including [saving coordinates](#) to a file. Further, since the molecule model is a [marker set](#), [Volume Tracer](#) can be used to control its display and save it as a [marker file](#). Marker files include colors and radii in addition to coordinates.

*Disadvantages* relate to the creation of physically unreasonable molecules with very large numbers of atoms and bonds. Chimera performs chemical perception, which may take up to hours in the presence of huge numbers of small rings. This essentially freezes the program. Any operations that trigger chemical perception should be avoided after **meshmol** is used, such as using the ball-and-stick or sphere [representations](#) or [adding hydrogens](#), even to other molecules. Atoms and bonds use more memory than surface points and mesh lines, and rendering large surfaces as cylinders may take a long time. Mesh molecules with more than 100,000 atoms cannot be [saved as PDB files](#) because the CONECT records only allow 5-digit atom serial numbers.

See also: [hkage](#), [shape](#), [volume](#), [Cage Builder](#), [Volume Tracer](#)

**Usage:****minimize** [options](#)

**Minimize** is the command-line implementation of [Minimize Structure](#); it energy-minimizes molecule models, optionally holding some atoms fixed. Minimization routines are provided by [MMTK](#), which is included with Chimera. [Amber](#) parameters are used for standard residues, and Amber's [Antechamber](#) module (included with Chimera) is used to assign parameters to nonstandard residues.

Before energy calculations can be performed, it is necessary to correct structural inconsistencies, add hydrogens, and associate atoms with force field parameters. **Minimize** calls [Dock Prep](#) to perform [several tasks](#) to to prepare the structure(s). [Dock Prep](#) may in turn call other tools ([AddH](#) and [Add Charge](#)). Currently, [Add Charge](#) (or the command [addcharge](#)) must be run prior to minimization because it plays a crucial role in assigning [parameters](#).

[Steepest descent](#) minimization is performed first to relieve highly unfavorable clashes, followed by [conjugate gradient](#) minimization, which is much slower but more effective at reaching an energy minimum after severe clashes have been relieved. Energies (kJ/mol) are reported in the [Reply Log](#). **\*\*Step numbers reported by MMTK are 2 greater than the actual numbers of minimization steps performed.** The additional "steps" are not minimization steps but operations required to obtain gradient values and updated coordinates.\*\*

Please consult the [Minimize Structure](#) manual page for further details.

**Options**

Option keywords for **minimize** can be truncated to unique strings and their case does not matter. A vertical bar "|" designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

**spec** [atom-spec](#)

Minimize/move only the atoms in [atom-spec](#). All other atoms will be held in place automatically, without explicit use of [freeze](#). However, energy calculations will include the entire molecule models containing the specified atoms, unless [fragment true](#) is also used. All included models are treated as a single system. If [atom-spec](#) contains any spaces, it must be enclosed in single or double quote marks.

**fragment** true | false

Instead of the entire models, include in energy calculations only the residues containing the [spec](#) atoms. Residues can only be included/excluded as a whole, and all included residues (even if from multiple models) are treated as a single system. Note, however, that any subset of the [spec](#) atoms can be [frozen](#) in place.

**cache** true | false

Whether to cache parameters such as partial charges for the current system (models or

[fragment](#) to be included in energy calculations). Caching allows reusing the data in subsequent rounds of minimization after simply updating the coordinates. Cached data for a system will be kept indefinitely unless Chimera detects a change in that system (atom addition or deletion); setting **cache false** helps to prevent overconsumption of memory.

**prep true | false**

Whether to call [Dock Prep](#) when parameters for the current system are not in the [cache](#). (If the parameters for the system are in the cache, [Dock Prep](#) will not be called, regardless of this setting.) The purpose of **prep false** is to avoid re-preparing structures that have already been prepared but not cached (for example, a different [fragment](#) than was previously minimized).

**freeze none | selected | unselected | [atom-spec](#)**

Which atoms (in addition to those [frozen automatically](#)) to hold in place during minimization: none (default), those which are [selected](#), those which are not selected, or those indicated with an [atom-spec](#). Previously this option was required to freeze any atoms, but now that atoms other than the [spec](#) atoms are frozen automatically, this option is only needed in the case of:

1. restricting the calculation to certain residues with the [spec](#) option and [fragment true](#)
2. additionally freezing some of the [spec](#) atoms

This combination of options is useful for preventing long or distorted bonds at the junction between included and excluded residues. For example, minimizing a linker between two domains that are excluded from the calculation could significantly degrade the linker-domain bonds (since those bonds are ignored) unless the linker endpoint atoms are frozen in addition to the domains.

**nsteps *N***

*N* (default **100**) is the number of steps of steepest descent minimization to perform before any conjugate gradient minimization.

**stepsize *S***

*S* (default **0.02 Å**) is the initial step length for steepest descent minimization.

**cgsteps *M***

*M* (default **10**) is the number of steps of conjugate gradient minimization to perform after finishing any steepest descent minimization.

**cgstepsize *D***

*D* (default **0.02 Å**) is the initial step length for conjugate gradient minimization.

**interval *I***

*I* (default **10**) is how often to update the display, in terms of minimization steps.

**nogui true | false**

Setting **nogui** to **true** suppresses the appearance of tool dialogs that might otherwise appear. [Dock Prep](#) and [AddH](#) are executed with their default settings (as shown when running **minimize** with **nogui false**), while [Add Charge](#) uses Amber **ff14SB** charges for standard residues ([details](#)) and for any nonstandard residues, the [method](#) last specified in its GUI (**AM1-BCC** if the GUI has not been used).

See also: [addh](#), [addcharge](#), [findclash](#), [Dock Prep](#), [Adjust Torsions](#), [Minimize Structure](#)

**Usage:**

( **modelcolor** | **modelcolour** ) *color\_name* [atom-spec](#)

**Usage:**

( **modelcolor** | **modelcolour** ) *color\_name* *model\_number(s)*

**Modelcolor** assigns color to molecule models at the model (or [submodel](#)) level. Atoms and ribbons will only show the model color when they lack color assignments of their own, because model color is overruled by more specific color settings (see [coloring hierarchy](#)).

*Color\_name* can be **none** or any [color name](#) that specifies a single color.

If the [atom-spec](#) is blank, all open molecule models will be affected; otherwise, the entire model(s) containing [atom-spec](#) will be affected. Alternatively, molecule models can be specified with one or more model numbers or ranges of model numbers separated by commas (with or without a preceding #).

Model-level color can also be changed in the [molecule model attributes panel](#).

See also: [color](#), [ribinsidecolor](#), [scolor](#), [transparency](#), [surfcOLOR](#)

**Usage:****modeldisplay** [atom-spec](#)**Usage:****~modeldisplay** [atom-spec](#)**Usage:****modeldisplay** *model\_number(s)***Usage:****~modeldisplay** *model\_number(s)*

**Modeldisplay** sets model-level display status (where "model" can indicate a [submodel](#) or a model that is not subdivided into submodels). If the [atom-spec](#) is blank, all open models will be affected. If an [atom-spec](#) is given, models with model numbers corresponding to the specified atoms will be affected. Alternatively, models can be specified with one or more model numbers or ranges of model numbers separated by commas (with or without a preceding #).

**Modeldisplay** toggles the overall display status of all types of models; it does not reset the display status of individual atoms, bonds, or atom surfaces, but overrides them (see [display hierarchy](#)). Changing the model-level display status can be thought of as enabling or disabling the display of its components; even if display is enabled at the model level, individual components may or may not be displayed.

Hiding a model without disrupting its atomic display states is often convenient. For example, it is common to display the backbone of a protein plus only the side chains of interest and a few important crystallographic waters. Attaining this state may take several commands or menu actions. **~Modeldisplay/****modeldisplay** can then be used to hide and show the model without disrupting the display setup.

Examples:

```
~modeldisp #0.1-5
```

- hide submodels 1 to 5 of model 0

```
~modeldisp #1
```

- hide model 1

See also: [display](#), [objdisplay](#), [surface](#), the [Model Panel](#)

**Usage:****molmap** [atom-spec](#) *resolution* [options](#)

The command **molmap** generates a density map from the [specified](#) atoms. Each atom is described as a 3D Gaussian distribution of width proportional to the *resolution* and amplitude proportional to the atomic number. A map of the combined densities is generated and opened as a data set in [Volume Viewer](#). A map corresponding to a symmetrical multimer of the structure can be generated with the [symmetry](#) option. Map display can be adjusted and the map saved to a file using [Volume Viewer](#) or the command [volume](#). See also: [measure correlation](#), [meshmol](#), [fitmap](#), [Fit to Segments](#)

The **molmap** command is based on the **pdb2mrc** program in the [EMAN package](#).

**Options**

Option keywords for **molmap** can be truncated to unique strings, and their case does not matter. Synonyms for true: True, 1. Synonyms for false: False, 0. A vertical bar "|" designates mutually exclusive options, and default settings are indicated with **bold**.

**gridSpacing** *s*

The grid spacing *s* (default *resolution/3*) is the separation of points along the X, Y, and Z axes of the generated map.

**edgePadding** *p*

The edge padding *p* (default **3\*resolution**) sets map dimensions relative to the bounding box of the atom centers. Each face of the volume data box is offset outward by *p* from the corresponding bounding box face.

**cutoffRange** *r*

How many standard deviations  $\sigma$  (default **5**) of each Gaussian distribution to include in the map. Omitting the tails speeds up map calculation.

**sigmaFactor** *f*

Together with the *resolution*, the sigma factor *f* determines the width of the Gaussian distribution used to describe each atom:

$$\sigma = f(\text{resolution})$$

By default,  $f = 1/(\pi * 2^{1/2}) \approx 0.225$  which makes the Fourier transform (FT) of the distribution fall to  $1/e$  of its maximum value at wavenumber  $1/\text{resolution}$ . Other plausible choices:

- $1/(\pi * (2/\log 2)^{1/2}) \approx 0.187$  makes the FT fall to half maximum at wavenumber  $1/\text{resolution}$
- $1/(2 * 2^{1/2}) \approx 0.356$  makes the Gaussian width at  $1/e$  maximum height equal the *resolution*

- $1/(2 * (2\log 2)^{1/2}) \approx 0.425$  makes the Gaussian width at half maximum height equal the *resolution*

**displayThreshold** *m*

Set the initial contour level to enclose a fraction *m* (default **0.95**) of the total mass in the map. The fraction equals the sum of grid point values above the contour level divided by the sum of all grid point values.

**modelId** *N*

Open the map as model number *N* (an integer, optionally preceded by #). Submodel specifications #*N.N* (# required) can also be given. If the source atoms belong to a single model, the default is the same model number as the atoms; if the atoms belong to multiple models, the default is the lowest unused number.

**replace** true|false

Whether to close any map previously created by **molmap** from the same set of atoms.

**showDialog** true|false

Whether to show the [Volume Viewer](#) dialog after creating the map.

**symmetry** *sym-type*

Create a map corresponding to a symmetrical multimer of the structure. By default, no symmetry is used. Specifications of *sym-type* are case-independent, and most types have additional sub-options or parameters:

- **biomt** - use [biological unit information](#), if any, from the molecule model containing the specified atoms
- symmetry of model #*N* - use [biomt information](#) from another molecule model or the [symmetry assignment](#) of a volume model
  - Example: **#4**
- [cage model](#) polygon symmetry #*N,pM* or #*N,pnM* - place copies at equivalent positions relative to each *M*-sided polygon in the [cage model](#) with ID number **N**. The **pM** form places one copy per *M*-sided polygon, whereas **pnM** places *M* copies per *M*-sided polygon using *CM* symmetry about the center of the *M*-sided polygon nearest the original copy.
  - Examples: **#2,p6** or **#2,pn5**
- cyclic symmetry **C<sub>n</sub>** around [axis](#) and [center](#)
  - Example: **C3**
- dihedral symmetry **D<sub>n</sub>** around [axis](#) and [center](#)
  - Example: **d7**
- tetrahedral symmetry **T**[,*orientation*] around [center](#)
  - Example: **t,z3**
 where *orientation* can be:
  - **222** (default) - with two-fold symmetry axes along the X, Y, and Z axes, a three-fold along axis (1,1,1)
  - **z3** - a three-fold symmetry axis along Z, another three-fold axis in the YZ plane such that rotation about the X axis by ~110° is a symmetry operation (EMAN convention)

- octahedral symmetry **O** around [center](#)
  - icosahedral symmetry **I**,*orientation*] around [center](#)
    - Example: **i,n25**
- where *orientation* can be:
- **222** (default) - with two-fold symmetry axes along the X, Y, and Z axes
  - **2n5** - with two-fold symmetry along X and 5-fold along Z
  - **n25** - with two-fold symmetry along Y and 5-fold along Z
  - **2n3** - with two-fold symmetry along X and 3-fold along Z
  - **222r** - same as 222 except rotated 90° about Z
  - **2n5r** - same as 2n5 except rotated 180° about Y
  - **n25r** - same as n25 except rotated 180° about X
  - **2n3r** - same as 2n3 except rotated 180° about Y
- helical symmetry **H**,*rise,angle,n*,*offset*] around [axis](#) and [center](#)
    - Example: **h,43.5,21,6,-2**
- where *rise* is the translation along the [axis](#) per subunit, *angle* is the rotation in degrees per subunit, and *n* is how many copies total (including the original) the resulting segment of infinite helix should contain. The integer *offset* (default **0**) allows extending the helix in both directions. The example above would give *n* = 6 copies total, with two copies in the negative axis direction, one at the identity position, and three in the positive axis direction.
- translational symmetry **shift**,*n,distance* along [axis](#) - or - **shift**,*n,x,y,z*
    - Example: **shift,3,26.7**
- where *n* is how many copies total (including the original) the result should contain. The translation can be expressed as a *distance* along the [axis](#) or as a vector *x,y,z* in the [reference coordinate system](#).
- the product of symmetry groups, each specified as described above and separated by \* to indicate multiplying each symmetry matrix of one group with each symmetry matrix of another; can be generalized to multiple symmetry groups (not just two)
    - Example: **c2\*h,42,21,9,-4**

### axis *axis*

Specify axis of symmetry (default **z**), where *axis* can be:

- **x** - X-axis
- **y** - Y-axis
- **z** - Z-axis
- *x,y,z* (three values separated by commas only) - an arbitrary vector in the [reference coordinate system](#)
- an [atom-spec](#) of exactly two atoms (not necessarily bonded or in the same model) or one bond. A bond can only be specified by [selecting](#) it and using the word **selected**, **sel**, or **picked**; any atoms also selected at the time will be ignored.

### center *center*

Specify center of symmetry (default **0,0,0**), where *center* can be:

- *x,y,z* (three values separated by commas only) - an arbitrary point in the [reference coordinate system](#)
- an [atom-spec](#) of any combination of atoms and [surface pieces](#). The center of the bounding sphere of the specified items will be used.

**coordinateSystem *N***

Specify a reference model (default is the molecule model containing the specified atoms) by model number *N* preceded by #. The reference coordinate system is used for interpreting specifications of [axis](#) and [center](#) of symmetry.

**Usage:**

**morph start** [atom-spec](#) [ name *morph-name* ] [options](#)

**Usage:**

**morph interpolate** [atom-spec](#) [ name *morph-name* ] [options](#)

**Usage:**

**morph movie** [ name *morph-name* ] [ steps *nsteps* ] [ nogui true | false ] [ minimize true | false ]

**Usage:**

**morph done** [ name *morph-name* ]

**Morph** is the command-line implementation of [Morph Conformations](#); it creates a trajectory that morphs between two or more structures (see [example systems](#)). Each sequential pair of input structures serves as the starting and ending points of one *segment* of the trajectory, and a morph trajectory can have multiple segments. Only the [atoms in common](#) among the input structures will be included in the morph trajectory. Morphing is under development and has [limitations](#).

By default, the [MD Movie](#) dialog will appear for showing the trajectory and optionally [recording](#) it as a movie. However, that dialog is suppressed when Chimera is in [nogui mode](#) or the [nogui true](#) option of **morph movie** is used, in which case the trajectory can be played with [coordset](#) instead. See also: [vop morph](#), [play](#), [making movies](#), the [ParM filament tutorial](#) at the Chimera web site

The different structures should be opened as separate models or submodels in Chimera and [superimposed](#). The [atom-spec](#) given in a **morph** command may refer to a whole molecule model or a subset of the atoms in a model, but either way, the entire model will be used. The models can have different numbers of residues or different sequences (homologs or mutants can be compared), but currently they must contain [equal numbers of chains](#). Chains are paired by chain ID if the sets of IDs are identical, otherwise by order of occurrence in the input files. Extra chains in the input models should be [deleted](#) beforehand or [split](#) into separate models not used in morphing.

The **morph start** command specifies the first structure (molecule model) in the trajectory and how to interpolate from that structure to the next. The entire molecule model containing the specified atoms will be used; atoms from more than one model should not be specified. A *morph-name* can be assigned with the **name** keyword, but is not necessary unless multiple morph trajectories will coexist. The default *morph-name* is **default**. All keywords to **morph** and their values except *morph-name* can be truncated to unambiguous strings.

The **morph interpolate** command specifies the next structure (molecule model) in the trajectory and how to interpolate from that structure to the one after it, if any. Since a morph trajectory can contain multiple segments, this command can be used multiple times for the same trajectory.

For both **morph start** and **morph interpolate**, additional options are:

Option keyword	Possible values (see <a href="#">details</a> )	Defaults	
		start	interpolate
<b>method</b>	corkscrew independent linear	corkscrew	(same as preceding segment)
<b>rate</b>	linear "ramp down" "ramp up" sinusoidal	linear	(same as preceding segment)
<b>frames</b>	(integers)	20	(same as preceding segment)
<b>cartesian</b>	true false	false	false

The **morph movie** command creates the trajectory. Normally [MD Movie](#) is started automatically to show the trajectory, but if Chimera is in [nogui mode](#) or the **nogui true** option is given, the [MD Movie](#) dialog is suppressed, and the command [coordset](#) can be used to play the trajectory instead. Whether to energy-minimize every interpolated conformation can be specified with **minimize** (default **false**, no minimization), and how many minimization steps to perform on each can be specified with **step nsteps** (default **60**). Note that minimization has its own set of [limitations](#) and increases computational demands significantly.

The **morph done** command deletes the trajectory and exits from [MD Movie](#).

**Usage:**

**move** *axis* [ *distance* [ *frames* ] ] [ **models** *model-spec* ] [ **coordinateSystem** *N* ]

**Move** translates the specified **models** by *distance* along the specified *axis* for a specified number of *frames* (default **1**). If a *distance* is not given, the length of the *axis* vector will be used. If no models are specified, all [active](#) models will be translated. See the [video mini-example](#). See also: [select](#), [movie-related commands](#)

A **coordinateSystem** for axis definition can be specified by reference model number *N*, optionally preceded by #. Otherwise, the laboratory frame of reference will be used, in which:

- the X-axis is horizontal in the plane of the screen, increasing to the right
- the Y-axis is vertical in the plane of the screen, increasing upward
- the Z-axis is perpendicular to the screen, increasing outward (toward the viewer)

The *axis* can be:

- **x** - unit vector along the X-axis (1,0,0)
- **y** - unit vector along the Y-axis (0,1,0)
- **z** - unit vector along the Z-axis (0,0,1)
- *x,y,z* (three values separated by commas only) - an arbitrary vector

The *distance* is in the units of length intrinsic to the data, usually Å, and can be positive or negative.

Keyword options can be used in any order and the keywords can be truncated.

Examples:

```
move x -1 25 models #1,2 move 5,2.5,0 coord #0 mod #0 30
```

Commands continue to be processed while the requested motion is in progress. To prevent processing of further commands until the motion is finished, use the [wait](#) command. To halt an ongoing **move**, use [freeze](#).

**Usage:****movie** *action* [options](#)

**Movie** is the command-line implementation of [Movie Recorder](#), which captures image frames from Chimera and assembles them into a movie file. See also: [making movies](#), [movie-related commands](#), [tips on preparing images](#)

One of the following *action* keywords must be specified:

- [record](#) - start recording frames
- [encode](#) - stop any ongoing recording and encode the saved image frames into a movie file
- **stop** - stop recording frames
- [reset](#) - reset status to zero frames saved
- **abort** - halt any encoding in progress and perform a **reset**
- [crossfade](#) - interpolate from the preceding frame to the following frame
- [duplicate](#) - expand the preceding frame into multiple frames
- [ignore](#) - start/stop ignoring subsequent **movie** commands
- **formats** - list the available movie file formats
- **status** - report status (number of saved frames, *etc.*) in the [Reply Log](#)

Actions may have specific options and arguments, described below. Action and option keywords can be truncated. Default values are indicated with **bold**. A vertical bar “|” designates mutually exclusive values.

- **movie record** options:

**supersample** *N*

Whether to render each image at a higher resolution and then sample it down to the final size (does not apply to [raytracing](#)). Increasing *N* increases the smoothness of edges within images and the calculation time with little effect on file size. *N* is the number of pixels sampled in the X and Y dimensions for each pixel in the final saved image. Useful values range from 1 (no supersampling) to 4 (4x4 supersampling), with 3 recommended when supersampling is done. In addition, a value of 0 can be used to indicate onscreen rather than offscreen rendering, without supersampling. A potential disadvantage of over-supersampling is that lines such as silhouettes or hydrogen-bond representations in the final images may become unacceptably thin because there is a limit to how wide they can be in the initial images. Smoothness is also affected by [other settings](#).

**raytrace** true | false

Whether to [raytrace with POV-Ray](#) rather than saving the contents of the Chimera graphics window. Raytraced images include fancier effects like shadows, but take longer to compute. Raytracing parameters can be adjusted in the [POV-Ray Options preferences](#) (see [balancing time requirements and results](#)).

**size** *width,height*

Generate images at the specified pixel *width* and *height* by offscreen rendering. This option

allows creating movies with image dimensions larger than the Chimera window; if size is not given, the dimensions of the Chimera graphics window will be used. If the specified aspect ratio (the ratio of *width* to *height*) does not match that of the graphics window, the movie will show more or less of the scene vertically than is visible in the graphics window.

**format** jpeg | png | ppm

The format of saved image files. Regardless of which format is specified, however, frames are always saved in the **png** format when [raytracing](#) is used.

**directory** *image-directory*

Where image files should be saved; the default varies by platform.

**pattern** *filename-pattern*

The *filename-pattern* is a string to use when naming the image files; the default is **chimovie\_XXXX-\***, where **XXXX** is replaced by random characters and **\*** is replaced by the frame number (for example, 00001, 00002, *etc.*). Any specified *filename-pattern* should include one and only one **\***, which will be replaced by the frame number.

**limit** *maxframes*

Record no more than *maxframes* image frames (default **15000**). This safeguards against creating too many image files, as may occur with certain scripting errors.

- **movie encode options:**

**output** *pathname*

The *pathname* is the name and location of the output movie file (default **~/Desktop/movie.mp4**). If file name but not location is given, the default location is the current directory, typically the same location as the movie-making script. The output file format can be indicated by [filename suffix](#) or with the [format](#) option. The **output** keyword can be omitted if the pathname precedes any other options, and in that case, multiple pathnames with different suffixes can be given to produce multiple outputs of different formats (example: **movie encode a.mp4 b.webm**). If the **output** keyword is used, the option can appear anywhere in the command, even multiple times in a single command, but only a single pathname can be given per use of the keyword.

**format** h264 | vp8 | theora | mov | avi | mp4 | mp2 | mpeg | wmv

Encode the movie in the specified format. Format keywords and their synonyms are listed in the table below. Alternatively, the format can be specified by [output filename suffix](#); where formats share a suffix, the newer format (higher in the table) will be used. The available formats can be listed with the command [movie formats](#).

format keyword, synonyms	format	filename suffix
h264	H.264 (default on Mac, Windows)	.mp4
vp8 webm	VP8/WebM (default on Linux)	.webm

theora ogg ogv	Theora	.ogv
mov qt quicktime	Quicktime	.mov
avi	AVI MSMPEG-4v2	.avi
mp4	MPEG-4	.mp4
mp2	MPEG-2	.mpg
mpeg	MPEG-1	.mpg
wmv	WMV2	.wmv

**quality** highest | higher | high | **good** | medium | fair | low

Encode the movie to play back at a variable bit rate appropriate for the specified quality level. Higher quality corresponds to a higher bit rate and a larger file (at a given window size and [frame rate](#)). For constant bit rate encoding, the [bitrate](#) option should be used instead.

**bitrate** *rate*

Encode the movie to play back at a constant bit rate, where *rate* is the target rate in Kbits/s. Generally, 200 Kbits/s gives low quality, 1000 medium, and 6000 high, but this strongly depends on image dimensions and the amount of motion. For HD movies (1920 x 1080 pixels), a rate of 25000 Kbits/s would be expected to give high quality. For variable bit rate encoding, the [quality](#) option should be used instead.

**framerate** *fps*

The frame rate is the rate of playback in terms of image frames. The frame rate *fps* is an integer and defaults to **25** frames/s.

**roundtrip** true | false

Whether to include the frames in reverse order as the second half of the movie.

**resetMode** clear | keep | none

By default, a *reset* of the frame counter to zero will occur when movie encoding is complete. At the same time, the image files can be deleted (**clear**, default) or retained (**keep**) for some other purpose, such as movie encoding with a different program. The third option, **none**, indicates that no reset should occur: the frame counter should not be zeroed and the images should not be deleted. This allows subsequently re-encoding the movie using the same set of images.

**wait** true | false

Whether to wait for encoding to finish before proceeding. If multiple movies are being recorded back-to-back, it is necessary to wait until one is encoded before starting to record the next, because any recording will be halted automatically when encoding finishes.

**preset** vcd | svcd | dvd

The **preset** option indicates that a built-in set of parameters should be used:

- vcd (Video Compact Disc) - MPEG-1, window size 352x240, constant bit rate 1150 Kbits/s
- svcd (Super Video Compact Disc) - MPEG-2, window size 480x480, constant bit rate 2040 Kbits/s
- dvd (Digital Versatile/Video Disc) - MPEG-2, window size 720x480, constant bit rate 6000 Kbits/s

Using a preset will resize the graphics window and override other specifications of the listed parameters.

- **movie reset** [ **resetMode** clear | keep ]

Reset the frame counter to zero, either deleting the image files (**clear**, default) or retaining them (**keep**) for some other purpose such as movie encoding with a different program.

- **movie crossfade** [ **frames** *N* ]

*N* is the number of frames in the crossfade transition (default **25**), and can be given without the **frames** keyword. The transition is a linear interpolation from the preceding image frame to the image frame specified right after the crossfade. The crossfade transition will not be shown during recording, but will be in the resulting movie. See the [video mini-example](#).

- **movie duplicate** [ **frames** *N* ]

*N* is the number of times to duplicate the preceding image frame (default **25**), and can be given without the **frames** keyword. The duplicates will not be shown during recording, but will be in the resulting movie.

- **movie ignore** [ on | off ]

Start ignoring subsequent **movie** commands (no keyword, or equivalently the keyword **on**, **true**, or **1**) or stop ignoring them (keyword **off**, **false**, or **0**). Ignoring **movie** commands may be useful during script (movie content) development.

**Usage:**

`msc multiscale-surf-id atom-spec distance`

**Usage:**

`~msc multiscale-surf-id`

**Msc** colors surfaces from [Multiscale Models](#) to match nearby atoms. It acts similarly to [Color Zone](#), except that **msc** will infer symmetry-related atom positions for which coordinates have not been loaded.

The *multiscale-surf-id* is the ID number (shown in the [Model Panel](#)) of the surface model created by [Multiscale Models](#). Surface vertices within the cutoff *distance* of an atom in [atom-spec](#) or of any of its inferred copies will be assigned the same color as the atom. When multiple atoms are within the cutoff, a vertex will be colored to match the closest one. Each surface triangle is colored by linearly interpolating its vertex colors. Colors are defined by red, green, blue and opacity/transparency components.

For example, atoms could be colored by B-factor using [rangecolor](#) or [Render by Attribute](#). All of the chain surfaces in [Multiscale Models](#) could then be colored to match, using a command something like:

```
msc #1 #0 5.0
```

By contrast, [Color Zone](#) would only color surfaces near the chain copies with loaded coordinates.

The command `~msc` restores the colors assigned in the [Multiscale Models](#) dialog. Note that changing surface parameters in [Multiscale Models](#) will also revert the coloring.

See also: [color](#), [rangecolor](#), [rainbow](#), [scolor](#), [mcopy](#), [Render by Attribute](#), [Color Zone](#)

**Usage:**

( **surfrepr** | **msms repr** ) *style model\_number(s)*

**Usage:**

( **surfcats** | **msms cat** ) *category\_name [atom-spec](#)*

The command **surfrepr** (**msms repr**) changes the [surface representation](#), whereas **surfcats** (**msms cat**) is used to specify sets or [categories](#) of atoms that should be grouped for surface calculation purposes. See also: [surface](#), [split](#), [transparency](#), the [Actions menu](#), [surface calculation failures and workarounds](#)

**Surfrepr** changes the representation [surface representation](#), of specified models to the specified *style*:

The *model\_number(s)* should be preceded by #, as in a normal [atom specification](#). If no model numbers are given, all surface models will be affected. Possible values for *style* are:

- **filled** or **solid** (solid surface)
- **mesh** (the surface polygon edges only are shown)
- **dot** (the surface polygon vertices only are shown)

**Surfcats** creates groupings of atoms for subsequent surface calculations:

The atoms in [atom-spec](#) are assigned to a category named *category\_name*. A category's surface can then be displayed with the command [surface](#).

In many cases, the [automatic categories](#) suffice and **surfcats** is not required. Categories are mutually exclusive, that is, an atom can only be in one category at a time. When atoms are placed in a new category, they are subtracted from the category they occupied previously.

In the following example, the sidechains and backbone of a peptide are placed in separate categories, and a surface is shown for the sidechain portion.

```
open 1gcn
surfcats schain #0
surfcats bbone @n,ca,c,o
surface schain
```

In the next example, the structure has two DNA chains (E and F) and three protein chains (A-C). [Automatic categorization](#) would put all of these chains in category **main** and enclose them in a single surface:

```
open 1tup; rainbow chain
surface
```

To separately enclose the DNA and the protein:

```
surfcat dna nucleic acid  
surf dna
```

To further enclose each protein chain separately:

```
surfcat one :.a & protein  
surfcat two :.b & protein  
surfcat three :.c & protein  
surf one | two | three
```

The “& protein” specifications are needed to exclude nonprotein residues of chains A, B, and C (in this case, several water molecules) from the surface categories.

**Usage:****namesel** [ *name* ]**Usage:****~namesel** *name*

The **namesel** command saves the current [selection](#) with the specified *name*. Omitting *name* indicates listing the names of previously saved selections in the [status line](#) and [Reply Log](#). The command **~namesel** “forgets” an existing named selection.

A *named selection* refers to the contents of a selection, not the process by which the selection was generated. For example, if a selection of all oxygens in a particular structure is saved, retrieving that named selection when *different* structures are present will not select their oxygens. See [alias](#) instead for making command-line equivalents of long [atom specification](#) strings and/or commands.

Named selections can be restored (as well as saved) using the [Select menu](#) or [Rapid Access](#) interface, and their names can be used for command-line [atom specification](#).

See also: [select](#)

-- not available on Windows systems --

### Usage:

**neon** *options*

The **neon** command produces a three-dimensional representation of the displayed molecule(s) with appropriate shadows. Molecules can be depicted as space-filling spheres, balls and sticks, sticks, or tubes. The current orientation and coloring is retained. This image is not interactive and may require up to several minutes to produce. After the image is displayed, clicking the left mouse button returns to the interactive display window.

The **neon** command is actually an alias that uses [pdbrun](#) to send data to the outside program [neon](#), which processes the data and then sends it, along with any command-line flags, to the program [conic](#). There are many options, which are detailed in the [neon](#) and [conic](#) manual pages.

\*On **Mac** systems, the **conic -o** flag or configuration file [output](#) option must be used to generate any output at all. If display is desired in addition to an output file, the **-s** flag should also be used.

See also: [pdbrun](#), [conic](#), [represent](#), [raytracing with POV-Ray](#)

**Usage:**

**nucleotides** *sidechain representation* [ [sidechain-options](#) ] [atom-spec](#)

**Usage:**

**nucleotides** *ndbcolor* [atom-spec](#)

**Usage:**

**~nucleotides** [atom-spec](#)

The **nucleotides** command generates special representations of nucleotide bases and sugars; it is the command-line implementation of the [Nucleotides](#) tool. Even if only particular atoms are [specified](#), the entire residue(s) containing the atoms will be affected. The **nucleotides** command does not control [ribbon](#) display. See also: [aromatic](#), [fillring](#), [represent](#), [ribbon](#), [ribrepr](#), [ribscale](#), [ribspline](#), the [Actions menu](#)

The command keywords and their sub-keywords can be truncated to unique strings, and their case does not matter. A vertical bar "|" designates mutually exclusive options, and default settings are indicated with **bold**.

The **sidechain** keyword indicates the base/sugar *representation* and may have [further options](#):

- [atoms](#) - remove special nucleotide representations to reveal atoms and bonds
- [fill/fill](#) - show the sugar (ribose) and base moieties as filled rings
- [fill/slab](#) - fill sugar rings, show bases as slabs
- [tube/slab](#) - show sugars as tubes connecting the bases to the backbone, show bases as slabs
- [ladder](#) - show H-bonded residue pairs as rods or ladder rungs

The **ndbcolor** keyword specifies coloring residues by the convention used in the [Nucleic Acid Database \(NDB\) Atlas](#): A red, T blue, C yellow, G green, and U cyan.

The command **~nucleotides** removes special representations of nucleotide residues to reveal atoms and bonds (equivalent to [nuc side atoms](#)).

## Sidechain Representations and Sub-Options

- **atoms**

Remove special representations, including ring fill, to reveal atoms and bonds (equivalent to [~nuc](#)). Atom/bond representations can be controlled separately (for example, with the command [represent](#)).

- **fill/fill** [ **orient true|false** ]

Show the sugar (ribose) and base moieties as filled rings. The **orient** option controls whether the positive faces of bases are shown with bumps. The *positive faces* point towards the 3' end of a strand in right-handed A- and B-DNA (the positive Z direction in the [standard reference frame](#)). Ring fill thickness tracks the thickness of the surrounding bonds: a thin layer for wires, thicker for sticks. The thickness of sticks can be adjusted by changing the [stick scale](#) (a molecule model [attribute](#)). Regardless of the **orient** setting, orientation bumps will not be shown where the fill is a thin layer.

- **fill/slab** [ **shape** box | tube | ellipsoid ] [ **style** big | fat | long | skinny | *custom-style* ] [ **thickness** *d* ] [ **hide** true|false ] [ **orient** true|false ]

Fill sugar rings and show bases as slabs of the specified **shape** (box, elliptical tube, or ellipsoid) and **style**. Elliptical tubes and ellipsoids are made large enough to enclose the corresponding box. [Custom styles](#) can be used. The default slab **thickness** is 0.5 Å. The **hide** option indicates whether to hide base atoms/bonds. The **orient** option controls whether the [positive faces](#) of bases are shown with bumps.

- **tube/slab** [ **shape** box | tube | ellipsoid ] [ **style** big | fat | long | skinny | *custom-style* ] [ **thickness** *d* ] [ **hide** true|false ] [ **orient** true|false ] [ **glycosidic** true|false ]

Show sugars as tubes connecting the bases to the backbone, and show bases as slabs of the specified **shape** (box, elliptical tube, or ellipsoid) and **style**. Elliptical tubes and ellipsoids are made large enough to enclose the corresponding box. [Custom styles](#) can be used. The default slab **thickness** is 0.5 Å. The thickness of sugar tubes can be adjusted by changing the [stick scale](#) (a molecule model [attribute](#)). The **hide** option indicates whether to hide base atoms/bonds. The **orient** option controls whether the [positive faces](#) of bases are shown with bumps. The **glycosidic** option applies only to base-anchored slab styles (long, skinny) and controls whether the sugar tubes should each consist of a single straight segment (default) or two segments, one along the glycosidic (sugar-base) bond.

- **ladder** [ **ignore** true|false ] [ **stubs** true|false ] [ **radius** *r* ]

Show H-bonded residue pairs as rods or ladder rungs. The **ignore** option indicates whether to ignore non-base atoms when identifying H-bonds; otherwise, all nucleic acid atoms will be considered (default). The **stubs** option sets whether to show bases without H-bonds to other bases as half-rungs (default) or not at all. The **radius** (default 0.45 Å) is used for ladder rungs showing H-bonds between bases, and for stubs representing bases that lack H-bonds to other bases. When **ignore** is **true**, any H-bonds involving non-base atoms (on either or both ends) will be shown with a smaller radius corresponding to the smallest ribbon dimension, or if the backbone is not shown as a ribbon, the model stick size.

## Custom Slab Styles

### [Usage:](#)

```
nucleotides add custom-style anchor sugar|base purine x1 y1 x2 y2
pyrimidine x1 y1 x2 y2
```

**Usage:****nucleotides delete *custom-style***

A *slab style* refers to the slab dimensions and location relative to the atoms in purine and pyrimidine bases. The settings define a virtual rectangle in the plane of the base, as shown in the schematic diagrams in the **Slab Style** section of the [Nucleotides](#) dialog. Ellipsoid slab shapes are drawn to enclose the the virtual rectangle.

The **add** keyword of the **nucleotides** command allows defining a new custom slab style or modifying one that already exists, where *custom-style* is the name of the style. Custom slab styles are saved in the Chimera [preferences file](#). The **delete** keyword allows removing a style from the [preferences file](#). All arguments are mandatory for adding or modifying a style, but **anchor**, **purine**, and **pyrimidine** can be given in any order. Example:

```
nuc add square pur 0 -3.5 3.5 0 pyr 0 -2.5 2.5 0 anchor base
nuc side tube/slab style square orient false
```

## Details:

**anchor** sugar | base

Indicate whether the point of reference for the virtual rectangle should be at the sugar end of the glycosidic bond (atom C1') or at the base end.

**purine** *x1 y1 x2 y2*

For purine bases, indicate virtual rectangle position (*x1* and *y1* specify the lower-left corner, *x2* and *y2* specify the upper right corner) in Å from the anchor; positive values are rightward along the X-axis and upward along the Y-axis in the [standard reference frame](#).

**pyrimidine** *x1 y1 x2 y2*

For pyrimidine and pseudopyrimidine bases, indicate virtual rectangle position (*x1* and *y1* specify the lower-left corner, *x2* and *y2* specify the upper right corner) in Å from the anchor; positive values are rightward along the X-axis and upward along the Y-axis in the [standard reference frame](#).

objdisplay

[Chimera Commands Index](#)

**Usage:**

**objdisplay** [atom-spec](#)

**Usage:**

**~objdisplay** [atom-spec](#)

**Usage:**

**objdisplay** *model\_number(s)*

**Usage:**

**~objdisplay** *model\_number(s)*

The **objdisplay** command controls the model-level display of *nonmolecular* [surface models](#) and [volume and VRML models](#). The command [modeldisplay](#) has the same function but additionally affects molecule and [molecular surface](#) models.

If the [atom-spec](#) is blank, all open objects will be affected. If an [atom-spec](#) is given, objects with model numbers corresponding to the specified atoms will be displayed. Alternatively, objects can be specified with one or more model numbers or ranges of model numbers separated by commas (with or without a preceding #).

Examples:

```
objdisplay 1
```

- show the object associated with model number 1

```
objdisplay #1,5-8
```

- show the objects associated with model numbers 1 and 5 through 8

```
~objdisplay :/isHelix
```

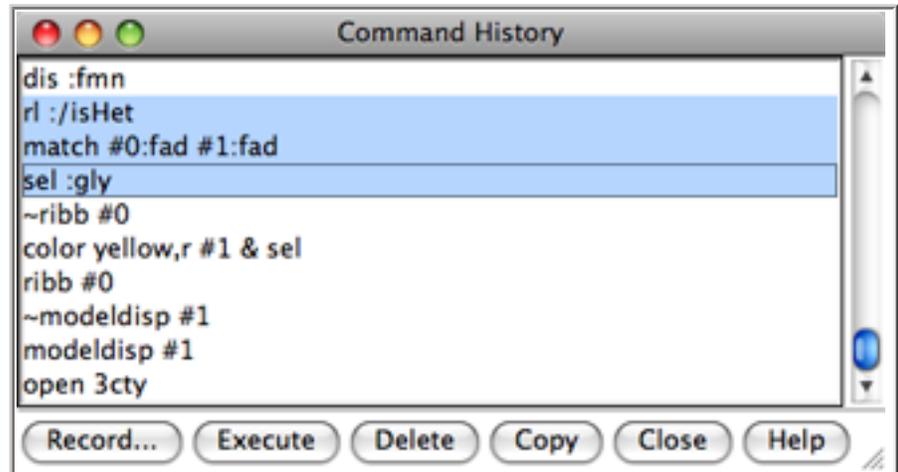
- hide any objects sharing a model number with amino acid residues in helices

See also: [display](#), [modeldisplay](#), [surface](#), the [Model Panel](#)

## Command History

The **Command History** lists previously used [commands](#). The commands can be re-executed and/or saved to a file. The **Command History** can be opened from the pulldown menu marked by a black triangle to the right of the [Command Line](#). The menu also shows the most recently entered commands. Commands are shown as entered even if they failed to execute due to mistyping or incorrect syntax.

In the **Command History**, individual commands or blocks of commands may be chosen (highlighted) using the left mouse button. **Ctrl-click** adds to an existing choice rather than replacing it. To choose a block of commands without dragging, click on the first (or last) and then **Shift-click** on the last (or first) in the desired block. The choice of a single command, but not a block of commands, can be moved up and down the list using the keyboard **up arrow** and **down arrow** or **Ctrl-p** and **Ctrl-n**. In the figure, three commands are chosen.



**Record...** brings up a [dialog](#) for saving or appending commands to a [file](#). The chosen commands or all commands in the history can be saved, as Chimera commands or their [Python translations](#).

**Execute** executes the chosen commands in the order shown; double-clicking an individual command executes that command. **Delete** removes the chosen commands from the history. **Copy** copies the chosen commands as plain text that can be pasted into another application window.

**Close** dismisses the **Command History** window; **Help** opens this manual page in a browser window.

Recently used commands are saved upon exiting from Chimera and will appear in the **Command History** of the next session that uses the same [preferences file](#). How many commands to remember between sessions is specified in the [Command Line preferences](#).

## Python Translations of Commands

A file containing the Python translations of commands cannot be executed in Chimera until statements to import any necessary modules have been added. In some cases, adding

```
import Midas
```

to the beginning of the file will be sufficient; however, additional modules may need to be imported, depending on the particular commands included and any further code that is added to the file.

A Python script can be executed by simply [opening](#) it in Chimera, or if it has command-line arguments,

using the command [runscript](#) or the startup option [--script](#).

---

UCSF Computer Graphics Laboratory / February 2010

## Chimera Command Usage Conventions

In the **Usage** sections of [Chimera command](#) manual pages:

- **keywords** to be taken literally are not italicized
- *parameters* that should be replaced with an appropriate value (such as a number or word) are italicized
- *atom-spec* refers to an [atom specification string](#), which generally specifies "all" when left blank
- optional keywords and parameters are referred to as *options* or are enclosed in square brackets: [ ]
- mutually exclusive choices are separated by a vertical bar: |
- mutually exclusive choices of a required item are grouped with parentheses: ( )

The brackets, vertical bars, and parentheses mentioned above are merely symbolic and should not be included in the typed command.

All commands can be truncated to unique strings. Keyword truncation is not allowed unless a command's manual page states otherwise.

Multiple commands can be combined on a single line, separated by semicolons. This may be useful because the display is not redrawn until the last command in a semicolon-separated series has been executed; that is, multiple graphical changes such as changes in representation, color, and display state can be made to appear simultaneously instead of one by one.

## Chimera Keyboard Shortcuts (Accelerators)

Keyboard shortcuts are disabled by default, but can be enabled:

- with the command [ac](#)
- by starting [Keyboard Shortcuts](#) (for example, with **Tools... General Controls... Keyboard Shortcuts**) and turning on **Enable keyboard shortcuts**; this also enables shortcuts automatically in subsequent uses of Chimera

The following keyboard shortcuts are included with Chimera (and more can be [defined](#)):

Visualization Framework	Molecules	Volume Data
<a href="#">Opening, Saving, Closing</a>	<a href="#">Display</a>	<a href="#">Opening, Saving, Closing</a>
<a href="#">Move Viewpoint or Models</a>	<a href="#">Coloring</a>	<a href="#">Appearance</a>
<a href="#">Background Color and Effects</a>	<a href="#">Zones and Selections</a>	<a href="#">Surface Display</a>
<a href="#">Selecting Objects</a>	<a href="#">Comparing Domain Orientation</a>	<a href="#">Measure Area, Volume, Length, ...</a>
<a href="#">Model Display and Clipping</a>	<a href="#">Multimers and Symmetry</a>	<a href="#">Segmenting and Filtering</a>
<a href="#">Commands, Mouse, Messages</a>		<a href="#">Fit or Build Models</a>

### Opening, Saving, Closing

op	<a href="#">Open file</a>
os	Open <a href="#">session</a>
ff	<a href="#">Fetch file</a> from web
ls	Open last session
ol	Open last file
ok	Open 2nd to last file
oj	Open 3rd to last file
o2	Open last 2 files
o3	Open last 3 files
lo	Show names of files last opened (via <a href="#">dialog</a> or accelerator only)
si	<a href="#">Save image</a>
Ss	Save <a href="#">session</a>
ss	Save session as
Ca	Close all models
Cs	Close session
Qt	Quit Chimera

### Move Viewpoint or Models

va	View all models
so	Standard orientation
fo	<a href="#">Focus</a>
x9	Turn 90 degrees about x axis
y9	Turn 90 degrees about y axis
z9	Turn 90 degrees about z axis
sv	Show <a href="#">Side View</a> dialog
cr	Set center of rotation ( <a href="#">pivot</a> )
a0	<a href="#">Activate</a> model 0 (toggle)
a1	Activate model 1 (toggle)
a2	Activate model 2 (toggle)
a3	Activate model 3 (toggle)
aa	Activate all models
ao	Activate only selected models
ar	Inactivate active models and activate inactive models

at	Activate all models and remember which were inactive; used again, inactivates remembered models
Op	Original model positions ( <a href="#">reset default</a> )
wt	Write relative transformation matrices
na nd nf nz	<a href="#">SpaceNavigator</a> <sup>TM</sup> (3D mouse) settings

### Background Color and Effects

bk	Set background to black
wb	Set background to white
dc	Toggle <a href="#">depth cueing</a>
sh	Toggle <a href="#">shadows</a>
se	Toggle <a href="#">silhouettes</a>
3d	Toggle sequential stereo <a href="#">camera mode</a>

### Selecting Objects

sa	Select all
cs	Clear selection
is	Invert selection (selected models)
iS	Invert selection (all models)

### Commands, Mouse, Messages

cl	Show <a href="#">Command Line</a>
oc	Type one command
hc	Hide <a href="#">Command Line</a>
af	Disable accelerators
ad	Show <a href="#">Keyboard Shortcuts</a> dialog
rl	<a href="#">Reply Log</a>
rt	Report graphics frame rate
ps	Python shell ( <b>IDLE</b> )
pf	<a href="#">Preferences dialog</a>
mm	<a href="#">Mouse preferences</a> (mouse button assignments)
sl	Show/Hide <a href="#">status line</a>
ug	<a href="#">User's Guide</a>

### Model Display and Clipping

mp	Show <a href="#">Model Panel</a> dialog
kl	Show <a href="#">Per-Model Clipping</a> dialog
ca	Toggle <a href="#">clipping</a> mouse mode for selected model
cc	Toggle <a href="#">per-model clipping</a> of selected and displayed models
cv	Toggle <a href="#">per-model clipping</a> of <a href="#">current set</a> of volume data
or	Toggle between orthographic and perspective <a href="#">projection</a>

### Molecule Display

da	Display atoms
ha	Hide atoms
Da	Delete atoms and bonds
wr	Wire representation
st	Stick representation
sp	Sphere representation
bs	Ball & stick representation
bb	Backbone only
ct	Show chain trace only
sx	Show side chains only
rr	Round ribbon
re	Edged ribbon
rf	Flat ribbon
hr	Hide ribbon
rh	Hide ribbon
sf	Show surface
sF	Surface selected atoms
hs	Hide surface

**Molecule Coloring**

c2	Color ribbons <a href="#">by secondary structure</a>
ce	Color atoms <a href="#">by element</a>
ra	Show <a href="#">Render by Attribute</a> dialog
rc	<a href="#">Rainbow chains</a>

**Comparing Domain Orientation**

<a href="#">ab</a>	Superimpose two structures using selected backbone atoms with matching residue/chain IDs
<a href="#">ai</a>	Show transformation between two structures using selected backbone atoms with matching residue/chain IDs

**Opening, Saving, Closing Volume Data**

ov	Open volume
vs	Show volume
fv	Show full volume
vh	Hide volume
vR	Remove volume
vv	Show <a href="#">Volume Viewer</a> dialog
wg	Write GRASP surface file
xs	<a href="#">Export scene</a>

**Volume Appearance**

ob	Toggle volume outline box display
px	Toggle solid style pixel display

**Molecule Zones and Selections**

sc	Select connected atoms/bonds
c3	Find 3-Å contacts between selected and unselected atoms
c5	Find 5-Å contacts between selected and unselected atoms
ri	Select residue interval
rn	Select next residue
rp	Select previous residue
zd	Show <a href="#">zone dialog</a>
zn	Select zone using <a href="#">zone dialog</a> settings

**Multimeric Molecules and Symmetry**

bu	Show molecule biological unit using <a href="#">Multiscale Models</a>
mB	Set molecule PDB BIOMT matrices for selected <a href="#">multiscale</a> chains ( <a href="#">details...</a> )
uc	Show <a href="#">Unit Cell</a> dialog
xc	Extend <a href="#">multiscale</a> selection to sequence copies

**[Surface](#) Display**

cm	Show <a href="#">Surface Color</a> dialog
co	Color selected surfaces
cp	Show <a href="#">Surface Capping</a> dialog
Ds	Delete selected surfaces
fs	Show selected surfaces in filled style
ms	Show selected surfaces using mesh style
Sc	Split selected surfaces into connected pieces (see also <a href="#">sop split</a> )
sz	Resize selected surfaces with mouse (one drag with button 3)
ts	Toggle surface selectability
t0	Make selected surfaces 0% transparent
t5	Make selected surfaces 50% transparent

**Segmenting and Filtering Volume Data**

dd	Delete selected <a href="#">segmentation regions</a>
eb	Erase volume data inside <a href="#">subregion selection</a> box
es	Erase volume data inside <a href="#">sphere</a>
eo	Erase volume data outside <a href="#">sphere</a>
ez	Copy volume with zeros inside <a href="#">zone</a>
FT	<a href="#">Show Fourier transform</a>
gg	Group selected <a href="#">segmentation regions</a>
<a href="#">lm</a>	Invert map values
LT	<a href="#">Show volume Laplacian</a>
rh	Hide selected <a href="#">segmentation regions</a>
rs	Shown only the selected <a href="#">segmentation regions</a>
sm	<a href="#">Split map by color zone</a>
<a href="#">u8</a>	Interpret MRC signed 8-bit map as unsigned
uu	Ungroup selected <a href="#">segmentation regions</a>
vm	Mask volume inside selected surfaces
wv	Make writable copy of volume data
zb	Zero volume boundary
zB2	Zero volume boundary for step size 2
zB4	Zero volume boundary for step size 4
zv	Copy volume with zeros outside <a href="#">zone</a>

**Measure Area, Volume, Length ...**

ma	<a href="#">Measure area</a> of selected surfaces
md	<a href="#">Measure mean, standard dev, rms</a> of volume data
mv	<a href="#">Measure volume</a> of selected surfaces
mw	Report molecular weight of selected atoms
pL	Show total length of selected bonds for each model
pl	Show total length of selected bonds
<a href="#">sd</a>	Measure distance from selected atoms/markers to surface, assign atom <a href="#">attribute distance</a>

**Placing Models in Volume Data**

fr	Rotate model to <a href="#">maximize density</a> at selected atoms
ft	Move model to <a href="#">maximize density</a> at selected atoms
oa	Find selected atoms outside contour surface
bv	<a href="#">Extend periodic map</a> to cover selected atoms
vp	Show <a href="#">Volume Tracer</a> dialog
dp	Display path tracer markers
hp	Hide path tracer markers
mz	Place marker at (0,0,0) in local coordinates of selected models, in global coordinates if no models selected
mc	Place marker at center of rotation
mC	Place marker at center of selected atoms
mk	Place marker under mouse
mS	Place markers on selected atoms

**Notes**

**ab** and **ai** - superimpose and compare backbone segments that have the same sequences, numbering, and chain identifiers.

Atoms from exactly two models must be [selected](#), and atoms other than CA (proteins) and P (nucleic acids) are ignored. Only atoms with the same name, residue type, residue number, and chain identifier are paired; **ab** performs a least-squares fit and applies the resulting transformation, whereas **ai** does not apply the transformation but shows it with two rectangular slabs colored to match the structures. Both **ab** and **ai** report the RMSD, number of atom pairs, and angle of rotation in the [status line](#) and [Reply Log](#).

One application is to measure changes in relative domain orientation in different conformations of a multidomain protein. For example, the thioredoxin reductase structures 1f6m and 1tde differ by a rotation of one domain relative to the other. They are different conformations of essentially the same protein and are numbered in the same way (chains other than A can be deleted). After selection of one domain (approximately residues 1-117, 245-320) in both structures, **ab** could be used to superimpose that domain. The selection could then be [inverted](#) to encompass the other domain (approximately residues 118-244) in both structures and **ai** used to compute the transformation, in this case a rotation of  $\sim 67^\circ$ . This conformational change is described in [Lennon \*et al.\*](#), Science 289:1190 (2000). The two accelerators do not have to be used together; the first domain could be superimposed manually or with [MatchMaker](#) before **ai** is used to compare the orientations of the second domain. See also: [measure rotation](#)

**bv** - Extend periodic map to cover selected atoms.

Creates a new map that covers the currently [selected](#) atoms plus 5 Å padding on all sides. The map is derived from the [current set](#) of data (active map) in [Volume Viewer](#). If the atoms plus padding extend beyond the bounds of the active map, then it is assumed that the active map is periodic along all three axes (such as a unit cell crystallographic density map). The new map is displayed using the thresholds and colors of the original map and the original map is undisplayed. See also: [vop cover](#)

**Im** - Invert map values.

If the map value type is signed (e.g. 32-bit float or signed 16-bit integer), each value is multiplied by -1. For unsigned 8-bit maps, the values are multiplied by -1 and 255 is added so that the values remain unsigned. For other unsigned maps, the values are multiplied by -1 and the maximum map value is added so that the map remains unsigned. The accelerator acts on the [current set](#) of data in [Volume Viewer](#). A copy of the map is made unless the map is itself a copy (for example, made using **wv**, **zb**, or [Volume Eraser](#)). The original file is not modified. Use the volume dialog menu ([File... Save map as...](#)) to save the inverted map.

**sd** - Measure distance from selected atoms/markers to surface.

The distance from each [selected](#) atom or [path tracer marker](#) to each displayed surface is printed in the [Reply Log](#). Example:

```
Distance from #0:18.water@0 to surface MSMS main surface of 1a0m.pdb
d = 2.09, surface point (3.14, -1.28, 7.85), side 1
```

Whether a surface is displayed is evaluated at the levels of [surface model](#) and [surface piece](#) (but not at the per-atom level of a [molecular surface](#)).

The coordinates of the nearest surface point and the side of the surface that the point lies on (+1 = outside, -1 = inside) are given. The closest surface point may lie within a triangle, on a triangle edge, or at a triangle vertex of the triangulated surface. If multiple surface points are equidistant, only one is reported. The calculation has been implemented in C++ for better performance. Each selected atom's distance to the surface last measured is assigned as the atom [attribute](#) named **distance**. To remove any ambiguity in interpreting the attribute values, it is best to perform the measurements with only a single surface present.

**u8** - Interpret MRC signed 8-bit map as unsigned.

The MRC volume file format does not support unsigned 8-bit map values. Some EM tomography programs use it to hold unsigned 8-bit values (0-255) with the data type in the file header incorrectly indicating that the values are signed 8-bit (-128 to 127). Use the u8 accelerator to reinterpret the data values as unsigned 8-bit. The original file is not modified. The accelerator acts on the [current set](#) of data in [Volume Viewer](#). It only works on MRC format maps.

---

UCSF Computer Graphics Laboratory / September 2014

**[Usage:](#)**

**pause [ end ]**

The **pause** command without arguments suspends [command script](#) execution. During the resulting pause, pressing the **End** or **Esc** key will abort script execution, whereas pressing any other key will resume execution. The **end** keyword indicates aborting immediately, as if the **End** key had been pressed.

Alternatively (without use of the **pause** command), command script execution can be paused/resumed by pressing **Shift-Esc** or aborted by pressing **Esc**.

The pause will not be included in any [movie](#) that is being recorded, because no new frames are drawn during that time; to generate pauses in a movie, [wait](#) should be used instead.

See also: [sleep](#)

**Usage:**

**pdb2pqr** [ **molecule** [atom-spec](#) ] [options](#)

Like the Chimera [PDB2PQR](#) tool, the **pdb2pqr** command runs [PDB2PQR](#), which prepares structures for further calculations by reconstructing missing atoms, adding hydrogens, assigning atomic charges and radii from specified force fields, and generating [PQR](#) files. A primary use is to prepare structures for [apbs](#) (Adaptive Poisson-Boltzmann Solver). The process can use either a web service provided by the [National Biomedical Computation Resource \(NBCR\)](#) or a locally installed copy of the program. Users should cite:

[PDB2PQR: expanding and upgrading automated preparation of biomolecular structures for molecular simulations.](#) Dolinsky TJ, Czodrowski P, Li H, Nielsen JE, Jensen JH, Klebe G, Baker NA. *Nucleic Acids Res.* 2007 Jul;35(Web Server issue):W522-5.

[PDB2PQR: an automated pipeline for the setup of Poisson-Boltzmann electrostatics calculations.](#) Dolinsky TJ, Nielsen JE, McCammon JA, Baker NA. *Nucleic Acids Res.* 2004 Jul 1;32(Web Server issue):W665-7.

If more than one molecule model is present, the [molecule](#) option should be used to specify which to act upon. Results are opened as a new model in Chimera, with **charge** and **radius** [attributes](#) assigned to the atoms. \*\* Any residues not handled by the designated [force field](#) will be omitted. Conversely, any unwanted residues such as waters should be [deleted](#) beforehand to ensure they do not appear in the result. \*\*

Although [Dock Prep](#) has a similar function, the overlap is only partial and even the seemingly shared functions (*e.g.*, repairing truncated sidechains, adding hydrogens) are done differently. It may be useful to run certain parts of [Dock Prep](#) beforehand, for example, to delete [solvent](#). However, only the charge and radius assignments from **pdb2pqr**, not those from Chimera or other Chimera tools, can be written to a [PQR](#) file. See also: [addh](#), [addcharge](#), [findhbond](#), [coulombic](#)

**Options**

Option keywords for **pdb2pqr** can be truncated to unique strings and their case does not matter. A vertical bar “|” designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

**molecule** [atom-spec](#)

Limit the calculation to the specified model (the entire molecule model containing the specified atoms). Only one model should be specified. If [atom-spec](#) includes any spaces, it must be enclosed in single or double quote marks.

**forcefield** *ff*

Which charge and radius parameters to use, where *ff* can be:

- **parse** (default) - **PAR**Ameters for **S**olvation **E**nergy ([Sitkoff, Sharp, and Honig](#), *J Phys Chem* **98**:1978 (1994) and [Tang et al.](#), *J Mol Biol* **366**:1475 (2007))

- **amber** - AMBER ff99 ([Wang, Cieplak, and Kollman](#), *J Comput Chem* **21**:1049 (2000))
- **swanson** - AMBER ff99 charges with optimized radii ([Swanson et al.](#), *J Chem Theory Comput.* **3**:170 (2007))
- **charmm** - CHARMM27 ([MacKerell et al.](#), *J Phys Chem B* **102**:3586 (1998))
- **peoepb** - a version of Gasteiger-Marsili Partial Equalization of Orbital Electronegativities, optimized for Poisson-Boltzmann calculations ([Czodrowski et al.](#), *Proteins* **65**:424 (2006))
- **tyl06** - a Poisson-Boltzmann-optimized force field ([Tang, Yang, and Luo](#), *J Phys Chem B* **110**:18680 (2006))

Note: residues not handled by the designated force field will be omitted.

#### **pqr file**

Pathname (name and location) of the output [PQR](#) file; if not specified, a temporary name and location will be used.

#### **propkaph pH**

Use [PROPKA](#) (version 3.0) to predict the pKa values of ionizable groups in protein at the specified *pH*. Users should cite:

[Improved treatment of ligands and coupling effects in empirical calculation and rationalization of pKa values.](#) Søndergaard CR, Olsson MHM, Rostkowski M, Jensen JH. *J Chem Theory Comput.* 2011;7(7):2284-95.  
[Very fast empirical prediction and rationalization of protein pKa values.](#) Li H, Robertson AD, Jensen JH. *Proteins.* 2005 Dec 1;61(4):704-21.

#### **neutralN true | false**

Whether to make the protein N-terminus neutral (only available for the PARSE [force field](#)).

#### **neutralC true | false**

Whether to make the protein C-terminus neutral (only available for the PARSE [force field](#)).

#### **debump true | false**

Whether to ensure that new atoms are not rebuilt too close to existing atoms [[details](#) at the PDB2PQR site].

#### **optHbond true | false**

Whether to adjust hydrogen positions and flip certain sidechains (His, Asn, Glu) as needed to optimize hydrogen bonds [[details](#) at the PDB2PQR site].

#### **distCutoff d**

Hydrogen bond distance cutoff (default 3.4 Å).

#### **angleCutoff a**

Hydrogen bond angle cutoff (default 30°).

**hbonds true | false**

Whether to send hydrogen bond information to the [Reply Log](#).

**ligands true | false**

Whether to use the PEOEPB approach (see [above](#)) to assign charges to any [ligand](#) residues, after protonation and conversion to Mol2 format (by Chimera) as currently required by the program [[details](#) at the PDB2PQR site]; the residue(s) will be renamed LIG and placed in chain L.

**apbs true | false**

Whether to generate an example [APBS](#) control file and show it in the [Reply Log](#); however, such a file is not needed to run the program from Chimera (using the [APBS](#) tool or [apbs](#) command).

**backend opal | local**

Whether to use an [Opal](#) web service (default) or a locally installed executable.

**location opal-URL | local-path**

Depending on the [backend](#) setting, the URL of the web service (default is the URL for the service provided by the [NBCR](#)) or the pathname of the local executable.

**wait true | false**

Whether to wait for the calculation to finish before starting to execute any subsequent commands.

## Usage:

**pdbrun** [ *all* ] [ *conect* ] [ *nouser* ] [ *nowait* ] *command* [*cmd\_args...*]

**Pdbrun** causes *command* and *cmd\_args* to be passed to the user's system shell for execution. A set of concatenated PDB files describing the current models (hereafter referred to as a PD Brun file) is also passed as standard input to *command*. Normal shell metacharacters (notably output redirection) can be used. Errors are ignored and any shell output is interpreted as commands and executed. For example, on a UNIX system,

```
pdbrun cat > file
```

would create a PD Brun file named *file* that describes the current models (see also the [save](#) and [write](#) commands).

The **all** option specifies that all atoms, not just those that are displayed, should be sent to *command*.

The **conect** [*sic*] option specifies that PDB-standard CONECT records should be generated for all residues, even if they have standard connectivity.

USER records are placed in the PD Brun file to indicate display information such as atom colors, view direction, clipping planes, etc. The **nouser** option specifies that no USER records should appear in the file.

By default, Chimera waits for *command* to exit before continuing. The **nowait** option causes Chimera to continue without waiting for *command* to exit.

Any output from *command* is interpreted as commands and executed. Control returns to the user interface when the **pdbrun** command terminates. The **pdbrun** command facilitates additions to the normal command set. For example, the [conic](#) command is implemented using **pdbrun**.

As mentioned above, a PD Brun file is passed as input to *command*. This file describes not only current model coordinates, but also other aspects of the modeling environment such as clipping plane positions. Modeling environment annotations are supplied in various types of USER records, described in detail in Appendix 1 of the MidasPlus User's Manual.

The PD Brun file has three principal sections:

1. *Annotation Headers* has global modeling information.
2. *Per Molecule Annotations* has molecular data for each model. Each model's data starts with a USER FILE record and is terminated with an END record. The serial numbers in ATOM, HETATM, and TER records increment continuously from one model to the next. Due to field widths, this imposes a limit of 99999 atoms in a scene. Any PDB records present in the model's original source file (and not normally interpreted for display purposes, such as REMARK records) are preserved and placed just after the USER FILE record.
3. *Annotation Trailers* contains records specifying the angles and distances being explicitly monitored in the current view.

Throughout, fields in PDB records that are not wide enough to hold their associated values are filled with asterisks instead. Details of the meaning and format of the `USER` records employed by `pdbrun` can be found in Appendix 1 of the MidasPlus User's Manual. However, as there are no Midas-style nonmolecular objects in the framework of the Midas emulator in Chimera, PDBRUN files created therein will not contain any `USER OBJ`, `USER ENDOBJ`, or `USER GFX` records.

A discussion of the motivation for and design criteria of the PDBRUN format can be found in: G. S. Couch, E. F. Pettersen, C. C. Huang and T. E. Ferrin "Annotating PDB files with scene information" *J. Mol. Graphics* **13**, 153-158 (1995).

See also: [runscript](#), [save](#), [system](#), [write](#), [export](#), [writesel](#)

**Usage:**

**perframe** *operation* [ **range** *start,end[,step]* ]<sub>M</sub> [ **frames** *N* ] [ **interval** *K* ]  
 [ **zeroPadWidth** *width* ] [ **showCommands** true|false ]

**Usage:**

~perframe

The **perframe** command specifies operations to be executed at each subsequent display frame, until a specified number of frames has elapsed, the end of a value range has been reached, or all per-frame operations have been discontinued collectively with ~**perframe**. Multiple per-frame operations can be active simultaneously. See the [video mini-examples](#). See also: [set maxFrameRate](#), [movie-related commands](#), [per-frame scripts](#) in [MD Movie](#)

A per-frame *operation* consists of one or more [Chimera commands](#), usually with embedded substitution arguments (*\$1*, *\$2*, etc.). If multiple commands are included, they should be combined into a single line with semicolon separators. If *operation* contains any spaces, it should be enclosed in double quotes. Alternatively, *operation* can be a previously defined single-word alias (defined using the [alias](#) command, with ^ before the name to indicate a beginning-of-line alias). During execution, substitution arguments in *operation* are replaced by the values of variables described in **range** specifications and/or a frame counter that starts at 1.

The number of frames *N* at which to perform *operation* can be indicated directly with the **frames** option or indirectly with **range**, where the substituted value will proceed from *start* to *end* in increments of *step*. If *step* is omitted, it will be calculated from the number of **frames** *N* (if supplied), otherwise assumed to be +1 if *start* < *end*, or -1 if *start* > *end*. The number of frames specified directly will win if it conflicts with the number of frames calculated from range information. If the end of the range is reached before the frames are complete, the substituted value will stay the same (equal to *end*) for the remaining frames. A range can be given for each substitution argument in *operation*, and multiple ranges will be used in order. If no range is given, the frame count will be substituted for \$1.

The **interval** option allows executing *operation* at every *K*<sup>th</sup> frame instead of every frame. Execution will start at the first frame and occur *N* times regardless of the interval *K* (where *N* is specified directly or indirectly, as explained above), but the total number of frames to elapse will vary:  $K(N - 1) + 1$ . The frame count for substitution purposes will only include frames at which *operation* is executed, from 1 to *N*.

The **zeroPadWidth** option allows padding the substituted value with leading zeroes up to *width* digits; this is typically used to generate output filenames from frame counts.

The **showCommands** option (default **false**) can be used to echo each expanded command to the [Reply Log](#) for debugging purposes.

Option keywords for **perframe** can be truncated to unique strings and their case does not matter.

Examples:

perframe

The following saves X3D files named 001.x3d, 002.x3d, ... 180.x3d as a structure is rotated in 2° increments about the Y axis:

```
alias ^getx3d export $1.x3d  
perframe getx3d zeropad 3; roll y 2 180; wait; ~perf
```

The following saves PDB files named 001.pdb, 002.pdb, ... 360.pdb as a structure is rotated in 1° increments about the Y axis:

```
perframe "turn y 1; write #0 ~/Desktop/$1.pdb" frames 360 zero 3
```

The following gradually changes a map contour level:

```
perframe "volume #0 level $1" range 2.5,0.5 frames 50
```

The following displays individual sidechains (hiding all others) along a protein ribbon, going to the next sidechain every 10<sup>th</sup> frame:

```
ribbon  
perframe "~disp protein; disp :$1" range 17,355 interval 10
```

The following was used to “walk” a DNA-binding motor protein along a DNA molecule by superimposing the backbone of template DNA in the protein structure onto successive segments of the longer DNA:

```
perframe "match #0:1-5.T@P #1:$1-$2.A@P" range 3,85 range 7,89
```

The following re-evaluates hydrogen bonding between two molecules as they are gradually changed from a previously saved position named **undocked** to a previously saved position named **docked**.

```
alias ^evalhb hb intramod false line 2  
reset undocked  
perframe evalhb; reset docked 40; wait; ~perf
```

**Usage:****play operation arguments**

The **play** command performs complex motions for creating animations. The *operation* can be:

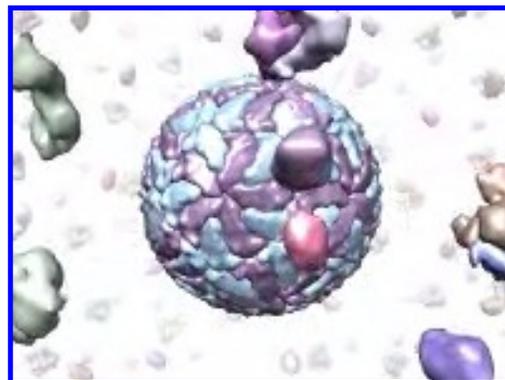
- [radial](#) - move [multiscale](#) chain surfaces radially
- [wave](#) - move [multiscale](#) chain surfaces from an initial to a final state
- [wiggle](#) - wiggle branches of a molecule
- [zipper](#) - move residues from an initial to a final state sequentially

See also: [reset](#), [fly](#), [morph](#), [coordset](#), [sym](#), [Multiscale Models](#), [movie-related commands](#)

Operation keywords and their sub-keywords described below can be truncated to unique strings, and their case does not matter. Synonyms for true: True, 1. Synonyms for false: False, 0. A vertical bar "|" designates mutually exclusive options, and default settings are indicated with **bold**.

- **play radial** *surfaces* [ **[factor]** *f* ] [ **[frames]** *N* ]

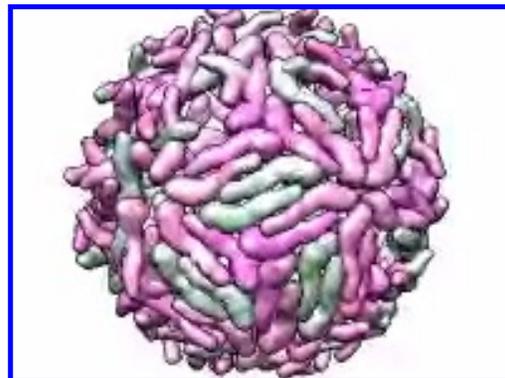
Move [Multiscale Models](#) chain surfaces radially from their collective center. The center of each chain surface is moved radially from the collective center to a final distance of *f* times the starting distance (default is expansion by a factor of **2.0**) over *N* frames (default **25**). Both keywords can be omitted, although if the **factor** keyword is used, the **frames** keyword should also be used if *N* is specified. This command was developed to peel off outer layers of proteins from virus capsid models. Examples of surface specifiers: #0, #1:1-30, #2: A, .C, #3: A-H, #4: 31-60.S-V



[Example animation](#)

- **play wave** *model1 model2 distanceStep* [ **[frames]** *N* ] [ **pairChains** *p* ] [ **groupChains** *g* ] [ **equivalentChains** *s* ] [ **pairingMethod** **match** | **push** | **pull** | **pushpull** ]

Move [Multiscale Models](#) chain surfaces from an initial position (*model1*) to a final position (as in *model2*) over *N* frames (default **25**). The *model1* surfaces are moved. The motion begins with the chains with loaded atoms and propagates outward in layers, where a layer consists of chains within *distanceStep* of chains in the previous layer. Chains in the initial state can be paired with "destination" chains in the final state in a way that minimizes the overall motion. This command was developed for showing the conformational rearrangement of dengue virus capsid proteins during virus maturation.



[Example animation](#)

Paired chains must contain identical sets of atoms because the motion is computed by minimum-RMSD match between corresponding atom positions. By default, pairing is by matching chain identifiers. Alternatively, the **pairChains** option can be used to specify pairing directly. For example, "pairChains A=S,B=T,C=U" identifies chains A,B,C in the initial state as matching chains S,T,U (respectively) in the final state. The **groupChains** option allows moving multiple chains as a single rigid object; for example, "groupChains A+D,B+E,C+F" makes chains A and D move as a single unit, likewise for B and E, and C and F. The

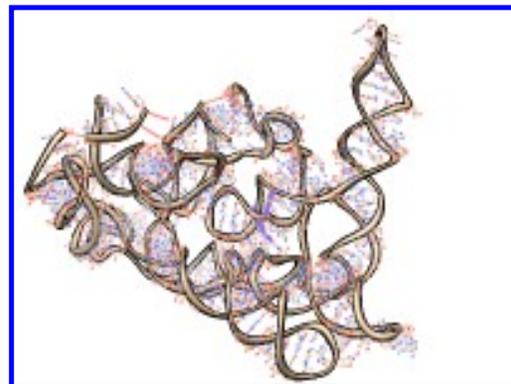
**equivalentChains** option lists chains (or groups) in the initial state that are equivalent to each other for purposes of pairing initial state and final state chains. For example, “equivalentChains A=B=C” indicates that chains A, B, and C are all interchangeable when pairing with final state chains. If A+D,B+C,C+F are grouped, “equivalentChains A+D=B+E=C+F” can be used to indicate that the three groups are equivalent for pairing purposes.

**Multiscale Models** typically generates multiple symmetry-related copies of individual chains. The copies of a given chain cannot be distinguished by chain ID, so for these, a **pairingMethod** is applied:

- **match** (default) - matches chain instance numbers, *e.g.*, chains 1-60 match 1-60, in order
- **push** - within a layer, matches each initial state chain to the nearest unpaired final state chain (that with the smallest maximum distance between corresponding atoms of the initial and final states). Initial state layers are used. These are defined as all chains with any atom within *distanceStep* of any atom in the previous layer, measured in the initial state.
- **pull** - within a layer, matches each final state chain to the nearest initial state chain that is not yet paired. Final state layers are used. The initial state layer is identified, and then the final state layer is defined as all chains (not in previous final state layers) that in the final state fall within *distanceStep* of chains in the initial state layer and all previous initial state layers in their initial positions.
- **pushpull** - matches by push-pairing the chains in the initial state layer, then pull-pairing any as yet unpaired chains in the final state layer; compared to **push** or **pull** alone, this may reduce excessively large movements

- **play wiggle** *atom-spec* *branches b* [ [frames] *N* ] [ *angle a* ] [ *speed s* ]

Wiggle branches of a molecule back and forth, where a branch is a contiguous range of residues. Each branch is rotated about an axis defined by its two end atoms, the atoms at each end of the branch that are directly bonded to residues outside the branch. The angle of rotation from the original position varies sinusoidally with time, and the speed of motion of a branch is inversely proportional to the number of atoms in the branch. This command was developed to [illustrate flexibility](#) of the 9000-nucleotide HIV RNA genome.



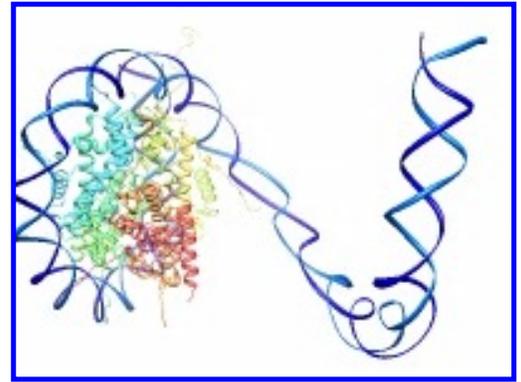
[Example animation](#)

The *atom-spec* delimits the scope of branch definitions (for example, to a specific model or chain). The **branches** argument *b* is a comma-separated list of residue ranges, each range defining a branch, for example, “branches 35-85,53-65,121-290”. The **angle** option specifies the maximum rotation in degrees (default 10). The number of frames for one oscillation of a given branch is first estimated by dividing the number of atoms in the branch by the **speed** parameter *s* (default 25), then adjusted to the nearest value that makes a half-integral number of cycles complete in the specified total number of frames *N* (default 25), so that when the wiggle ends, all atoms will be in their original positions.

- **play zipper** *residueList1 residueList2* [ [spacing] *d* ] [ [step] *s* ]

Interpolate atom positions from an initial state to a final state, starting at one end of a chain of residues and progressing over time to the other end, much like the action of closing a zipper. This command was originally used to make an [HIV RNA movie](#) showing RNA being synthesized from a DNA template.

Each residue list should contain only residues from a single molecule model, and the two lists should contain equal numbers of residues. Residues are paired in the order in which they are specified; the residue numbers need not match. Residues in the first list are moved. Within paired residues, atoms are paired by matching atom names. Unpaired atoms are not moved. The interpolation fraction  $f$  varies with residue number as a piecewise linear function. If  $f = 0$  represents the initial state and  $f = 1$  the final state, completely moved residues are at  $f = 1$ , subsequent residues fall within a linear ramp down to 0, and residues that have not been moved yet are at  $f = 0$ . The linear ramp is moved along the residues like a sliding window to animate the change. At each frame, the ramp advances by a number of residues equal to the **step**  $s$  (default **1**). The step value can be non-integer; for example, 0.1 could be used to produce a slower motion. Negative step values make the motion begin at the last residue and proceed to the first. The ramp width in number of residues is recalculated at each frame by identifying the residue at the bottom of the ramp and dividing its initial-to-final distance by the **spacing**  $d$  (default **3.0**). The bottom of the ramp is where it first hits  $f = 0$ . If the bottom of the ramp falls between residues, the pro-rated average of the distances of the flanking residues is used.



[Example animation](#)

**Usage:****preset apply** *type* ( *number* | *name* )**Usage:****preset list** [ *type* ]

A *preset* is a predefined combination of display settings. The **preset** command can be used to **apply** a preset or to **list** the available presets in the [Reply Log](#). See also: [alias](#), [represent](#), [ribrepr](#), [ribscale](#), [surface](#), [set](#), [mcopy](#), [copy](#), [tips on preparing images](#)

Presets are grouped by *type*:

- [interactive](#) - for interactive manipulation and analysis; may change which items (atoms, ribbons, surfaces) are displayed and how they are colored. The [background color](#) is set to black and the [ribbon path method](#) to B-spline.
  - **1** (ribbons)
  - **2** (all atoms)
  - **3** (hydrophobicity surface)
- [publication](#) - for generating presentation and publication images; do not change which items are displayed or their colors, but may change their styles. The [background color](#) is set to white.
  - **1** (sihouette, rounded ribbon)
  - **2** (sihouette, licorice)
  - **3** (depth-cued, rounded ribbon)
  - **4** (depth-cued, licorice)

Publication presets may decrease interactive performance because they increase [smoothness](#) by using finer divisions to depict curved objects (ribbons, [molecular surfaces](#), etc.). Individual display parameters are discussed in more detail in the [tips on preparing images](#).

- **custom** - user-defined; directories in which to look for custom preset scripts are specified in the [Presets preferences](#)

The command **preset apply** has the same effect as choosing the corresponding entry from the [Presets menu](#). A specific preset is indicated by *type* and either *number* or *name* (case-dependent). The command **preset list** sends a list of presets of the specified *type* (all types, if not specified) to the [Reply Log](#). The keyword (**apply** or **list**), *type*, and *name* can be truncated to unique strings.

Examples:

```
preset app int all
```

- apply the “all atoms” **interactive** preset

```
preset apply pub 1
```

preset

- apply **publication** preset #1

### **preset list c**

- list all **custom** presets in the [Reply Log](#)

---

UCSF Computer Graphics Laboratory / October 2012

**Usage:****rainbow** [*level*] [*color1,color2, ...*] [atom-spec](#)

The command **rainbow** colors successive residues, chains, models, or protein secondary structure elements over a specified color range.

If [atom-spec](#) is blank, every biopolymer chain will be colored, with the color changing per residue. The default color list is **blue**, **cyan**, **green**, **yellow**, and **red**, in that order. Thus, **rainbow** without any arguments will color the residues in each chain from blue through the rainbow spectrum to red (peptides from N-terminus to C-terminus, nucleic acids from 5' end to 3' end). Although a narrower [atom-spec](#) may be supplied, the entire model(s) containing the specified atoms will be affected. The coloring cannot be limited to a subset of the polymer chains in a model or to a subset of the residues in a chain except as noted below for the *level*:

- **residue** or **residues** (default) - assign a different color to each residue, starting over for each chain; [water and HET](#) without chain IDs are ignored
- **chain** or **chains** - assign a different color to each chain, starting over for each model; [water and HET](#) without chain IDs are ignored
- **model** or **models** - assign a different color to each model
- **sse** - assign a different color to each protein secondary structure element (helix or strand), starting over for each chain; coil (non-helix, non-strand) and nonproteins are ignored
- **helix** or **helices** - assign a different color to each protein  $\alpha$ -helix, starting over for each chain; strand, coil, and nonproteins are ignored
- **strand** or **strands** - assign a different color to each protein  $\beta$ -strand, starting over for each chain; helix, coil, and nonproteins are ignored

Two or more color names *color1*, *color2*, etc. can be supplied as a comma-separated list. Each can be any [color name](#) that specifies a single color, except that R,G,B[,A] tuples cannot be used. If two names are given, they designate the starting and ending colors; if more than two names are given, they designate the starting color, intermediate color(s), and ending color. The [Rainbow](#) tool is a graphical implementation of **rainbow** in which five colors are used.

Apparent color is determined by a [hierarchy](#); briefly, atom colors assigned on a per-atom basis and ribbon colors assigned on a per-residue basis override the model-level color. When coloring per residue or chain, **rainbow** sets individual atom colors and per-residue ribbon colors; when coloring per model, it sets model-level colors and removes any atom-level and ribbon color assignments within the models.

See also: [color](#), [rangecolor](#), [ribbon](#), [ribcolor](#), [scolor](#), [transparency](#), [msc](#), [PipesAndPlanks](#)

**Usage:****ramachandran** [ **assign** ] [atom-spec](#)

The **ramachandran** command shows the distribution of peptide  $\phi, \psi$  angles (a [Ramachandran plot](#)) for each specified model that contains protein. It is the command-line implementation of the [Ramachandran plot](#) function in the [Model Panel](#). A separate plot will be generated for each protein-containing model with any atom in [atom-spec](#), and the plot for a model will contain all of its peptide  $\phi, \psi$  angles regardless of which atoms were specified.

The **assign** keyword indicates assigning an [attribute](#) named **ramaProb** to the amino acid residues, with values taken from the appropriate [dataset](#) for each residue (**Proline** for prolines, *etc.*).

Whether or not a Ramachandran plot is shown, **phi** ( $\phi$ ) and **psi** ( $\psi$ ) are automatically assigned as residue [attributes](#), and as such can be viewed/changed in the [Selection Inspector](#) or changed with [setattr](#). See also: [angle](#), [rotation](#), [swapaa](#)

**Usage:**

( **rangecolor** | **rangecolour** ) *attr\_name*[,a][,r][,s][,v][,l][,la][,lr] [**key**] *value1 color1 value2 color2* [... *valueN colorN*] [atom-spec](#)

**Rangecolor** shows the values of an [attribute](#) such as B-factor or hydrophobicity with color gradations. The attribute is specified by [name](#) (*attr\_name*) and must be numerical (see [examples](#)). The related tool [Render by Attribute](#) provides a graphical interface for coloring by attribute values. See also: [color](#), [colorkey](#), [rainbow](#), [coulombic](#), [scolor](#), [transparency](#), [defattr](#), [setattr](#), [msc](#), [Render by Attribute](#)

At least two value/color pairs must be supplied. Pairs do not need to be ordered by value. Values (*value1*, *value2*, etc.) can be specified directly or with the special words:

value indicator	meaning
<b>min</b>	minimum value among the <a href="#">specified</a> atoms
<b>max</b>	maximum value among the <a href="#">specified</a> atoms
<b>mid</b>	average of min and max
<b>novalue</b>	no value assigned for the attribute

A pair specified with **novalue** does not contribute toward the minimum of two pairs because the corresponding color is not involved in any ranges; it is only used to color items with no value for the specified attribute (for example, nucleic acid residues do not have [kdHydrophobicity](#) values). When an [atom attribute](#) is used to color ribbons, the ribbon segment for each residue is colored according to the average of the values for atoms in the residue. In all cases, the reported **min**, **mid**, and **max** values describe the original attribute rather than any derived quantity such as an average.

Each color name (*color1*, *color2*, etc.) can be any [color name](#) that specifies a single color.

The [atom-spec](#) indicates which atoms (and/or associated labels, surfaces, or ribbon) are to be colored. If no [atom-spec](#) is given, all atoms (and/or their associated labels, surfaces, or ribbon) are affected.

While [atom-spec](#) indicates the scope of coloring, one or more specifiers can be used to indicate which types of items should be colored:

- **a** (or **b**) - atoms (and [normally](#), bond halves are colored to match the flanking atoms)
- **r** - ribbons
- **s** - [molecular surfaces](#)
- **v** - [VDW surfaces](#)
- **l** - atom and residue labels
- **la** (or **al**) - atom labels
- **lr** (or **rl**) - residue labels

In the absence of a specifier, all of the above are affected. Other kinds of surfaces, [2D Labels](#), and bond labels are not affected.

The **key** option brings up the [Color Key](#) tool, filled in with the appropriate colors and values, and set to **Use mouse for key placement** for creating/positioning the color key in the graphics window. The mouse setting can be toggled to allow moving models with the mouse.

In Chimera, visible color is determined by a [hierarchy](#). **Rangecolor** sets atom, atom label, and surface colors at the atom level, and ribbon and residue label colors at the residue level.

Examples:

**rangecol bfactor,s key min blue 20 white 40 red**

- color the molecular surface (only) by the [atom attribute](#) named **bfactor**, from blue for the minimum value, to white for 20, to red for 40 and above; call [Color Key](#), filled in with those values and colors

**range kdHydrophobicity min lime green max orange novalue magenta**

- color from lime green to orange by increasing values of the [residue attribute](#) named [kdHydrophobicity](#); color any residues without hydrophobicity assignments magenta

read

[Chimera Commands Index](#)

**Usage:**

**read** *filename*

**Read** executes a [command file](#) without automatically updating the display until the end. It is intended for executing scripts that accomplish some end result rather than continuous display.

The related command [open](#) automatically updates the display after each line of commands as needed.

See also: [pdbrun](#), [Command History](#)

**Usage:**

**represent** *style* [atom-spec](#)

**Represent** changes the [draw modes](#) of the [specified](#) atoms (and the bonds connecting them) according to *style*. *Style* can be:

- **wire** (default) - wire [representation](#)
- **stick** - stick [representation](#)
- **bs** or **b+s** - ball-and-stick [representation](#); ball radii equal the [VDW radii](#) multiplied by the [ball scale](#) factor
- **sphere** or **cpk** - sphere or CPK [representation](#); sphere radii equal the [VDW radii](#)

[Draw modes](#) can also be changed with the [Actions menu](#), the [molecule model attributes panel](#), and the [Selection Inspector](#).

See also: [linewidth](#), [bondrepr](#), [ribbon](#), [ribrepr](#), [surface](#), [surfrepr](#), [preset](#)

**Usage:**

```
reset [ view_name [ frames ]] [ holdSteady model ] [ moveModels other-models ]
```

**Usage:**

```
reset list
```

**Reset** restores models to a default position or one previously saved with [savepos](#). See the [video mini-example](#). Positions can also be restored from the [Rapid Access](#) interface. See also: [fly](#), [play](#), [window](#), [matrixset](#), [scene](#), [movie-related commands](#)

A *view\_name* is specified when a position is saved with [savepos](#). A *position* includes:

- the transformations of models (their rotations and translations relative to input coordinates)
- the overall scale
- [clipping plane](#) status, locations, and orientations
- the [center of rotation method](#)
- the [horizontal field of view](#)

If no *view\_name* is given, the name **default** is used. Unless changed with [savepos](#), the **default** position is one in which the models have not been rotated or translated.

Saved positions are included in saved [Chimera sessions](#), and a position named **session-start** is created automatically when a session is restored.

**Reset** restores the named position over the specified number of image update *frames* (default **1**). For *frames* > 1, a path from the current position to the named position is generated by interpolation ([details...](#)). Whereas **reset** interpolation considers only a pair of positions at a time (start and end), the related command [fly](#) can generate a smooth interpolated path over a series of multiple saved positions.

Depending on the current and named positions, the camera viewpoint and the transformation of each model could change. The **holdSteady** option allows keeping the viewpoint fixed on a specified model while moving other models to their transformations relative to that model in the named position. The movement can apply to all other models (default) or only those specified with the **moveModels** keyword.

Models may have been closed or opened since the position was saved. How these situations are handled (assuming no **holdSteady** restriction):

- If a model that was included in a saved position is no longer present when the position is restored, everything else will still be restored.
- If some new model has been opened and assigned the same ID number as a model that was closed, the new model (but not any [per-model clipping plane](#)) will be treated as if it were the one included in the saved position.
- Otherwise, new models will be transformed to maintain their positions relative to the lowest-numbered old model.

**Reset list** lists the names of all saved positions in the [status line](#) and [Reply Log](#).

**Interpolation details.** All models with the same relative transformation (rotation and translation) between the initial and final positions are grouped together as a rigid unit. The center of the bounding sphere of the group will move along a straight line between the initial and final positions, while the group rotates from the initial orientation to the final orientation about a fixed axis (and that center) at a constant angular rate.

**Usage:**

**resrenumber** *start* [atom-spec](#)

**Resrenumber** assigns new residue numbers to residues with any atom in [atom-spec](#). It is the command version of [Renumber Residues](#). The specified residues from *each* chain will be renumbered starting from the integer *start*. In other words, numbering will start over for each chain rather than incrementing from the previous chain.

If the renumbering would produce one or more duplicate residue numbers within a chain, an error message will appear and the numbering will not be applied.

See also: [changechains](#), [write](#)

**Usage:****ribbackbone** [atom-spec](#)**Usage:****~ribbackbone** [atom-spec](#)

By default, when [ribbon](#) is shown, [mainchain atoms](#) (backbone atoms) are hidden and their locations are adjusted to fall on the ribbon. The ribbon [residue class](#) determines which atoms are considered mainchain. The **amino acid** and **nucleic acid** classes are built into Chimera, but additional classes can be created with [Ribbon Style Editor](#).

**Ribbackbone** allows displaying backbone atoms at their unadjusted positions at the same time as ribbon (of course, these atoms can still be undisplayed). **~Ribbackbone** restores the default behavior. If the [atom-spec](#) is blank, all open molecule models will be affected; otherwise, the entire model(s) containing [atom-spec](#) will be affected. Whether ribbon and backbone atoms can be displayed simultaneously for residues within a model can also be controlled in the [molecule model attributes panel](#).

The ribbon follows a smooth, interpolated path that in some places may not overlap well with the backbone atoms. Displayed in their true positions, these atoms may appear to be “floating” away from the ribbon. The method of path calculation can be controlled with [ribspline](#); a cardinal spline may be preferred over the default B-spline for more closely tracking the backbone atom positions ([more...](#)).

See also: [ribbon](#), [ribrepr](#), [ribscale](#), [ribclass](#), [ribspline](#), [display](#)

**Usage:****ribbon** [atom-spec](#)**Usage:****~ribbon** [atom-spec](#)

**Ribbon** shows a [ribbon representation](#) of proteins and nucleic acids (and chains of any other residue types for which a ribbon [residue class](#) has been defined). **~Ribbon** hides the ribbon. Residues with any atoms in [atom-spec](#) will be affected. See the [video mini-example](#). See also: [color](#), [ribcolor](#), [ribinsidecolor](#), [shape ribbon](#), [shape tube](#)

Ribbon style can be controlled with [ribrepr](#), ribbon scaling (secondary-structure-specific dimensions) with [ribscale](#), ribbon residue class with [ribclass](#), and ribbon path method with [ribspline](#). New styles, scalings, and classes can be created with [Ribbon Style Editor](#).

By default, a residue's [mainchain atoms](#) are hidden when its ribbon segment is displayed; the command [ribbackbone](#) enables showing both at the same time.

Protein helix and strand assignments are taken from the input structure file or generated with [ksdssp](#). For nucleic acids, the ribbon simply follows the phosphodiester backbone.

Ribbon color is assigned on a per-residue basis (for example, using [rainbow](#)) and may differ from the apparent color of the residue's atoms; see [coloring hierarchy](#). If ribbon colors have not been assigned, the ribbon defaults to the model-level color.

**Usage:**

**ribclass** *class-name* [atom-spec](#)

**Ribclass** sets [residue class](#): which atoms in a residue control [ribbon](#) path and which are considered part of the backbone. The *class-name* can be **none**, indicating default behavior, or the name of a [residue class](#) previously created and named with [Ribbon Style Editor](#). The default behavior is to use the [built-in residue class](#) most appropriate for a given residue, based on the atoms it contains. Residue class names containing spaces should be enclosed in single or double quote marks, for example:

```
ribclass "beta peptide" :1-8.b
```

Residues with any atoms in [atom-spec](#) will be affected.

The command [ribspline](#) sets how ribbon path is calculated from atom positions. Backbone atoms are hidden when ribbon is shown unless [ribbackbone](#) is used.

See also: [ribbon](#), [ribrepr](#), [ribscale](#)

**Usage:**

( **ribcolor** | **ribcolour** ) *color\_name* [atom-spec](#)

**Usage:**

( **~ribcolor** | **~ribcolour** ) [atom-spec](#)

**Ribcolor** is equivalent to [color](#) with the ,r specifier.

**Ribcolor** assigns ribbon colors at the residue level. Even if only particular atoms are [specified](#), the ribbon segment(s) corresponding to the entire residue(s) containing the atoms will be affected. The command **~ribcolor** sets the ribbon colors to none (no color assigned), revealing the model-level color (see [coloring hierarchy](#)).

*Color\_name* can be **none** or any [color name](#) that specifies a single color.

See also: [ribbon](#), [color](#), [ribinsidecolor](#), [modelcolor](#), [rainbow](#), [rangeicolor](#), the [Actions menu](#)

**Usage:**

( **ribinsidecolor** | **ribinsidecolour** ) *color\_name* [atom-spec](#)

**Usage:**

( **~ribinsidecolor** | **~ribinsidecolour** ) [atom-spec](#)

**Ribinsidecolor** specifies a color to be used for the insides of peptide/protein helix ribbon segments. Only a single [ribbon inside color](#) can be defined per molecule model. Even if only particular atoms are [specified](#), the entire molecule model will be affected. The command **~ribinsidecolor** sets the color to none (no color assigned), making the ribbon insides match the outsides.

*Color\_name* can be **none** or any [color name](#) that specifies a single color.

Ribbon inside color can also be changed with [Actions... Color](#) in the menu, the [Selection Inspector](#), and the [molecule model attributes panel](#).

See also: [ribbon](#), [ribcolor](#), [color](#), [modelcolor](#), [rainbow](#)

**Usage:**

**ribrepr** *style* [atom-spec](#)

**Ribrepr** controls [ribbon](#) display style. Possible *style* settings include:

- **flat** or **ribbon** (no thickness)
- **edged** or **sharp** (thick with rectangular edges)
- **rounded**, **round**, or **smooth** (thick with rounded edges)
- **supersmooth** (same as **rounded**, except [defined](#) with a greater number of points; useful when generating [raytraced](#) images)

Additional [styles](#) created and named with the [Ribbon Style Editor](#) can also be used. Other ways to switch among ribbon styles are with the [Actions menu](#), the [Selection Inspector](#), and the [molecule model attributes panel](#).

Residues with any atoms in [atom-spec](#) will be affected.

See also: [ribbon](#), [ribbackbone](#), [ribscale](#), [ribclass](#), [ribspline](#), [preset](#)

**Usage:****ribscale** *scaling* [atom-spec](#)

**Ribscale** controls which [scaling](#) (named set of secondary-structure-specific dimensions) is used for [ribbons](#). Possible *scaling* settings include:

- **Chimera default** - varies with secondary structure; beta-strand directionality is shown with arrowheads
- **licorice** - does not vary with secondary structure

Their definitions can be viewed in the [Ribbon Style Editor](#). Additional [scalings](#) created and named with the [Ribbon Style Editor](#) can also be used. Other ways to switch among ribbon scalings are with the [Selection Inspector](#) and the [molecule model attributes panel](#).

Scaling names containing spaces should be enclosed in single or double quote marks, for example:

```
ribscale "Chimera default" :1-8.b
```

Residues with any atoms in [atom-spec](#) will be affected.

**Known problem:** reassigning secondary structure (for example, by using [ksdssp](#)) sets the scaling of some residues to **Chimera default**; any other scalings previously in effect need to be reapplied.

See also: [ribbon](#), [ribbackbone](#), [ribrepr](#), [ribclass](#), [ribspline](#), [preset](#)

**Usage:**

**ribspline** *method* [ **spec** [atom-spec](#) ] [ **smoothing** none | strand | coil | both ]  
 [ **stiffness** *s* ]

**Rihspline** sets the *method* of [ribbon](#) path calculation:

- **bspline** or **default** - path smoothed over five successive residues, may not coincide exactly with any [mainchain atoms](#)
- **cardinal** - path smoothed over four successive residues, follows [guide atom](#) positions exactly unless additional smoothing is applied; to minimize the appearance of kinks, a tubular ribbon (for example, command: [ribrepr rounded](#); [ribscale licorice](#)) may be preferred

The options can be given in any order. Option keywords can be truncated to unique strings and their case does not matter.

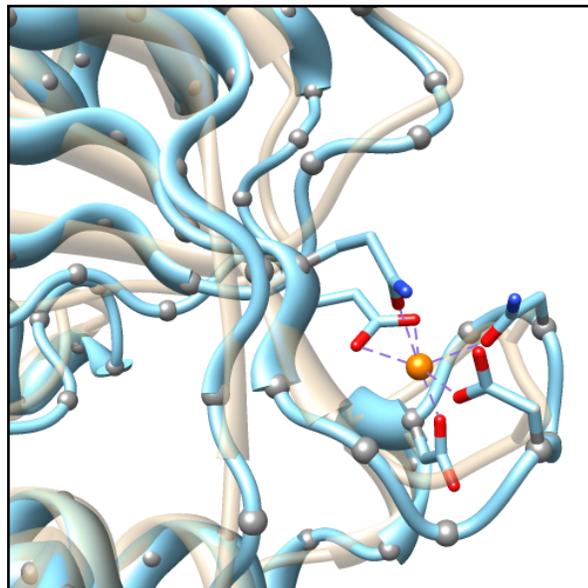
The **spec** option allows limiting the command to certain models; molecule models containing any atom in [atom-spec](#) will be affected. Only one method can be used per molecule model.

The remaining options affect only the **cardinal** spline:

- whether to perform additional **smoothing**, and if so, for which types of secondary structure (helix and strand assignments are taken from the input structure file or generated with [ksdssp](#)):
  - **none** (default)
  - **strand** - peptide/protein strand
  - **coil** - peptide/protein nonhelix, nonstrand; nucleic acid
  - **both** - strand and coil
- **stiffness** or degree of resistance against changes in path direction (range 0.0-1.0, default 0.8)

Other ways to control the ribbon path method are with the [Selection Inspector](#) and the [molecule model attributes panel](#).

See also: [ribbon](#), [ribbackbone](#), [ribrepr](#), [ribscale](#), [ribclass](#), [nucleotides](#), [Ribbon Style Editor](#)



**bspline:** tan;  
**cardinal** (unsmoothed): light blue;  
 gray balls:  $\alpha$ -carbon positions

**Usage:**

rlabel [ **offset default** | *x,y,z* ] [atom-spec](#)

**Usage:**

~rlabel [atom-spec](#)

**Rlabel** displays labels for residues with any atom in [atom-spec](#). **~Rlabel** removes residue labels. The default label contents are residue name and specifier, but custom residue labels can be defined using [labelopt](#) or the [custom residue labeling](#) dialog. Label font, size, and whether to draw labels in front regardless of their Z-offsets (default true) can be set in the [Labels preferences](#). The command [label](#) shows atom labels. See also: [color](#), [2dlabels](#), the [Actions menu](#)

The **offset** keyword allows specifying an absolute offset or a return to the **default** behavior, which depends on the [method](#) of residue label positioning. The word **default** can be truncated. An absolute offset is specified by three comma-separated values *x,y,z* representing distances along the X-, Y-, and Z-axes in the laboratory coordinate system:

- the X-axis is horizontal in the plane of the screen, increasing to the right
- the Y-axis is vertical in the plane of the screen, increasing upward
- the Z-axis is perpendicular to the screen, increasing outward (toward the viewer)

Negative distances can be used. Label offsets can also be adjusted [with the mouse](#).

The method of [residue label positioning](#) can be set in the [molecule model attributes panel](#) or with the command [setattr](#). Residue label position can be based on:

- the centroid of the displayed atoms in the residue (default method; default offset **0,0,1**)
- the centroid of the displayed backbone atoms ([mainchain atoms](#)) in the residue, and for residues without such atoms, the centroid of the displayed atoms in the residue (default offset **0.5,0,0.5**)
- the residue *primary atom*, generally CA in an amino acid, C5' in a nucleotide, the first nonhydrogen atom in residues of other types (default offset (*display\_radius* + **0.2**),**0,0.5**, automatically adjusted along with [display style](#))

When ribbon is displayed instead of [mainchain atoms](#), the mainchain atoms are considered displayed for the purposes of computing centroids.

**Usage:**

**rmsd** [atom-spec1](#) [atom-spec2](#)

The **rmsd** command evaluates the root-mean-square deviation (RMSD) between specified sets of atoms, without any fitting. The related command [match](#) performs least-squares fitting. See also: [superimposing structures](#)

[Atom-spec1](#) and [atom-spec2](#) must specify equal numbers of atoms.

Atoms are paired in the order specified, *i.e.*, the first atom in the first specification is matched to the first atom in the second specification, the second atom to the second atom, and so on. If atom order is not specified, for example,

```
rmsd #1:fad #0:fad  
rmsd #2:246,295 #0:195,221
```

the atoms within a residue are ordered first by name, and where these are identical, by alternate location identifier, and where these are also identical, by serial number.

The two sets of atoms can be specified collectively as a [selection](#) using the word **selected**, **sel**, or **picked**. An ordered selection can be created by [picking](#) atoms one by one, first the atoms of the first set, then those of the second in the corresponding order.

**Usage:****rna operation arguments**

The **rna** command builds rough but potentially large-scale models of single-stranded RNA, given base-pairing information. Nucleotides are laid out schematically to form stems and loops in three dimensions. **Rna** also builds DNA and RNA/DNA double helices along a specified path. This command was used to make 9000-nucleotide RNA and DNA models for an [HIV virus animation](#), along with the command [play wiggle](#).

For building DNA, RNA, and hybrid double helices in standard conformations (A-form or B-form), see [Build Structure](#). For more detailed RNA model-building and refinement from secondary structure information, see the [Assemble2](#) plugin (available separately from Chimera).

The *operation* can be:

- [duplex](#) - create double-helical DNA or RNA/DNA following a given path
- [minimizeBackbone](#) - energy-minimize nucleic acid backbones, leaving bases fixed
- [model](#) - create single-stranded RNA from secondary structure (base-pairing) information
- [path](#) - create a path of [markers](#) representing stems and loops
- [smoothPath](#) - create a smoother version of a path

Operation keywords and their sub-keywords described below can be truncated to unique strings, and their case does not matter. Synonyms for true: True, 1. Synonyms for false: False, 0. A vertical bar “|” designates mutually exclusive options, and default settings are indicated with **bold**. Examples are provided [below](#).

- **rna duplex** *sequence path* [ **startSequence** *i* ] [ **type** **DNA** | **RNADNA** ]

Create an atomic model of double-helical DNA or hybrid RNA/DNA with a specified sequence on one strand and the complementary sequence on the other, following a given path. The *sequence* can be a string of upper-case letters (such as AGCTU) or the pathname of a FASTA file containing the desired sequence. The **startSequence** option (default **1**) indicates where to start within the supplied sequence. The *path* is [specified](#) as a series of atoms or [markers](#) (such as from [rna path](#) or [rna smoothPath](#)), where each atom or marker represents the center of one base pair. The **type** of double helix can be **DNA** (default) or **RNADNA** (hybrid RNA/DNA).

**Algorithm:** Basepair atomic templates are placed along the path, with a twist of 36° per base pair. The templates are based on PDB entry [1BNA](#).

- **rna minimizeBackbone** *model* [ **chunkSize** *c* ] [ **steps** *steps* ] [ **conjugateGradientSteps** *cgsteps* ] [ **updateInterval** *i* ] [ **noGui** true | false ]

Energy-minimize the backbone of RNA and/or DNA in *model* (a molecule model specified by ID number), keeping the bases fixed.

For computational expediency, minimization is applied to successive nonoverlapping segments of **chunkSize** (default **10**) consecutive residues. The purpose is to fix long bonds and bad angles generated by joining template fragments (as is done by [rna model](#) and [rna duplex](#)), rather than to resolve any nonlocal conflicts. The calculation uses the [minimize](#) command with the specified number of steepest-descent **steps** (default **100**), **conjugateGradientSteps** (default **100**), **updateInterval** (default **10**), and **nogui** setting (default **true**). These correspond to the [minimize](#) command options **nsteps**, **cgsteps**, **interval**, and **nogui**, respectively.

- **rna model** *sequence* ( **path** [spec](#) | **pairs** *p* ) [ **startSequence** *i* ] [ **length** *l* ] [ **circle** true | false ] [ **randomBranchTilt** *angle* ] [ **stemColor** *color* ] [ **loopColor** *color* ] [ **name** *n* ]

Create an atomic model of single-stranded RNA. The *sequence* can be a string of upper-case letters (such as AGCTU) or the pathname of a FASTA file containing the desired sequence. The **startSequence** option (default **1**) indicates where to start within the supplied sequence. Either a **path** or **pairs** must be given. The **path** to be followed by the RNA is [specified](#) as a series of [markers](#) from [rna path](#) (which also assigns nucleotide orientations required to create the model). The **pairs** specification and associated options **length**, **circle**, and **randomBranchTilt** are the same as described for [rna path](#). Base-paired and non-base-paired nucleotides in the resulting structure will be colored according to the **stemColor** and **loopColor** values, respectively, where each *color* can be any [color name](#) that specifies a single color. The **name** option gives the name of the resulting molecule model (default **RNA**).

**Algorithm:** The path of the RNA is either precomputed with [rna path](#) or computed implicitly using the same algorithm as that command. The algorithm generates a series of markers and assigns an orientation to each marker ([details...](#)). Atomic model templates are placed in the computed orientations, with template coordinates 0,0,0 (atom N9 in A and G nucleotides, atom N1 in C and U nucleotides) on top of the markers. In stem regions, orientations are chosen to form base-pairing hydrogen bonds. In loops (which are shaped like lobes), orientations are chosen to stack the bases perpendicular to the plane of the lobe and to extend them toward the center line of the lobe or the center of the semicircle that forms the lobe end. The atomic templates are in the file rna-templates.pdb within the Chimera RNALayout module.

- **rna path** *pairs* [ **length** *l* ] [ **circle** true | false ] [ **radius** *r* ] [ **randomBranchTilt** *angle* ] [ **stemColor** *color* ] [ **loopColor** *color* ] [ **name** *n* ]

Create a path of [markers](#), one marker per nucleotide, to represent an RNA molecule with specified secondary structure (base-pairing interactions). Consecutive markers and base-paired markers are connected by links. Stretches of base-paired nucleotides are *stems*, whereas nucleotides that are not base-paired are said to belong to *loops*. Stems are described with triples of integers. For example, 1,50,10 indicates pairing nucleotide 1 with nucleotide 50 at the start of a 10-bp stem, such that 2 and 49, 3 and 48, ... 10 and 41 are also paired. Multiple stems can be specified with additional triples (*e.g.*, 1,50,10,60,70,2 describes two stems), and any number of stems can be given. A limitation is that residues within a stem range cannot be paired with residues beyond that range; for example, if a stem starts at 1,50, no other stem can pair residues in the range 1-50 with residues outside that range. The required *pairs* argument can be either a comma-separated list of numbers or the pathname of a text file containing three columns, one line per stem.

Residue numbers in the resulting marker set start at 1. The last residue number is the highest number specified for a stem, unless a higher number is given with the **length** option. The **circle** option (default **false**) controls whether the overall RNA layout is circular or linear. The **radius** option specifies marker radius (default **2.0 Å**). Stem and loop orientations are produced by random rotations, where each angle of rotation is obtained by multiplying the **randomBranchTilt** *angle* (default **0°**) by a random number uniformly distributed between -1 and 1. Paired and unpaired markers are colored according to the **stemColor**

and **loopColor** values, respectively, where each *color* can be any [color name](#) that specifies a single color. The **name** option gives the name of the resulting marker model (default **RNA path**).

**Algorithm:** Stem regions are generated as twisted double helices rotating 36° per base pair, with pairs 4 Å apart along the helix axis and the markers for paired nucleotides 9 Å apart. The line between the paired markers is perpendicular to the helix axis and 4 Å from that axis. Markers for loop nucleotides are 5 Å apart. Loops are laid out as lobes in which two parallel lines of nucleotides are joined by a semicircle of nucleotides at one end. The semicircle contains 8 nucleotides, and the straight segments may contain up to 10 nucleotides each. Loops of more than 28 nucleotides will be laid out as two side-by-side lobes. The RNA between the two nucleotides at the end of a stem forms a series of loops and stems, or *cycle*. A cycle is laid out in a circle, that is, the points at which loops and stems connect to the cycle are placed on a planar circle. The loops and stems radiate outward within same plane unless a nonzero **randomBranchTilt** *angle* is given. If **circle** is **false**, the top-level cycle is laid out in a straight line instead of a circle, and each stem or loop is rotated about the previous one by 145° in addition to any random tilt. Except for the random tilt, the geometric parameters are currently hard-coded.

- **rna smoothPath** *path* [ **radius** *r* ] [ **spacing** *s* ] [ **kinkInterval** *ki* ] [ **kinkRadius** *kr* ] [ **name** *n* ]

Create a new path of [markers](#) that is a smoothed version of an input [path](#) (such as from [rna path](#)). The smoothing assures a minimum radius of curvature (**radius** *r*, default 50 Å) and produces a set of markers with uniform **spacing** *s* (default 3.33 Å). The original use was to create a path for duplex DNA approximating the path of a single-stranded RNA, but without sharp kinks. To allow for some sharper kinks, a different minimum radius of curvature given by **kinkRadius** can be applied every **kinkInterval** markers (default is no kinks). The **name** option gives the name of the resulting marker model (default **smooth path**).

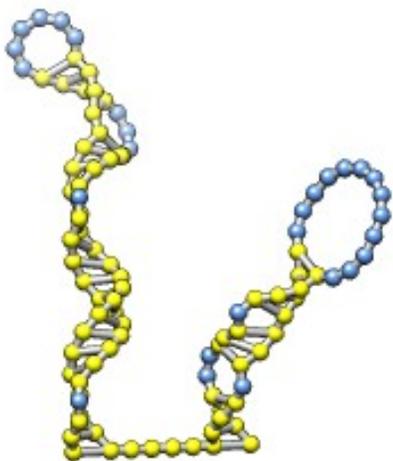
**Algorithm:** The first new marker is placed at the first marker of the input path. Subsequent markers are placed in the direction from the most recent new marker to the next input marker. If that direction would cause a bend with radius of curvature less than required, as judged by the angle made by the last segment of the new path and the next segment, the angle is expanded to give the minimum allowed radius of curvature (a bend of  $2 \cdot \arcsin(\text{spacing}/2 \cdot \text{radius})$ ). The next marker is placed in this direction, at the specified distance from the previous marker.

## Examples

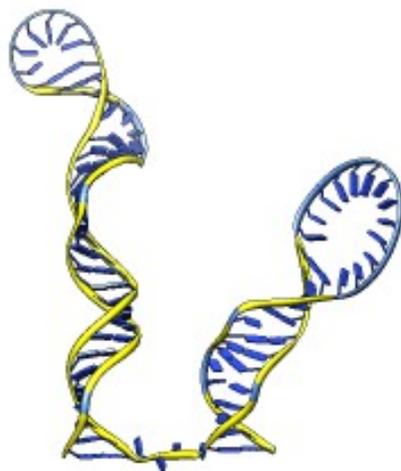
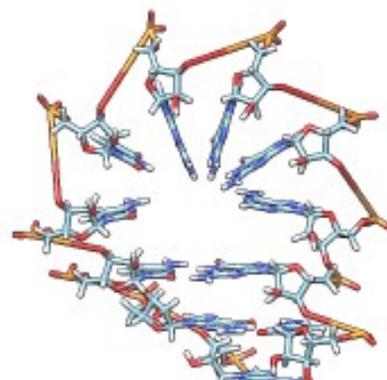
Each of the **rna** operations is illustrated here for the first 100 nucleotides of HIV RNA, with secondary structure as described in:

[Architecture and secondary structure of an entire HIV-1 RNA genome](#). Watts JM, Dang KK, Gorelick RJ, Leonard CW, Bess JW Jr, Swanstrom R, Burch CL, Weeks KM. *Nature*. 2009 Aug 6;460(7256):711-6.

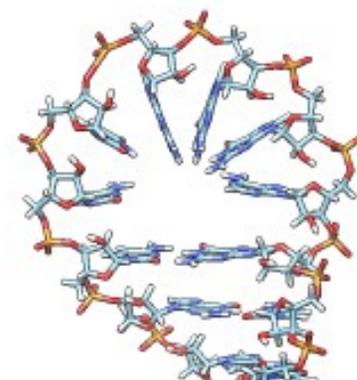
The full-length HIV RNA model can be viewed in this [HIV virus animation](#).



rna path pairings.txt length 100

rna model hiv-pNL4-3.fasta #0 start  
455

Without minimization.



rna min #1



rna smooth #0

rna duplex #2 hiv-pNL4-3.fasta  
start 455

with base pairing file pairings.txt containing

1	57	3
5	54	11
17	43	5
25	38	4
58	104	8
67	94	3
70	90	4

and sequence file hiv-pNL4-3.fasta containing

```
>gi|296556482|gb|AF324493.2| HIV-1 vector pNL4-3, 1-9709
TGGAAGGGCTAATTTGGTCCCAAAAAGACAAGAGATCCTTGATCTGTGGATCTACCACACACAAGGCTA
CTTCCCTGATTGGCAGA ACTACACACCAGGGCCAGGGATCAGATATCCACTGACCTTTGGATGGTGCTTC
AAGTTAGTACCAGTTGAACCAGAGCAAGTAGAAGAGGCCAATGAAGGAGAGAACAACAGCTTGTTACACC
CTATGAGCCAGCATGGGATGGAGGACCCGGAGGGAGAAGTATTAGTGTGGAAGTTTGACAGCCTCCTAGC
ATTTTCGTCACATGGCCCGAGAGCTGCATCCGGAGTACTACAAAGACTGCTGACATCGAGCTTTCTACAAG
GGACTTTCCGCTGGGGACTTTCCAGGGAGGTGTGGCCTGGGCGGGACTGGGGAGTGGCGAGCCCTCAGAT
GCTACATATAAGCAGCTGCTTTTTGCCTGTACTGGGTCTCTCTGGTTAGACCAGATCTGAGCCTGGGAGC
TCTCTGGCTAACTAGGGAACCCACTGCTTAAGCCTCAATAAAGCTTGCCTTGAGTGCTCAAAGTAGTGTG
```

**Usage:**

```
rock [ axis [ angle [ frames ]]] [ cycle frames-per-cycle ] [ models model-spec ]
[ coordinateSystem N ] [ center center ]
```

**Rock** wags the specified **models** back and forth around a specified *axis* (default **y**) for a specified number of *frames* (default **infinity**). A full cycle of motion, analogous to a pendulum swing in one direction and then the other, takes *frames-per-cycle* frames (default **136**). Larger values of *frames-per-cycle* will give a slower rocking motion. See the [video mini-example](#). See also: [roll](#), [turn](#), [select](#), [cofr](#), [set independent](#), [movie-related commands](#)

The *angle* (default **3°**) is 1/20 of the total angle of rotation from one extreme to the other. It can be positive or negative, corresponding to the initial direction of motion.

The **models** to rock can be specified by model number(s) or ranges separated by commas and preceded by #. If no models are specified, all [active](#) models will be moved.

A **coordinateSystem** can be specified by reference model number *N*, optionally preceded by #. Otherwise, the laboratory frame of reference will be used. Any *axis* or *center* specification of the form *x,y,z* will be interpreted in the reference coordinate system, and when further motions are applied to an ongoing rotation, the center and axis will remain pinned relative to the reference model.

The *axis* can be:

- **x** - X-axis (if laboratory coordinate system, horizontal in the plane of the screen)
- **y** - Y-axis (if laboratory coordinate system, vertical in the plane of the screen)
- **z** - Z-axis (if laboratory coordinate system, perpendicular to the screen)
- *x,y,z* (three values separated by commas only) - an arbitrary vector
- an [atom-spec](#) of exactly two atoms (not necessarily bonded or in the same model) or one bond. If two atoms, the order of specification defines a handedness, and right-handed rotations are positive. If a bond, the handedness is not under user control. A bond can only be specified by [selecting](#) it and using the word **selected**, **sel**, or **picked**; any atoms also selected at the time will be ignored. If a center is not specified separately, it will lie on the atom-atom or bond axis. If a separate center is supplied, only the direction of the axis will be used. The first model in the axis [atom-spec](#) will be used for reference frame pinning unless a **center** [atom-spec](#) or a **coordinateSystem** is also given.

The *center* can be:

- *x,y,z* (three values separated by commas only) - an arbitrary point
- an [atom-spec](#) of any combination of atoms and [surface pieces](#). The center of the bounding sphere of the specified items will be used. The first model in [atom-spec](#) will be used for reference frame pinning unless a **coordinateSystem** is also given.
- any of the following keywords, to specify a [center of rotation method](#) for computing the center:
  - **view** (center of view method)
  - **models** (center of models method)

rock

- **front** (front center method)

This does not change the current [center of rotation method](#), but allows using a different one for the purposes of executing the command.

If a center of rotation is not specified directly or by using an [atom-spec](#) to define the axis, the current Chimera [center\(s\) of rotation](#) will be used.

Keyword options can be used in any order and the keywords can be truncated. Example:

```
rock z 5 136 center 0,0,0 coord 0
```

Commands continue to be processed while the requested motion is in progress. To pause command processing until the motion is finished, use the [wait](#) command. To halt an ongoing **rock**, use [freeze](#).

**Usage:**

**roll** [ *axis* [ *angle* [ *frames* ]]] [ **models** *model-spec* ] [ **coordinateSystem** *N* ]  
 [ **center** *center* ] [ **precessionTilt** *tilt* ]

**Roll** rotates the models by *angle* degrees (default 1.5) around a specified *axis* (default **y**) for a specified number of *frames* (default **infinity**). **Roll** is the same as [turn](#) except for the default number of frames. See also: [rock](#), [select](#), [cofr](#), [set independent](#), [movie-related commands](#)

The *angle* can be positive or negative. The **models** to rotate can be specified by model number(s) or ranges separated by commas and preceded by #. If no models are specified, all [active](#) models will be rotated.

A **coordinateSystem** can be specified by reference model number *N*, optionally preceded by #. Otherwise, the laboratory frame of reference will be used. Any *axis* or *center* specification of the form *x,y,z* will be interpreted in the reference coordinate system, and when further motions are applied to an ongoing rotation, the center and axis will remain pinned relative to the reference model.

The *axis* can be:

- **x** - X-axis (if laboratory coordinate system, horizontal in the plane of the screen)
- **y** - Y-axis (if laboratory coordinate system, vertical in the plane of the screen)
- **z** - Z-axis (if laboratory coordinate system, perpendicular to the screen)
- *x,y,z* (three values separated by commas only) - an arbitrary vector
- an [atom-spec](#) of exactly two atoms (not necessarily bonded or in the same model) or one bond. If two atoms, the order of specification defines a handedness, and right-handed rotations are positive. If a bond, the handedness is not under user control. A bond can only be specified by [selecting](#) it and using the word **selected**, **sel**, or **picked**; any atoms also selected at the time will be ignored. If a center is not specified separately, it will lie on the atom-atom or bond axis. If a separate center is supplied, only the direction of the axis will be used. The first model in the axis [atom-spec](#) will be used for reference frame pinning unless a **center** [atom-spec](#) or a **coordinateSystem** is also given.

The *center* can be:

- *x,y,z* (three values separated by commas only) - an arbitrary point
- an [atom-spec](#) of any combination of atoms and [surface pieces](#). The center of the bounding sphere of the specified items will be used. The first model in [atom-spec](#) will be used for reference frame pinning unless a **coordinateSystem** is also given.
- any of the following keywords, to specify a [center of rotation method](#) for computing the center:
  - **view** (center of view method)
  - **models** (center of models method)
  - **front** (front center method)

This does not change the current [center of rotation method](#), but allows using a different one for the purposes of executing the command.

roll

If a center of rotation is not specified directly or by using an [atom-spec](#) to define the axis, the current Chimera [center\(s\) of rotation](#) will be used.

The **precessionTilt** option specifies an additional rotation about a moving axis that is carried along by the main rotation. The moving axis is offset from the main *axis* by *tilt* degrees (toward the vertical if *axis* is along the line of sight). The two rotations cycle at the same rate but in opposite directions around their respective axes. See the [Wobble motion](#) movie in the Chimera Animation Gallery.

Keyword options can be used in any order and the keywords can be truncated. Example:

```
roll 1,1,0 center 8.5,0,0 coord #1
```

Commands continue to be processed while the requested motion is in progress. To pause command processing until the motion is finished, use the [wait](#) command. To halt an ongoing **roll**, use [freeze](#).

**Usage:**

**rotation** [ *rot-id* ] [ *reverse* ] *atom1 atom2*

**Usage:**

**rotation** *rot-id angle* [ *frames* ]

**Usage:**

**~rotation** *rot-id*

The **rotation** command activates a torsion (makes a bond rotatable) or adjusts a previously activated torsion (rotates the bond). Active torsions are listed in [Adjust Torsions](#) and can also be controlled with that tool. See the [video mini-example](#). See also: [angle](#), [adjust](#), [ramachandran](#), [roll](#), [turn](#), [movie-related commands](#)

The command **~rotation** freezes a bond in its current position and makes it no longer rotatable.

When *atom1* and *atom2* are [specified](#), **rotation** makes the bond between them rotatable. If *rot-id* is not supplied, the next available integer will be used. The order in which the atoms are specified is not important. If the two atoms (only) are [selected](#), they can be specified collectively with the word **selected**, **sel**, or **picked**. The two atoms must be bonded, and the bond cannot be terminal (not attached to further atoms on either end) or part of a ring. Two rotations cannot affect a common set of atoms unless one affected set is a complete subset of the other. That is, rotations must be properly nested. The ability to activate a rotation does not imply that the bond is freely rotatable in the chemical sense; for example, it could be a double bond. By default, the smaller group of atoms will move during bond rotation while the larger group remains fixed. The **reverse** keyword indicates the larger group should move instead. This keyword can only be specified at the time of activation, but an already active torsion can be reversed using the [Adjust Torsions dialog](#).

An already active torsion can be adjusted by specifying its *rot-id* along with the *angle* of change in degrees. The rotation can be performed repeatedly for a specified number of *frames* (default 1).

Examples:

```
rotation 2 :16.b@ca,cb
```

- activate a rotation with ID number 2 around the CA-CB bond of residue 16 in chain B

```
rotation 2 -40
```

- adjust rotation 2 by -40°

```
rotation #0:1@c:2@n
```

rotation

- activate a rotation around the bond between C in residue 1 and N in residue 2 of model 0

**[Usage:](#)**

**runscript** *script script-arguments*

**Runscript** allows executing a Python script with command-line arguments from within Chimera. The script file is specified by *script*, either a pathname to open the file directly or the word **browse** or **browser** to elicit a [dialog](#) for opening the file. The script runs in Chimera's current directory, which can be changed with the command [cd](#) (other files referenced in the script or its arguments can be specified by pathnames relative to that location or by absolute pathnames). Arguments that include spaces must be enclosed by quotes, but quotes should not be nested, with the exception that single quotes can appear inside a double-quoted string and double quotes can appear inside a single-quoted string.

The **chimera** module is pre-imported into the script's global namespace, and the arguments are made into a tuple variable in the global namespace called "arguments" containing however many strings were supplied. The arguments are also available in the standard `sys.argv` location.

The command [open](#) can also execute Python scripts, but it does not include a way of supplying command-line arguments.

See also: startup option [--script](#), [pdbrun](#)

[Chimera Commands Index](#)

**Usage:**

**save** [ *session\_name* ]

**Usage:**

**save** [ *session\_name.py* ]

**Save** saves a [Chimera session](#). If *session\_name* is omitted, a [dialog](#) for specifying the name and location will appear. A session file consists of Python code that reconstitutes the state of Chimera by displaying data and performing other operations; if no **.py** suffix is entered, it will be added to the name automatically.

Molecular coordinate and sequence alignment data are included in the session file; however, volume data files must still be present to restart a session in which they were open. When a session with volume data is restarted, the user may be queried about file location(s) if the data or the session file have been moved since the session was saved.

There are multiple ways to [save sessions](#) and [restart sessions](#).

See also: [savepos](#), [open](#), [export](#), [write](#), [pdbrun](#), the [File menu](#)

**Usage:****savepos** [ *view\_name* ]**Usage:****~savepos** [ *view\_name* ]**Usage:****savepos list**

**Savepos** saves the current position and associates *view\_name* with it. If *view\_name* is missing, the name **default** is used. Saved positions can be restored with [reset](#) or from the [Rapid Access](#) interface, and the command [fly](#) can generate a smooth interpolated path over a series of saved positions.

A *position* includes:

- the transformations of models (their rotations and translations relative to input coordinates)
- the overall scale
- [clipping plane](#) status, locations, and orientations
- the [center of rotation method](#)
- the [horizontal field of view](#)

Unless changed with **savepos**, the **default** position is one in which the models have not been rotated or translated.

Saved positions are included in saved [Chimera sessions](#), and a position named **session-start** is created automatically when a session is restored. To decrease the size of session files, positions can be “forgotten” using **~savepos** *view\_name*.

The **list** keyword indicates that the names of all saved positions should be listed in the [status line](#) and [Reply Log](#).

See also: [reset](#), [fly](#), [save](#), [scene](#), [movie-related commands](#)

**Usage:**

**scale** *factor* [ *frames* ]

**Usage:**

**~scale**

**Scale** changes the size of the view by *factor* for the specified number of image update *frames* (default 1). The scaling factor must be positive; a scaling factor less than one will decrease the size of the displayed view. This command does not modify the model coordinates. See the [video mini-example](#). See also: [3D manipulation](#), the [Side View](#), [movie-related commands](#)

**~Scale** will halt an ongoing scaling process. Commands continue to be processed while the requested motion is in progress. To pause command processing until the motion is finished, use the [wait](#) command.

**Usage:**

```
scene scene_name ( save | reset [ frames ] )
```

**Usage:**

```
~scene ( scene_name | all )
```

**Usage:**

```
scene list
```

The **scene** command saves and restores scenes.

\*\* Scenes are under development and do not include all aspects of Chimera. Including scenes in saved [sessions](#) may significantly increase session file size and time to restore. \*\*

A Chimera *scene* includes:

- locations and orientations of models and clipping planes (*etc.*, as in a saved [position](#))
- display styles, colors, and visibilities of molecule models, molecular surfaces, and their parts
- [volume](#) displays
- 3D labels and [2D labels](#)
- certain global parameters such as [background](#), [lighting](#), and [effects](#)

The **save** keyword indicates saving the current scene as *scene\_name*. The **reset** keyword indicates restoring the named scene over the specified number of image update *frames* (default **1**). Scenes can also be saved and restored using the [Animation](#) tool, or restored from [Rapid Access](#).

Saved scenes are included in saved [Chimera sessions](#). To decrease the size of session files, scenes can be “forgotten” (deleted) using **~scene**, either individually by name or **all** at once.

The **list** keyword indicates that the names of all saved scenes should be listed in the [Reply Log](#) and the [Animation](#) dialog (if [active](#), otherwise the Chimera [status line](#)).

See also: [savepos](#), [reset](#), [fly](#), [save](#), [movie-related commands](#)

**Usage:****( scolor | colour ) surf-model method [options](#)**

**Scolor** colors [surface models](#) and their [caps](#). It is the command-line implementation of [Surface Color](#) and [Color Zone](#). See also: [color](#), [colorkey](#), [rangecolor](#), [rainbow](#), [transparency](#), [coulombic](#), [apbs](#), [sop](#), [volume](#), [msc](#)

Examples:

```

scolor #0 geometry cylindrical
scolor #0 geom radial cmap 100,gray:150,blue:200,purple
scolor #0 volume #1 cmap -10,red:0,white:10,blue
scolor #1 zone @ca range 5

```

The *surf-model* argument can be a specific model number or range of model numbers preceded by #, or simply # to indicate all applicable surfaces. The *method* can be:

- [color](#) - a single color (can also be done with the [color](#) command)
- [zone](#) - to match the colors of nearby atoms or markers
- [volume](#) - by [volume data](#) such as electron density or [electrostatic potential](#)
- [gradient](#) - by [volume data](#) gradient
- [geometry radial](#) - by distance from a point
- [geometry cylindrical](#) - by distance from an axis
- [geometry height](#) - by distance from a plane

Method keywords and their sub-keywords (described below) can be truncated to unique strings, and their case does not matter. Synonyms for true: True, 1. Synonyms for false: False, 0. A vertical bar “|” designates mutually exclusive options, and default settings are indicated with **bold**.

- **scolor surf-model color color\_name**

Simply use the specified color, where *color\_name* can be any [color name](#) that specifies a single color. The [general options](#) do not apply to this method.

- **scolor surf-model zone [atom-spec](#) [ range cutoff ] [ autoUpdate true | false ]**
- **~scolor surf-model**

Color surface patches to match the [specified](#) atoms or [markers](#). The **range** option indicates a *cutoff* (default **2.0**) in physical distance units, typically Å. The **autoUpdate** option indicates whether to update the coloring of a surface automatically when its shape changes. The **~scolor** command discontinues coloring by zone. See also: [Color Zone](#)

- **scolor surf-model volume volume-model [ perPixel true | false ]**

[ **colorOutsideVolume** *color\_name* ] [ **offset** *d* | *start,end,N* ] [general-options](#)

Color by [volume data](#) such as electron density or [electrostatic potential](#), where *volume-model* is the model number (preceded by #) of the volume data set. The **perPixel** option indicates whether to determine color separately for each surface pixel instead of interpolating across surface triangles (see [details](#)). Per-pixel coloring tends to give smoother color gradations. Values at surface vertices are determined by interpolation except when per-pixel coloring is used (see [details](#)). The **colorOutsideVolume** option specifies how to color for surface vertices that fall outside the volume data grid. The *color\_name* can be any [color name](#) that specifies a single color; the default is **#800080008000** (a dark gray). The **offset** option specifies how far out from each surface vertex, along its normal, to evaluate the data (default **0.0**):

- *d* - a single offset distance
- *start,end,N* (three values separated by commas only) - starting distance, ending distance, and how many offsets to evaluate within that range, respectively. For example, **offset 1.5,3.0,3** specifies offsets of 1.5, 2.25, and 3.0. Values from multiple offsets are averaged per surface vertex. Specifying multiple offsets turns off any [per-pixel coloring](#).

• **scolor** *surf-model* **gradient** *volume-model* [ **colorOutsideVolume** *color\_name* ] [ **offset** *d* | *start,end,N* ] [general-options](#)

Color by [volume data](#) gradient norm, where *volume-model* is the model number (preceded by #) of the volume data set. Gradients (how steeply the values change in space) at surface vertices are determined by interpolation (see [details](#)). The [colorOutsideVolume](#) and [offset](#) options are as described above for [volume](#).

• **scolor** *surf-model* **geometry radial** [ **center** *center* ] [ **coordinateSystem** *N* ] [general-options](#)

Color by distance from a point. The point can be specified with **center**, where *center* can be:

- *x,y,z* (three values separated by commas only) - an arbitrary point
- an [atom-spec](#) of any combination of atoms and [surface pieces](#). The center of the bounding sphere of the specified items will be used.

The default is the center of the bounding box of the surface. The **coordinateSystem** option indicates that *x,y,z* specifications of **center** should be interpreted in the coordinate system of a reference model. *N* is the reference model number preceded by #. The default coordinate system is that of the surface being colored.

• **scolor** *surf-model* **geometry cylindrical** [ **center** *center* ] [ **axis** *axis* ] [ **coordinateSystem** *N* ] [general-options](#)

Color by distance from an axis. The axis is defined by any point on the axis and a direction. The point can be specified with **center** and the direction with **axis**, where *axis* can be:

- **x** - X-axis
- **y** - Y-axis
- **z** - Z-axis
- *x,y,z* (three values separated by commas only) - an arbitrary vector
- an [atom-spec](#) of exactly two atoms (not necessarily bonded or in the same model) or one bond. A bond can only be specified by [selecting](#) it and using the word **selected**, **sel**, or **picked**; any atoms also selected at the time will be ignored.

The default is the Z-axis in the coordinate system of the surface. The **coordinateSystem** option indicates that specifications of [center](#) and/or [axis](#) should be interpreted in the coordinate system of a reference model. *N* is the reference model number preceded by #. The default coordinate system is that of the surface being colored.

- **scolor** *surf-model geometry height* [ *center center* ] [ *axis axis* ] [ *coordinateSystem N* ] [general-options](#)

Color by distance from a plane, or topographic height. The plane is defined by any point on the plane and a vector normal to the plane. The point can be specified with [center](#) and the vector with [axis](#). The **coordinateSystem** option indicates that specifications of [center](#) and/or [axis](#) should be interpreted in the coordinate system of a reference model. *N* is the reference model number preceded by #. The default coordinate system is that of the surface being colored.

## General Options

**cmap** *palette* | *value1,color1:value2,color2:...*

Use the specified color mapping, which can be either a pre-defined *palette* (colors listed in ascending value order):

- **redblue** (default) - red, white, blue
- **rainbow** - red, yellow, green, cyan, blue
- **cyanmaroon** - #0f1bc7adcf5b (a dark cyan), white, #9eb820005eb8 (a dark maroon)
- **gray** - black, white

or a series of *value,color* pairs (no spaces) separated by colons only. Values can be expressed directly on the scale of the data ([cmapRange](#) not used) or as fractional values 0.0-1.0 across a range of data specified with [cmapRange](#). Values below the lowest given will map to the same color as the lowest, and values above the highest will map to the same color as the highest. A *color* can be any [color name](#) that specifies a single color.

**cmapRange** *low,high* | **full**

What data values should map to the extremes of the [cmap palette](#), or to values 0.0 and 1.0 when [cmap value,color](#) pairs are used. The *low* and *high* values should be separated by a comma only (no spaces). The **full** keyword indicates using the minimum and maximum values found at surface vertices (default for pre-defined palettes). When multiple surfaces are colored in a single command, the **full** range is determined separately for each surface.

**reverseColors** true | false

Whether to reverse the order of the colors in the [palette](#).

**capOnly** true | false

Whether to color only [surface caps](#) and not the rest of a surface.

**autoUpdate** true | false

Whether to update the coloring of a surface automatically when its shape changes. For example, the shape of a [surface cap](#) changes as [clipping](#) is adjusted, and a [molecular surface](#) changes shape when it is recomputed with a different probe radius. This updating only accounts for changes in the shape of a surface, not changes in its position or orientation relative to other models.

**Usage:****section** *distance* [ *frames* ]

**Section** moves both [global clipping planes](#) in the same direction by the specified *distance*. The *distance* is a value in the current display units, usually Å; a positive number displays a cross section closer to the user, while a negative number displays a cross section farther away from the user.

**Section** moves the clipping planes in the specified manner for the specified number of image update *frames* (default **1**). Commands continue to be processed while the requested motion is in progress. To pause command processing until the motion is finished, use the [wait](#) command. To halt an ongoing **section** motion, use [freeze](#).

See also: [clip](#), [thickness](#), the [Side View](#), [clipping](#), [movie-related commands](#)

**Usage:****segment** *operation arguments*

The command **segment** acts on *segmentation models*, which are [surface models](#) and associated information from [Segment Map](#), and *segmentation regions*, [surface pieces](#) within a segmentation model. See also: [mask](#), [volume](#)

Possible values of *operation*:

- [copygroups](#) - copy the grouping of regions in one segmentation model to another
- [directioncolor](#) - color regions according to principal axis orientation
- [exportmask](#) - export segmentation region index mask as a map file
- [sliceimage](#) - create montage images of density map slices for segmented regions
- [unbin](#) - generate a segmentation model for a full [volume](#) data set from a segmentation model based on a binned version of the data set

Each operation keyword has different arguments, described below. Operation keywords and their sub-keywords can be truncated to unique strings, and their case does not matter. Synonyms for true: True, 1. Synonyms for false: False, 0. A vertical bar "|" designates mutually exclusive options, and default settings are indicated with **bold**.

Segmentation models, segmentation regions, and volume data sets can be specified by model number (e.g., #1) or as a selection (e.g., sel).

- **segment copygroups** *segmodel1 segmodel2*

Copy the grouping from one segmentation model (*segmodel1*) to another (*segmodel2*). Regions of *segmodel2* are grouped together if their maximum-value grid points lie within the same (grouped) region of *segmodel1*.

- **segment directioncolor** *regions* [ **pattern circle** | circle111 | rgb | rgb111 | rgb2 | cmy | cmy2 | cmz ] [ **spherekey true|false** ]

Color regions according to long axis direction. The long axis is the principal axis with minimum inertia, calculated with equal weighting of each grid point in the region. Many color schemes (**pattern** settings) are available for showing the axis orientations within the segmentation model coordinate system. The **spherekey** option indicates whether a sphere depicting the corresponding color key should be displayed.

See also: [scolor](#), [measure inertia](#), [Render by Attribute](#)

- **segment exportmask** *segmodel* [ **savePath path** ] [ **format mrc** | dsn6 | cmap | netcdf ] [ **sequentialIds true|false** ] [ **binSize bi,bj,bk** ]

Export a segmentation as an integer-valued density map in which the value at each grid point is the ID number of the segmentation region that contains it. A location and filename for saving the map can be specified with **savePath**, and the file format specified with **format** (default **mrc**). If no **savePath** is given, the map will be opened in Chimera but not saved to a file. Segmentation region ID numbers are often nonsequential. If **sequentialIds** is **true** (default), the regions will be renumbered starting from 1 in the created map; otherwise, the original ID numbers will be used. By default, the map will be the same size as the map that was segmented. If the segmentation was done on a binned map, however, the **binSize** option is useful for exporting to the original, unbinned size.

- **segment sliceimage regions volume** [ **traceSpacing** *spacing* ] [ **traceTipLength** *tlength* ] [ **unbendSize** *size* ] [ **unbendYAxis** *x,y,z* ] [ **unbendGridSpacing** *g* ] [ **sliceSpacing** *sx,sy,sz* ] [ **xyTrim** *trim* ] [ **panelAspect** *a* ] [ **imageSpacing** *pixels* ] [ **showImage** **true|false** ]

Create a montage image containing slices of unbent segmentation regions along three orthogonal axes. For each region, a center-line is computed with the [measure spine](#) command, an unbent region of the specified density map is calculated along that line with the [vop unbend](#) command, and a montage image is composed of gray-level rendering of slices along the X, Y, and Z axes of the unbent volume. The **traceSpacing** and **traceTipLength** parameters control how [measure spine](#) spaces points along the center-line, with default values as defined by that command. The **unbendSize**, **unbendYAxis**, and **unbendGridSpacing** options set parameters for [vop unbend](#):

- The center-line is mapped to the Z axis of the unbent map; **unbendYAxis** specifies what axis of the existing volume (default **0,0,1**) is mapped to the Y axis.
- The **unbendSize** value (default **1.5**) is multiplied by the region diameter at the center-line midpoint to determine the width and height (X and Y dimensions) of the unbent volume around the center-line.
- The **unbendGridSpacing** value (default **1**) is multiplied by the minimum volume grid plane spacing to determine the grid plane spacing in the unbent volume.

The **sliceSpacing** values are in physical units and determine the spacing of the slice images taken from the unbent volume, with defaults equal to 10 times its grid spacing. The **xyTrim** parameter is the fraction (default **0.3**) of the unbent volume width and height that should be trimmed. It limits the range of slices perpendicular to the X and Y axes. The **panelAspect** parameter (default **0.5**) is the minimum image aspect ratio (width divided by height) for each of the three panels of tiled slice images; it controls the numbers of rows and columns in each panel. The **imageSpacing** parameter specifies the number of pixels of padding to leave between the three panels (default **20**).

By default, the image will be displayed in a separate window with several buttons:

- **Fit** - resize image as large as possible to fit within the current window dimensions
- **Full** - show the image at full size with scrollbars on the window if the image is larger than will fit in the window
- **Window** - adjust the window size to match the image size
- **Save** - save image as JPEG, PNG, or TIFF file

The separate image window can be suppressed by setting **showImage** to **false**. The image

is assigned as the region attribute named **slices** and can be accessed from the [Region Attributes dialog](#).

See also: [vop tile](#)

- **segment unbin** *binseg fulldata*

From a segmentation model (*binseg*) based on a binned [volume](#) data set, generate a new segmentation model for the corresponding full [volume](#) data set (*fulldata*).

See also: [vop bin](#), [Volume Filter](#)

The **select** command has been enhanced in Chimera and now has dual functions. The [newer function](#) is [selection](#). The [older function](#) is [activation for motion](#).

## Selection

### Usage:

**select** [atom-spec](#)

### Usage:

**~select** [atom-spec](#)

### Usage:

**select** [ **up** | **down** | **invert** | **invert sel** ]

The newer function of **select** is [selection](#) of atoms and/or [surface models](#). The [atom-spec](#) can specify any combination of [atoms](#) and [surfaces](#); a blank specification selects all. The effect of “**select** [atom-spec](#)” on an existing selection depends on the [selection mode](#). **~Select** removes the specified atoms from the selection regardless of [selection mode](#).

Keyword arguments:

- **up** - [broaden](#) the selection
- **down** - [narrow](#) the selection
- **invert** - [invert](#) the selection within all models, like **Select... Invert (all models)** in the menu
- **invert sel** - [invert](#) the selection within models containing selections only, like **Select... Invert (selected models)** in the menu

Examples:

```
select #1
```

- select all atoms and bonds in model 1

```
~select :/isHelix
```

- deselect any atoms in amino acid residues in helices

See also the [keyboard shortcuts](#) **rn**, **rp**, **ri** for shifting a selection to the next residue in a chain, the previous residue, and to include intervening residues, respectively. These can be used as commands with **ac** (for example: **ac rn**). See also: [namesel](#), [zonesel](#), [arrow key shortcuts](#)

## Activation for Motion

### Usage:

`select [ all | model_number(s) ]`

### Usage:

`~select [ all | model_number(s) ]`

The original function of `select` is indicated by use of the keyword `all` or the *absence* of a pound sign `#` before *model\_number(s)*. This function, now usually called [activation](#), is to control whether a model can be [manipulated interactively](#). Activation/deactivation allows manually moving one model relative to another. The activation status of models can also be controlled using the [Model Panel](#) or checkboxes under the [Command Line](#).

The *model\_number(s)* argument should be one or more model numbers or ranges separated by commas, *not preceded by #*. The keyword `all` indicates that all models should be activated. `~Select all` will then revert to the previous activation state. This feature is convenient for switching back and forth between moving models relative to one another and global motion of all models.

Examples:

```
select 1,5-8
```

- activate models 1 and 5 through 8 for motion

See also: [split](#), [movie-related commands](#)

**Usage:**

**set** *keyword*

**Usage:**

( **~set** | **unset** ) *keyword*

**Usage:**

**set** *keyword value*

The following keywords indicate on/off toggles, where **set** *keyword* enables the option and **~set** *keyword* or **unset** *keyword* disables it:

- **autoColor** - new color for each new model
- **bgTransparency** - transparent background
- **depthCue** - front-to-back shading (mist)
- **echo** - echoing lines in command files as they are executed
- **flatTransparency** - non-angle-dependent transparency
- **fullscreen** - make graphics window fill entire screen
- **independent** - make models rotate about individual centers
- **shadows** - interactive shadows
- **silhouette** - outlines
- **singleLayer** - single-layer transparency

The following are used for value assignments with **set** *keyword value*:

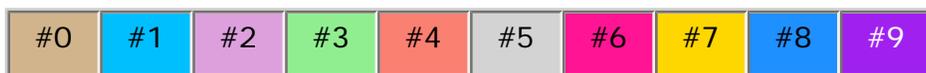
- **bgColor** - background color
- **dcColor** - depth-cueing color
- **dcStart** - front of depth-cueing ramp
- **dcEnd** - back of depth-cueing ramp
- **fieldOfView** - angular field of vision
- **maxFrameRate** - target redraw rate
- **projection** - whether to use **perspective** or **orthographic** projection
- **silhouetteColor** - outline color
- **silhouetteWidth** - outline thickness
- **subdivision** - fineness of facets (triangles) in balls, spheres, sticks, ribbons
- **subdivisionPixels** (can also be **~set** or **unset**) - level-of-detail adjustment in triangle fineness

Keywords can be truncated to unique strings, and their case does not matter. Where multiple keywords start with the same word, an ambiguous truncation will be interpreted as the simplest keyword (for example, **sil** will be interpreted as **silhouette**).

See also: [preset](#), [linewidth](#), [lighting](#), [setattr](#), [Effects](#), [tips on preparing images](#), [movie-related commands](#)

- **set autoColor**

Whether to give each newly opened model a different color; same as **use new color...** in the [New Molecules preferences](#). For example, if the background is solid **black** or **white**:



See also: [modelcolor](#)

- **set bgColor** *color*

Sets the [background color](#); *color* can be **none** or any [color name](#) that specifies a single color. Any transparency in the color will be ignored. Equivalent to [background solid](#).

- **set bgTransparency**

Whether to include opacity values (alpha) in PNG and TIFF image files [saved](#) from Chimera, to facilitate combining the images with different backgrounds in image-editing applications. With background transparency, if the Chimera [background](#) is a single solid color, it will be completely invisible in the saved images. (**Note:** TIFF images with background transparency may not be interpreted correctly by Adobe Photoshop.) Background transparency can also be enabled in the [Save Image dialog](#) or with the startup option [--bgopacity](#).

- **set depthCue**

Whether to use [depth-cueing](#), front-to-back shading of the scene.

- **set echo**

Whether to echo the line number and contents of each line in subsequently opened [Chimera command files](#) to the [Reply Log](#) and [status line](#) as that line is executed. Blank lines and comments (lines starting with #) are included.

- **set dcColor** *color*

Sets the depth-cueing color; *color* can be **none** or any [color name](#) that specifies a single color. Any transparency in the color will be ignored.

- **set dcStart** *start*

- **set dcEnd** *end*

Set the position of the depth-cueing ramp. Depth-cueing shading increases linearly from *start* to *end*, each expressed as a position relative to the front (0.0) and back (1.0) [global clipping planes](#).

- **set fullscreen**

Whether to use [fullscreen mode](#), in which the graphics window fills the entire screen (available in Windows, Linux, and Mac-X11 versions of Chimera).

- **set independent**

Whether to rotate models about their individual centers or a single collective center; **set independent** changes the [center of rotation method](#) to **independent** while **~set independent** changes the [center of rotation method](#) to **front center**.

See the [video mini-example](#). See also: [cofr](#), [focus](#), [tile](#), [Rotation](#)

- **set fieldOfView *angle***

Set [horizontal field of view](#) in degrees (allowed range  $0^\circ < \textit{angle} < 180^\circ$ ).

See also: [Camera](#)

- **set flatTransparency**

Whether to make apparent transparency independent of the viewing angle. Otherwise, transparent triangles (forming objects as well as surfaces) will appear more opaque when viewed edged-on than when viewed face-on.

- **set maxFrameRate *N***

Set the maximum frame rate in Chimera to *N* frames per second (default **60**). Although *N* is the target value, the actual rate may be lower depending on the rendering speed of the computer and the complexity of the scene. The actual frame rate can be reported in the [status line](#) using the [accelerator](#) **rt** (command **ac rt**).

See also: [frame rate and movie speed](#)

- **set projection *mode***

Which *mode* of [projection](#) to use: **perspective**, which makes farther-away objects smaller, or **orthographic**, which does not.

See also: [Camera](#)

- **set shadows**

Whether to show [interactive shadows](#).

- **set silhouette**

Whether to show [silhouettes](#), outlines that emphasize borders and discontinuities. This setting is global (applies to all models), but when it is on, silhouettes can be toggled for individual models with the command [setattr](#), the [Selection Inspector](#), or the [molecule model attributes panel](#).

- **set silhouetteColor** *color*

Sets the color of silhouettes; *color* can be **none** (equivalent to **black**) or any [color name](#) that specifies a single color.

- **set silhouetteWidth** *linewidth*

Sets the linewidth of silhouettes.

- **set singleLayer**

Whether to render only the topmost layer of all transparent items. Showing only the topmost layer rather than all transparent layers simplifies the display and is recommended for de-emphasizing transparent parts.

- **set subdivision** *value*

Sets [subdivision](#) level, where a higher value increases the apparent smoothness of stick, ball-and-stick, sphere, and ribbon [representations](#) by increasing how many planar facets are used to approximate them.

- **set subdivisionPixels** *pixels-per-length*

By default, the fineness of faceting of stick, ball-and-stick, sphere, and ribbon [representations](#) is adjusted dynamically based on:

1. the [subdivision](#) parameter
2. the pixel width of the graphics window divided by the width of the view at the [center of rotation](#)

Setting **subdivisionPixels** substitutes the specified static *pixels-per-length* for the second value above and suspends dynamic adjustment. This can enhance responsiveness to manual zooming (by precluding further subdivision calculations) and can speed up rendering by avoiding overly fine subdivisions that may occur with large windows and/or highly zoomed-in views. Setting *pixels-per-length* to zero or a negative value restores the dynamic subdivision behavior, as does **~set subdivisionPixels**.

**Usage:**

`setattr ( a | r | m | M | b | p | g | s ) attr_name attr_value atom-spec`

**Usage:**

`~setattr ( a | r | m | M | b | p | g | s ) attr_name atom-spec`

**setattr** sets an [attribute](#) (existing or new) to a specified single value. The attribute may remain multi-valued, however, as the assignment may affect only a subset of the entities with the attribute. To assign multiple values simultaneously, use [defattr](#). For *attr\_name* (attribute name) lookup, see the [balloon help](#) of [attribute inspector](#) dialogs ([more...](#)).

Although **setattr** does not contain restrictions on which attributes can be affected, it may not make sense to change certain attributes. It is possible, however, to set an attribute that is also used in the [atom-spec](#) because the specification will be evaluated before the attribute value is changed.

The attribute level can be specified as:

- **a** (atom) - set the value for all atoms in [atom-spec](#) (see [atom attributes](#))
- **r** (residue) - set the value for all residues with any atoms in [atom-spec](#) (see [residue attributes](#))
- **m** (molecule model) - set the value for all molecule models with any atoms in [atom-spec](#) (see [molecule model attributes](#))
- **M** (Models of all relevant types) - set the value for all specified models that can have the attribute (for example, the text-valued attribute **note** is an annotation that can be shown for any type of model in the [Model Panel](#))
- **b** (bond) - set the value for all bonds with both flanking atoms in [atom-spec](#)
- **p** (pseudobond) - set the value for all [pseudobonds](#) with both flanking atoms in [atom-spec](#)
- **g** (pseudobond group) - set the value for all [pseudobond groups](#) containing pseudobonds with both flanking atoms in [atom-spec](#) (see the [pseudobond group attributes panel](#))
- **s** (surface) - set the value for the specified [molecular surface](#) models (see the [molecular surface attributes panel](#)). Molecular surface attributes include:
  - **probeRadius**
  - (vertex) **density**
  - **allComponents** (true or false) - whether to include multiple disconnected parts such as interior bubbles

For other (nonmolecular) [surface models](#), see the command [sop](#).

The [attribute name](#) (*attr\_name*) can be existing or new. It should be an alphanumeric string without spaces; it can include underscores, but cannot begin with a digit, underscore, or capital letter. An attribute with *attr\_name* ending in **color** (case-independent) will be interpreted as a color-valued attribute.

If the attribute already exists, *attr\_value* should be of the appropriate type. Allowed attribute types:

- real number (float)

setattr

- integer
- boolean - **true** or **false**, case-independent, or **0** or **1**
- color - **none** or any [color name](#) that specifies a single color
- string - a string of any characters except tabs; strings with spaces, or that might be interpreted as one of the other attribute types, should be enclosed in single or double quotes

~**setattr** sets the value to **None** (no value), if possible. Certain attributes (for example, atom **radius**) and attribute types (for example, strings) cannot have no value. String-valued attributes such as atom **label** should be set to an empty string with "" instead.

Examples:

```
setattr a label "pick me!" element.S
```

- label all sulfur atoms with the text **pick me!**

```
setattr s density 5.0
```

- set molecular surface vertex density to 5.0

```
setattr r isHelix false :30-45.a
```

- remove any helix assignment from residues 30-45 in chain A (without changing the structure itself)

```
setattr p color hot pink ions za<5
```

- color pseudobonds between atoms within 5 Å of [ions](#) hot pink

```
setattr g lineWidth 3
```

- use linewidth 3 for all pseudobond groups

**Usage:****shape** *shape* [options](#)

The **shape** command creates a [surface model](#) of the specified shape. See also: [define](#), [mask](#), [meshmol](#), [ribbon](#), [sop](#), [geometric objects](#)

Examples:

```
shape sphere center ligand color dodger blue radius 10
shape cylinder center 12.5,15,15 coord 1 rad 2 caps true
shape icosahedron radius 100 ori 2n3 lattice 1,2 sphere 0.2 line 2
```

The *shape* can be:

- [sphere](#)
- [ellipsoid](#)
- [cone](#)
- [cylinder](#)
- [icosahedron](#)
- [rectangle](#)
- [ribbon](#) - a ribbon connecting a series of atoms or [markers](#)
- [tube](#) - a tube connecting a series of atoms or [markers](#)
- [boxpath](#) - zigzag “beams” connecting a series of atoms or [markers](#)

Option keywords and their subkeywords can be truncated to unique strings, and their case does not matter. Synonyms for true: True, 1. Synonyms for false: False, 0. A vertical bar “|” designates mutually exclusive options, and default settings are indicated with **bold**.

- **shape sphere** [ **radius** *r* ] [general-options](#)

The default **radius** *r* is **10**.

- **shape ellipsoid** [ **radius** *r* | *rx,ry,rz* ] [general-options](#)

The default **radius** *r* is **10** along each axis. If a single value is specified, the result is a sphere. Specifying three values (*rx,ry,rz*) sets ellipsoid radii along the X, Y, and Z axes, respectively (see [coordinateSystem](#)).

See also: [measure inertia](#)

- **shape cone** [ **radius** *rbase* ] [ **height** *h* ] [ **topRadius** *rtop* ] [ **caps** true|false ] [general-options](#)

The default **radius** *r*<sub>base</sub> and **height** *h* are **10** and **40**, respectively. The default orientation is with the axis of symmetry (**height** dimension) along the Z axis and top towards +Z (see [coordinateSystem](#)). The **topRadius** *r*<sub>top</sub> defaults to **0**, giving a pointed cone, but values > 0 can be used to produce a truncated cone. The **caps** option indicates whether to cap the end(s) of the cone or leave them open.

- **shape cylinder** [ **radius** *r* ] [ **height** *h* ] [ **caps** true|false ] [general-options](#)

The default **radius** *r* and **height** *h* are **10** and **40**, respectively. The default orientation is with the axis of symmetry (**height** dimension) along the Z axis (see [coordinateSystem](#)). The **caps** option indicates whether to cap the cylinder or leave it open-ended.

- **shape icosahedron** [ **radius** *r* ] [ **orientation** *type* ] [ **sphereFactor** *f* ] [ **lattice** *h,k* ] [general-options](#)

The **radius** *r* is the distance from the center to a 5-fold vertex (default **10**). The **orientation** *type* can be:

- **222** (default) - with two-fold symmetry axes along the X, Y, and Z axes
- **2n5** - with two-fold symmetry along X and 5-fold along Z
- **n25** - with two-fold symmetry along Y and 5-fold along Z
- **2n3** - with two-fold symmetry along X and 3-fold along Z
- **222r** - same as 222 except rotated 90° about Z
- **2n5r** - same as 2n5 except rotated 180° about Y
- **n25r** - same as n25 except rotated 180° about X
- **2n3r** - same as 2n3 except rotated 180° about Y

The **sphereFactor** option allows generating a shape that is an interpolation between an icosahedron and a sphere of equal radius. The factor *f* is the weight of the sphere component in the interpolation and can range from **0** (default, icosahedron) to **1** (sphere). The interpolation only affects vertex positions and will not generate curved mesh lines or curved surface triangles. The **lattice** option allows showing the icosahedron surface with hexagons and pentagons instead of triangles. A shape with icosahedral symmetry (like many virus capsids) can be idealized as a sheet of hexagons in which curvature is introduced by replacing certain hexagons with pentagons, as in a geodesic dome. The pentagons occupy the points of the icosahedron, while the indices *h* and *k* refer to the number and arrangement of hexagons in each face ([details...](#)). Hexagons are bent where they cross from one triangular face to another. The indices *h* and *k* can each be zero (but not both zero) or a positive integer. Surface generation can be slow if large values are used.

See also: [hkcage](#), [Icosahedron Surface](#), [Cage Builder](#)

- **shape rectangle** [ **width** *w* ] [ **height** *h* ] [ **widthDivisions** *dw* ] [ **heightDivisions** *dh* ] [general-options](#)

The **width** *w* and **height** *h* are the X and Y dimensions of the rectangle (both default to **10**). The number of surface points in the rectangle along the X dimension will equal 1 + *dw* and along the Y dimension will equal 1 + *dh*. Both the **widthDivisions** *dw* and **heightDivisions** *dh* default to **10**, and these settings override the **divisions** [general option](#).

- **shape ribbon** [atom-spec](#) [ **width** *w* ] [ **yaxis** *axis* ] [ **twist** *t* ] [ **segmentSubdivisions** *divisions* ] [ **bandLength** *length* ] [ **followBonds** true|false ] [general-options](#)\*

A **ribbon** is a smooth path with flat cross-section that connects a series of atoms or [markers](#). The default **width** *w* is **1.0**. The **yaxis** *axis* (default none) and **twist** *t* control ribbon orientation and how it varies along the path. If neither is specified, the ribbon normal varies along the path so that there is no local twist. If an *axis* is given, the ribbon normal is aligned with that axis as closely as possible within the constraints of the path. A constant **twist** *t* to be applied at each point along the path can also be specified (default 0°). The *axis* can be given as:

- **x** - X-axis
- **y** - Y-axis
- **z** - Z-axis
- *x,y,z* (three values separated by commas only) - an arbitrary vector
- an [atom-spec](#) of exactly two atoms (not necessarily bonded or in the same model) or one bond. A bond can only be specified by [selecting](#) it and using the word **selected**, **sel**, or **picked**; any atoms also selected at the time will be ignored.

Coordinate specifications of *axis* are interpreted in the system of the model containing the first atom in the path. The **segmentSubdivisions** setting controls how many straight segments are used to form the curve between a consecutive pair of atoms; higher values give smoother curves. The number of straight segments forming the curve between a pair of atoms will equal *divisions* + 1 (default **10** + 1 = 11). The **bandLength** option specifies what *length* of ribbon centered on an atom should be colored to match that atom (default **0.0**). The **followBonds** option indicates whether the ribbon should follow the directions of the bonds connecting the atoms; the option should be set to false for atoms that are not bonded.

\*The [general options](#) **center**, **rotation**, **qrotation**, **coordinateSystem**, and **slab** do not apply to **ribbon** shapes.

See also: [ribbon](#), [ribspline](#), [Volume Tracer](#)

- **shape tube** [atom-spec](#) [ **radius** *r* ] [ **segmentSubdivisions** *divisions* ] [ **bandLength** *length* ] [ **followBonds** true|false ] [general-options](#)\*

A **tube** is a smooth path with circular cross-section that connects a series of atoms or [markers](#). The default **radius** *r* is **1.0**. The **segmentSubdivisions** setting controls how many straight segments are used to form the curve between a consecutive pair of atoms; higher values give smoother curves. The number of straight segments forming the curve between a pair of atoms will equal *divisions* + 1 (default **10** + 1 = 11). The **bandLength** option specifies what *length* of tube centered on an atom should be colored to match that atom (default **0.0**). The **followBonds** option indicates whether the tube should follow the directions of the bonds connecting the atoms; the option should be set to false for atoms that are not bonded.

\*The [general options](#) **center**, **rotation**, **qrotation**, **coordinateSystem**, and **slab** do not

apply to **tube** shapes.

See also: [ribbon](#), [ribspline](#), [Volume Tracer](#)

- **shape boxpath** [atom-spec](#) [ **width** *w* ] [ **twist** *t* ] [ **reportCuts** true|false ] [ **cutScale** *s* ] [general-options](#)\*

A **boxpath** is a 3D zigzag of “beam” segments connecting a series of atoms or [markers](#).

The beam has a square cross-section of **width** *w* (default **1.0**), and the **twist** *t* (default **0.0°**) sets the rotational orientation of the segments about their long axes. To make a physical replica of the boxpath (see the [sculpture at DePauw](#)), the segments could be generated by angled cuts along a single straight beam. Setting **reportCuts** to **true** gives the locations of such cuts along the four edges of a hypothetical beam; the **cutScale** *s* (default **1.0**) is a factor for converting these distances from Å to the appropriate physical units.

\*The [general options](#) **center**, **rotation**, **qrotation**, **coordinateSystem**, **divisions**, and **slab** do not apply to **boxpath** shapes.

See also: [Box-Beam Protein Sculpture Design](#)

## General Options

**center** [atom-spec](#) | *x,y,z*

The **center** can be set to the center of specified atom(s) or to a point (*x,y,z*) in the coordinate system specified with [coordinateSystem](#). The default center is **(0,0,0)**.

**rotation** *ax,ay,az,angle*

Rotate the shape *angle* degrees around the specified axis (*ax,ay,az*) in the coordinate system specified with [coordinateSystem](#). The default is no rotation.

**qrotation** *qx,qy,qz,qw*

Apply the rotation specified as a quaternion in the coordinate system specified with [coordinateSystem](#). The default is no rotation.

**coordinateSystem** *N*

The coordinate system is indicated by reference model ID number, optionally preceded by #. The coordinate system is used for interpreting the [center](#), [rotation](#), [qrotation](#), and [ellipsoid radius](#) arguments. The default is the coordinate system of the new or existing surface model (see [modellId](#)).

**color** *colorname*

Set the surface color to *colorname*, which can be any [color name](#) that specifies a single color (default **gray**).

**modelName** *name*

Name to use for the newly created surface model. If no name is specified, a generic shape

name is used (for example, "sphere").

**modelId** *N*

Open the surface as model number *N* (an integer, optionally preceded by #). Submodel specifications #*N.N* (# required) can also be given. The default is the lowest unused number. If a [surface model](#) with the same ID number already exists, the new shape is added as another [surface piece](#).

**divisions** *d*

Set the fineness of surface triangulation; *d* is the number of square mesh cells around the circumference (default **72**). The number of triangles around the circumference is roughly 2-4 times higher, depending on the shape and on the value of *d*. This setting does not apply to an [icosahedron](#) with **lattice** specified, as it will be shown with the indicated numbers of hexagons and pentagons rather than with triangles.

**mesh** true|false

Whether to display the surface as a mesh or as a solid surface. The default is **false** except for an [icosahedron](#) with **lattice** specified.

**linewidth** *width*

The *width* is the pixel linewidth of mesh display (default **1.0**).

**slab** *width* | *d1,d2*

The **slab** option indicates that a shell or slab of finite thickness should be created instead of a single layer of surface. If a single value (*width*) is supplied, the inner and outer layers of the slab will be offset from the nominal radius *r* by  $\pm\frac{1}{2}(\textit{width})$ . Alternatively, two values separated by a comma but no spaces can be used to specify the offsets of the two layers independently. Offsets can be positive (outward) or negative (inward).

show

[Chimera Commands Index](#)

**Usage:**

**show** [atom-spec](#)

**Usage:**

**~show** [atom-spec](#)

**Show** displays the specified atoms and removes all others in those models from the display. Models not involved in [atom-spec](#) will be unaffected. Note that this differs from the [display](#) command in that the **show** command displays *only* those atoms specified (and will undisplay all others within the same model). **~Show** removes the specified atoms from the display.

See also: [chain](#), [display](#)

sleep

[Chimera Commands Index](#)

**[Usage:](#)**

**sleep** *number\_of\_seconds*

**Sleep** suspends [command script](#) execution for *number\_of\_seconds*. Such pauses will not be included in any [movie](#) that is being recorded, because no new frames are drawn during that time; to generate pauses in a movie, [wait](#) should be used instead.

See also: [pause](#)

**Usage:**

**solvate box|oct** *solvent\_model box\_size*

**Usage:**

**solvate cap** *solvent\_model center radius*

**Usage:**

**solvate shell** *solvent\_model thickness*

The **solvate** command adds solvent around molecule models using [AmberTools](#). It is the command-line implementation of [Solvate](#). Thanks to Wei Zhang for adding this command.

Deleting any existing solvent and ions and closing other molecule models is recommended before running **solvate**. Explicit hydrogens must be present for solvent addition; if the molecule model does not include hydrogens, a dialog for running [AddH](#) beforehand will appear.

One of the following methods must be specified:

- **box** - rectangular box with edges no closer than *box\_size* (Å) to any atom in the solute
- **oct** - truncated octahedron with edges no closer than *box\_size* (Å) to any atom in the solute
- **cap** - spherical cap with the specified *center* and *radius* (Å). The *center* can be:
  - a residue identified by the residue ID (starting from 1)
  - an atom identified by a string like **ddd.xxx**, where **ddd** is the residue ID and **xxx** is the name of the atom
- **shell** - a layer of solvent extending *thickness* (Å) from the solute. The shell will be irregular in shape since it reflects the contours of the solute.

The *solvent\_model* can be any of the following:

- **CHCL3BOX** - chloroform
- **MEOHBOX** - methanol
- **NMABOX** - N-methylacetamide
- **POL3BOX** - POL3 water model
- **SPCBOX** - SPC/E water model
- **SPCFWBOX** - SPC/Fw water model
- **QSPCFWBOX** - qSPC/Fw water model
- **TIP3PBOX** - TIP3P water model
- **TIP3PFBOX** - TIP3P/F water model
- **TIP4PBOX** - TIP4P water model
- **TIP4PEWBOX** - TIP4P/Ew water model

See the [AmberTools](#) documentation for further details.

See also: [addh](#), [delete](#), [minimize](#), [Solvate](#)

## Usage:

### *sop operation arguments*

The **sop** command performs operations on one or more [surface models](#). See also: [surface](#), [scolor](#), [setattr](#), [shape](#), [define](#), [volume](#) (surface and mesh options), [vop](#), [surface attributes](#)

The *operation* can be:

- [cap](#) - adjust capping of clipped surfaces
- [clip](#) - spherically clip volume isosurfaces, with curved caps
- [finerMesh](#) - subdivide surface mesh
- [hideDust](#) - hide small blobs (disconnected parts)
- [hidePieces](#) and [showPieces](#) - hide/show specified [pieces](#) (independently selectable parts)
- [smooth](#) - smooth surfaces by moving each vertex toward the average position of its neighbors
- [split](#) - split disconnected parts of a surface into independently selectable [pieces](#)
- [transform](#) - scale, rotate, and shift surfaces
- [zone](#) - show only surface parts within a cutoff distance of specified atoms

Operation keywords and their sub-keywords described below can be truncated to unique strings, and their case does not matter. Surface models can be specified by a model number or range of model numbers preceded by #, or simply # to indicate all applicable surfaces. Synonyms for true: True, 1. Synonyms for false: False, 0. A vertical bar "|" designates mutually exclusive options, and default settings are indicated with **bold**.

- **sop cap** on | off [ **mesh** true | false ] [ **offset** *d* ] [ **subdivision** *f* ] [ **color** same | *color-name* ]

Adjust capping, in which planar caps are drawn to hide the interior of a [surface model](#) sliced by a [clipping plane](#). Capping can be toggled on or off, and caps can be shown as a solid surface or as **mesh**. The **offset** distance *d* is how far to displace caps from the clipping plane (default **0.01** Å). Some displacement is needed, or the cap itself can be clipped (invisible) depending on floating-point rounding errors and the specific graphics hardware being used. If *d* is large, however, the mismatch between the cap and the cut edge of the surface will be evident. The **subdivision** factor *f* indicates how finely to subdivide cap surfaces (default **1.0**, triangles approximately the same size as in the surface being capped); larger values yield smaller triangles and finer color gradations (when the cap is multicolored, for example with [Surface Color](#)), but increase computational demands. There is no reason to increase the subdivision factor when caps are a single solid color. The cap **color** can be the **same** as that of the clipped surface model ([details](#)) or indicated with any [color name](#) that specifies a single color. See also: [Surface Capping](#)

- **sop clip** *volume-model(s)* [ **center** [atom-spec](#) | *x,y,z* ] [ **radius** *r* ] [ **coordinateSystem** *N* ] [ **mesh** true | false ] [ **color** *color-name* ] [ **replace** true | false ]

Spherically clip [volume](#) isosurfaces, with curved caps covering holes. The clipping sphere

**center** and **radius** can be specified; defaults are the geometric center of the isosurface and 40% of the largest dimension of its bounding box, respectively. The coordinate system for interpreting center  $x,y,z$  specifications is indicated by reference model ID number preceded by #. Caps can be shown as a solid surface or as **mesh**. Cap **color** defaults to the same as the isosurface, but can be indicated with any [color name](#) that specifies a single color, except with spaces collapsed (e.g., **hotpink** instead of **hot pink**). The **replace** option indicates whether to remove any pre-existing spherical clipping caps.

- **sop finerMesh** *surf-model(s)* **spacing** *d* [surf-creation-options](#)

Subdivide the surface mesh to achieve the specified vertex **spacing** *d*. If the input surface already has a finer mesh than what is specified, it will simply be duplicated.

- **sop hideDust** *surf-model(s)* **size** *s* [ **metric** **size** | **area** | **volume** | **sizeRank** | **areaRank** | **volumeRank** ] [ **update** **true** | **false** ]
- **~sop hideDust** *surf-model(s)*

Hide smaller blobs (disconnected parts) of a surface. One use is to simplify the display of noisy [volume data](#). The cutoff **size** for hiding must be specified; blobs smaller than *s* will be hidden, where the units depend on the **metric**:

- **size** (default) - bounding box dimension (largest of X, Y, Z)
- **area** - surface area
- **volume** - enclosed volume
- **sizeRank** - rank by largest **size** (a value of *N* indicates showing the *N* largest blobs by the **size** criterion)
- **areaRank** - rank by largest **area**
- **volumeRank** - rank by largest **volume**

Size measurements will include any blob parts that have been undisplayed or hidden by [zoning](#) or [clipping](#), and blobs at or above the cutoff size will be displayed completely (although possibly still clipped) even if they had been hidden beforehand. The **update** option indicates whether to update hiding automatically if the surface is changed (default **true**). The command **~sop hideDust** stops hiding blobs of the specified surfaces. See also: [Hide Dust](#)

- **sop hidePieces** *surf-piece(s)*
- **sop showPieces** *surf-piece(s)*

Hide and show [pieces](#) (independently selectable parts) of a surface. Surface pieces can be specified by model number or as a [selection](#) ([details...](#)).

- **sop smooth** *surf-piece(s)* [ **factor** *f* ] [ **iterations** *i* ] [surf-creation-options](#)

Smooth [surface pieces](#) by moving each vertex toward the average position of its neighboring vertices (those connected by triangle edges). In each of *i* iterations (default **2**),

each vertex is moved a fraction  $f$  (default **0.3**) of the way toward the average position of its neighbors. This will generally “shrink” a surface, *i.e.*, smoothing an enclosed surface will make it enclose a smaller volume. A [molecular surface](#) can be smoothed without shrinkage by instead increasing its vertex density (for example, command: [setattr s density 5](#)). See also: [smoothness tips](#)

- **sop split** *surf-model(s)* [surf-creation-options](#)

Copy a surface model into a new model in which the disconnected blobs (components) are independently selectable [pieces](#).

- **sop transform** *surface(s)* [ **scale**  $s$  | **radius**  $r$  ] [ **rotate**  $angle$  ] [ **move**  $x,y,z$  ] [ **center**  $x,y,z$  | [atom-spec](#) ] [ **axis**  $x$  |  $y$  |  $z$  |  $x,y,z$  | [atom-spec](#) ] [ **coordinateSystem**  $N$  ]

Scale, rotate, and shift surface models and/or [pieces](#). Surface pieces can be specified by model number or as a [selection](#) ([details...](#)). Scaling about the center is applied first, then rotation about the center and axis, and then translation (shifting). Scaling can be done with a multiplicative **scale** factor  $s$ , or such that the largest distance from a surface vertex to the center is **radius**  $r$  (see the [video mini-example](#)). The **rotate** option indicates a rotation of  $angle$  degrees about the center and axis, and **move** indicates a shift of  $x$ ,  $y$ , and  $z$  distance units along the X, Y, and Z axes, respectively. The **center** for scaling and rotation defaults to the area-weighted center of the surface, but can be given as  $x,y,z$  coordinates or an [atom-spec](#). The rotation **axis** can be given as:

- **x** - X-axis
- **y** - Y-axis
- **z** (default) - Z-axis
- $x,y,z$  (three values separated by commas only) - an arbitrary vector
- an [atom-spec](#) of exactly two atoms (not necessarily bonded or in the same model) or one bond. A bond can only be specified by [selecting](#) it and using the word **selected**, **sel**, or **picked**; any atoms also selected at the time will be ignored.

The coordinate system for interpreting center, axis, and shift  $x,y,z$  specifications is indicated by reference model ID number preceded by #. By default, the coordinate system of the first specified surface is used.

- **sop zone** *surf-model(s)* [atom-spec](#) *radius* [ **bondPointSpacing**  $s$  ] [ **maxComponents**  $N$  ] [ **update** **true** | **false** ]
- **~sop zone** *surf-model(s)*

Show only parts of the surface within  $radius$  Å of any atom in [atom-spec](#). If

**bondPointSpacing**  $s$  is specified, use points along bonds in addition to the atoms to define the zone. The points along the bonds will be placed  $s \times$  ([bond radius](#)) apart. [Link radii](#) in [Volume Tracer](#) are equivalent to bond radii, except when a link radius is 0.0, the corresponding bond radius is 1.0. The **maxComponents** option indicates hiding all but the  $N$  largest connected surface patches in the zone; by removing the visual clutter of small patches, this can significantly enhance viewing of pocket surfaces. Ranking to determine

the largest is based on maximum dimension (along X, Y, or Z for a given patch), *including* any portions within the zone that are hidden by [clipping](#). If **maxComponents** is not supplied, all patches within the zone will be shown. The **update** setting indicates whether to update the zone automatically if the surface changes shape. The command **~sop zone** stops hiding the surface parts outside the zone. See also: [Surface Zone](#)

## Surface Creation Options

The following options apply to operations that create a new surface model.

### **modelId** *N*

Open the new surface as model number *N* (an integer, optionally preceded by #). The default is the lowest unused number.

### **inPlace** true|false

Whether to overwrite the existing surface in Chimera instead of creating a new one.

**Usage:**

**split** [ *model\_number(s)* ] [ **chains** ] [ **ligands** ] [ **connected** ] [ **atoms** [atom-spec](#) ]<sub>n</sub>

The **split** command partitions a molecule model into separate [submodels](#), by default according to chain. The *model\_number(s)* argument can be one or more model numbers or ranges separated by commas and preceded by #. If no models are specified, all molecule models will be affected.

How to split can be indicated with one or more of the following:

- **chains** - by chain ID
- **ligands** - each [ligand](#) in a separate submodel, the remaining atoms in another
- **connected** - into covalently bonded sets of atoms
- **atoms** [atom-spec](#) - can be used multiple times in the same command, for example:

**split #0 atoms :1-100 atoms :101-150 atoms :151-200**

... to place each set of specified atoms, as well as the set of unspecified atoms, in a separate submodel.

If none of the above are given, the **chains** method will be used. If there is nothing to split, the original model will be left unaltered.

Placing parts of a model in separate submodels allows for more independent control. For example, using the [Model Panel](#), each can be activated/deactivated for motion or hidden/shown at the model level (without changing atomic display settings) independent of the others.

There is no way to undo this operation except by reopening the original structure.

See also: [combine](#), [surfcats](#), [select](#), the [accelerator](#) Sc, the [Model Panel](#), [surface calculation failures and workarounds](#)

start

[Chimera Commands Index](#)

### **Usage:**

**start** *tool\_name*

**Start** invokes the [Chimera tool](#) named *tool\_name*. The same pattern of capitalization and spaces (if any) as shown in the [Tools menu](#) should be used.

Example:

```
start side View
```

- start the [Side View](#)

See also: [open](#)

**Usage:****stereo** *mode***Usage:**~**stereo**

**Stereo** controls the [camera mode](#) of the graphics window ([details...](#)). The command ~**stereo** restores mono viewing. Possible values of *mode* are listed in **bold**:

- **mono** or **off**, no stereo (can also be set with the command ~**stereo**)
- **left-eye** (equivalents: **left**, **left eye**)
- **right-eye** (equivalents: **right**, **right eye**)
- **cross-eye** (equivalents: **cross**, **crosseye**, **cross eye**)
- **wall-eye** (equivalents: **wall**, **walleye**, **wall eye**)
- **red-cyan** or **anaglyph** - overlapping left-eye and right-eye views in different colors, to be viewed with “glasses” (often inexpensive cardboard/plastic) with colored filters
- **green-magenta** or **trioscopic** - similar to red-cyan, except with different colors and trade-offs
- **sequential** (equivalents: **seq**, **on**) - rapid flickering between left-eye and right-eye views, to be viewed with special synchronised glasses. Sequential stereo needs [special hardware](#).
- **reverse sequential** (equivalents: **reverse seq**, **rev seq**) - as above, but with the opposite convention for the two views
- **row even** (equivalents: **row**, **row interleaved**) - row-interleaved with even rows used for the right-eye view
- **row odd** - row-interleaved with odd rows used for the right-eye view
- **dome** [**tilt** *t-angle*] [**parallax** *p-angle*] - angular fisheye of the hemisphere in front of the camera, with [horizontal field of view](#) locked to 90°, optional tilt, and optional parallax:

When no tilt is specified, the Chimera Z-axis appears at the dome zenith (the center of the circular view in the graphics window). Setting the dome tilt angle to 60° places the Chimera Z-axis 30° above the dome horizon, the approximate focus of attention of an audience in a planetarium dome with unidirectional seating. In the graphics window, this means the Z-axis direction is  $\frac{1}{2}$  of the way from the bottom of the circular display.

Left- and right-eye stereo views for dome display can be generated by specifying parallax angles of opposite signs. The views are rotated by *p-angle* degrees about the Y (vertical) axis through the center of the focal plane. For example, views recorded with dome parallax  $\pm 5^\circ$  can be combined to give a stereo effect when viewed in the forward ( $-Z$ ) direction.

- **truncated dome** - same as the dome mode, except with the bottom of the hemisphere cut off, and no tilt or parallax options
- **dti** (equivalents: **DTI**, **dti side-by-side**, **DTI side-by-side**)

See also: startup option [--stereo](#), [Camera](#), the [Side View](#)

stop

[Chimera Commands Index](#)

### **Usage:**

**stop** [confirmed | noask | now | really | yes]

**Stop** ends the current session without saving any of its contents. Whether exiting from Chimera requires confirmation can be controlled in the [General preferences](#). The exit confirmation dialog can also be suppressed with the keyword **confirmed**, **noask**, **now**, **really**, or **yes**.

See also: [save](#)

**Usage:**

**struts** [atom-spec](#) [ **length** *maxlen* ] [ **loop** *maxloopen* ] [ **fattenRibbon** true|false ]  
[other-options](#)

**Usage:**

**~struts** [ [atom-spec](#) ]

The **struts** command adds [pseudobonds](#) to a molecular structure to strengthen it for 3D printing. Pairs of atoms within the maximum strut length of each other are considered in order of increasing distance. A strut is added if the shortest through-bond distance between the two atoms, including any existing struts, is greater than the maximum “loop” length. By default, ribbons are also thickened to make them more robust. The **~struts** command removes struts connected to the specified atoms, or all struts if no atoms are specified. See also: [export](#), the [Chimera 3D Model Gallery](#)

Protein example:

```
struts @ca length 7 loop 30 color blue
```

Example for tRNA (see the [gallery](#) for a picture):

```
struts @p length 15 loop 80 rad 1.5
```

**Options**

Keyword options, including those given above, can be used in any order and the keywords can be truncated. A vertical bar “|” designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

**length** *maxlen*

In effect, *maxlen* is the maximum length of a strut (default **7.0 Å**). Pairs of atoms in [atom-spec](#) within *maxlen* of each other are considered in order of increasing distance. A strut is added if the shortest through-bond distance between the two atoms, including any existing struts, is greater than that specified with the **loop** option.

**loop** *maxloopen*

A strut is added between a candidate pair of atoms when the shortest through-bond path between them, including any existing struts, is longer than *maxloopen* (default **30.0 Å**).

**fattenRibbon** true | false

Whether to apply a fat [scaling](#) to any displayed ribbon. Ribbon width and height can be fine-tuned with the [Ribbon Style Editor](#).

**radius** *strut-radius*

The struts will be shown as sticks of radius *strut-radius* (default 0.75 Å).

**color** *strut-color*

The *strut-color* can be any [color name](#) that specifies a single color (default 0.7,0.7,0.7,1, a medium gray), except that names with spaces should be enclosed in quote marks or their spaces stripped.

**name** *name*

Name to use for the strut [pseudobond group](#) and listing in the [Model Panel](#).

**modelId** *N*

Add the strut [pseudobonds](#) as model number N (an integer, optionally preceded by #). The default is the lowest unused number.

**replace** **true** | false

Whether to replace struts previously calculated for the same set of atoms rather than creating an additional entry in the [Model Panel](#). Regardless of this setting, struts previously determined for a different set of atoms will not be replaced.



The surfaces are created with embedded software from the [MSMS package](#), described in:

[Reduced surface: an efficient way to compute molecular surfaces](#). Sanner MF, Olson AJ, Spohner JC. *Biopolymers*. 1996 Mar;38(3):305-20.

MSMS surface calculations may fail numerically, especially on large structures; see [surface calculation failures and workarounds](#).

A molecule model is automatically divided into the following mutually exclusive categories:

Automatic Categories	
name	membership rules, in order of application
<b>solvent</b>	of the following two, the set with the greater number of residues: <ul style="list-style-type: none"> <li>“small solvent” candidate set: residues of up to 3 atoms named WAT, HOH, and DOD, plus singleton atoms (<i>i.e.</i>, not covalently bonded to other atoms) of atomic number 6-8 in single-atom residues</li> <li>“other solvent” candidate set: excluding residues in the “small solvent” set, the most prevalent type of residue that is not covalently bonded to other residues, has <math>\leq 10</math> atoms per residue, and is present in at least 10 copies in the structure</li> </ul>
<b>ions</b>	non-solvent singleton atoms other than noble gases, plus covalently bonded groups of $\leq 4$ atoms (not counting hydrogens) in the same residues as those singletons
<b>ligand</b>	singleton atoms that are noble gases; single residues or bonded sequences of residues with $< 10$ residues per bonded sequence, $< 250$ atoms, and $< 1/4$ the number of atoms in the largest bonded sequence of residues in the model; residues bonded to a chain but not included in its main sequence ( <i>e.g.</i> , retinal in rhodopsin, glycosylations)
<b>main</b>	all remaining atoms

Each [category name](#) can be used in [atom-spec](#) strings. The surface will be calculated for each entire category that contains any [specified](#) atoms, but only the surface corresponding to those atoms will be displayed. Using **surface** with a blank [atom-spec](#) displays the surface for all atoms in category **main**.

Examples:

**surface #0**

- display the surface for all categories in model 0

**surface**

- display the surface for category **main** in all models

### **surface ligand**

- display the surface for category **ligand** in all models

### **surface #1:5-38**

- display the surface for residues 5-38 in model 1

Categories can be defined manually with [surfcats](#) when the automatic categories do not give the desired results. By default, protein and nucleic acid chains are classified as **main**, and if multiple chains are in contact, they will be enclosed in a single surface. If separate surfaces for the separate chains are preferred, approaches include:

- using [split](#) to make each chain a separate model, then generating surfaces for the new models
- using [surfcats](#) to define custom surface categories, then **surface** to generate surfaces for these categories

**Usage:**

( **surfcOLOR** | **surfcolour** ) ( **byatom** | **bymodel** ) *model\_number(s)*

The command **surfcOLOR** does not color surfaces. Commands that color surfaces include [color](#), [sCOLOR](#), [rainbow](#), [rangecolor](#), and [coulombic](#).

Instead, **surfcOLOR** sets the color source (effectively the level at which coloring is defined) of a [molecular surface](#) to **byatom** or **bymodel**. The color of the surface will not necessarily match the color of the items at the same level, since at each level the “self” and surface color assignments may differ (see [coloring hierarchy](#)). If no *model\_number* is supplied, all open models will be affected. The *model\_number(s)* should be preceded by #, as in a normal [atom specification](#).

The color source is normally **byatom** unless changed explicitly or as a side effect of custom coloring (for example, with [sCOLOR](#)). Surface color source can also be set in the [molecular surface attributes panel](#).

See also: [modelcolor](#), [surface](#), [setattr](#), [coloring](#)

**Usage:**

**surftransparency** *percent* [ [atom-spec](#) [ *frames* ] ]

**Usage:**

**~surftransparency** [ [atom-spec](#) [ *frames* ] ]

The command **surftransparency** changes the transparency of a surface to the specified *percent*, where 0% is completely opaque and 100% is completely transparent. It is equivalent to [transparency](#) with the *,s* specifier. For [molecular surfaces](#), this overrides any transparencies included in the definitions of the surface colors, but they are remembered and can be restored with **~surftransparency**. For other [surface models](#), **~surftransparency** sets the transparency to 0%. A change in surface transparency can be performed abruptly, or over a specified number of *frames*. See the [video mini-example](#). See also: [surface](#), [surfrepr](#), [colordef](#), the [Actions menu](#), [movie-related commands](#)

The transparencies of per-atom patches of [molecular surface](#) can be adjusted independently by specifying the corresponding atoms. Other [surface models](#) can only be specified as a whole.

Surfaces colored with [tcolor](#) will not be affected by **surftransparency**.

## Usage:

`swapaa type atom-spec options`

**Swapaa** replaces amino acid sidechains using information from a [rotamer library](#); it is the command-line implementation of [Rotamers](#). A residue can be changed to a different sidechain conformation (rotamer) of the same type of amino acid or mutated into a different type. Rotamers can be [chosen](#) automatically based on lowest clash score, most H-bonds, best fit to an electron density map, and/or highest probability according to the library, or interactively from a [rotamer list](#). See also: [addaa](#), [ramachandran](#), [rotation](#), [swapna](#), [Build Structure](#), [Dock Prep](#)

Although sidechains at multiple positions can be replaced simultaneously, **swapaa** is not recommended for predicting the conformations of multiple sidechains in an interacting cluster. Programs such as [SCWRL](#) are more appropriate for that purpose.

The sidechain of each residue with at least one atom in [atom-spec](#) is replaced with a rotamer of *type*. The *type* can be specified with a three-letter code for one of the 20 standard amino acids (case is unimportant) or the word **same**, which allows substitutions at multiple residues of different types to be performed simultaneously.

In the **Dynameomics** [rotamer library](#) only, there are multiple choices of *type* for cysteine and histidine depending on the oxidation or protonation state of the sidechain:

- CYH - cysteine reduced free sulfhydryl
- CYS - cysteine oxidized disulfide-bonded (half-cystine)
- HID - histidine neutral  $\delta$ -protonated
- HIE - histidine neutral  $\epsilon$ -protonated
- HIS - histidine neutral (HID and HIE combined)
- HIP - histidine positive protonated on both sidechain nitrogens

These refer to the species for which conformational data were collected, but the rotamers do not include hydrogens and will be given standard residue names (CYS or HIS) if incorporated into a structure.

For nonstandard amino acids, see the [SwissSidechain Chimera plugin](#).

Bond lengths and angles are taken from the [Amber](#) parameter files **all\*94.lib**, and hydrogens are not included.

Only the sidechain atoms of a rotamer are evaluated. For [clash](#) and [H-bond](#) detection, interactions with other rotamers in the same set and the current residue at that position are disregarded, but all other atoms in the vicinity will be included unless [ignoreOtherModels](#) is used. In addition, atoms in the same model that are unwanted for such calculations (for example, [solvent](#)) should be deleted beforehand.

Examples:

**swapaa leu #0:248.a**

- replace amino acid residue 248 in chain A of model 0 with leucine

**swapaa his #0:248.a lib dnameomics**

- replace amino acid residue 248 in chain A of model 0 with a rotamer of neutral histidine from the **Dnameomics** [rotamer library](#)

**swapaa same sel**

- replace the sidechains of all selected residues with the best rotamers without mutating their types

**swapaa tyr :trp.b retain true**

- incorporate tyrosines wherever there are tryptophan residues in chain B, but keep the existing tryptophan sidechains too

**Options**

Option keywords for **swapaa** can be truncated to unique strings and their case does not matter. A vertical bar "|" designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

**lib rotamer-library**

What rotamer library to use; the source of rotamer torsion angles and probabilities. Possible values of *rotamer-library* (capitalization optional):

- **Dunbrack** (default) - [Dunbrack backbone-dependent rotamer library](#) (for chain-terminal residues, the Dunbrack backbone-independent version is used instead):

[Rotamer libraries in the 21st century](#). Dunbrack RL Jr. *Curr Opin Struct Biol*. 2002 Aug;12(4):431-40.

- **Dnameomics** - [Dnameomics rotamer library](#):

[The Dnameomics rotamer library: amino acid side chain conformations and dynamics from comprehensive molecular dynamics simulations in water](#). Scouras AD, Daggett V. *Protein Sci*. 2011 Feb;20(2):341-52.

The Dnameomics library includes multiple choices of residue type for cysteine and histidine, depending on the oxidation or protonation state; see [above](#).

- **Richardson.common** - common-atom values (author-recommended) from the [Richardson backbone-independent rotamer library](#):

[The penultimate rotamer library](#). Lovell SC, Word JM, Richardson JS, Richardson DC. *Proteins*. 2000 Aug 15;40(3):389-408.

- **Richardson.mode** - mode values from the [Richardson backbone-independent rotamer library](#)

**criteria** *method* | manual

How to choose the rotamer. If **manual**, all rotamers will be displayed in the graphics window and listed in a [dialog](#) so that the user can choose interactively; otherwise (default), a single rotamer will be chosen automatically according to the *method*. The [preserve](#) option can be used to filter the set of rotamers by chi angle similarity to the current sidechain before the *method* is applied. The *method* can be any combination, without spaces, of one or more of the following letters (default **dchp**):

- **d** - by best fit into [density](#)
- **c** - by lowest [clash score](#)
- **h** - by highest number of [H-bonds](#)
- **p** - by highest probability according to the [rotamer library](#) (probabilities are simply taken from the library and are not affected by the structural environment, except by phi and psi angles when the Dunbrack library is used)

Each successive method is only used when the previous method(s) have produced a tie. For example, with the default criteria (**dchp**), if no density map is [specified](#), clashes will be evaluated; if the [clash scoring method](#) is **num** and more than one rotamer ties for the lowest number of clashes, H-bonds will be evaluated to break the tie; if the lowest-clashing rotamers also have equal numbers of H-bonds, the one with the highest probability will be used. Alternatively, an integer argument *N* can be used instead of letters to indicate the rotamer with the *N*th highest probability, or **0** (zero) to indicate the rotamer with the lowest probability, regardless of any local interactions or density data.

**preserve** true | false

Whether to discard rotamers (regardless of the [criteria](#), except ignored if **manual**) with any chi angle > 40 ° different from that in the current sidechain. If the current sidechain has symmetrical branching (as in Asp, Glu, Phe, Tyr), the chi angle for comparison is calculated in both possible ways.

**retain** true | false | sel

What to do with the pre-existing sidechain(s): retain, replace (default), or retain only those with any atom [selected](#) (keyword **sel**). Regardless of this setting, sidechains will always be replaced where the incoming residue type is glycine or alanine. When there will be multiple sidechains at a given residue position, the new sidechain(s) will be assigned different alternative location identifiers.

**log** true | false

Whether to report torsion angles in the [Reply Log](#). Values are reported for the backbone (phi, psi, and whether the peptide bond is cis or trans) and the chosen sidechain rotamer (chi angles) for each swapped residue. Pre-swap chi angles are also reported when the

[preserve](#) option is used.

**ignoreOtherModels** true | false

In [clash](#) and [H-bond](#) detection, whether to include only atoms in the same model as the residue being swapped; useful for preventing superimposed related proteins or additional copies of the starting structure from affecting the results.

Density parameters:

**densitySpec** *mapmodel*

The *mapmodel* is the model number preceded by # of the [density map](#) open in Chimera.

Clash parameters:

**overlapCutoff** *cutoff*

The *cutoff* is how much VDW overlap should count as a clash (default 0.6 Å). A larger positive *cutoff* restricts the results to more severe clashes ([details](#)).

**hbondAllowance** *allowance*

When VDW overlap is calculated, an *allowance* (default 0.4 Å) is subtracted for atom pairs comprised of a possible hydrogen bond donor (or its hydrogen) and a possible acceptor ([details](#)).

**scoreMethod** sum | num

How to calculate the clash score: as a simple count of the number of clashes (**num**) or a sum of all overlaps  $\geq$  [cutoff](#) (**sum**).

H-bond parameters:

**relax** true | false

Whether to relax the [precise criteria](#) for hydrogen bonding.

**distSlop** *tolerance*

The *tolerance* is how much to relax the distance criteria if [relax](#) is true (default 0.4 Å).

**angleSlop** *tolerance*

The *tolerance* is how much to relax the angle criteria if [relax](#) is true (default 20.0 degrees).

**Usage:**

**swapna** *new\_type* [atom-spec](#) [ **preserve true** | **false** ]

**Swapna** "mutates" nucleic acid residues by replacing their base moieties. Every residue with at least one atom in [atom-spec](#) is changed to a *new\_type* residue. *New\_type* is a one-letter code representing one of the standard nucleic acid bases: A, T, G, C, or U (case is unimportant). The geometries of the new residues may not be optimal.

The **preserve** option (default **true**) retains the existing torsion angle around the base-sugar (glycosidic) bond and the position of the base nitrogen involved in that bond.

The temperature factor for the new residue is set to the highest currently found in the model.

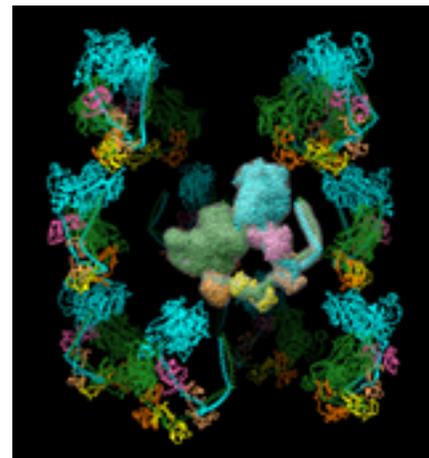
See also: [swapaa](#), [Adjust Torsions](#), [Build Structure](#)

**Usage:****sym** [ *molmodel* ] [options](#)**Usage:****~sym** [ *molmodel* ]

The **sym** command generates symmetry-related copies of a molecule model. The command **~sym** removes the copies. See also:

[crystalcontacts](#), [matrixcopy](#), [measure symmetry](#), [fitmap](#), [Unit Cell](#), [Multiscale Models](#), [Cage Builder](#), the [biological unit](#) function in the [Model Panel](#), [fetching](#) PQS files

The molecule model to copy (*molmodel*) is specified by model number, optionally preceded by #. This specification can be omitted from the **sym** command when there is only one molecule model. If **sym**-created copies of *molmodel* already exist, they will be replaced. The **~sym** command without arguments closes all **sym**-created copies.

(see [Quicktime movie](#))

One application of **sym** is to facilitate symmetrical placement of copies of a structure (molecule model) within related [volume data](#), usually a density map. In that case, the volume model should be specified as the [reference coordinate system](#). After the structure has been placed approximately as desired in the density, **sym** can be used with [update true](#) to create copies that update automatically when the original model is moved. For example, see a [Quicktime movie](#) of myosin copies moving symmetrically.

For memory and display efficiency, the copies can be generated as [low-resolution surfaces](#) (from [Multiscale Models](#)) instead of atomic coordinates by specifying [surfaces true](#). This may be helpful for large multimers. See also: [play](#), [msc](#)

**Options**

Option keywords (but not their arguments) can be truncated to unique strings, and their case does not matter. Synonyms for true: True, 1. Synonyms for false: False, 0. A vertical bar “|” designates mutually exclusive options, and default settings are indicated with **bold**.

**group** *symmetry*

Specifications of *symmetry* are case-independent, and most types have additional sub-options or parameters:

- **biomt** (default) - use [biological unit information](#), if any, in the *molmodel* input file
- symmetry of model #*N* - use [biomt information](#) from another molecule model or the [symmetry assignment](#) of a volume model
  - Example: **#4**
- [cage model](#) polygon symmetry #*N*,*pM* or #*N*,*pnM* - place copies at equivalent

positions relative to each M-sided polygon in the [cage model](#) with ID number **N**. The **pM** form places one copy per M-sided polygon, whereas **pnM** places M copies per M-sided polygon using *CM* symmetry about the center of the M-sided polygon nearest the original copy.

- Examples: **#2,p6** or **#2,pn5** ([more...](#))
- cyclic symmetry **C<sub>n</sub>** around [axis](#) and [center](#)
  - Example: **C3**
- dihedral symmetry **D<sub>n</sub>** around [axis](#) and [center](#)
  - Example: **d7**
- tetrahedral symmetry **T[,orientation]** around [center](#)
  - Example: **t,z3**

where *orientation* can be:

- **222** (default) - with two-fold symmetry axes along the X, Y, and Z axes, a three-fold along axis (1,1,1)
- **z3** - a three-fold symmetry axis along Z, another three-fold axis in the YZ plane such that rotation about the X axis by ~110° is a symmetry operation (EMAN convention)
- octahedral symmetry **O** around [center](#)
- icosahedral symmetry **I[,orientation]** around [center](#)
  - Example: **i,n25**

where *orientation* can be:

- **222** (default) - with two-fold symmetry axes along the X, Y, and Z axes
- **2n5** - with two-fold symmetry along X and 5-fold along Z
- **n25** - with two-fold symmetry along Y and 5-fold along Z
- **2n3** - with two-fold symmetry along X and 3-fold along Z
- **222r** - same as 222 except rotated 90° about Z
- **2n5r** - same as 2n5 except rotated 180° about Y
- **n25r** - same as n25 except rotated 180° about X
- **2n3r** - same as 2n3 except rotated 180° about Y
- helical symmetry **H,rise,angle,n[,offset]** around [axis](#) and [center](#)
  - Example: **h,43.5,21,6,-2**

where *rise* is the translation along the [axis](#) per subunit, *angle* is the rotation in degrees per subunit, and *n* is how many copies total (including the original) the resulting segment of infinite helix should contain prior to any filtering by [contact](#) or [range](#). The integer *offset* (default **0**) allows extending the helix in both directions.

The example above would give *n* = 6 copies total, with two copies in the negative axis direction, one at the identity position, and three in the positive axis direction.

- translational symmetry **shift,n,distance** along [axis](#) - or - **shift,n,x,y,z**
  - Example: **shift,3,26.7**

where *n* is how many copies total (including the original) the result should contain prior to any filtering by [contact](#) or [range](#). The translation can be expressed as a *distance* along the [axis](#) or as a vector *x,y,z* in the [reference coordinate system](#).

- the product of symmetry groups, each specified as described above and separated by \* to indicate multiplying each symmetry matrix of one group with each symmetry matrix of another; can be generalized to multiple symmetry groups (not just two)
  - Example: **c2\*h,42,21,9,-4**

### axis axis

Specify axis of symmetry (default **z**), where *axis* can be:

- **x** - X-axis
- **y** - Y-axis
- **z** - Z-axis
- *x,y,z* (three values separated by commas only) - an arbitrary vector in the [reference coordinate system](#)
- an [atom-spec](#) of exactly two atoms (not necessarily bonded or in the same model) or one bond. A bond can only be specified by [selecting](#) it and using the word **selected**, **sel**, or **picked**; any atoms also selected at the time will be ignored.

#### **center** *center*

Specify center of symmetry (default **0,0,0**), where *center* can be:

- *x,y,z* (three values separated by commas only) - an arbitrary point in the [reference coordinate system](#)
- an [atom-spec](#) of any combination of atoms and [surface pieces](#). The center of the bounding sphere of the specified items will be used.

#### **coordinateSystem** *N*

Specify a reference model (default is the original molecule model, *molmodel*) by model number *N* preceded by #. The reference coordinate system is used for [dynamic updating](#) and for interpreting coordinate specifications such as of [axis](#) and [center](#) of symmetry.

#### **biomtSet** **true** | **false**

Whether to generate [BIOMT matrices](#) for *molmodel* according to the specified [symmetry](#), replacing any pre-existing BIOMT information. The BIOMT matrices will be included if *molmodel* is subsequently [saved as PDB](#).

The [accelerator](#) **mB** (command **ac mB**) sets [BIOMT matrices](#) for molecule models associated with the currently selected [multiscale](#) chains to only the symmetries represented by those chains. This allows writing only a subset of the original BIOMT matrices (or symmetries assigned with [sym](#)).

#### **update** **true** | **false**

Whether to dynamically update the positions of the copies to preserve symmetry when the original model is moved relative to the [reference coordinate system](#). If the reference model is closed, the copies will cease to update.

#### **modelId** *N*

Open the copy or copies as model number *N* (an integer, optionally preceded by #). The default is the lowest unused number. When multiple copies are loaded, they will be opened as submodels of the specified model. When only a single copy is loaded, it will not receive a submodel number unless a specification of the form #N.N (# required) is given instead of *N*.

#### **contact** *contact-dist*

Only generate copies with any atom within *contact-dist* of the original molecule model.

#### **range** *range-dist*

Only generate copies with centers within *range-dist* of the center of the original molecule model. A model's center is defined as the center of its bounding box.

### occupancy *f*

Only generate copies at a fraction of the positions that would otherwise be filled. Random number generation is used to fill each position at a probability *f* (a value ranging from 0 to 1). This approach may yield different overall fractions filled for multiple uses of the same command, but distributed about *f*.

### surfaces all | true | false [ resolution *r* ]

Whether to generate the additional copies as [low-resolution surfaces](#) (from [Multiscale Models](#)) instead of atomic coordinates. The surfaces may be preferred for large multimers because they require much less memory and are more efficient to display than atomic coordinates. The keyword **all** indicates making a surface for the original structure as well as for the additional copies. A low-resolution surface is created by counting the atoms in each cell of a 3D grid and then making an isosurface of this occupancy map. The **resolution** *r* is the grid spacing (default 8 Å). The isosurface is smoothed to reduce artifacts associated with using an arbitrarily aligned grid.

## BIOMT matrices (biological unit information)

BIOMT matrices are included in some [Protein Data Bank](#) (PDB) entries, for example, [1fav](#). (The corresponding information in mmCIF format can also be used.) BIOMT matrices can be added to PDB files using a text editor or the command [sym](#) with [biomtSet true](#).

The [image](#) shows twelve copies of myosin arranged helically, as specified by the following twelve matrices added to PDB entry [1i84](#) (the first is simply an identity matrix that does not specify an additional copy):

```
REMARK 350 BIOMOLECULE: 1
REMARK 350 APPLY THE FOLLOWING TO CHAINS: S, T, U, V, W, Z
REMARK 350 BIOMT1 1 1 0 0 0
REMARK 350 BIOMT2 1 0 1 0 0
REMARK 350 BIOMT3 1 0 0 1 0
REMARK 350 BIOMT1 2 0 -1 0 0
REMARK 350 BIOMT2 2 1 0 0 0
REMARK 350 BIOMT3 2 0 0 1 0
REMARK 350 BIOMT1 3 -1 0 0 0
REMARK 350 BIOMT2 3 0 -1 0 0
REMARK 350 BIOMT3 3 0 0 1 0
REMARK 350 BIOMT1 4 0 1 0 0
REMARK 350 BIOMT2 4 -1 0 0 0
REMARK 350 BIOMT3 4 0 0 1 0
REMARK 350 BIOMT1 5 0.866025 -0.5 0 0
REMARK 350 BIOMT2 5 0.5 0.866025 0 0
REMARK 350 BIOMT3 5 0 0 1 145
REMARK 350 BIOMT1 6 -0.5 -0.866025 0 0
REMARK 350 BIOMT2 6 0.866025 -0.5 0 0
REMARK 350 BIOMT3 6 0 0 1 145
REMARK 350 BIOMT1 7 -0.866025 0.5 0 0
REMARK 350 BIOMT2 7 -0.5 -0.866025 0 0
REMARK 350 BIOMT3 7 0 0 1 145
```

sym

```
REMARK 350  BIOMT1  8  0.5 0.866025 0 0
REMARK 350  BIOMT2  8 -0.866025 0.5 0 0
REMARK 350  BIOMT3  8  0 0 1 145
REMARK 350  BIOMT1  9  0.866025 0.5 0 0
REMARK 350  BIOMT2  9 -0.5 0.866025 0 0
REMARK 350  BIOMT3  9  0 0 1 -145
REMARK 350  BIOMT1 10 -0.5 0.866025 0 0
REMARK 350  BIOMT2 10 -0.866025 -0.5 0 0
REMARK 350  BIOMT3 10  0 0 1 -145
REMARK 350  BIOMT1 11 -0.866025 -0.5 0 0
REMARK 350  BIOMT2 11  0.5 -0.866025 0 0
REMARK 350  BIOMT3 11  0 0 1 -145
REMARK 350  BIOMT1 12  0.5 -0.866025 0 0
REMARK 350  BIOMT2 12  0.866025 0.5 0 0
REMARK 350  BIOMT3 12  0 0 1 -145
```

system

[Chimera Commands Index](#)

**Usage:**

**system** *command*

**System** executes *command* under the system shell. *Command* cannot be an interactive program. If Chimera was [started](#) from a shell window, the output will appear there.

See also: [pdbrun](#)

**Usage:**

( **tcolor** | **tcolour** ) *color\_name*[,b][,l][,s][,v] [atom-spec](#)

The syntax of the **tcolor** command is the same as that of the [color](#) command. The [atom-spec](#) indicates which atoms (and/or associated labels or surfaces) are to be colored with a texture color rather than a conventional color. The default 2D texture can contain five colors. New textures can be created with the [texture new](#) command. Colors can be assigned to textures with the [texture map](#) or [texture color](#) commands. The "current" texture affected by [texture](#) commands and used by **tcolor** can be changed with [texture use](#).

Example:

```
texture map 1=red 2=blue 3=yellow 4=gray
tcolor gray
tcolor red @o=
tcolor blue @n=
tcolor yellow @s=
```

See also: [color](#), [texture](#)

**Usage:**

```
texture new texture_name [num_colors]
```

**Usage:**

```
texture map index=name ...
```

**Usage:**

```
texture color index color
```

**Usage:**

```
texture use texture_name
```

**Texture new** creates a new texture with the given name. If *num\_colors* is not given, the texture will be able to contain five distinct colors. If given, *num\_colors* must be 4 or 5. The principal advantage of using a four-color texture is that color transitions occur midway between vertices in a four-color texture, but not in a five-color texture.

**Texture map** assigns names to the texture colors. This command can be used in two ways. It can be used to assign mnemonics to the texture colors (e.g., **texture map 1=hydrophilic 2=hydrophobic ...**), in which case the **texture color** command would be used to assign actual color values to the texture colors, or it can be used to directly assign color values to the texture colors (e.g., **texture map 1=red 2=green 3=blue ...**). At this time, redefining a color with the [colordef](#) command *after* that color has been used in **texture map** will not cause a change in the texture color.

**Texture color** assigns *color* to the texture color corresponding to *index*.

**Texture use** makes the texture named *texture\_name* the current texture for purposes of further **texture** and [tcolor](#) commands. The initial default texture is named **default**.

Example:

```
texture map 1=red 2=blue 3=yellow 4=gray
tcolor gray
tcolor red @o=
tcolor blue @n=
tcolor yellow @s=
```

See also: [tcolor](#)

**Usage:****thickness** *increment* [ *frames* ]

**Thickness** changes the separation between the [global clipping planes](#) by the specified *increment*. The *increment* is a value in the current display units, usually Å; a positive number increases the distance between the clipping planes, whereas a negative number decreases it.

**Thickness** moves the clipping planes in the specified manner for the specified number of image update *frames* (default **1**). Commands continue to be processed while the requested motion is in progress. To pause command processing until the motion is finished, use the [wait](#) command. To halt an ongoing **thickness** motion, use [freeze](#).

See also: [clip](#), [section](#), the [Side View](#), [clipping](#), [movie-related commands](#)

**Usage:**

**tile** *models* [ **columns** *N* ] [ **spacingFactor** *f* ] [ **independentRotation** true|false ]  
[ **viewAll** true|false ]

**Usage:**

**~tile** *models* [ **viewAll** true|false ]

The command **tile** spreads models out into a plane; it is the command-line implementation of [Tile Structures](#). The *models* argument should be a comma-separated list of model numbers or ranges preceded by #, or simply # or blank (nothing) to indicate all models. A blank specification can only be used when no keywords are given.

The **columns** keyword specifies arranging the models into *N* columns in the viewing plane. If not specified, the numbers of rows and columns that best fit the window dimensions will be used.

The **spacingFactor** *f* indicates how far apart the structures should be spaced, with larger values giving greater separations. The default of **1.0** spaces the models so that their bounding spheres abut, while **0.0** superimposes the bounding sphere centers.

The **independentRotation** option (default **true**) makes models rotate about individual centers rather than a collective center; it sets the [center of rotation method](#) to **independent**. Specifying this option as false will not turn off independent rotation, however. Instead, rotation about a collective center can be restored with the command **~tile** or [~set independent](#).

The **viewAll** option (default **true**) adjusts the view to enclose all displayed models regardless of whether they were specified, equivalent to clicking **View All** in the [Side View](#).

The command **~tile** sets the transform of all of the specified models to be the same as the first model. Any prior transformation of one model relative to another such as [superposition](#) or docking in Chimera will be lost. If such manipulations have been performed, using [savepos](#) to save the position before tiling is recommended. If the [center of rotation method](#) is **independent**, the command **~tile** changes it to **front center**.

Default or previously saved positions can be restored with the command [reset](#), while previous model transformations (but not scale or camera center) can be restored with [Undo Move](#).

See also: [align](#), [center](#), [matrixcopy](#), [window](#), [vop tile](#), [superimposing structures](#)

## Usage:

### **topography** *model-spec* [options](#)

The **topography** command creates a surface in which height represents the values in a single plane of [volume data](#). The volume data should first be [opened](#) and only a [single plane](#) displayed. *Model-spec* is the model number of the data, preceded by #.

The topographic surface is opened as a new [surface model](#) named after the volume data but with the word **height** appended. Creating or rotating the surface can be slow. Default options for a 512 x 512 data plane create a 1 million-triangle surface. Such a surface took 5 seconds to create and rendered at 2 frames per second on a 2005 Mac G5 PPC. Surface size can be reduced by adjusting the volume plane display beforehand to a larger [step size](#) and/or smaller [subregion](#).

## Options

Option keywords for **topography** can be truncated to unique strings and their case does not matter. Synonyms for true: True, 1. Synonyms for false: False, 0. A vertical bar "|" designates mutually exclusive options, and default settings are indicated with **bold**.

### **height** *h*

Values will be mapped linearly to surface heights of up to *h* (in the physical units of the volume data) from the volume data plane. The surface minimum will coincide with the volume data plane. The default *h* is 10% of the data size along the X axis.

### **interpolation** *cubic|none*

Cubic interpolation generates additional data points in the plane to make a surface with four times as many triangles. The finer triangulation improves surface appearance but slows rendering.

### **meshType** *isotropic|slash|backslash*

Whether to represent each data grid cell with four triangles and an added center point (**isotropic**, default) or with two triangles defined with a diagonal line from bottom left to top right (slash) or from top left to bottom right (backslash) when the axis normal to the data plane is positive towards the user. Using a two-triangle option speeds rendering but can yield anisotropic artifacts.

### **colorMap** *rainbow|none*

Whether to color the surface by height (default is **rainbow** coloring from red for the lowest values to blue for the highest) or a uniform gray (none). The surface can be recolored with [color](#).

### **smoothingIterations** *i*

How many cycles of surface smoothing by moving surface vertices toward the average position of neighboring vertices should be performed (default **0**, or no smoothing). Smoothing will improve surface appearance but alter its shape, degrading the mapping of values to heights.

**smoothingFactor** *f*

What fraction of the distance to the average position of neighboring vertices to move each vertex in a single cycle of smoothing (default **0.3**).

**replace** **true** | **false**

Whether to replace any existing **topography** surfaces with the new one rather than opening an additional model.

See also: [volume](#), [scolor](#), [vop tile](#), [Volume Viewer](#)

**Usage:**

**transparency** *percent*[,a][,f][,r][,s][,v][,l][,la][,lr][,lb][,b] [atom-spec](#) [ frames *N* ]

**Usage:**

**~transparency**[,a][,f][,r][,s][,v][,l][,la][,lr][,lb][,b] [atom-spec](#) [ frames *N* ]

The **transparency** command sets transparency to the specified *percent*, where 0% is completely opaque and 100% is completely transparent. It can affect atoms, bonds, [ring fill](#), [ribbons](#), labels (3D), and various [surfaces](#). The **frames** option indicates making the change gradually over *N* frames. The **~transparency** command sets transparency to 0%. See the [video mini-example](#). See also: [colordef](#), [color](#), [modelcolor](#), [scolor](#), [rainbow](#), [rangecolor](#), [background](#), [set](#), [transparency-related settings](#), [movie-related commands](#)

The assignment can be restricted with one or more specifiers:

- **a** - atoms (and [normally](#), bond halves match the flanking atoms; see **b** for bonds-only control)
- **f** - [ring fill](#)
- **r** - [ribbons](#)
- **s** - [molecular surfaces](#) and nonmolecular [surface models](#)
- **v** - [VDW surfaces](#)
- **l** - atom and residue labels
- **la** (or **al**) - atom labels
- **lr** (or **rl**) - residue labels
- **lb** (or **bl**) - bond labels
- **b** - bonds only (not the flanking atoms); however, bond-only transparency will not be visible unless [halfbond mode](#) has been turned off, for example, by using [color](#) with the **,b** specifier

In the absence of a specifier, all of the above are affected except bonds-only. However, VRML models, [2D Labels](#), and [pseudobonds](#) and their labels are not affected. A pseudobond or its label can be made transparent by [selecting](#) the pseudobond and using the [Selection Inspector](#) to assign a transparent color.

The [atom-spec](#) indicates which atoms (and/or associated bonds, ribbon segments, surface patches, *etc.*) should be affected, and a [similar specification](#) can be used for nonmolecular [surface models](#). If no specification is given, all applicable items are affected.

Except for [molecular surfaces](#), subsequent color changes will reset transparency to zero.

Examples:

**transparency 80,a,r ligand zr>5**

- make the atoms/bonds and ribbon segments of residues that are more than 5 Å from [ligand](#) 80% transparent

**transp 75,s #3:60-100 frames 50**

- make the molecular surface patches of residues 60-100 in model 3 75% transparent over 50 frames

**Usage:**

**turn** [ *axis* [ *angle* [ *frames* ]]] [ **models** *model-spec* ] [ **coordinateSystem** *N* ]  
 [ **center** *center* ] [ **precessionTilt** *tilt* ]

**Turn** rotates models by *angle* degrees (default 1.5) around a specified *axis* (default **y**) for a specified number of *frames* (default 1). **Turn** is the same as [roll](#) except for the default number of frames. See the [video mini-example](#). See also: [rock](#), [align](#), [select](#), [cofr](#), [set independent](#), [movie-related commands](#)

The *angle* can be positive or negative. The **models** to rotate can be specified by model number(s) or ranges separated by commas and preceded by #. If no models are specified, all [active](#) models will be rotated.

A **coordinateSystem** can be specified by reference model number *N*, optionally preceded by #. Otherwise, the laboratory frame of reference will be used. Any *axis* or *center* specification of the form *x,y,z* will be interpreted in the reference coordinate system, and when further motions are applied to an ongoing rotation, the center and axis will remain pinned relative to the reference model.

The *axis* can be:

- **x** - X-axis (if laboratory coordinate system, horizontal in the plane of the screen)
- **y** - Y-axis (if laboratory coordinate system, vertical in the plane of the screen)
- **z** - Z-axis (if laboratory coordinate system, perpendicular to the screen)
- *x,y,z* (three values separated by commas only) - an arbitrary vector
- an [atom-spec](#) of exactly two atoms (not necessarily bonded or in the same model) or one bond. If two atoms, the order of specification defines a handedness, and right-handed rotations are positive. If a bond, the handedness is not under user control. A bond can only be specified by [selecting](#) it and using the word **selected**, **sel**, or **picked**; any atoms also selected at the time will be ignored. If a center is not specified separately, it will lie on the atom-atom or bond axis. If a separate center is supplied, only the direction of the axis will be used. The first model in the axis [atom-spec](#) will be used for reference frame pinning unless a **center** [atom-spec](#) or a **coordinateSystem** is also given.

The *center* can be:

- *x,y,z* (three values separated by commas only) - an arbitrary point
- an [atom-spec](#) of any combination of atoms and [surface pieces](#). The center of the bounding sphere of the specified items will be used. The first model in [atom-spec](#) will be used for reference frame pinning unless a **coordinateSystem** is also given.
- any of the following keywords, to specify a [center of rotation method](#) for computing the center:
  - **view** (center of view method)
  - **models** (center of models method)
  - **front** (front center method)

This does not change the current [center of rotation method](#), but allows using a different one for the purposes of executing the command.

turn

If a center of rotation is not specified directly or by using an [atom-spec](#) to define the axis, the current Chimera [center\(s\) of rotation](#) will be used.

The **precessionTilt** option specifies an additional rotation about a moving axis that is carried along by the main rotation. The moving axis is offset from the main *axis* by *tilt* degrees (toward the vertical if *axis* is along the line of sight). The two rotations cycle at the same rate but in opposite directions around their respective axes. See the [Wobble motion](#) movie in the Chimera Animation Gallery.

Keyword options can be used in any order and the keywords can be truncated. Example:

```
turn 1,1,0 180 center 0,2.5,2.5 coord 2 models #1
```

Commands continue to be processed while the requested motion is in progress. To pause command processing until the motion is finished, use the [wait](#) command. To halt an ongoing **turn**, use [freeze](#).

**Usage:****vdw** [atom-spec](#)**Usage:****~vdw** [atom-spec](#)

**Vdw** displays dot van der Waals surfaces for the atoms in [atom-spec](#).

The [default radii](#) depend on whether explicit hydrogen atoms are present. When there are no explicit hydrogens, the radii of some atoms are enlarged to compensate. VDW radii can be altered with [vdwdefine](#) and the density of dots can be changed using [vdwdensity](#).

Details of the **vdw** algorithm can be found in:

[Van der Waals Surfaces in Molecular Modeling: Implementation with Real-Time Computer Graphics](#). Bash PA, Pattabiraman N, Huang C, Ferrin TE, Langridge R. Science. 1983 Dec 23;222(4630):1325-1327.

See also: [surface](#), [vdwdefine](#), [vdwdensity](#)

**Usage:**

**vdwdefine** [+|-]*value* [atom-spec](#)

**Usage:**

**~vdwdefine** [atom-spec](#)

**Vdwdefine** changes the [VDW radii](#) of the [specified](#) atoms. If a plus (+) or minus (-) sign precedes *value*, the radii are increased or decreased, respectively, by *value* Å; otherwise, the radii are set to *value* Å. There should not be any space between the plus or minus sign and *value*.

**~Vdwdefine** restores the VDW radii of the of the [specified](#) atoms to their [default values](#). The defaults depend on whether explicit hydrogen atoms are present; thus, the VDW radius of an atom may change when hydrogens are [added](#) or [deleted](#).

Examples:

```
vdwdefine 2.0 @c=
```

- assign a radius of 2.0 Å to all atoms whose names begin with C

```
vdwdefine -0.5 :gly@ca
```

- decrease the radii of glycine CA atoms by 0.5 Å

See also: [vdw](#), [vdwdensity](#), [changing VDW radii](#)

vdwdensity

[Chimera Commands Index](#)

**Usage:**

**vdwdensity** *value*

**Vdwdensity** controls the dot density of the displayed surface. *Value* is the number of dots per 0.2 Å<sup>2</sup>, and the default is 5 (thus 25 dots/Å<sup>2</sup>).

See also: [vdw](#), [vdwdefine](#), [setattrr](#), the [molecule model attributes panel](#)

version

[Chimera Commands Index](#)

## **Usage:**

### **version**

**Version** shows copyright information and what version of Chimera is being used. The Chimera version number should be included in bug reports.

**Usage:****viewdock** *dock-file* [*dock-type*]

The **viewdock** command starts [ViewDock](#) and loads the specified file of docking results (*dock-file*). Possible *dock-type* settings are:

- **Dock 4, 5 or 6** (default) - from recent versions of [DOCK](#)
- **Dock 3.5.x search** - search mode results from the variant developed in the [Shoichet laboratory](#)
- **Dock 3.5.x single** - single mode results from the variant developed in the [Shoichet laboratory](#)
- **Dock 3 or 3.5** - from older versions of [DOCK](#)
- **Mordor** - from [MORDOR](#), MOlecular Recognition with a Driven dynamics OptimizeR

Any argument containing a space, including *dock-type*, should be enclosed in quotation marks. Example:

```
viewdock ../docking/test_single.eel1 "Dock 3.5.x single"
```

See also: [cd](#), [ViewDock](#)

**Usage:**

**vina docking receptor** [recmodel](#) **ligand** [ligmodel](#) [options](#)

Like the Chimera [AutoDock Vina](#) tool, the **vina** command runs single-ligand docking calculations with [AutoDock Vina](#). The process can use either a web service provided by the [National Biomedical Computation Resource \(NBCR\)](#) or a locally installed copy of the program. Users should cite:

[AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading.](#) Trott O, Olson AJ. *J Comput Chem*. 2010 Jan 30;31(2):455-61.

The receptor and ligand structures should be opened as separate models in Chimera and specified with the **receptor** and **ligand** keywords, respectively. The entire models containing the specified atoms will be used.

If the receptor contains MSE (selenomethionine) residues, incomplete side chains, or atoms with alternate locations, running [Dock Prep](#) beforehand to correct those issues is recommended. The **vina** command runs [AutoDock accessory scripts](#) locally to (further) prepare the structures, such as to add hydrogens if they have not been added already with Chimera.

Docking results will be shown automatically in [ViewDock](#). Please see the [AutoDock Vina manual](#) for a description of the output values.

**Options**

Option keywords for **vina** can be truncated to unique strings and their case does not matter. A vertical bar “|” designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

**output** *name*

Pathname (location and filename prefix) of output files. If the run is successful, the following files will be generated:

- *name* - docking results in [PDBQT](#) format, automatically read into [ViewDock](#) when the calculation finishes
- *name.receptor.pdb* - receptor PDB file from Chimera, input to the [AutoDock receptor preparation script](#)
- *name.receptor.pdbqt* - processed receptor in [PDBQT](#) format, input to AutoDock Vina
- *name.ligand.pdb* - ligand PDB file from Chimera, input to the [AutoDock ligand preparation script](#)
- *name.ligand.pdbqt* - processed ligand in [PDBQT](#) format, input to AutoDock Vina
- *name.conf* - AutoDock Vina configuration file

If *name* is not specified, a temporary filename prefix and location will be used.

The default box in which to sample ligand positions will enclose the entire receptor, with extra space on all sides. If the region of interest is smaller, the desired box center and size should be specified to allow for a more efficient search:

**search\_center** *x,y,z*

In the receptor coordinate system, location of the center of the box in which to sample ligand positions.

**search\_size** *xlen,ylen,zlen*

In the receptor coordinate system, dimensions along X, Y, and Z of the box in which to sample ligand positions.

Settings for the [receptor preparation script](#):

**receptor\_addh** true | false

Whether to add hydrogens in Chimera (see [addh](#)) before calling the script. The receptor prep script will check for hydrogens and add them if they are missing. AutoDock Vina needs the polar (potentially H-bonding) hydrogens to identify atom types for scoring purposes.

**receptor\_nphs** true | false

Whether to merge charges and remove nonpolar hydrogens. Note AutoDock Vina does not use charges or nonpolar hydrogens, so this setting is not expected to affect results except for the presence or absence of nonpolar hydrogens in the processed receptor.

**receptor\_lps** true | false

Whether to merge charges and remove lone pairs. Note AutoDock Vina does not use charges or lone pairs, so this setting is not expected to affect results except for the presence or absence of lone pairs in the processed receptor (and there may not have been any lone pairs to start with).

**receptor\_waters** true | false

Whether to remove water residues.

**receptor\_nonstdres** true | false

Whether to remove chains composed entirely of residues other than the 20 standard amino acids.

**receptor\_nonstd** true | false

Whether to remove all residues other than the 20 standard amino acids.

Settings for the [ligand preparation script](#):

The ligand prep script will check for hydrogens and add them if they are missing. AutoDock Vina needs the polar (potentially H-bonding) hydrogens to identify atom types for scoring purposes.

**ligand\_nphs** true | false

Whether to merge charges and remove nonpolar hydrogens. Note AutoDock Vina does not use charges or nonpolar hydrogens, so this setting is not expected to affect results except for the presence or absence of nonpolar hydrogens in the ligand output files.

**ligand\_lps** *true* | *false*

Whether to merge charges and remove lone pairs. Note AutoDock Vina does not use charges or lone pairs, so this setting is not expected to affect results except for the presence or absence of lone pairs in the ligand output files (and there may not have been any lone pairs to start with).

Docking parameters:

**num\_modes** *N*

Maximum number of binding modes to generate (up to 20, default **9**).

**exhaustiveness** *M*

Thoroughness of search, roughly proportional to time (default **8**).

**energy\_range** *range*

Maximum difference in score (default **3.0** kcal/mol); binding modes with scores not within *range* of the best score will be discarded.

Executable location:

**backend** *opal* | *local*

Whether to use an [Opal](#) web service (default) or a locally installed executable.

**location** *opal-URL* | *local-path*

Depending on the [backend](#) setting, the URL of the web service (default is the URL for the service provided by the [NBCR](#)) or the pathname of the local executable.

**Usage:****volume** *model-spec* [options](#)

**Volume** is the command-line implementation of many features in [Volume Viewer](#), a tool for visualizing [volume data](#). See also: [open](#), [vop](#), [vseries](#), [mask](#), [molmap](#), [topography](#), [measure](#), [scolor](#), [sym](#), [fitmap](#), [meshmol](#), [Volume Viewer](#)

*Model-spec* can be a specific model number or range of model numbers (preceded by #), or simply # or the word **all** to indicate all volume models. Several of the [sampling and size options](#) apply to all volume models, regardless of which are specified.

Option keywords for **volume** can be truncated to unique strings, and their case does not matter. Specifications of true (synonyms t, yes, y, on, 1) and false (synonyms f, no, n, off, 0) are also case-independent. A vertical bar “|” designates mutually exclusive options, and factory default settings are indicated with **bold** (different defaults can be saved from [Volume Viewer](#), however).

Examples:

```

volume #0 style mesh level 0.8 color red level 1.2 color 0,.5,.8
volume #2 level 10,0 level 100,1 level 400,1 color hotpink style solid
vol all hide

```

There are many options, here grouped into categories:

[General Display Options](#)[Sampling and Size Options](#)[Dimension and Scale Options](#)[Planes Options](#)[File-Saving Options](#)[Surface and Mesh Display Options](#)[Solid Display Options](#)

← **General Display Options** (Usage: [volume](#) *model-spec options*)

- **show**

Display the volume model.

- **hide**

Undisplay the volume model.

- **toggle**

Of the specified volume models, show those that are hidden and hide those that are shown.

- **style** surface | mesh | solid

Designate the [style](#) of display: the **surface** and **mesh** modes depict isosurfaces (contour surfaces), while the **solid** mode shows data as a semitransparent solid.

Separate sets of [level](#), [color](#), [brightness](#), and [transparency](#) information are maintained for the surface/mesh and solid styles of a volume model; switching to solid from surface or mesh (or *vice versa*) restores any previous assignments for that style. See also: [surface and mesh display options](#), [solid display options](#)

- **level** *threshold-level*

Place a *threshold* for mapping data values to the display. See the [video mini-example](#).

- For **surface** and **mesh** displays, *threshold-level* is a single number, the contour level of the surface. This corresponds to horizontal placement on a histogram in the [graphical interface](#). To adjust the threshold level automatically so that the contour encloses a specified spatial volume, see the [encloseVolume](#) and [fastEncloseVolume](#) options.
- For **solid** displays, *threshold-level* consists of two numbers separated by a comma (no spaces). The first number indicates a data value and the second number indicates an intensity ranging from 0.0 to 1.0. These correspond to horizontal and vertical placement, respectively, on a histogram in the [graphical interface](#).

- **color** *threshold-color*

Assign [threshold](#) color. The *threshold-color* can be any [color name](#) that specifies a single color (with any spaces stripped). The default color is an opaque medium gray (0.7,0.7,0.7,1.0) for surface and mesh displays, **white** for solid. See also: [scolor](#)

Multiple [level](#) and [color](#) specifications can be included in a single command. If a color is specified but no levels, the color applies to all existing levels and becomes the default color for the volume model. If levels are given but no color, the model's current default color is used for the levels, and all old levels are removed. If one color and one or more levels are given, that color applies to all levels but does not become the default color. Otherwise, if multiple levels and colors are given, there must be an equal number of each. Levels and colors are paired in the order given, but they do not need to be interleaved; only the ordering of each type of specification (levels or colors) is significant.

- **encloseVolume** *volume*

Automatically set **surface** or **mesh** [threshold level](#) to enclose the specified *volume* in distance units cubed (*e.g.*, Å<sup>3</sup> if the grid spacing is expressed in Å). Multiple volume models can be specified in the same command to make their isosurfaces enclose the same spatial volume. The level is determined by an iterative procedure. In each iteration, the density value midway between the upper and lower bounds is tested. If the resulting enclosed volume is larger (smaller) than the target, that midpoint value becomes the new lower (upper) bound. In the first iteration, the upper and lower bounds are the maximum and minimum values in the map. Iteration stops when the actual enclosed volume differs from the target by less than 0.001 of the target or 30 iterations have been performed. The [fastEncloseVolume](#) option is similar but uses a noniterative approximation. See also: [measure volume](#), [vseries align](#)

- **fastEncloseVolume** *volume*

Automatically set **surface** or **mesh** [threshold level](#) to enclose the specified *volume* in distance units cubed (e.g., Å<sup>3</sup> if the grid spacing is expressed in Å). Multiple volume models can be specified in the same command to make their isosurfaces enclose the same spatial volume. The number of grid points to enclose is estimated by dividing the target volume by the volume of one grid cell, and then the level corresponding to that number of points is estimated by sorting the data values into 10,000 bins of equal width and identifying the value bin that attains (cumulatively) that number of points. The [encloseVolume](#) option is similar but uses an iterative procedure with increased accuracy at the cost of increased computation time. See also: [measure volume](#), [vseries align](#)

- **brightness** *value*

Brightness scales the intensity of the color of the display. Values can range from 0.01 to 10.0, where **1.0** (the default) produces no change relative to the [specified colors](#).

- **transparency** *value*

Transparency values range from 0.0 (fully opaque) to 1.0 (fully transparent).

- For **surface** and **mesh** displays, transparency is the fraction of light transmitted from behind the surface or a line of mesh (default **0.0**). By default, these objects are dimmed as they are made more transparent (see [dimTransparency](#).)
- For **solid** displays, this transparency setting further modulates initial transparencies obtained from the [transfer function](#) in a way that compensates for the thickness of the display ([details](#)). Values range from 0.0 (no modulation) to 1.0, default **0.5**. By default, more transparent voxels are made dimmer (see [dimTransparentVoxels](#)).

- **showOutlineBox** true | false

Outline the bounding box of the [current display region](#).

- **outlineBoxRgb** *outline-color*

Assign a color to the [outline box](#). The *outline-color* can be any [color name](#) that specifies a single color (with any spaces stripped). Any transparency in the color will be ignored. The default color is **white**.

## ← Sampling and Size Options (Usage: [volume](#) *model-spec options*)

- **step** *N* | *Nx,Ny,Nz*

Step values indicate sampling density; a step of 1 means all data points are used to generate the display, while 2 means every other data point is taken along each axis. Step sizes must be integers. If a single number is supplied, it is used in all three directions; if three numbers are supplied (separated by commas but not spaces), they are used in the X, Y, and Z directions, respectively. Changing a step value will change the data size limit for automatic step adjustment (see [voxelLimit](#)).

- **limitVoxelCount** true | false

Automatically adjust [step size](#) so that no more than the [specified voxel limit](#) is displayed.

- **voxelLimit** *limit*

Set the maximum number of Mvoxels to be displayed (default **1.0**) when [limitVoxelCount](#) is set to true.

The remaining options in this section apply to all volume models, regardless of which are [specified](#):

- **showOnOpen** true | false

Automatically display a data set when it is opened if it does not exceed a [specified size](#).

- **voxelLimitForOpen** *size*

Set the data size limit in Mvoxels below which data should be automatically displayed when opened (default **256.0**) when [showOnOpen](#) is set to true.

- **showPlane** true | false

Initially display just a single plane (normal to the Z axis) of a data set if it exceeds a [specified size](#).

- **voxelLimitForPlane** *size*

Set the data size limit in Mvoxels above which a single plane of the data should be initially displayed (default **256.0**) when [showPlane](#) is set to true.

- **dataCacheSize** *size*

Set how much memory in Mb should be dedicated to volume data (default **512**). A cache can improve performance, since accessing cached data is faster than reading it from disk. The least recently displayed data values are purged to maintain the specified *size*. The data cache only accounts for approximately 1/3 to 1/2 of the memory used in viewing volume data, as additional memory is occupied by surfaces and color arrays.

#### ← Dimension and Scale Options (Usage: [volume](#) *model-spec options*)

- **region** all | *name* | *i1,j1,k1,i2,j2,k2*

Show the full data set (specified with **all**), or the data region previously [assigned](#) *name*, or the data region with grid indices *i1-i2* along the X axis, *j1-j2* along the Y axis, and *k1-k2* along the Z axis. Grid indices must be integers separated by commas but not spaces.

- **nameRegion** *name*

Assign *name* to the currently displayed region.

- **origin** *x,y,z*

Place the grid origin at coordinates *x,y,z* (numbers separated by commas but not spaces).

- **originIndex** *i,j,k*

Place the coordinate origin (0,0,0) at grid indices *i,j,k* (numbers separated by commas but not spaces). Fractional and negative values are allowed, as the origin is not required to coincide with a grid point or even to fall within the grid.

- **voxelSize** *S* | *Sx,Sy,Sz*

Voxel size indicates the scale of the data set, the spacing of points in units of distance. If a single number is supplied, it is used in all three directions; if three numbers are supplied (separated by commas but not spaces), they are used in the X, Y, and Z directions, respectively. The grid is anchored at the coordinate origin ([originIndex](#) remains unchanged).

- **symmetry** *sym-type*

Assign the specified symmetry to the volume data set. This information is retained in files [saved](#) in [Chimera map](#) format and can be used by other commands such as [sym](#) and [fitmap](#). For automatic symmetry detection and assignment, see the command [measure symmetry](#). Specifications of *sym-type* are case-independent, and most types have additional sub-options or parameters:

- symmetry of model #*N* - use [biomt information](#) from a molecule model or the symmetry assignment of another volume model
  - Example: **#4**
- [cage model](#) polygon symmetry #*N*,*pM* or #*N*,*pnM* - place copies at equivalent positions relative to each M-sided polygon in the [cage model](#) with ID number **N**. The *pM* form places one copy per M-sided polygon, whereas *pnM* places M copies per M-sided polygon using *CM* symmetry about the center of the M-sided polygon nearest the original copy.
  - Examples: **#2,p6** or **#2,pn5**
- cyclic symmetry *C<sub>n</sub>* around [axis](#) and [center](#)
  - Example: **C3**
- dihedral symmetry *D<sub>n</sub>* around [axis](#) and [center](#)
  - Example: **d7**
- tetrahedral symmetry *T*[*orientation*] around [center](#)
  - Example: **t,z3**
 where *orientation* can be:
  - **222** (default) - with two-fold symmetry axes along the X, Y, and Z axes, a three-fold along axis (1,1,1)
  - **z3** - a three-fold symmetry axis along Z, another three-fold axis in the YZ plane such that rotation about the X axis by ~110° is a symmetry operation (EMAN convention)
- octahedral symmetry *O* around [center](#)
- icosahedral symmetry *I*[*orientation*] around [center](#)
  - Example: **i,n25**
 where *orientation* can be:
  - **222** (default) - with two-fold symmetry axes along the X, Y, and Z axes
  - **2n5** - with two-fold symmetry along X and 5-fold along Z
  - **n25** - with two-fold symmetry along Y and 5-fold along Z
  - **2n3** - with two-fold symmetry along X and 3-fold along Z
  - **222r** - same as 222 except rotated 90° about Z
  - **2n5r** - same as 2n5 except rotated 180° about Y
  - **n25r** - same as n25 except rotated 180° about X
  - **2n3r** - same as 2n3 except rotated 180° about Y
- helical symmetry *H*,*rise*,*angle*,*n*[*offset*] around [axis](#) and [center](#)
  - Example: **h,43.5,21,6,-2**
 where *rise* is the translation along the [axis](#) per subunit, *angle* is the rotation in

degrees per subunit, and  $n$  is how many copies total (including the original) the resulting segment of infinite helix should contain. The integer *offset* (default **0**) allows extending the helix in both directions. The example above would give  $n = 6$  copies total, with two copies in the negative axis direction, one at the identity position, and three in the positive axis direction.

- translational symmetry **shift**, $n$ ,*distance* along [axis](#) - or - **shift**, $n$ , $x$ , $y$ , $z$ 
    - Example: **shift,3,26.7**
- where  $n$  is how many copies total (including the original) the result should contain. The translation can be expressed as a *distance* along the [axis](#) or as a vector  $x,y,z$  in the [reference coordinate system](#).
- the product of symmetry groups, each specified as described above and separated by \* to indicate multiplying each symmetry matrix of one group with each symmetry matrix of another; can be generalized to multiple symmetry groups (not just two)
    - Example: **c2\*h,42,21,9,-4**

- **axis axis**

Specify axis of symmetry (default **z**), where *axis* can be:

- **x** - X-axis
- **y** - Y-axis
- **z** - Z-axis
- $x,y,z$  (three values separated by commas only) - an arbitrary vector in the [reference coordinate system](#)
- an [atom-spec](#) of exactly two atoms (not necessarily bonded or in the same model) or one bond. A bond can only be specified by [selecting](#) it and using the word **selected**, **sel**, or **picked**; any atoms also selected at the time will be ignored.

- **center center**

Specify center of symmetry in physical coordinates (default **0,0,0**), where *center* can be:

- $x,y,z$  (three values separated by commas only) - an arbitrary point in the [reference coordinate system](#)
- an [atom-spec](#) of any combination of atoms and [surface pieces](#). The center of the bounding sphere of the specified items will be used.

This option is overridden by [centerIndex](#).

- **centerIndex** [  $i$  |  $i,j,k$  ]

Specify center of symmetry in grid coordinates, given as a single value or three values separated by commas only. Fractional and negative values can be used. If a single value is given, it is used as the grid coordinate along all three axes. This option overrides [center](#).

- **coordinateSystem N**

Specify a reference model by model number  $N$  preceded by #. The reference coordinate system is used for interpreting specifications of [axis](#) and [center](#) of symmetry.

- **planes** *axis,start[,end[,increment[,depth]]]*

Sequentially display slabs *depth* planes thick along the specified data *axis* (x,y, or z) starting from index *start* and repositioning the slab by *increment* grid units per frame until the index at the next frame would exceed *end*. The indices refer to the first displayed plane along the axis. If no *end* is supplied, only a single slab will be shown. The default *increment* and *depth* are both 1. Only planes that are multiples of the [step size](#) will be shown, and the *depth* refers to the number of planes shown rather than a grid index range. Although an *increment* smaller than the step size can be specified, at each frame the grid index will be rounded down to a multiple of the step size, resulting in the same plane being displayed in more than one frame. To avoid this, change the step size to 1 beforehand or specify *start*, *end* and *increment* values consistent with the current step size. The *start*, *end*, *increment*, and *depth* parameters can be floating-point numbers. For example, an *increment* of 0.25 with step size 1 will show each plane for 4 frames. See the [video mini-example](#). See also: [vop tile](#)

- **expandSinglePlane** true | false

Expand a single-plane display along its perpendicular axis to the full thickness of the data. This option does not apply to [orthogonal planes](#).

- **orthoplanes** xyz | xy | yz | xz | off [ **positionPlanes** *i,j,k* ]

Display planes perpendicular to the X, Y, and/or Z data axes within the [current region](#). If only a single plane is shown beforehand, it will be [expanded](#) to define the region. The **positionPlanes** option specifies plane locations along the axes by integer grid indices (default 0,0,0 or the lowest indices in the [current region](#)). Orthogonal planes automatically use the **solid** [display style](#), opaque [color mode](#), and an [outline](#), although these can be overridden with the corresponding command options. Turning orthogonal planes **off** restores the **auto8** [color mode](#).

- **boxFaces** true | false

Display orthogonal planes as the six box faces of the [current region](#). If only a single plane is shown beforehand, it will be [expanded](#) to define the region. Box-face planes automatically use the **solid** [display style](#), opaque [color mode](#), and an [outline](#), although these can be overridden with the corresponding command options. Turning box-face planes **off** restores the **auto8** [color mode](#).

## ← File-Saving Options (Usage: [volume](#) *model-spec options*)

- **save** *filename* [ **append** true|false ] [ **saveStep** *N* | *Nx,Ny,Nz* ]

[ **saveRegion** all | *name* | *i1,j1,k1,i2,j2,k2* ] [ **maskZone** true|false ] [ **baseIndex** *M* ]

Write the data to a file in [MRC](#) (default), NetCDF, [Chimera map](#), or BRIX format. The desired format can be indicated with the corresponding [filename suffix](#) **.mrc**, **.nc**, **.cmap** or **.cmp**, or **.brix** (overridden by [saveFormat](#)). For [Chimera map](#) format only, **append true** can be used to append the data to an existing file instead of overwriting it.

The output pathname *filename* cannot contain spaces. However, specifying *filename* as **browse** or **browser** will raise a [dialog](#) for saving the file. Multiple data sets can be written

to multiple files by including a %d integer format specification in *filename*:

```
volume #2,3,7 save data%d.mrc
volume #2,3,7 save data%03d.mrc
```

The first example would generate the files data1.mrc, data2.mrc, and data3.mrc, while the second would instead use names like data001.mrc. Successive integers starting with **baseIndex** *M* (default 1) will be used in the new names regardless of the volume model numbers. If the format specification is incorrect, *filename* will be interpreted as the name of a single file. Only the [Chimera map](#) format can accommodate multiple data sets in a single file.

The output file header will include information for converting between grid indices and Cartesian coordinates, such as the [origin](#) and [scale](#).

- **saveStep** indicates whether to write the full resolution of the data (step size 1, default) or a specified [subsample](#) (step size > 1). Step sizes must be integers. A step size of 1 indicates all data points, 2 indicates every other data point, 3 every third point, *etc.* If a single number is supplied, it is used along all three axes; if three numbers are supplied (separated by commas but not spaces), they are used along the X, Y, and Z axes, respectively.
- **saveRegion** indicates whether to write the full extents of the data or a specified [subregion](#). The full extents are specified by **all**, while a subregion can be specified by *name* (previously [assigned](#)) or by grid indices *i1-i2* along the X axis, *j1-j2* along the Y axis, and *k1-k2* along the Z axis. Grid indices must be integers separated by commas but not spaces. If the **saveRegion** option is not supplied, the currently displayed subregion will be written. In this context, a zone is not considered a subregion.
- If **maskZone** is **true** and zoning is in effect (using [Surface Zone](#) or the [Zone](#) feature in [Volume Viewer](#)), data will be written out for a region enclosing the zone, with values outside the zone set to zero.

- **saveFormat** `mrc` | `netcdf` | `cmap` | `dsn6`

Specify the **save** format as [MRC](#) (default), NetCDF, [Chimera map](#), or BRIX; the keywords are the corresponding [filename prefixes](#). This option overrides any [filename suffix](#) supplied with **save**.

When Chimera map format is [saved](#):

- multiple data sets can be saved to a single file (multiple models can be [specified](#))
- the data layout can be controlled with [chunkShapes](#)
- the file can be compressed with [compress](#)
- the file will include any [symmetry assignment](#)

- **chunkShapes** *order*

Control data layout when [saving](#) Chimera map format. Layout affects the efficiency of later reading the data, primarily a concern for very large data sets (hundreds of Mb). The *order* can be one or more of the following, separated by commas but not spaces:

- **zyx** (default)
- zxy
- yxz
- yzx
- xzy
- xyz

Data are written in blocks of up to 64 Kb. The blocks are shaped according to the specified *order*: smallest along the first axis and largest along the third. Data planes are read from a file with different efficiencies depending on their orientations and the layout. For example, from data written in the default order (**zyx**), XY planes will be read the most efficiently and YZ planes the least efficiently. When multiple orders are specified, multiple copies of the data are written to the same file. When the file is read, the most efficient copy available will be used. Saving a file with an *order* ending in "z" can be very slow because one Z-plane is written at a time.

- **compress** true | false

Compress the file when [saving](#) Chimera map format. Most useful for volume masks (values 0/1), as other data sets tend to be noisy and compress very little, if at all.

- **dumpHeader** true | false

Write the file header contents (if any) of a map read from MRC or CCP4 format to the [Reply Log](#).

## ← Surface and Mesh Display Options (Usage: [volume](#) *model-spec options*)

The **surface** and **mesh** [display styles](#) both depict isosurfaces.

- **surfaceSmoothing** true | false

Whether to smooth surface and mesh displays. Smoothing entails moving each vertex a [specified fraction](#) of the way toward the average position of its neighbors a [specified number](#) of times.

- **smoothingIterations** *N*

How many iterations of smoothing to perform (default **2**) when [surfaceSmoothing](#) is set to true. Each vertex is moved once per iteration.

- **smoothingFactor** *f*

How far to move each vertex when [surfaceSmoothing](#) is set to true. In each [iteration](#), each vertex is moved a fraction *f* (ranging from 0.0 to 1.0, default **0.3**) of the way toward the average position of the vertices connected to it by triangle edges.

- **subdivideSurface** true | false

Whether to subdivide each triangle in surface and mesh displays into four smaller triangles a [specified number](#) of times. A triangle is subdivided by connecting the midpoints of its edges. Subdivision can help to produce smoother surfaces when combined with the

[surfaceSmoothing](#) option.

- **subdivisionLevels** *j*

How many times to subdivide triangles when [subdivideSurface](#) is set to true. The number of triangles is increased by a factor of  $4^j$ , where *j* is a positive integer (default 1).

- **smoothLines** true | false

Turn on anti-aliasing to smooth lines in mesh displays. Mesh lines with [transparency](#) > 0.0 can only be smoothed when [dimTransparency](#) is true. A side effect of OpenGL anti-aliasing is that dense meshes look brighter from some viewpoints and darker from others, depending on the order in which the lines were drawn.

- **squareMesh** true | false

Display only a subset of the lines in the triangular mesh. Lines in the square mesh show the intersection of the XY, YZ, and XZ grid planes with the contour surface.

- **lineThickness** *width*

Set pixel linewidth used in mesh displays. The *width* must be a positive integer (default 1).

- **dimTransparency** true | false

Decrease the brightness of surface and mesh displays as their [transparency](#) is increased. When dimming is on, OpenGL (alpha,1-alpha) blending is used instead of (1,1-alpha) blending.

- **meshLighting** true | false

Make the inside of a mesh-enclosed volume dimmer than the outside by varying the brightness according to the angle between each surface point normal and the line of sight. Brightness is maximal when the outward-facing normal is parallel to the line of sight and pointing at the user (see more on the definition of "outward" under [flipNormals](#)). When this option is off, brightness is uniform regardless of the angle between the normal and the line of sight.

- **twoSidedLighting** true | false

Light both sides of surface displays. Otherwise, only the outside of a surface-enclosed volume will be lit (see more on the definition of "outside" under [flipNormals](#)). The brightness of each lit side varies according to the angle between a surface point normal and the line of sight; brightness is maximal when the normal is parallel to the line of sight.

- **flipNormals** true | false

Affects surface displays when [twoSidedLighting](#) is set to false, mesh displays when [meshLighting](#) is set to true. When **flipNormals** is true, the side toward larger or more positive values is treated as the outside for negative [thresholds](#) and the side toward smaller or more negative values is treated as the outside for positive [thresholds](#) (appropriate for data in which the sign is meaningful, such as [electrostatic potential](#)). When **flipNormals** is false, the side toward smaller or more negative data values is always treated as the outside.

- **capFaces true | false**

Cover the faces of the volume data box where high values would be exposed.

## ← Solid Display Options (Usage: [volume](#) *model-spec options*)

The **solid** [display style](#) shows data as a semitransparent solid.

- **colorMode** *cmode*

Specify color pixel format (OpenGL texture format). Possible values of *cmode* combine a string describing the types of information:

- rgba - multiple colors and transparency
- rgb - multiple colors, opaque
- la - luminance (single color) and transparency
- l - luminance (single color), opaque
- auto - set the mode based on the current display

with a number of bits: 4, 8, 12, 16. For example, the default *cmode* is **auto8**.

- **projectionMode** *pmode*

Specify projection mode for memory-efficient display of large data sets such as tomograms. Possible values of *pmode*:

- **auto** (default)
- 2d-x
- 2d-y
- 2d-z
- 2d-xyz
- 3d

Displaying just the planes perpendicular to one data axis (2d-x, 2d-y, or 2d-z) uses less memory than automatically switching to those along the data axis most perpendicular to the screen at a given time (2d-xyz). The auto setting uses 2d-z for volumes with X or Y dimensions at least 4 times greater than Z, otherwise 2d-xyz. The 3d option (3D texture mapping) uses planes perpendicular to the current line of sight, which may not lie along any data axis. If this option is chosen but not supported by the computer hardware, an empty red outline box will be shown; if it is supported but the texture is too large, an empty yellow outline box will be shown.

- **maximumIntensityProjection true | false**

At each pixel, display the the most intense color value underlying the pixel along the line of sight. The maximum intensities of the red, green, and blue color components are determined separately, and transparency is ignored. This option can be useful for enhancing detail. Unphysical effects can result, but are usually not very noticeable; examples include the disappearance of a dim spot when it passes in front of a brighter spot and the simulation of a single spot when the maximal values of different color components under the same pixel actually come from different spots.

- **dimTransparentVoxels true | false**

Scale voxel brightness in solid displays by a factor of (1–transparency). Otherwise, increasing the transparency also makes a volume appear brighter, because less light is blocked.

- **btCorrection** true | false

Correct brightness and transparency for the viewing angle. Without this correction, the apparent brightness and transparency of solid displays (in [projection modes](#) other than 3d) will depend on the viewing angle relative to the data axes. For a cube-shaped volume with equal resolution in the X, Y, and Z dimensions, the brightness drops and the transparency increases by a factor of  $3^{1/2}$  (approximately 1.7) as the viewing angle is changed from along any axis to along the cube diagonal. The brightness correction remedies this, but doubles rendering time.

- **minimalTextureMemory** true | false

Reuse a single 2D texture for solid displays (in [projection modes](#) other than 3d) instead of allocating separate textures for every plane of the data. This is useful for viewing large data sets that would otherwise fail to display, but can degrade interactive response.

- **linearInterpolation** true | false

Linearly interpolate brightness and transparency between voxels in solid displays. Turning interpolation off may yield a pixelated appearance but speed up rendering, depending on the graphics hardware.

**Usage:****vop** *operation arguments* [options](#)

The **vop** command edits [volume data](#) to create a new volume data set. The original map is undisplayed and the new map is displayed with the same threshold and color as the original. Map display can be adjusted and the map saved to a file using [Volume Viewer](#) or the command [volume](#). See also: [mask](#), [sop](#), [Volume Filter](#), [Volume Eraser](#), [Segment Map](#)

Examples:

```
vop add #1-25 onGrid #0
vop add #1,2,5 onGrid #5 inPlace true
vop add #1,2 boundingGrid false
vop gaussian #3 sd 5
vop subtract #2 #4 modelId #5
vop unbend #0 #1 z 200 200
```

The *operation* can be:

- [add](#) - add two or more maps
- [bin](#) - reduce data size by averaging over cells of multiple grid points
- [boxes](#) - extract cubic regions centered on markers
- [cover](#) - extend map to cover specified atoms or box
- [falloff](#) - smooth the boundaries of masked density
- [flatten](#) - scale values to flatten map baseline
- [fourier](#) - Fourier transform
- [gaussian](#) - Gaussian filtering
- [laplacian](#) - Laplacian filtering
- [localCorrelation](#) - calculate map-map correlation over a sliding box
- [maximum](#) - take the maximum values pointwise from two or more maps
- [median](#) - set each value to the median of values in a surrounding box
- [minimum](#) - take the minimum values pointwise from two or more maps
- [morph](#) - morph (interpolate) between two or more maps
- [multiply](#) - multiply values in two or more maps
- [~octant](#) - erase positive octant
- [octant](#) - erase all but the positive octant
- [permuteAxes](#) - permute axes
- [ridges](#) - skeletonize; emphasize ridges or filaments in the density
- [resample](#) - resample on the grid of another map
- [scale](#) - scale, shift, normalize, and/or cast to a different data value type
- [subtract](#) - subtract another map from the first
- [threshold](#) - reassign values that are below a specified minimum and/or above a specified maximum
- [tile](#) - make a single-plane volume from tiled slices of another volume
- [unbend](#) - unbend a map near a path formed by [markers/links](#) or atoms/bonds

- [unroll](#) - unroll a cylindrical slab into a flat slab
- [zFlip](#) - reverse order of Z planes
- [zone](#) - zero the values at grid points within or beyond a cutoff distance from specified atoms

Volume models (maps) are specified by a comma-separated list of model numbers or ranges of model numbers preceded by #. Operation keywords and their sub-keywords described below can be truncated to unique strings, and their case does not matter. Synonyms for true: True, 1. Synonyms for false: False, 0. A vertical bar “|” designates mutually exclusive options, and default settings are indicated with **bold**.

- **vop add** *volume-spec* [ **scaleFactors** *f1,f2,...* ] [ **onGrid** *gridmap* ] [ **boundingGrid** true|false ] [ **gridStep** *N* | *Nx,Ny,Nz* ] [ **gridSubregion** *name* | *i1,j1,k1,i2,j2,k2* | **all** ] [general-options](#)

Add two or more maps to create a new map. Option keywords are the same as for [vop minimum](#), [vop maximum](#), and [vop multiply](#):

The **scaleFactors** keyword specifies a multiplier for each map (default **1.0**); as many values as input maps must be supplied, separated by commas but not spaces.

The new map can be created on the grid of another, where *gridmap* is a model number preceded by #. If *gridmap* is not specified, it defaults to the first in *volume-spec* (the first of the maps being added). The input maps are resampled on the grid by trilinear interpolation, and the resulting values summed for each grid point. Further options related to *gridmap*:

- **boundingGrid** - whether to adjust (extend or shrink) the grid of *gridmap* to bound the input maps (default **true** when adding maps without specifying a *gridmap*, otherwise **false**)
- **gridStep** - whether to use the full resolution of *gridmap* (step size **1**, default) or a specified [subsample](#) (step size > 1). Step sizes must be integers. If a single number is supplied, it is used along all three axes; if three numbers are supplied (separated by commas but not spaces), they are used along the X, Y, and Z axes, respectively.
- **gridSubregion** - whether to use the full extents of *gridmap* (**all**, default) or a specified [subregion](#). A subregion can be specified by:
  - *name* previously assigned with [volume](#) (see [nameRegion](#)) or [Volume Viewer](#) (see [Named regions](#))
  - grid indices *i1-i2* along the X axis, *j1-j2* along the Y axis, and *k1-k2* along the Z axis. Grid indices must be integers separated by commas but not spaces.

If the new map is large, for example a whole tomogram, the command may fail for lack of memory. The whole new map must fit in memory.

- **vop bin** *volume-spec* [ **binSize** *N* | *Nx,Ny,Nz* ] [general-options](#)

Average over cells of multiple grid points in the original map to produce a smaller map. Supplying a single integer *N* (default **2**) indicates partitioning the map into bins of  $N \times N \times N$  grid points and averaging the  $N^3$  values per bin to produce a new map with  $1/N$  as many points in each dimension. Cells with different numbers of grid points in each dimension can

be specified by supplying three integers  $N_x, N_y, N_z$  separated by commas only.

See also: [segment](#)

- **vop boxes** *volume-spec* [atom-spec](#) size *d* [ **useMarkerSize** true|false ] [general-options](#)

For each [marker](#) or atom in [atom-spec](#), extract a surrounding cube of data. The edge length of each cube is the size *d* plus (if **useMarkerSize** is true) the diameter of its central marker or atom. The **size** is specified in physical units of length and can be omitted (default  $d=0.0$ ) only when **useMarkerSize** is true.

- **vop cover** *volume-spec* [ **atomBox** [atom-spec](#) [ **pad** *d* ] ] [ **box** *x1,y1,z1,x2,y2,z2* ] [ **x** *x1,x2* ] [ **y** *y1,y2* ] [ **z** *z1,z2* ] [ **fbox** *a1,b1,c1,a2,b2,c2* ] [ **fx** *a1,a2* ] [ **fy** *b1,b2* ] [ **fz** *c1,c2* ] [ **ibox** *i1,j1,k1,i2,j2,k2* ] [ **ix** *i1,i2* ] [ **iy** *j1,j2* ] [ **iz** *k1,k2* ] [ **cellSize** *nx,ny,nz* ] [ **useSymmetry** true|false ] [general-options](#)

Extend a map to cover specified atoms or to fill a rectangular box, using map symmetries and periodicity. The output dimensions can be specified as:

- **atomBox** spanning the [specified](#) atoms plus any extra **pad** in each dimension (*d* is in units of physical distance, default **5.0**)
- **box** or just individual dimensions **x**, **y**, and/or **z** in the X,Y,Z coordinate system of the input map
- **fbox** or just individual dimensions **fx** *etc.* in fractional coordinates where 0.0-1.0 spans each dimension of the input map
- **ibox** or just individual dimensions **ix** *etc.* in grid indices of the input map. The input map's grid indices start at 0.

Unspecified dimensions will be kept the same as the input map. The output grid will have the same spacing and alignment as the grid of the input map. The **cellSize** option specifies unit cell dimensions in grid units along the X, Y, and Z axes. The default unit cell dimensions correspond to the full size of the map, or for CCP4 and MRC maps, are taken from the header. The **useSymmetry** option indicates whether to use any symmetries associated with the map (default **true**); if false, only unit cell periodicity will be used. Map symmetries are read from the CCP4 or MRC file header, or can be assigned manually with the [symmetry option of volume](#) or automatically with [measure symmetry](#).

Values from symmetry copies are determined by trilinear interpolation. Where symmetries and periodicity give multiple copies of the input map overlapping a grid point, the average value will be assigned. The maximum difference between values from different copies at a grid point will be reported in the [Reply Log](#). If there are grid points not covered by symmetry or unit cell periodicity, a message will be sent to the [Reply Log](#) and [status line](#), and the points will be assigned values of 0.0. To ensure complete coverage by symmetry copies, the asymmetric unit of the map should extend far enough that its symmetry copies overlap by a non-zero amount. For example, if the unit cell is 100 grid points wide and there is two-fold symmetry along the X axis, the asymmetric unit would need to contain at least 52 grid points along the X axis. If it contains only 50, the two symmetric copies will not overlap, and values in the space between the copies cannot be determined because the current code cannot interpolate between different copies of the map. If it contains 51,

the two copies will have a single plane of grid points in common. Although that would be sufficient with exact arithmetic, the copies still might not overlap given the rounding errors inherent in computer calculations.

- **vop falloff** *volume-spec* [ **iterations** *M* ] [general-options](#)

Smooth the boundaries of a masked map by replacing the value at each grid point outside the boundary with the average of the values of its six nearest neighbors, for *M* iterations (default **10**). All grid points with values of zero before the first iteration are taken to be outside the boundary, thus assigned a new value at each iteration. Thanks to Greg Pintilie for the initial implementation.

- **vop flatten** *volume-spec* [ **method** **multiply**|**divide** ] [ **fitregion** *name* | *i1,j1,k1,i2,j2,k2* | **all** ] [general-options](#)

If the **method** is **multiply**, scale data values by factor  $(a*i + b*j + c*k + d)$  where *i,j,k* are the grid indices and *a,b,c,d* are calculated to zero out the first moments of the resulting map (make its mass balance at the center of the grid). If the **method** is **divide**, data values are divided by the factor  $(a*i + b*j + c*k + d)$ , which is a least-squares fit to the map data values. For both methods, the *a,b,c,d* coefficients are scaled to make  $(a*i + b*j + c*k + d)$  equal to 1 at the center of the map. If a **fitregion** is specified, the calculation of the *a,b,c,d* coefficients uses only the data values in the specified region, while the scaling operation applies to the entire map or the part specified by the **subregion** [general option](#). The **fitregion** can be the full extents of the data (**all**, default) or a subregion specified by:

- *name* previously assigned with [volume](#) (see [nameRegion](#)) or [Volume Viewer](#) (see [Named regions](#))
- grid indices *i1-i2* along the X axis, *j1-j2* along the Y axis, and *k1-k2* along the Z axis. Grid indices must be integers separated by commas but not spaces.

- **vop fourier** *volume-spec* [ **phase** **true**|**false** ] [general-options](#)

Calculate the [3D Fourier transform](#). If **phase** is false (default), generate a magnitude map; if **phase** is true, generate a map of phase values ( $-\pi$  to  $\pi$ ) instead.

- **vop gaussian** *volume-spec* [ **sDev**  $\sigma_x,\sigma_y,\sigma_z$  ] [ **valueType** *value-type* ] [general-options](#)

Perform [Gaussian filtering](#) with half-width  $\sigma$ , one standard deviation of the 3D Gaussian function in physical units such as Å (default **1.0**). Different half-widths along X,Y,Z can be specified as three values separated by commas only. The *value-type* defaults to the [current type](#) and can be 8-, 16-, or 32-bit signed integer (**int8**, **int16**, or **int32**), 8-, 16-, or 32-bit unsigned integer (**uint8**, **uint16**, or **uint32**), or 32- or 64-bit floating-point (**float32** or **float64**).

- **vop laplacian** *volume-spec* [general-options](#)

Perform [Laplacian filtering](#).

- **vop localCorrelation** *map othermap* [ **windowSize** *N* ] [ **subtractMean** true | false ] [ **modelId** *M* ]

Calculate the correlation between two maps over a sliding box of  $N \times N \times N$  grid points, generating a new map by assigning the correlation value to the box center. The sliding box is based on the grid of the first map, and  $N = 5$  grid units by default. If the grids of the two input maps do not coincide, the values of the second map will be interpolated. The **subtractMean** option specifies subtracting the mean of the values in the window from each value in the window before calculating the correlation. The output map will be  $N-1$  smaller in each dimension than the first map. It will be opened as model number  $M$ , or if not specified, the lowest unused model number.

- **vop maximum** *volume-spec* [ **scaleFactors** *f1,f2,...* ] [ **onGrid** *gridmap* ] [ **boundingGrid** true|false ] [ **gridStep** *N* | *Nx,Ny,Nz* ] [ **gridSubregion** *name* | *i1,j1,k1,i2,j2,k2* | all ] [general-options](#)

Set each value to the maximum at that point in two or more input maps. See [vop add](#) for descriptions of the options.

- **vop median** *volume-spec* [ **binSize** *N* | *Nx,Ny,Nz* ] [ **iterations** *M* ] [general-options](#)

Smooth the data by setting each value to the median of the values in a box centered at that point. Values at points for which the surrounding box extends outside the data are simply set to zero. Box dimensions are specified in grid units with **binSize** and must be odd integers. Supplying a single integer  $N$  (default **3**) indicates a box size of  $N \times N \times N$  grid points. Boxes with different numbers of grid points in each dimension can be specified by supplying three integers  $N_x, N_y, N_z$  separated by commas only. The **iterations** option indicates how many cycles of smoothing to perform (default **1**).

- **vop minimum** *volume-spec* [ **scaleFactors** *f1,f2,...* ] [ **onGrid** *gridmap* ] [ **boundingGrid** true|false ] [ **gridStep** *N* | *Nx,Ny,Nz* ] [ **gridSubregion** *name* | *i1,j1,k1,i2,j2,k2* | all ] [general-options](#)

Set each value to the minimum at that point in two or more input maps. See [vop add](#) for descriptions of the options.

- **vop morph** *volume-spec* [ **start** *start-fraction* ] [ **playStep** *increment* ] [ **frames** *N* ] [ **playDirection** 1 | - 1 ] [ **playRange** *low-fraction,high-fraction* ] [ **scaleFactors** *f1,f2,...* ] [ **constantVolume** true|false ] [ **addMode** true|false ] [ **hideOriginalMaps** true|false ] [ **interpolateColors** true|false ] [general-options](#)

Morph between two or more maps (this is the command-line implementation of [Morph Map](#), except that more than two maps can be handled). For a reasonable result, the input maps should have the same grids: dimensions, spacing, and numbers of points. Note [vop resample](#) can be used to make a copy of one map that has the same grid as another. A morphing fraction of 0.0 corresponds to the first map and a fraction of 1.0 corresponds to the last, with intermediate maps evenly spaced within that range. There is smooth interpolation between each adjacent pair of maps. See the [video mini-example](#).

The morph display will proceed from *start-fraction* (default **0.0**) in steps of *increment* (default **0.04**) for *N frames* (default **25**). By default (**playDirection 1**), the initial direction of play is from low to high fractions. If the number of frames and step increment are more than needed to reach the **playRange** bounds (default is the entire range: **0.0,1.0**), the morph display will “bounce” back and forth. The **scaleFactors** keyword specifies a multiplier for each map (default **1.0**); as many values as input maps must be supplied. The **constantVolume** option specifies adjusting the [threshold](#) (contour level) automatically to keep the enclosed volume constant. The **addMode** option specifies treating the second map as a delta to be added to the first instead of linearly interpolating between the two. It is not recommended for inputs of >2 maps. The **hideOriginalMaps** option specifies hiding the input maps. The **interpolateColors** option only applies when the maps have the same number of [coloring thresholds](#) (contour levels for surface/mesh display, coloring control nodes for solid display).

See also: [morph](#), [vseries](#), [movie-related commands](#), the [ParM filament tutorial](#) at the Chimera web site

- **vop multiply** *volume-spec* [ **scaleFactors** *f1,f2,...* ] [ **onGrid** *gridmap* ] [ **boundingGrid** true|false ] [ **gridStep** *N* | *Nx,Ny,Nz* ] [ **gridSubregion** *name* | *i1,j1,k1,i2,j2,k2* | all ] [general-options](#)

Multiply the values pointwise in two or more maps. This is used to apply a mask (values 0,1) to a map. See [vop add](#) for descriptions of the options.

- **vop ~octant** *volume-spec* [ **center** *x,y,z* | **iCenter** *i,j,k* ] [ **fillValue** *value* ] [general-options](#)

Erase values inside the positive octant (all grid points with X,Y,Z coordinates greater than the center). The center can be specified in physical units (such as Å) with **center** or in grid units with **iCenter**. The default is the center of the volume data box. The coordinates should be separated by commas but not spaces, and the values can be fractional. **iCenter** overrides **center** if both are given. The values in the erased regions will be set to *value* (default **0**). A different value may improve contour surface appearance; for example, large negative values produce flatter surfaces where an octant has been cut away from a map of positive values.

- **vop octant** *volume-spec* [ **center** *x,y,z* | **iCenter** *i,j,k* ] [ **fillValue** *value* ] [general-options](#)

Erase values outside the positive octant. Options are as described for [~octant](#) above.

- **vop permuteAxes** *volume-spec* [ **axisOrder** *order* ] [general-options](#)

Permute grid axes to the specified *order*, which can be any of the 6 ordered combinations of **x**, **y**, and **z**. The original order is **xyz**.

- **vop ridges** *volume-spec* [ **level** *minimum* ] [general-options](#)

Skeletonize map(s) by tracing along high-density grid points to identify ridges or filamentous structures in the density. At each grid point, the value is compared to the values of all the points in the surrounding 3x3x3 box, and the count of how many directions (up to 13) along which the value is a local maximum is assigned as that point's value in the new map. The **level** keyword indicates a *minimum* value in the original map below which to automatically set the new value to 0, essentially ignoring those points in the skeletonization. The default *minimum* is the lowest display [threshold](#) (contour level) in the original map. Viewing the new skeleton map with a threshold of 6-10 highlights ridgelike features in the original map.

See also: [Volume Tracer](#)

- **vop resample** *volume-spec onGrid gridmap* [ **boundingGrid** true|false ] [ **gridStep** *N* | *Nx,Ny,Nz* ] [ **gridSubregion** *name* | *i1,j1,k1,i2,j2,k2* | **all** ] [general-options](#)

Resample values on the grid of another map, where *gridmap* is a model number preceded by #. Values on the grid are obtained by trilinear interpolation of the input map. The other arguments are as described [above](#) for [add](#).

- **vop scale** *volume-spec* [ **shift** *constant* ] [ **factor** *f* ] [ **rms** *new-rms* | **sd** *new-std-dev* ] [ **valueType** *value-type* ] [general-options](#)

Shift values by adding a *constant* (default **0.0**), scale values by a multiplicative factor *f* (default **1.0**), and/or cast them to a different data value type. When values are both shifted and scaled, the shift is applied first. Two normalization options calculate a scaling factor from the data:

- **rms** - scale values to make  $new-rms = ((\sum x^2)/N)^{1/2}$  where *x* is each value and *N* is the total number of values
- **sd** - first shift values so that the mean is 0.0, then scale values to make  $new-std-dev = ((\sum x^2)/N)^{1/2}$

If a **factor** *f* is also specified, it is applied last. The *value-type* defaults to the [current type](#) and can be 8-, 16-, or 32-bit signed integer (**int8**, **int16**, or **int32**), 8-, 16-, or 32-bit unsigned integer (**uint8**, **uint16**, or **uint32**), or 32- or 64-bit floating-point (**float32** or **float64**).

See also: [measure mapStats](#)

- **vop subtract** *map othermap* [ **scaleFactors** *f1,f2* ] [ **minRMS** true|false ] [ **onGrid** *gridmap* ] [ **boundingGrid** true|false ] [ **gridStep** *N* | *Nx,Ny,Nz* ] [ **gridSubregion** *name* | *i1,j1,k1,i2,j2,k2* | **all** ] [general-options](#)

Subtract the values of *othermap* from *map*, both specified by model number preceded by #. The **scaleFactors** keyword specifies multipliers *f1* and *f2* for *map* and *othermap*, respectively; two values must be supplied, separated by a comma but not spaces. Alternatively, the **minRMS** option can be used to scale *othermap* automatically to minimize

the root-mean-square sum of the resulting (subtracted) values at grid points within the lowest contour of *othermap*.

The new map can be created on the grid of another, where *gridmap* is a model number preceded by #. If *gridmap* is not specified, it defaults to *map*. The input maps are resampled on the grid by trilinear interpolation, and the resulting values subtracted for each grid point. The remaining arguments are as described [above](#) for [add](#), except that **boundingGrid** always defaults to **false**.

- **vop threshold** *volume-spec* [ **minimum** *min* ] [ **set** *newmin* ] [ **maximum** *max* ] [ **setMaximum** *newmax* ] [general-options](#)

Replace all values that are below a **minimum** value (*min*) with *newmin* (default equal to *min*), and/or replace all values that are above a **maximum** value (*max*) with *newmax* (default equal to *max*).

- **vop tile** *volume-spec* [ **axis** *x|y|z* ] [ **pstep** *plane-step* ] [ **trim** *i* ] [ **rows** *r* ] [ **columns** *c* ] [ **fillOrder** *order* ] [general-options](#)

Create a single-plane volume by tiling slices of a specified volume perpendicular to the specified **axis** (default **z**). The spacing of slices (default **1** grid unit) is given with the **pstep** keyword. The **trim** keyword indicates each slice should be trimmed on all four edges by *i* grid units (default **0**). The slices are arranged into a single plane with number of **rows** *r* and number of **columns** *c*. If neither the number of rows nor the number of columns is supplied, they are computed to produce as near a square tiling as possible. If one or the other is supplied, the remaining parameter is adjusted to accommodate the total number of slices. The **fillOrder** setting (default **ulh**) specifies the tiling pattern, including the starting corner, the tiling direction (horizontal or vertical), and whether to reverse the order of slices. The first two characters specify the corner for the first tile, the first character being **u** for upper or **l** for lower and the second being **l** for left or **r** for right. These directions are defined with the specified axis pointing at the viewer and the remaining two axes pointing up and right. The third character is **h** for horizontal tiling or **v** for vertical tiling. The optional fourth character **r** indicates that the order of the slices should be reversed. The resulting volume data set has the same origin and orientation of axes as the original volume, and grid size 1 along the specified axis.

See also: [segment sliceimage](#), [topography](#), [tile](#)

- **vop unbend** *volume-spec path-spec new-y xsize ysize* [ **gridSpacing** *s* ] [general-options](#)

Unbend a map near a path formed by [markers/links](#) or equivalently, atoms/bonds. The *path-spec* should be an [atom-spec](#) that specifies a single chain of atoms (markers) connected by bonds (links). The path will be mapped to the Z axis of the result. The *new-y* parameter defines what axis in the existing volume will be mapped to the Y axis of the result, and can be given as:

- **x** - X-axis
- **y** - Y-axis

- **z** - Z-axis
- *x,y,z* (three values separated by commas only) - an arbitrary vector
- an [atom-spec](#) of exactly two atoms (not necessarily bonded or in the same model) or one bond. A bond can only be specified by [selecting](#) it and using the word **selected**, **sel**, or **picked**; any atoms also selected at the time will be ignored.

The *xsize* and *ysize* parameters give the X and Y dimensions of the new map in physical units (typically Å). The **gridSpacing** *s* is the separation between grid points in the new map (default is the minimum spacing along the three axes of the input map). A cubic spline is placed through the path points and the input volume is interpolated on planes perpendicular to the splined path.

See also: [segment sliceimage](#)

- **vop unroll** *volume-spec* [ **center** *x,y,z* ] [ **axis** *x | y | z | x,y,z | atom-spec* ] [ **coordinateSystem** *N* ] [ **length** *d* ] [ **innerRadius** *r1* ] [ **outerRadius** *r2* ] [ **gridSpacing** *s* ] [general-options](#)

Unroll a hollow cylindrical section of the map into a flat slab. The cylinder **axis** can be given as:

- **x** - X-axis
- **y** - Y-axis
- **z** (default) - Z-axis
- *x,y,z* (three values separated by commas only) - an arbitrary vector
- an [atom-spec](#) of exactly two atoms (not necessarily bonded or in the same model) or one bond. A bond can only be specified by [selecting](#) it and using the word **selected**, **sel**, or **picked**; any atoms also selected at the time will be ignored.

Cylinder **axis** and **center** (default **0,0,0**) coordinates are interpreted in the coordinate system of the input map, unless another reference model is specified with **coordinateSystem**. The dimensions of the cylindrical slab are given in physical units of length, typically Å: **length** *d* (default is the map extent parallel to the cylinder axis) and inner and outer radii *r1* and *r2* (defaults are 90% of the smallest radius and 110% of the largest radius of the displayed isosurface, respectively, given the cylinder center and axis direction). The flattening is done by interpolating values from the original map on a cylindrical grid of points, then unwrapping the cylindrical grid into a rectangular grid. The cylinder radial direction becomes the X-axis of the new map, circumference the Y-axis, and cylinder axis direction the Z-axis. The **gridSpacing** *s* is the requested separation of grid points along each axis in the new map (default is the minimum spacing along the three axes of the input map). The actual spacing may be slightly different because the dimensions of the new map may not be an exact multiple of the requested value; the number of grid divisions along each axis is chosen to give spacing as close as possible to the requested value without being smaller.

- **vop zFlip** *volume-spec* [general-options](#)

Reverse the order of the Z planes.

- **vop zone** *volume-spec* [atom-spec](#) *radius* [ **invert** true|false ] [ **minimalBounds** true|false ] [ **bondPointSpacing** *s* ] [general-options](#)

Set the values of grid points farther than *radius* Å from any atom in [atom-spec](#) (those beyond the zone) to zero, or if **invert** is true, set the values of grid points within the zone to zero. If **minimalBounds** is true, make the resulting map as small as possible while enclosing the zone; otherwise, the dimensions will be the same as for the input map. If **bondPointSpacing** *s* is specified, use points along bonds in addition to the atoms to define the zone. The points along the bonds will be placed  $s \times$  ([bond radius](#)) apart. [Link radii](#) in [Volume Tracer](#) are equivalent to bond radii, except when a link radius is 0.0, the corresponding bond radius is 1.0.

See also: the [Zone](#) feature in [Volume Viewer](#)

## General Options

### **modelId** *N*

Open the new data set as model number *N* (an integer, optionally preceded by #). The default is the lowest unused number.

### **step** *N* | *Nx,Ny,Nz*

Whether to use the full resolution of the data (step size **1**, default) or a specified [subsample](#) (step size > 1). Step sizes must be integers. A step size of 1 indicates all data points, 2 indicates every other data point, 3 every third point, *etc.* If a single number is supplied, it is used along all three axes; if three numbers are supplied (separated by commas but not spaces), they are used along the X, Y, and Z axes, respectively.

### **subregion** *name* | *i1,j1,k1,i2,j2,k2* | **all**

Whether to use the full extents of the data (**all**, default) or a specified [subregion](#). A subregion can be specified by:

- *name* previously assigned with [volume](#) (see [nameRegion](#)) or [Volume Viewer](#) (see [Named regions](#))
- grid indices *i1-i2* along the X axis, *j1-j2* along the Y axis, and *k1-k2* along the Z axis. Grid indices must be integers separated by commas but not spaces.

### **inPlace** true|false

Whether to overwrite the existing data set in Chimera instead of creating a new one. Not all [operations](#) accept this option. Regardless of this setting, the existing data will only be overwritten if it was created in Chimera (for example with a previous **vop** command) rather than read from a file. In the case of map [addition](#), the model to overwrite is the *gridmap* (the model whose grid will be used for the result).

**Usage:****vseries** *action arguments*

The **vseries** command manipulates an ordered sequence of [volume data](#) sets. It is the command-line implementation of [Volume Series](#). See also: [vop morph](#), [volume](#), [movie-related commands](#)

Possible values of *action*:

- [open](#) - open data
- [close](#) - close data
- [play](#) - series playback
- [stop](#) - stop looping playback
- [align](#) - align the maps in the series
- [measure](#) - track surface area, enclosed volume, and centroid position
- [save](#) - save to [Chimera map](#) format, optionally with cropping, normalization, alignment

Examples:

```
vseries open myseries*.mrc
vseries open ~/Desktop/myseries.cmap
vseries play #0 loop true
vseries stop #0
vseries save #0 ~/Desktop/test.cmap subregion 100,0,0,200,511,150 threshold 140 valueType
uint8
```

Arguments for each action are described below. Option keywords for **vseries** can be truncated to unique strings and their case does not matter. A vertical bar “|” designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

- **vseries open** *filename*

Read the volume series from one or more files specified by *filename* (including path/location). The wildcard \* can be used to specify multiple files. If *filename* includes spaces, it should be enclosed in quotes. The members of a series are indexed 0, 1, 2, ... and these indices are referred to as the *time*. Any numbers in the individual filenames are not used.

- **vseries close** *volume-spec*

Close the volume series, where *volume-spec* is the model number of any member of the series.

- **vseries play** *volume-spec* [ *play-options* ]

Play back the [specified](#) volume series. The *play-options* are as follows:

**jumpTo** *time*

Go directly to the specified [time](#).

**direction** *mode*

Specify the playback *mode*:

- **forward** (default) - in increasing order, then back to the first when [looping](#) is on (01230123...)
- **backward** - in decreasing order, then back to the last when [looping](#) is on (32103210...)
- **oscillate** - alternating increasing and decreasing order (01232101...)

**loop** true | false

Whether to loop playback continuously until it is halted with **vseries stop**.

**maxFrameRate** *rate*

Specify a maximum playback *rate* in steps per second. By default, playback is as fast as possible, which can be fairly slow for large data. This option is used to slow playback when it is too fast.

**normalize** true | false

Whether to adjust the [thresholds](#) (contour levels) to keep the enclosed volume constant throughout the series. This is useful when the signal level in the data changes over time or between states.

**showMarkers** true | false [ **precedingMarkerFrames** *N* ] [ **followingMarkerFrames** *M* ]

Whether to display markers previously created manually with [Volume Tracer](#) (or automatically with [vseries measure](#)) to trace spatial and temporal paths. Each marker is labeled with the [time](#) of the data set on which it was placed. The labels are not shown, but used by **vseries** to limit marker display to the corresponding [time](#). Simultaneously, markers can also be displayed for *N* earlier time points (default 0) and *M* later time points (default 0) using **precedingMarkerFrames** and **followingMarkerFrames**, respectively.

**colorRange** *cutoff*

Whether to color volume contour surfaces to match markers within a specified distance *cutoff* (regardless of whether the markers are [shown](#)). All of the markers associated with the current [time](#) are used to color the current surface. The coloring does not apply to solid displays.

**cacheFrames** *K*

Whether to store volume rendering (surface triangle or solid voxel) information for the *K* most recent displays (default 1). This can speed playback because less time is spent recalculating display information. There is no hard limit to the memory used to cache rendering information. Surface renderings use memory proportional to the number of triangles composing the surface. Solid renderings use memory proportional to the number of data voxels displayed, and it is generally only feasible to cache solid display information for small data sets.

- **vseries stop** *volume-spec*

Halt playback of the [specified](#) volume series.

- **vseries align** *volume-spec* [ **encloseVolume** *volume* | **fastEncloseVolume** *volume* ]

Align the maps in the [specified](#) volume series. (Alternatively, alignment can be done with the [align](#)

option of [vseries save](#).) The **encloseVolume** option indicates setting the [threshold](#) (contour level) to enclose the specified *volume* in distance units cubed (e.g., Å<sup>3</sup>) before aligning the maps. The contour level affects alignment because only values above the contour level are used. The level is determined by an iterative procedure ([details...](#)). The **fastEncloseVolume** option is similar but uses a faster, noniterative approximation ([details...](#)).

- **vseries measure** *volume-spec* [ **output** *filename* ] [ **centroids** true | false ] [ **radius** *centroid-radius* ] [ **color** *centroid-color* ]

For each [time](#) in the [specified](#) volume series, calculate centroid (x,y,z) coordinates, distance from the previous centroid (“step”), cumulative distance along the piecewise linear path from the first centroid, surface-enclosed volume, and surface area. The results are saved as plain text in *filename* specified with the **output** option, otherwise given in the [Reply Log](#).

The calculation uses the [step size](#) and [threshold](#) (contour level) of the first map in the series for every time point; *i.e.*, the contour level is not adjusted to maintain a constant volume. The centroid is the center of mass of the density map based on map regions above the threshold. The surface area is a sum over the triangles of the contour surface. The surface-enclosed volume does not include interior bubbles (if any), and any holes in the surface are treated as if covered by planar caps.

If **centroids** is **true**, a [marker](#) will be placed at each centroid, with radius *centroid-radius* (default is the minimum grid-spacing in the maps) and color *centroid-color* (default **gray**). The *centroid-color* can be any [color name](#) that specifies a single color, enclosed in quotes if it contains spaces. Successive markers are linked. The markers can be shown during playback with the [showMarkers](#) option of [vseries play](#).

See also: [measure](#)

- **vseries save** *volume-spec filename* [ *save-options* ]

Save the [specified](#) series as a single file (*filename*) in [Chimera map format](#), optionally with processing such as cropping, normalization, and alignment. The *save-options* are listed below in order of application when used together:

**subregion** *i1,j1,k1,i2,j2,k2*

Instead of saving the full dimensions, save the subregion delimited by grid indices *i1-i2* along the X axis, *j1-j2* along the Y axis, and *k1-k2* along the Z axis.

**valueType** *value-type*

Change grid value type before any processing with other save options. The related option [finalValueType](#) sets the value type after any processing. The *value-type* can be:

- **int8** - 8-bit signed integer
- **int16** - 16-bit signed integer
- **int32** - 32-bit signed integer
- **uint8** - 8-bit unsigned integer
- **uint16** - 16-bit unsigned integer
- **uint32** - 32-bit unsigned integer
- **float32** - 32-bit floating-point
- **float64** - 64-bit floating-point

**threshold** *minimum*

Replace all values below the specified *minimum* with zero. If this option is used, the output will be compressed.

**zeroMean** true | false

Subtract the mean from each value so that the new mean will be zero.

**scaleFactor** *f*

Scale values by a multiplicative factor *f*.

[ **encloseVolume** *volume* | **fastEncloseVolume** *volume* ]

Set the contour level of each map to enclose a specified *volume* in distance units cubed. Either of two methods can be used, as described [above](#). The contour level affects [alignment](#) because only values above the contour level are used.

**normalizeLevel** *value*

Scale map values so that the current [threshold](#) (contour level) equals the specified *value*.

**align** true | false

Align the maps before saving them. Only values above the contour level are used for alignment; contour levels can be set to [enclose a specified volume](#).

**onGrid** *gridmap*

Create the new map on the grid of another, where *gridmap* is a model number preceded by #. This allows using a consistent grid for maps after [alignment](#).

**mask** *maskmap*

Mask by *maskmap* (multiply by the 0,1 values within that map). The *maskmap* must be on the same grid and have the same dimensions as the series maps.

**finalValueType** *value-type*

Set grid value type after any processing with other save options. Possible types are as listed for the related option [valueType](#), which sets the value type before processing.

**compress** true | false

Compress the output file ([Chimera map format](#)).

**Usage:****wait** [ *wait\_frames* ]**Wait** suspends command processing for the given number of *wait\_frames*. It is used:

- to enforce the ordered rather than simultaneous execution of commands in [scripts](#) and [demos](#)
- to generate duplicate image frames in [movies](#)

See also: [sleep](#), [pause](#), [movie-related commands](#)**Ordered Command Execution**

When the number of wait frames is not specified, **wait** suspends command processing until [certain commands](#) have finished. For example, in the following, **wait** prevents the rock from starting until the roll has finished:

```
roll y 3 120; wait
rock x 2 68
```

It is also acceptable to give the frame argument where it could have been omitted. The following is equivalent to the first example:

```
roll y 3 120; wait 120
rock x 2 68
```

**Wait** without arguments will automatically wait for the following multi-frame commands to finish: [move](#), [turn](#), [rock](#), [roll](#), [scale](#), [clip](#), [thickness](#), [section](#), [reset](#), [fly](#), [2dlabels](#) visibility changes. Other multi-frame commands require the second approach (specifying the appropriate number of wait frames) to enforce their ordered execution: [coordset](#), [perframe](#), [play](#), [scene](#), [transparency](#), [volume planes playback](#), [vop morph](#), and others.

A shorter wait can be applied to allow overlapping but staggered execution, for example:

```
2dlab change title2 visibility show frames 20; wait 10
coordset #2 1,26,1; wait 26
```

When a [command script](#) is executed, a single-frame display update (implicit **wait 1**) will be added at the end of each line that: (a) contains one or more commands that could change the display, and (b) does not already end with an explicit **wait** of any length. (Exception: display updates are not added when a script is executed with [read](#).)

**Duplicate Image Frames in Movies**

Stretches of duplicate image frames correspond to periods of time where objects remain static. If motion occurs in the last few frames of a [movie](#), due to the encoding process it may appear to end on a blur. This effect can be avoided by including a few wait frames (**wait 5** or similar) at the end of scripted [movie content](#).

**Usage:**

**window** [ [atom-spec](#) ]

If no arguments are supplied, **window** turns off [global clipping](#) and adjusts the scale to include all displayed atoms, bonds, ribbon segments, and [surfaces](#). This is equivalent to clicking **View All** in the [Side View](#).

Otherwise, **window** turns on [global clipping](#) and adjusts the scale and clipping plane positions to enclose the specified [atoms](#) and [surface pieces](#), regardless of whether they are displayed.

Model rotations and translations are left unchanged.

See also: [align](#), [center](#), [focus](#), [reset](#), the [Side View](#)

windoworigin

[Chimera Commands Index](#)

### [Usage:](#)

**windoworigin** [*x y*]

**Windoworigin** sets the location of the Chimera graphics window, where *x* and *y* are the pixel offsets of the top left corner of the window from the top left corner of the screen. Using **windoworigin** without arguments reports the current offsets in the [status line](#) and [Reply Log](#).

See also: startup option [--geometry](#), [windowsize](#)

**Usage:****windowsize** [*width height*]**Usage:****~windowsize**

**Windowsize** adjusts the dimensions of the graphics window to the specified *width* and *height* in pixels, thus also controlling the dimensions of images from [copy](#) and recorded [movies](#). Using **windowsize** without arguments reports the current pixel dimensions of the window in the [status line](#) and [Reply Log](#).

Depending on the display of other parts of the [Chimera window](#) such as the [Command Line](#), **windowsize** may not reduce the dimensions of the *overall* window below certain lower bounds. If smaller dimensions are specified, the graphics area will be drawn at the requested size, but will be flanked by unused margins as needed to fill out the overall window. Manually resizing the overall window will get rid of the unused margins, making the graphics area again fill the entire window.

**Windowsize** can be used in [nogui mode](#) (useful when offscreen rendering is possible).

See also: startup option [--geometry](#), [copy](#), [movie](#), [2dlabels](#), [windoworigin](#), [movie-related commands](#)

**Usage:**

**write** [options](#) *model\_number* [ *filename* ]

**Write** saves the atomic coordinates of the specified molecule model in a file named *filename* of the specified [format](#). The *model\_number* is optionally preceded by #. If *filename* is omitted, a [dialog](#) for specifying the name and location will appear. To save multiple models at once, use the [Save PDB](#) or [Save Mol2](#) dialog instead.

Only atomic coordinates are saved, not atomic display status, color, or radius. Non-molecule models (surfaces, VRML models, *etc.*) are not saved.

**Options****format** `pdb` | `mol2`

Which output file format to generate: [PDB](#) or [Mol2](#). Files saved in `pdb` format (default) include [HELIX and SHEET records](#) reflecting the current protein [secondary structure assignments](#), as well as any other header lines read from PDB input. Even if protein helix/strand assignments have not been changed in Chimera, the output HELIX and SHEET records may differ from the input because helices are written assuming the right-handed  $\alpha$  type, and strands are written as if each were a separate sheet.

**selected**

Write only the coordinates of [selected](#) atoms.

**displayed**

Write only the coordinates of displayed atoms.

**relative** *n*

Write coordinates relative to the *untransformed* coordinates of model number *n*. Otherwise, the coordinates will be written as currently transformed. The model number is optionally preceded by #. The **relative** option is useful for preserving the spatial relationship between models. For example, if model 1 has been matched or docked to model 0, saving 1 relative to 0 results in the models being matched or docked in the same way when the model 0 file (original) and the model 1 file (saved relative to 0) are reopened.

**trajectory**

If the model contains a trajectory, write all frames *that have been* [read in](#) with [MD Movie](#). Otherwise, only the current frame will be written out.

**resnum**

If writing `mol2` [format](#), generate substructure names that include residue numbers (for example, ALA85 rather than ALA).

**atomTypes** `amber` | `gaff` | `sybyl`

For `mol2` [format](#), setting **atomTypes** (or **atomtypes**) to `amber` or `gaff` indicates that the

output file should include Amber/GAFF atom types instead of [Sybyl atom types](#). Amber/GAFF atom types are assigned by [Add Charge](#) or the command [addcharge](#). If any atom being written lacks an Amber/GAFF type assignment, file-writing will halt and an error message will be displayed. \*\* Mol2 files with Amber/GAFF types are not suitable for reading back into Chimera, as the types are likely to be misinterpreted. \*\*

See also: [getcprd](#), [pdbrun](#), [save](#), [export](#), [writesel](#), [Write DMS](#), [Write Prmtop](#), [saving data](#)

## Usage:

**writesel** *sel-file* [options](#)

**Writesel** writes a parsable list of the currently [selected](#) (or unselected) items. It is the command-line implementation of [Actions... Write List](#).

The output file name/pathname (*sel-file*) must be specified before any [options](#). If *sel-file* includes spaces, it must be enclosed in single or double quote marks. If *sel-file* is **browse** or **browser**, a [dialog](#) for saving the file will appear. If *sel-file* is a single dash, the information will be sent to the [Reply Log](#) instead of saved to a file.

## Options

Option keywords for **writesel** can be truncated to unique strings and their case does not matter. A vertical bar "|" designates mutually exclusive options, and default values are indicated with **bold**. Synonyms for true: True, 1. Synonyms for false: False, 0.

**namingStyle** simple|command|serial  
Style to use for specifications:

- **simple** - residue name, residue specifier, and atom name (for example, HIS 16.A ND1)
- **command** or **command-line** - [command-line specifier](#) (for example, :16.A@ND1)
- **serial** or **serialnumber** - atom serial number (for example, 126)

Model number will be included when multiple models are present. If **namingStyle** is not supplied, the **Atomspec display style** specified in the [Labels preferences](#) will be used.

**selected** true|false

Whether to write out specifications of items that are [selected](#); if **false**, descriptions of the unselected items will be written out.

**itemType** atom|bond|pseudobond|**residue**|molecule|model

Whether to write specifications of atoms, bonds, pseudobonds, residues, molecule models, or non-molecule (molecular surface, surface, VRML, and volume) models, respectively.

See also: [Actions... Write List](#), [pdbrun](#), [write](#), [saving PDB files](#)

**Usage:**

**zonesel** [ref-spec](#) *cutoff* [other-spec](#) [ **extend true** | **false** ]

The **zonesel** command [selects](#) atoms and/or [surface pieces](#) in [other-spec](#) that are within a *cutoff* distance of the atoms and/or [surface pieces](#) in [ref-spec](#). Surface models and their pieces can be specified by model number or as a [selection](#) ([details...](#)). The **extend true** setting (default **false**) indicates the items in [ref-spec](#) should also be selected.

See also: [zone specifiers](#), [select](#), [measure distance](#), [Surface Zone](#)

## Unsupported Midas Commands

Midas was Chimera's predecessor program; although Chimera was not built upon Midas, its command set was developed with backwards compatibility in mind. Most Midas commands are now available in Chimera, plus a significant number of commands that are new in Chimera or have been made more powerful. Most of the Midas commands unsupported by Chimera have been rendered obsolete or replaced by other functionality.

- addgrp - see [Build Structure](#)
- assign
- brotation - see [rotation reverse](#)
- bs - see [represent bs](#)
- colorrename
- delegate, pickatom, pickabort

Midas delegates included:

- density - see [Volume Viewer](#)
  - discern
  - gd
  - label3d
  - midasmovie - see [MD Movie](#)
  - mrotate - see [reset](#)
- devopt
- fix - see [Adjust Torsions](#)
- fixreverse - see [Adjust Torsions](#)
- intensity - see **yon intensity** in the [Effects](#) tool
- link - see [bond](#)
- makems - see [surface](#)
- mark, makemark - see [alias](#) and [naming selections](#)
- midaspush, midaspop
- noeshow
- pdb2site
- pdbopen
- record - see **Record** in the [Command History](#) tool
- redraw
- reverse - see [Adjust Torsions](#)
- ribbonjr - see [ribbon](#) and [Nucleotides](#)
- run - see [pdbrun](#), [system](#)
- setcom - but [rock](#), [roll](#), [turn](#) have been enhanced for much easier control over center of rotation
- speed
- stereoimg - see [saving stereo images](#)
- update
- vdwopt - see [vdwdefine](#) and [vdwdensity](#)
- watch, watchopt - see [findclash](#)

## Other Unsupported Midas Features

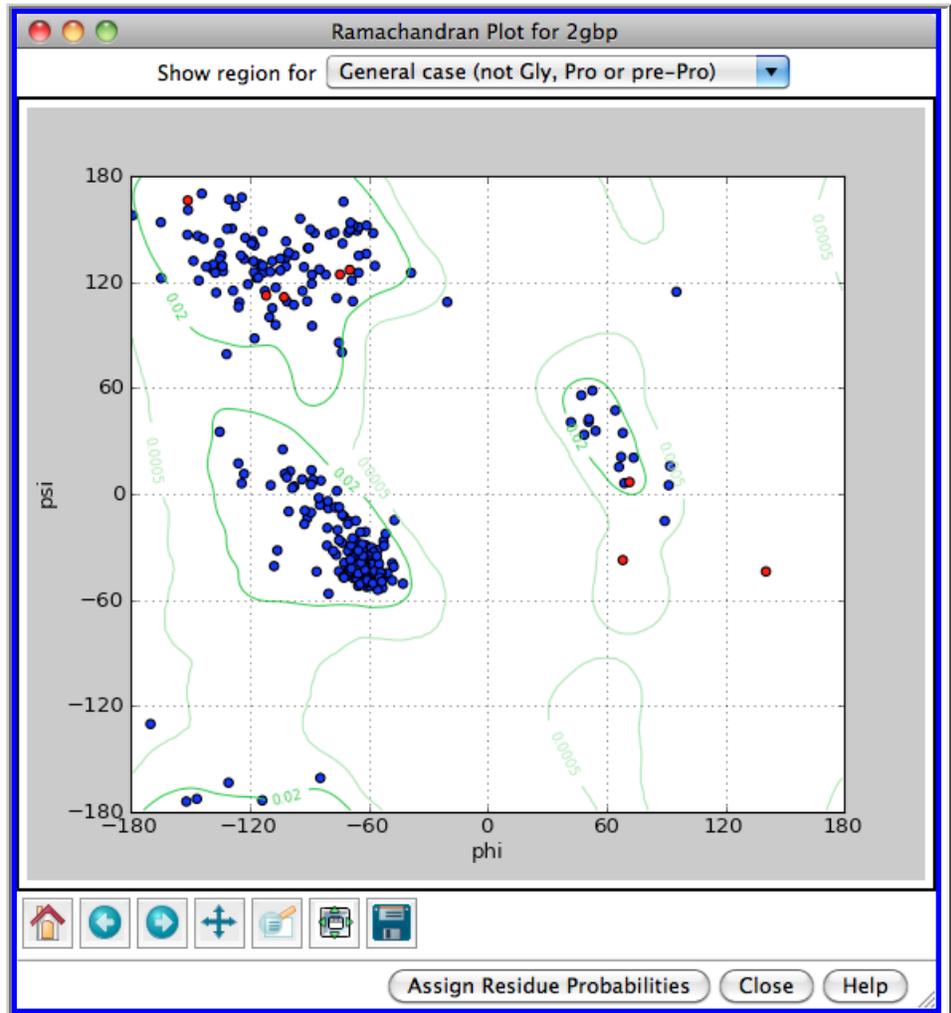
- **some Ctrl-key command-line editing options** - in Chimera, **Ctrl-u** erases the command line contents, **Ctrl-p** switches to the previous command, and **Ctrl-n** switches to the next command, but further [text-editing shortcuts](#) are system-dependent
- **Midas-style objects** - however, the Chimera [BILD format](#) is similar and more extensive: planes and solid shapes can be defined in addition to points and lines

## Ramachandran Plot

The [Ramachandran plot](#) function in the [Model Panel](#) plots the distribution of amino acid backbone conformations in peptide and protein structures. It is also implemented as the command [ramachandran](#). See also: [RR Distance Maps](#), [Rotamers](#), [Structure Measurements](#), [ksdssp](#)

Each amino acid residue is shown as a dot in a graph of  $\phi$  vs.  $\psi$ , more commonly known as a Ramachandran plot or Ramachandran map. Residues are shown as blue dots, or when [selected](#), as red dots. Conversely, clicking a single dot on the plot will [select](#) the corresponding residue in the structure.

Probability contours based on a reference set of high-resolution proteins can be shown on the plot as green lines. The reference set and resulting  $\phi, \psi$  distributions are described in:



### [Structure validation by C \$\alpha\$](#)

[geometry:  \$\phi, \psi\$  and C \$\beta\$  deviation.](#) Lovell SC, Davis IW, Arendall WB 3rd, de Bakker PI, Word JM, Prisant MG, Richardson JS, Richardson DC. *Proteins*. 2003 Feb 15;50(3):437-50

Probability contours can be displayed for different subsets of the amino acid residues in the reference proteins, as indicated next to **Show region for**:

- **None** - no probability contours
- **Alanine (no repet sec struct)** - alanine residues not in helix or sheet
- **General case (no repet sec struct)** - residues not in helix or sheet
- **General case (not Gly, Pro or pre-Pro)** - residues except for glycine, proline, and those immediately preceding prolines in sequence
- **Glycine (sym)** - glycine residues, probabilities symmetrized
- **Glycine (sym, no repet sec struct)** - glycine residues not in helix or sheet, probabilities symmetrized
- **Proline** - proline residues
- **pre-Proline (not Gly or Pro)** - residues immediately preceding prolines in sequence, except for glycine and proline

When the plot has mouse focus, the X and Y coordinates (in this case,  $\phi$  and  $\psi$  values) of the cursor location on the graph are reported on the lower right. Below the plot are standard [navigation icons](#) provided by

[matplotlib](#).

**Assign Residue Probabilities** assigns an [attribute](#) named **ramaProb** to the amino acid residues, with values taken from the appropriate [dataset](#) for each residue (**Proline** for prolines, *etc.*), and opens the [Render/Select by Attribute](#) tool.

**Close** removes the Ramachandran plot. **Help** opens this manual page in a browser window.

Whether or not a Ramachandran plot is shown, **phi** ( $\varphi$ ) and **psi** ( $\psi$ ) are automatically assigned as residue [attributes](#), and as such can be viewed/changed in the [Selection Inspector](#) or changed with [setattr](#).

---

UCSF Computer Graphics Laboratory / September 2014

```
#  
# "color" affects atoms, ribbons, surfaces, and atom and residue labels  
#  
color dark gray #  
color orange red :asp  
color red :glu  
color hot pink :asn  
color deep pink :gln  
color dodger blue :lys  
color blue :arg  
color deep sky blue :his  
color light green :gly  
color light green :pro  
color pink :ser  
color plum :thr  
color gold :cys  
color goldenrod :met  
color goldenrod :mse  
color dark khaki :ala  
color dark khaki :val  
color tan :ile  
color tan :leu  
color sandy brown :phe  
color sandy brown :trp  
color rosy brown :tyr
```

```
open receptor.pdb
open GCP.pdb
preset apply interactive 1
color aquamarine #1
disp #1 & #0 z<5
color orange,a #1 @o=
color medium blue,a #1 @n=
color magenta #2
repr bs #2
```

```
#  
# commands for Chimera image tutorial: Similar Binding Sites  
# (additional stuff to be done interactively: defining ribbon style,  
# generating positions, adding 2D labels, saving images)  
#  
window size 831 544  
open 1exp  
open 1cel  
delete :.b  
preset apply int 1  
~disp  
color gold #0  
color cyan #1  
background solid white  
alias site1 #0:84,126,233,171,205  
alias site2 #1:367,141,217,145,228  
alias both site1 | site2  
disp both  
match iterate 2.0 site2 site1  
focus both  
color byhet  
set subdivision 10  
transparency 75,r  
set flatTransparency  
repr wire @ca
```

# UCSF CHIMERA

## an Extensible Molecular Modeling System

### Web Services Used by UCSF Chimera

Web services accessed by Chimera fall into two general categories:

- [Data Retrieval Services](#)
- [Computational Services](#)

See also: [Sites that provide Chimera web data](#)

#### • Data Retrieval Web Services Used by Chimera

##### [ASTRAL Compendium](#)

Allows [fetching](#) domain structures specified by [SCOP domain identifier](#).

##### [Computed Atlas of Surface Topography of proteins](#) (CASTp)

Allows [fetching](#) precomputed [pocket measurements](#) and corresponding structures specified by PDB ID.

##### [Electron Density Server](#) (EDS)

Allows [fetching](#) electron density maps (2fo-fc and fo-fc) specified by PDB ID.

##### [Electron Microscopy Data Bank](#) (EMDB)

Allows [fetching](#) electron density maps by EMDB identifier and finding maps by keyword search.

##### [ModBase](#)

Allows [fetching](#) *modeled* structures ([comparative models](#)) specified by SwissProt, TrEMBL, GenPept or PIR accession code.

##### [Nucleic Acid Database](#) (NDB)

Translates NDB ID codes into PDB ID codes, which are then used to [fetch](#) structures from the [PDB](#).

##### [Protein Data Bank](#) (RCSB PDB)

Allows [fetching](#) structures in PDB and mmCIF formats by PDB ID. Also provides lookup of other PDB entry information by [Blast Protein](#) and [PDB/UniProt Info](#).

##### [Protein Quaternary Structure](#) (POS)

Allows [fetching](#) *predicted* biological units specified by PDB ID.

##### **Pub3D**

Allows [fetching](#) *modeled* organic molecule structures specified by [PubChem](#) CID from the **Pub3D** database. This database contains structures for millions of PubChem

entries and is described in [Willighagen \*et al.\*, BMC Bioinformatics 8:487 \(2007\)](#). **Pub3D** and the [service](#) for accessing it are provided by the [Chemical Informatics and Cyberinfrastructure Collaboratory \(CICC\)](#) at Indiana University.

### **[UniProt](#)**

Allows [fetching](#) protein sequences and their feature annotations by UniProt ID, either for PDB structure chains by [PDB/UniProt Info](#), or independent of structure.

### **[Virus Particle Explorer database](#)**

Allows [fetching](#) icosahedral virus capsid structures (transformed into a standard orientation) specified by PDB ID.

## • **Computational Web Services Used by Chimera**

### **APBS**

Chimera provides an [interface](#) to calculating electrostatic potential maps using an [APBS](#) (Adaptive Poisson-Boltzmann Solver) web service provided by the [National Biomedical Computation Resource \(NBCR\)](#).

### **AutoDock Vina**

Chimera provides an [interface](#) to single-ligand docking calculations using an [AutoDock Vina](#) web service provided by the [NBCR](#).

### **BLAST Protein**

The [Blast Protein](#) tool performs protein sequence searches using a [BLAST](#) web service hosted by the [UCSF Resource for Biocomputing, Visualization, and Informatics \(RBVI\)](#).

### **Clustal Omega**

Multiple sequence alignment with [Clustal Omega](#) can be performed using a web service hosted by the [UCSF RBVI](#). The sequences of structure chains in Chimera can be submitted via the [Align Chain Sequences](#) tool, and/or sequences already in an alignment shown in [Multalign Viewer](#) can be submitted for realignment.

### **Modeller**

Chimera provides an [interface](#) for comparative (homology) modeling, loop remodeling, and building missing segments using [Modeller](#), either run locally or using a web service hosted by the [UCSF RBVI](#).

### **MultiFit**

The [MultiFit](#) tool performs simultaneous rigid fitting of multiple atomic-resolution structures into density maps using the program [MultiFit](#) (courtesy of Keren Lasker, [Sali group](#)) via a web service hosted by the [UCSF RBVI](#).

### **MUSCLE**

Multiple sequence alignment with [MUSCLE](#) can be performed using a web service hosted by the [UCSF RBVI](#). The sequences of structure chains in Chimera can be submitted via the [Align Chain Sequences](#) tool, and/or sequences already in an alignment shown in [Multalign Viewer](#) can be submitted for realignment.

## PDB2PQR

Chimera provides an [interface](#) to structure cleanup, parameter assignment, and [PQR](#) file generation using a [PDB2PQR](#) web service provided by the [NBCR](#).

## [SaliLab Model Evaluation Server](#)

Can be called to calculate additional scores for comparative models using the **Fetch Scores** menu in the [Model List](#) dialog.

## Small-Angle X-Ray Profile

The [Small-Angle X-Ray Profile](#) tool calculates small-angle X-ray scattering (SAXS) profiles from structures using the program [FoXS](#) (courtesy of Dina Schneidman, [Sali group](#)) via a web service hosted by the [UCSF RBVI](#).

## smi23d

Generates *modeled* organic molecule structures specified by [SMILES](#) string with the command [open](#) or in [Build Structure](#). The [smi23d web service](#) is provided by the [Chemical Informatics and Cyberinfrastructure Collaboratory \(CICC\)](#) at Indiana University and deploys the same procedure used to populate **Pub3D** ([above](#)).

## [StrucTools](#)

Used by [Area/Volume from Web](#) to calculate atomic surface areas (various methods) and Voronoi volumes from coordinates.

## 2D Chemical Diagram

Used by [Structure Diagram](#) and [ViewDock](#) to draw 2D diagrams of small organic molecules given coordinates or a [SMILES](#) string. This web service is implemented in Java using the [Chemistry Development Kit](#) and is provided by the [UCSF RBVI](#).

# Sites That Provide Chimera Web Data

Accessing [Chimera web data](#) (such as by clicking a link or choosing a display option at one of these sites) automatically displays database contents or server results in a locally installed copy of Chimera. See also: [Web services used by Chimera](#)

## [ConSurf Server](#)

Clicking the link to view results in Chimera displays the query structure and the multiple sequence alignment from ConSurf, both colored by the server's conservation measure. The ConSurf-calculated phylogenetic tree and custom headers are shown along with the sequence alignment (see [screenshot and details](#)).

## [ModBase](#)

The database contains predicted protein structures from comparative (homology) modeling. The "Chimera (Structure/Alignment)" option displays the modeled structure along with the template structure and the sequence alignment between the two. The "Visualize Surface Cavities (Chimera)" option shows a molecular surface of the modeled structure colored by [ConCavity](#) binding site prediction score.

## [MultiFit Webserver](#)

The server performs simultaneous rigid fitting of multiple structures into a density

map; clicking a thumbnail image in the results page opens the corresponding fitting solution in Chimera.

### [PhosphoSitePlus® \(PSP\)](#)

Clicking “Get ChimeraX Script” on the Structure Viewer will show the structure in Chimera with experimentally observed sites of post-translational modification (PTM) colored and labeled. PSP is continuously updated with data on phosphorylation and other commonly studied PTMs.

### [SALIGN Webservice](#)

The SALIGN server for multiple protein sequence/structure alignment can launch Chimera to display the superimposed structures and associated multiple sequence alignment.

### [Structure-Function Linkage Database](#)

Chimera web data files are provided for viewing enzyme structures, active sites (Chimera sessions), and sequence alignments.

### [SwissDock Server](#)

The SwissDock server allows launching Chimera (and its [ViewDock](#) tool) to display the results; the site includes a short example video.

### [UCSC Genome Browser](#)

Chimera links are provided in UCSC Gene pages (see [example](#)) and SNP track details pages. Clicking a Chimera link displays the corresponding structure with nonsynonymous SNP residues colored and labeled with the dbSNP identifier.

# UCSF CHIMERA

---

## an Extensible Molecular Modeling System

### Licensing

UCSF Chimera is licensed for noncommercial use at no cost by the Regents of the University of California. You must agree to the terms and conditions of use as specified in the license agreement before you will be permitted to download Chimera ([preview the agreement](#)).

Chimera extension developers please note: The purpose of the **Ownership and Assignment of Copyright** section of the license agreement is to protect UCSF's copyright in the existing code and derived works based on our code. It does not cover original code developed by others. Extensions to Chimera that are not modifications, improvements, or derivatives of UCSF copyrighted code (programming examples we provide are explicitly not copyrighted) are considered the property of their authors.

Commercial use of Chimera is covered by a separate, written license agreement. Commercial licensing costs are tier-based, depending on the number of users. Please [contact us](#) if you are interested in using Chimera for commercial purposes.

Chimera software and documentation is copyrighted by the Regents of the University of California. All rights reserved. Chimera is provided pursuant to a license agreement containing restrictions on its disclosure, duplication and use.

# UCSF CHIMERA

## an Extensible Molecular Modeling System

### Citing UCSF Chimera

Chimera development by the UCSF Resource for Biocomputing, Visualization, and Informatics is funded by the [National Institutes of Health](#). NIH tracks publications that make use of our Resource and its software, and hence your cooperation is appreciated in citing the grant number and reference(s) as described below. Thank you!

*Publications include scientific papers, posters, films, videos, exhibits, and artwork.* Publications with images or results from Chimera should include an acknowledgement similar to the following:

Molecular graphics and analyses were performed with the UCSF Chimera package. Chimera is developed by the Resource for Biocomputing, Visualization, and Informatics at the University of California, San Francisco (supported by NIGMS P41-GM103311).

and should cite one or more of the [Chimera references](#), such as:

[UCSF Chimera--a visualization system for exploratory research and analysis.](#)

Pettersen EF, Goddard TD, Huang CC, Couch GS, Greenblatt DM, Meng EC, Ferrin TE. *J Comput Chem.* 2004 Oct;25(13):1605-12.

The Chimera home page <http://www.cgl.ucsf.edu/chimera> can also be cited. In addition:

- Solvent-excluded [molecular surfaces](#) are created with the help of the [MSMS package](#):

[Reduced surface: an efficient way to compute molecular surfaces.](#) Sanner MF, Olson AJ, Spehner JC. *Biopolymers.* 1996 Mar;38(3):305-20.

- Raytraced images are produced with [POV-Ray](#): [\[citation information\]](#)
- Several Chimera tools use published methods or software, and their manual pages provide the appropriate citation information

For permission to use images from the Chimera web site, please contact [chimera@cgl.ucsf.edu](mailto:chimera@cgl.ucsf.edu).

### Authors



Chimera Dev Team circa February 2010: (left to right), Conrad Huang, Zheng Yang (“YZ”), Greg Couch, Elaine Meng, Eric Pettersen, Tom Goddard

Chimera is the work of many individuals:

- [Thomas Ferrin](#) - Principal investigator
- Conrad Huang - Project leader
- Greg Couch - OpenGL graphics, molecule rendering, [raytracing](#), ...
- Eric Pettersen - Structure analysis and [superposition](#), [sequence viewer](#), ...
- Tom Goddard - [Volume data](#) display and analysis, molecular assemblies, ...
- Elaine Meng - [User's Guide](#), [tutorials](#), user training, web pages
- Zheng Yang - [Modeller interface](#)
- Sam Hertig - [MultiDomain Assembler](#)
- Darren Weber, David Mischel - [Animation tool](#)
- Victor Muñoz Robles, Jean-Didier Maréchal (The Computational Biotechnological Chemistry Team, Universitat Autònoma de Barcelona; funding from Ministerio de Ciencia e Innovación, Generalitat de Catalunya) - normal mode analysis, molecular dynamics simulation
- Daniel Greenblatt - Collaboratory, [movie recorder](#), bug reporter
- David Konerding - Collaboratory, early version of [trajectory viewer](#)

- Greg Pintilie - [Segger](#) map segmentation tools
- Wei Zhang - [Counterion](#) and [solvent](#) addition, [morph map](#)
- Therese Lang - [Dock Prep](#) ideas
- Sam Schreiber - [Intersurf](#), ...
- Jonathan E. Chen - [Protein contact maps](#)
- Ben Morris - [Raytracing](#)
- Thomas Margraf - [Space Navigator](#) support on Linux

... and many more on related projects (see also [Chimera plug-ins on the web](#) and [sites with Chimera web data](#)):

- Scooter Morris - Cytoscape plugin [structureViz](#)
- Nadezhda Doncheva - Cytoscape plugin [RINalyzer](#)
- [Structure-Function Linkage Database \(SFLD\) development team](#) - Chimera web data and sessions in the [SFLD](#)
- Ursula Pieper - Chimera web data in [ModBase](#)
- Elana Erez, Fabian Glaser - Chimera web data produced by the [ConSurf Server](#)
- Forbes Burkowski - Structural bioinformatics [course materials](#) and textbook: [Computational and Visualization Techniques for Structural Bioinformatics Using Chimera](#) (2014)
- Andrew Jewett - [MinRMS](#)
- Cathy Lawson - [Protein Data Bank](#) curation
- Suzuki Hirofumi - [EM Navigator](#) database images
- Richard Newman - [Electron Microscopy Data Bank images](#)
- Ardan Patwardhan - [Electron Microscopy Data Bank \(EMDB\)](#)
- Gabe Lander, Padma Natarajan - [VIPERdb](#) image creation scripts
- Heidi Houtkooper - [Web page](#) development
- Teri Klein - Scientific collaborations
- Kris Casler, Frank Federico, Robin Parsons, Willa Crowell, Norma Belfer - Administrative support, event planning

... and undoubtedly others.

## Other Software and Databases

Chimera incorporates [many publicly available software packages](#) and accesses [several web services](#).

# UCSF CHIMERA

## an Extensible Molecular Modeling System

### UCSF Chimera References

Primary reference:

[UCSF Chimera--a visualization system for exploratory research and analysis.](#)

Pettersen EF, Goddard TD, Huang CC, Couch GS, Greenblatt DM, Meng EC, Ferrin TE. *J Comput Chem.* 2004 Oct;25(13):1605-12.

Additional papers on features and methodology:

[Enhancing UCSF Chimera through web services.](#) Huang CC, Meng EC, Morris

JH, Pettersen EF, Ferrin TE. *Nucleic Acids Res.* 2014 Jul;42(Web Server issue): W478-84.

[UCSF Chimera, MODELLER, and IMP: An integrated modeling system.](#) Yang Z,

Lasker K, Schneidman-Duhovny D, Webb B, Huang CC, Pettersen EF, Goddard TD, Meng EC, Sali A, Ferrin TE. *J Struct Biol.* 2012 Sep;179(3):269-78.

[Quantitative analysis of cryo-EM density map segmentation by watershed and scale-space filtering, and fitting of structures by alignment to regions.](#)

Pintilie GD, Zhang J, Goddard TD, Chiu W, Gossard DC. *J Struct Biol.* 2010 Jun;170(3):427-38.

[structureViz: linking Cytoscape and UCSF Chimera.](#) Morris JH, Huang CC,

Babbitt PC, Ferrin TE. *Bioinformatics.* 2007 Sep 1;23(17):2345-7.

[Visualizing density maps with UCSF Chimera.](#) Goddard TD, Huang CC, Ferrin

TE. *J Struct Biol.* 2007 Jan;157(1):281-7.

[Tools for integrated sequence-structure analysis with UCSF Chimera.](#) Meng

EC, Pettersen EF, Couch GS, Huang CC, Ferrin TE. *BMC Bioinformatics.* 2006 Jul 12;7:339. PMID: 16836757

[Nucleic acid visualization with UCSF Chimera.](#) Couch GS, Hendrix DK, Ferrin

TE. *Nucleic Acids Res.* 2006 Feb 14;34(4):e29.

[Software extensions to UCSF chimera for interactive visualization of large molecular assemblies.](#) Goddard TD, Huang CC, Ferrin TE. *Structure.* 2005

Mar;13(3):473-82.

# UCSF CHIMERA

---

## an Extensible Molecular Modeling System

### Contact Us

Image permission requests and inquiries into commercial [licensing](#) should be sent to [chimera@cgl.ucsf.edu](mailto:chimera@cgl.ucsf.edu).

We welcome feedback about UCSF Chimera:

#### To ask a question about using Chimera, discuss features, or suggest improvements:

You can send mail to the Chimera users mailing list, [chimera-users@cgl.ucsf.edu](mailto:chimera-users@cgl.ucsf.edu). This list is monitored closely by the Chimera developers, as well as other list subscribers, and responses are usually quick and thorough.

You can [subscribe](#) to the chimera-users mailing list, [view](#) the whole archive, or search the archive ([search syntax](#)):

#### To report a problem:

If you can, please use the **Report a Bug** dialog found in the **Help** menu. Or you can send mail to [chimera-bugs@cgl.ucsf.edu](mailto:chimera-bugs@cgl.ucsf.edu) or use our [online submission form](#).

After being processed, your report will appear in our [bug tracking system](#). If you supply your email address, you will be notified when your bug is fixed.

#### To ask a question about writing your own Chimera extensions:

You can send mail to the Chimera development mailing list, [chimera-dev@cgl.ucsf.edu](mailto:chimera-dev@cgl.ucsf.edu).

While this list is not as lively as the chimera-users mailing list, it is actively monitored by the Chimera developers, and can be a very useful resource for getting sample code and other helpful pointers for writing your own extensions in Chimera.

You can [subscribe](#) to the chimera-dev mailing list, [view](#) the whole archive, or search the archive ([search syntax](#)):

### **Chimera announcements list:**

The Chimera announcements list is used by the Chimera team to inform users of releases and upcoming events such as workshops (~2-3 messages/year).

Although you cannot send messages to the chimera-announce mailing list, you can [subscribe](#) to it.