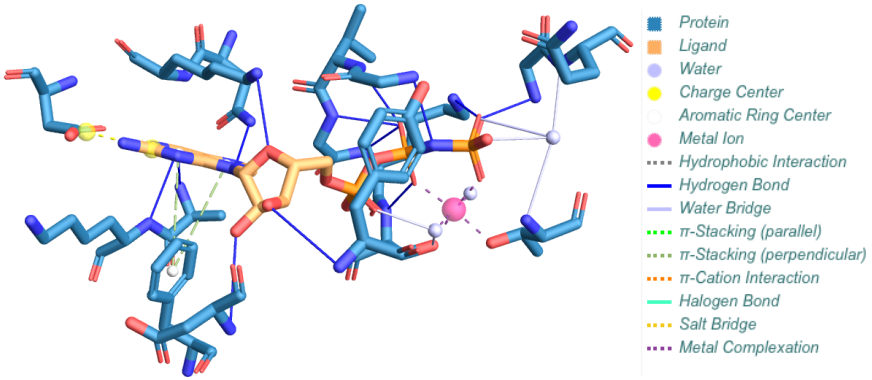
PyMOL: Advanced

Misc. useful stuff, Using PyMOL scripts, Wizards and Plug-Ins
Making Movies

http://www.pymolwiki.org/index.php/MovieSchool_1

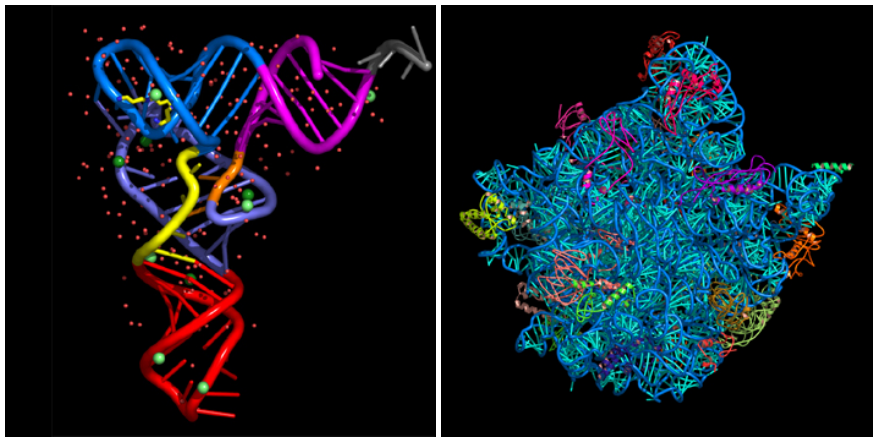
Protein-Ligand Interaction Profiler PLIP

A web server analyzing the interaction between proteins and ligands that uses PyMOL to generate figures and allows to download the results as a PyMol .pse file
https://projects.biotec.tu-dresden.de/plip-web/plip/index



Salentin S, Schreiber S, Haupt VJ, Adasme MF, Schroeder M.
PLIP : fully automated protein-ligand interaction profiler.
Nucleic Acids Res. 2015 Jul 1;43(W1):W443-7. doi: 10.1093/nar/gkv315. Epub 2015 Apr 14.

Program NUCCYL allows to generate highly simplified Nucleic Acid Cartoon Representations in PyMOL



http://www.biosci.ki.se/groups/ljo/software/nuccyl.html

Pymol scripts (*.pml)

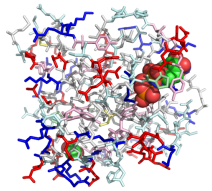
either:
copy-and-paste script into the command line

or:
run the script with the command
@script.pml

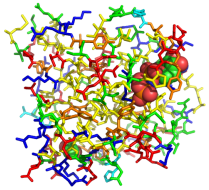
@pml_scripts/AHo_BindingPocket.pml

Running a simple script downloaded from the web

http://pymolwiki.org/index.php/Category:Script_Library
https://github.com/Pymol-Scripts/Pymol-script-repo
http://pldserver1.biochem.queensu.ca/~rlc/work/pymol/



run py_scripts/color_by_restype.py
color_by_restype 3K8Y



run py_scripts/AHo_color_by_restype.py
color_by_restype 3K8Y

@AAcolor.pml

If you put the "run" commands into a file called "pymolrc" (visible) or ".pymolrc" (invisible), they will be executed automatically when PyMOL is started, and you will have the commands available in every PyMOL session

Finding a specified Subsequence in a Protein

Using selection operators to find the subsequence "GAD":
select DDrF0016 and pepseq GAD
also finds "G", "GA", "AD" and "D", but not "A"

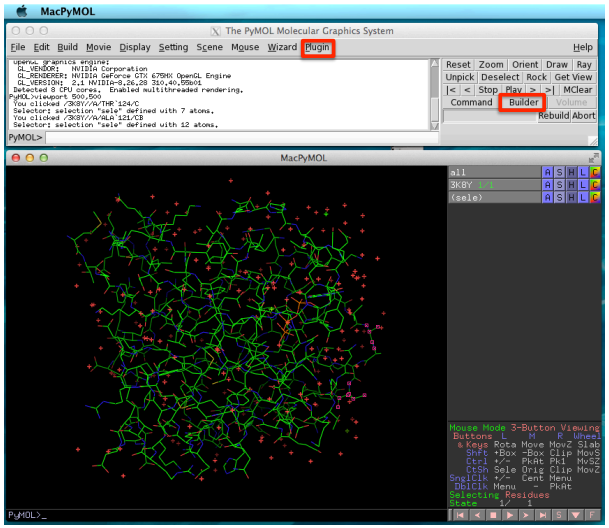
Using a python script:
run ~/pymol/py_scripts/findseq.py
findseq GAD, DDrF0016
appears to work fine on a single object, but does not work properly if the selection contains multiple objects

The findseq script interprets **regular expressions***

*: *http://en.wikipedia.org/wiki/Regular_expression*

Mac: Using the Hybrid Version of PyMOL

The Hybrid mode and X-Windows (Darwin) mode of MacPyMOL offer some advanced options not available in the desktop mode: e.g.
Builder,
Plugin support



Needs X11:https://www.xquartz.org

PlugIns

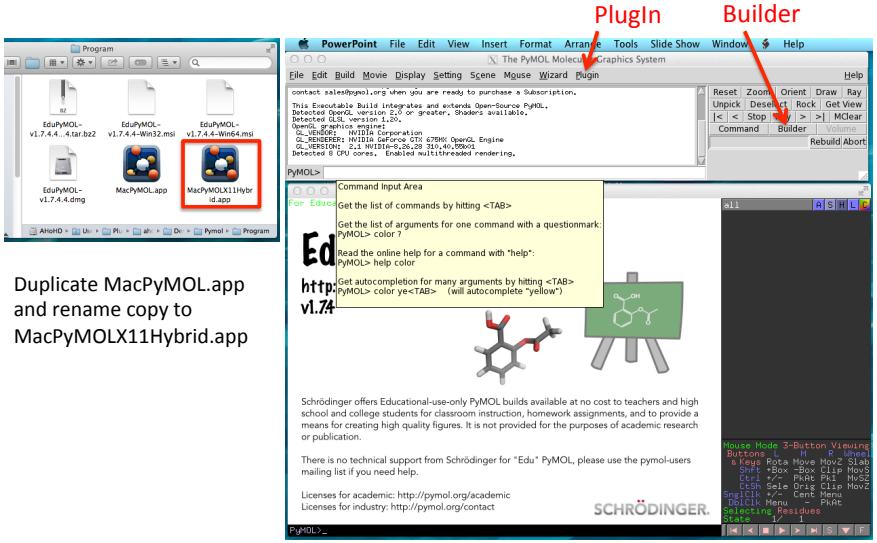
MacOSX: Duplicate MacPyMOL.app, rename to PyMOLX11Hybrid.app
The standard OS X Pymol application, MacPyMOL.app does not run with the Tcl/Tk interface which is required for plugins to work. However, a quick renaming of the program from MacPyMOL.app to PyMOLX11Hybrid.app makes the application run as an X11 application, and plugins are now available.

To rename the executable, right click (or control click) on MacPyMOL and choose "Get Info" in the Panel. Change the Name & Extension to PyMOLX11Hybrid.app. This name can also be changed using the mv command in Terminal.app.

Once this change is made, half of the program will show up under the X11 icon, and half will show up under the MacPyMOL icon.

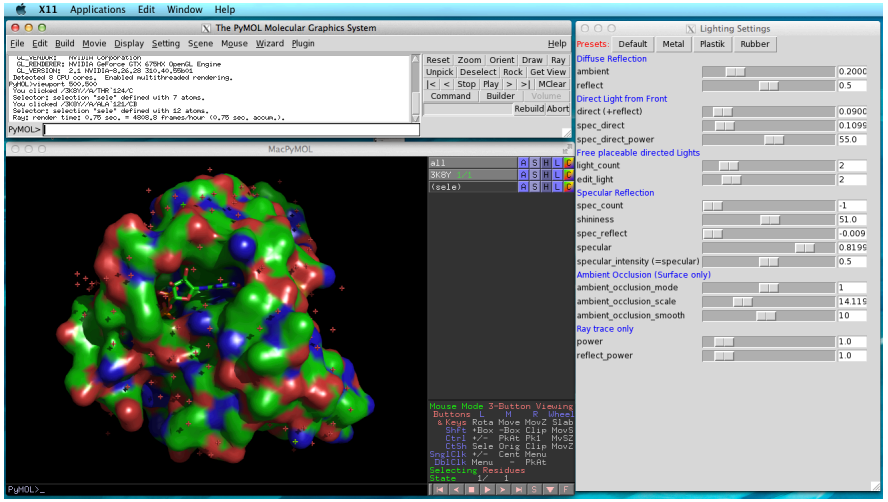
http://pymolwiki.org/index.php/Category:Plugins

Mac: Using the Hybrid Version of PyMOL



Duplicate MacPyMOL.app and rename copy to MacPyMOLX11Hybrid.app

Plugin: Lighting Settings

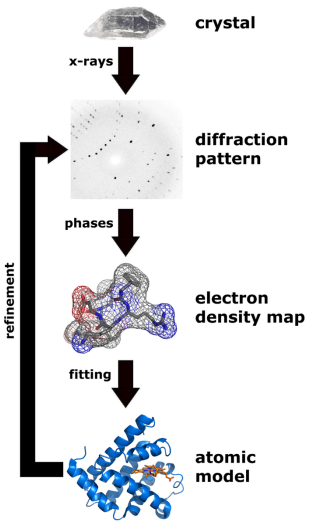


Interactive control over illumination and reflectivity settings

Looking at Electron Density

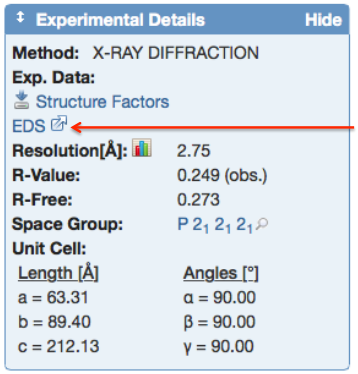
Using the Density Wizard

X-Ray Diffraction: Display Electron Density



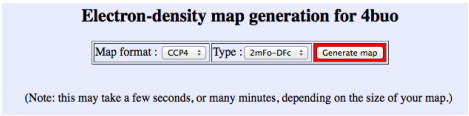
Download Electron Density Map

Find your protein of interest on http://www,rcsb.org



Follow link to Electron Density Server

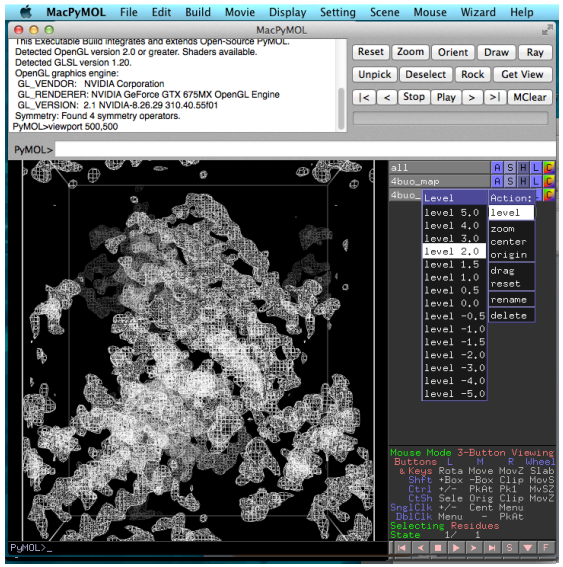
Download: Maps



Download maps in CCP4-format for use in PyMOL

For 4BUO.pdb (rat NTR1) download files 4buo.ccp4 and 4buo_diff.ccp4
Rename 4buo.ccp4 to 4buo_map.ccp4

Open Map-File with PyMOL



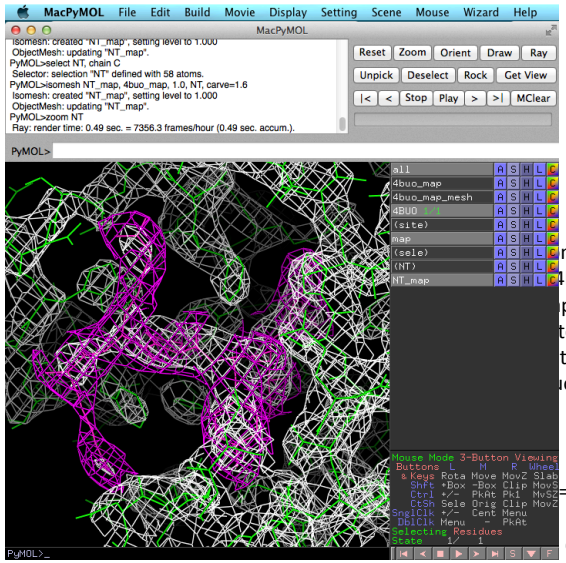
Click on 4buo_map
=> an empty cube is displayed

Open file 4buo_map.ccp4
=> nothing visible on screen

select Action:Mesh:@level 3.0
for object 4buo_map
⇒ a new item called
4buo_map.mesh appears

To change the map level,
select Action:Mesh:@level 2.0
for object 4buo_map.mesh

Add the Coordinate File



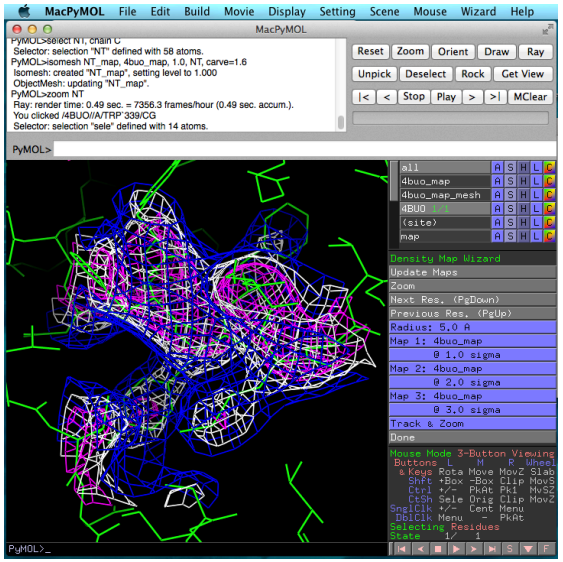
open file 4BUO.pdb

mmand line:
4buo
p, 4buo_map, 2.0, site, carve=1.6
tes a map that just covers the
tes
uo_map_mesh to see)

=1.6

different color to the local map)

Density Wizard

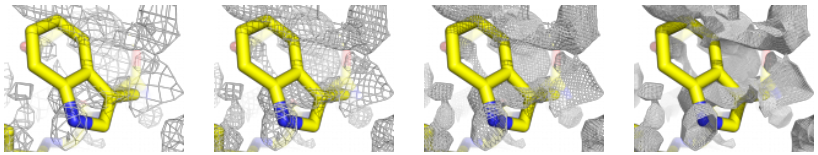


With the density wizard,
you can move through the
maps residue by residue,
using the <pg up> and
<pg down> keys

For a publication quality figure the following are suggestions:

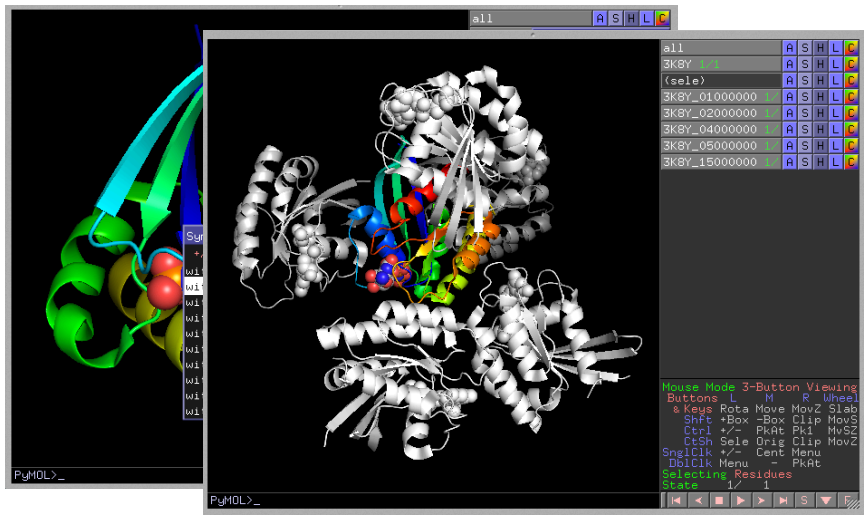
```
color grey50, map          # sets map to 50% gray
set mesh_width, 0.5        # makes meshes thinner for ray-tracing
bg_color white             #sets background to white
set ray_trace_fog, 0       # turns off raytrace fog--optional
set depth_cue, 0           # turns off depth cueing--optional
set ray_shadows, off       # turns off ray-tracing shadows
ray 1024,1024 # this would create a 1024x1024 pixel ray-traced image
png image.png              # output final image
```

map_double resamples a map at twice the current resolution. The amount of memory required to store the map will increase eight-fold.



Std. map mesh spacing | Map doubled | Map double, doubled | Map double, double, doubled

Creating symmetry related molecules

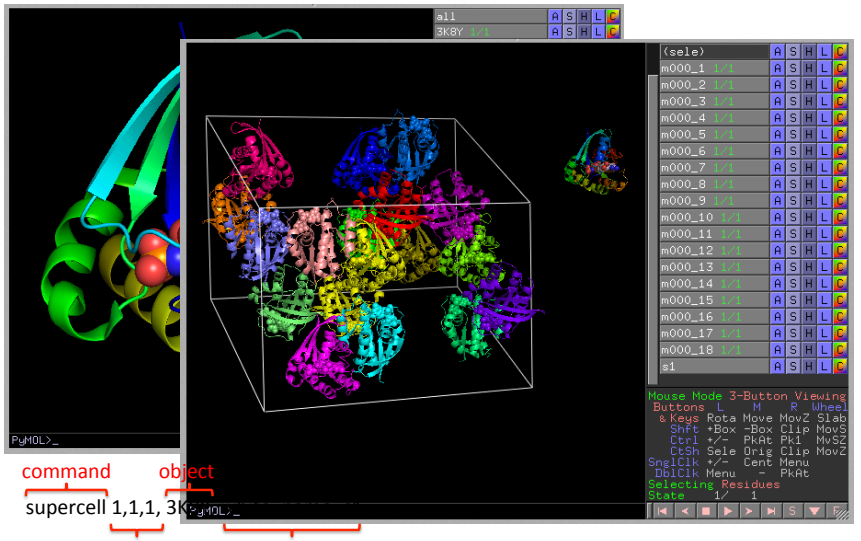


symexp prefix, object, selection, cutoff [, segi]

symexp 3K8Y_, 3K8Y, 3K8Y, 5.0

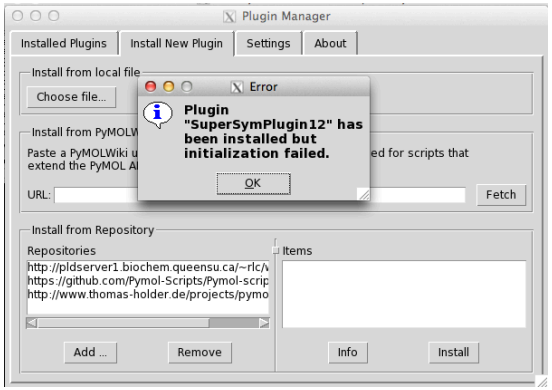
http://pymolwiki.org/index.php/Symexp

Python Script: Supercell



http://pymolwiki.org/index.php/Supercell

PlugIn: SuperSym



This plugin requires cctbx and numeric python (numpy).

Computational Crystallography Toolbox (CCTBX)
http://cctbx.sourceforge.net

Looking at Electrostatic Potentials

Using the ABPS PlugIn

Adaptive Poisson-Boltzmann Solver (APBS) PlugIn

Adaptive Poisson-Boltzmann Solver (APBS) is a software package for modeling biomolecular solvation by solving the Poisson-Boltzmann equation (PBE). The PBE is a popular continuum model used to describe electrostatic interactions between solutes in salty, aqueous media. http://www.poissonboltzmann.org. **Work through the explanations, examples and tutorials on this site to understand what you are doing!**

PDB2PQR: an automated pipeline for the setup of Poisson–Boltzmann electrostatics calculations http://nar.oxfordjournals.org/content/32/suppl_2/W665.full

Adding a limited number of missing heavy atoms to biomolecular structures
Determining side-chain pKas
Placing missing hydrogens
Optimizing the protein for favorable hydrogen bonding
Assigning charge and radius parameters from a variety of force fields

PDB2PQR Server: http://nbcr-222.ucsd.edu/pdb2pqr_2.0.0/

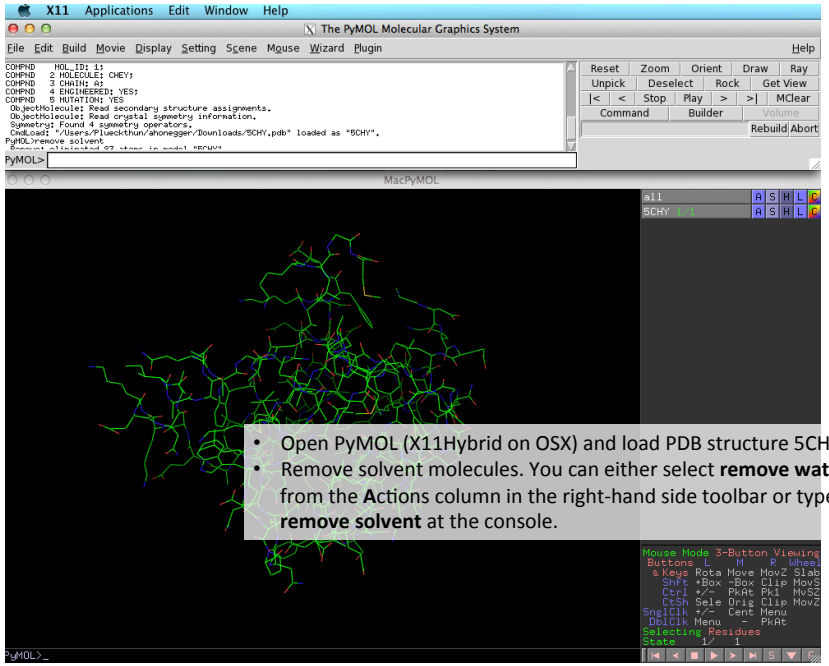
Uses **PROPKA** (or PDN2PKA) to determine the titration state of proteins at specific pH.
Uses **APBS web solver** to solve the Poisson-Boltzmann and related equations to calculate solvation energies and electrostatic properties for analysis and visualization.

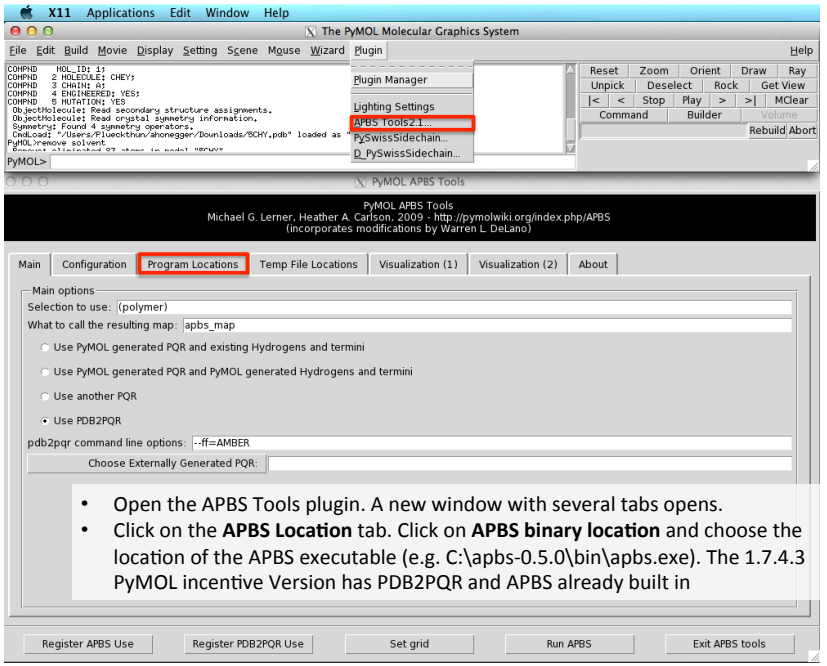
PDB2PQR and APBS are part of the SBgrid package, set APBS-Plugin paths to the appropriate binaries

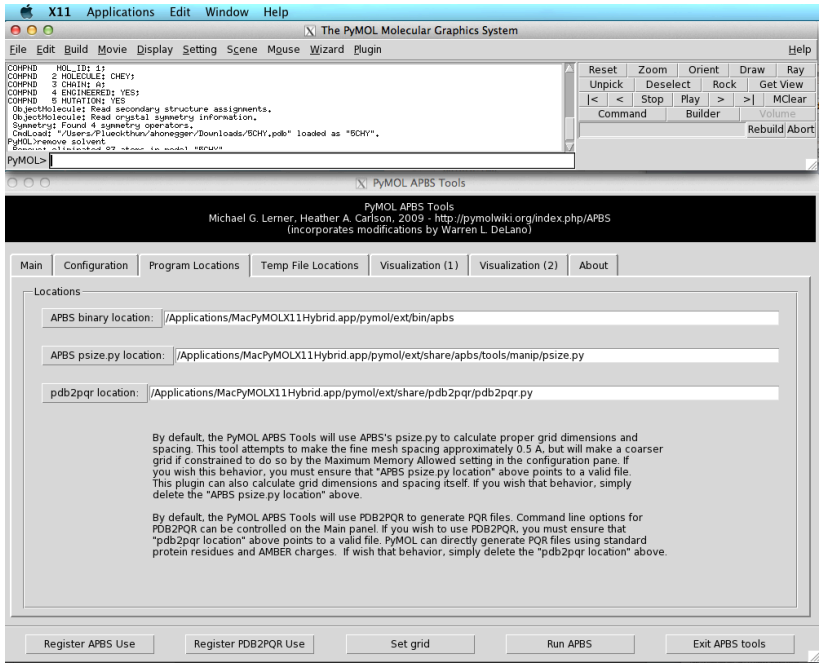
Getting started

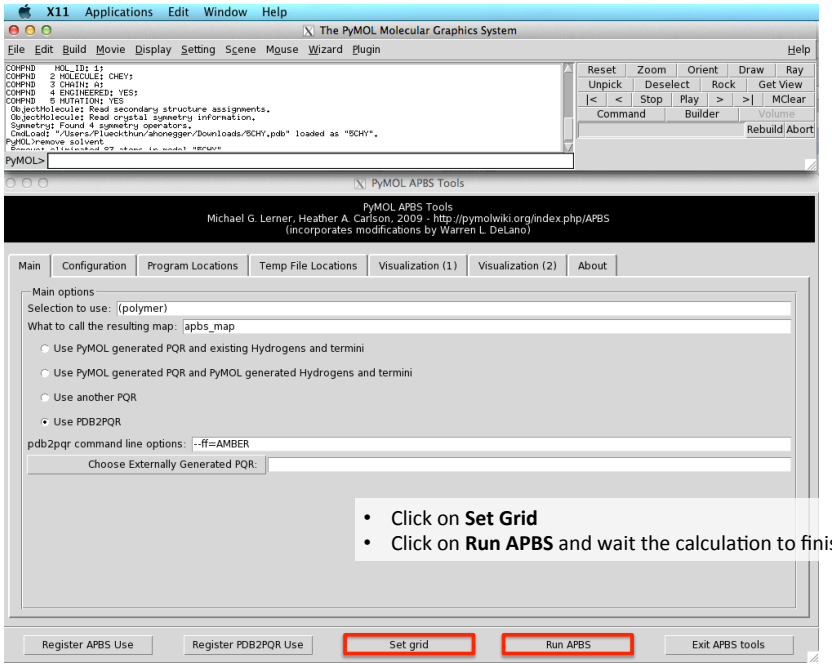
1. Open PyMOL (X11Hybrid on OSX) and load PDB structure 5CHY
2. Remove solvent molecules. You can either select **remove waters** from the **A**ctions column in the right-hand side toolbar or type **remove solvent** at the console.
3. Open the APBS Tools plugin. A new window with several tabs opens.
4. Click on the **APBS Location** tab. Click on **APBS binary location** and choose the location of the APBS executable (e.g. C:\apbs-0.5.0\bin\apbs.exe). The 1.7.4.3 PyMOL incentive Version has PDB2PQR and APBS already built in
5. Click on **Set Grid**
6. Click on **Run APBS** and wait the calculation to finish
7. Click on the **Visualization** tab and hit **Update**
8. In the Molecular Surface area, click on **Show**

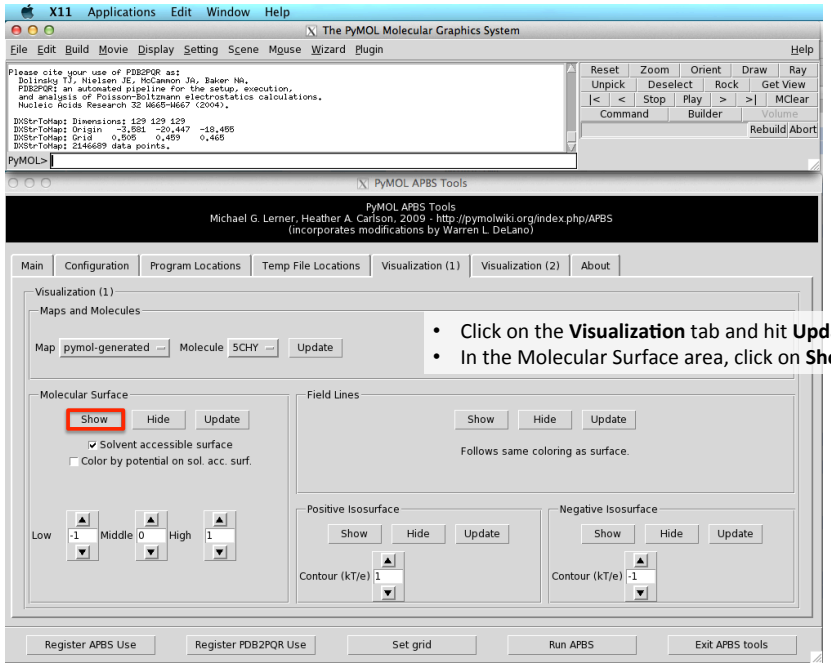
http://web.chem.ucsb.edu/~kalju/CSUPERB/public/prot_prot_A1.html


X11 Applications Edit Window Help
The PyMOL Molecular Graphics System
File Edit Build Movie Display Setting Scene Mouse Wizard Plugin Help
Reset Zoom Orient Draw Ray
Unpick Deselect Rock Get View
Stop Play MClear
Command Builder Volume
Rebuild Abort
PyMOL>
MacPyMOL
• Open PyMOL (X11Hybrid on OSX) and load PDB structure 5CHY
• Remove solvent molecules. You can either select **remove waters** from the **A**ctions column in the right-hand side toolbar or type **remove solvent** at the console.

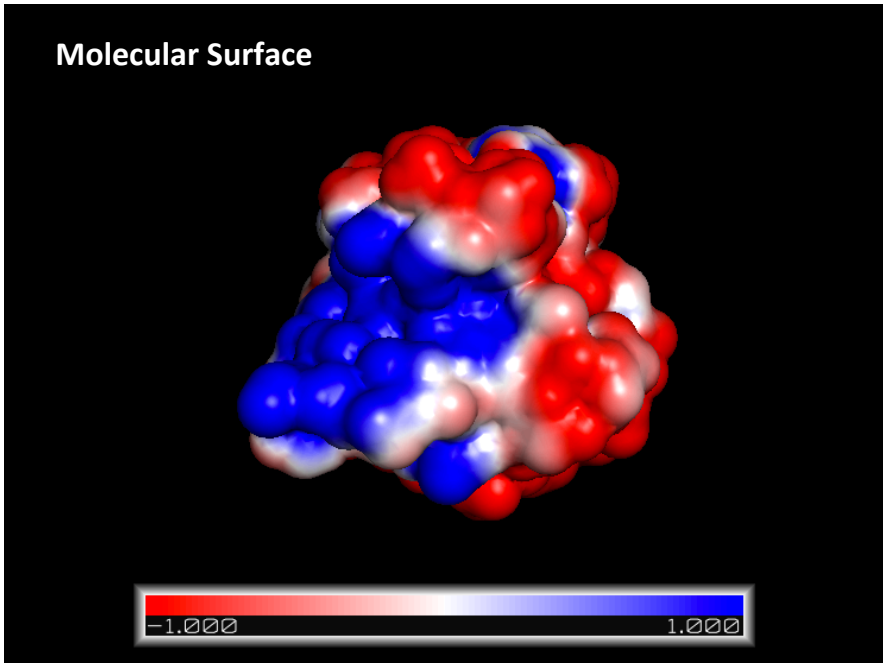

X11 Applications Edit Window Help
The PyMOL Molecular Graphics System
Plugin
Plugin Manager
Lighting Settings
APBS Tools2.1...
PySwissSidechain...
D_PySwissSidechain...
PyMOL APBS Tools
PyMOL APBS Tools
Michael G. Lerner, Heather A. Carlson, 2009 - http://pymolwiki.org/index.php/APBS
(incorporates modifications by Warren L. DeLano)
Main Configuration Program Locations Temp File Locations Visualization (1) Visualization (2) About
Main options
Selection to use: (polymer)
What to call the resulting map: apbs_map
Use PyMOL generated PQR and existing Hydrogens and termini
Use PyMOL generated PQR and PyMOL generated Hydrogens and termini
Use another PQR
Use PDB2PQR
pdb2pqr command line options: --ff=AMBER
Choose Externally Generated PQR:
Register APBS Use
Register PDB2PQR Use
Set grid
Run APBS
Exit APBS tools
• Open the APBS Tools plugin. A new window with several tabs opens.
• Click on the **APBS Location** tab. Click on **APBS binary location** and choose the location of the APBS executable (e.g. C:\apbs-0.5.0\bin\apbs.exe). The 1.7.4.3 PyMOL incentive Version has PDB2PQR and APBS already built in

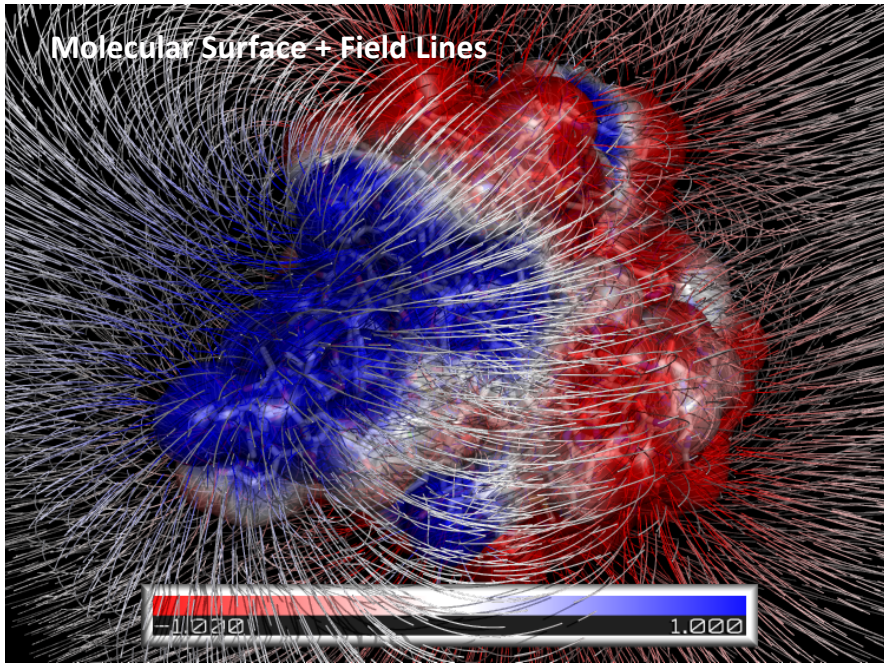

X11 Applications Edit Window Help
The PyMOL Molecular Graphics System
PyMOL APBS Tools
PyMOL APBS Tools
Michael G. Lerner, Heather A. Carlson, 2009 - http://pymolwiki.org/index.php/APBS
(incorporates modifications by Warren L. DeLano)
Main Configuration Program Locations Temp File Locations Visualization (1) Visualization (2) About
Locations
APBS binary location: /Applications/MacPyMOLX11Hybrid.app/pymol/ext/bin/apbs
APBS psize.py location: /Applications/MacPyMOLX11Hybrid.app/pymol/ext/share/apbs/tools/manip/psize.py
pdb2pqr location: /Applications/MacPyMOLX11Hybrid.app/pymol/ext/share/pdb2pqr/pdb2pqr.py
By default, the PyMOL APBS Tools will use APBS's psize.py to calculate proper grid dimensions and spacing. This tool attempts to make the fine mesh spacing approximately 0.5 A, but will make a coarser grid if constrained to do so by the Maximum Memory Allowed setting in the configuration pane. If you wish this behavior, you must ensure that "APBS psize.py location" above points to a valid file. This plugin can also calculate grid dimensions and spacing itself. If you wish that behavior, simply delete the "APBS psize.py location" above.
By default, the PyMOL APBS Tools will use PDB2PQR to generate PQR files. Command line options for PDB2PQR can be controlled on the Main panel. If you wish to use PDB2PQR, you must ensure that "pdb2pqr location" above points to a valid file. PyMOL can directly generate PQR files using standard protein residues and AMBER charges. If wish that behavior, simply delete the "pdb2pqr location" above.
Register APBS Use
Register PDB2PQR Use
Set grid
Run APBS
Exit APBS tools

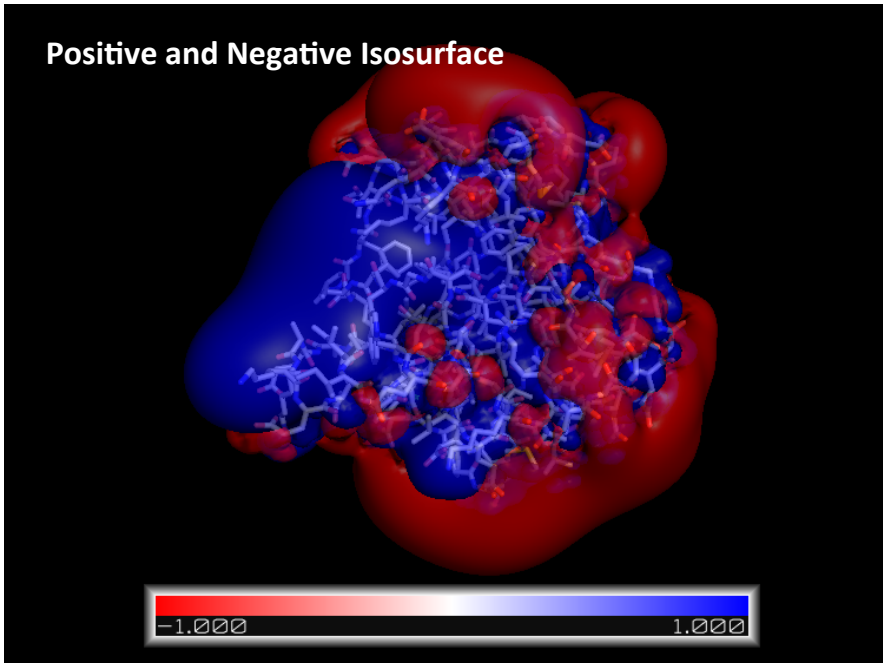

X11 Applications Edit Window Help
The PyMOL Molecular Graphics System
PyMOL APBS Tools
PyMOL APBS Tools
Michael G. Lerner, Heather A. Carlson, 2009 - http://pymolwiki.org/index.php/APBS
(incorporates modifications by Warren L. DeLano)
Main Configuration Program Locations Temp File Locations Visualization (1) Visualization (2) About
Main options
Selection to use: (polymer)
What to call the resulting map: apbs_map
Use PyMOL generated PQR and existing Hydrogens and termini
Use PyMOL generated PQR and PyMOL generated Hydrogens and termini
Use another PQR
Use PDB2PQR
pdb2pqr command line options: --ff=AMBER
Choose Externally Generated PQR:
Register APBS Use
Register PDB2PQR Use
Set grid
Run APBS
Exit APBS tools
• Click on **Set Grid**
• Click on **Run APBS** and wait the calculation to finish

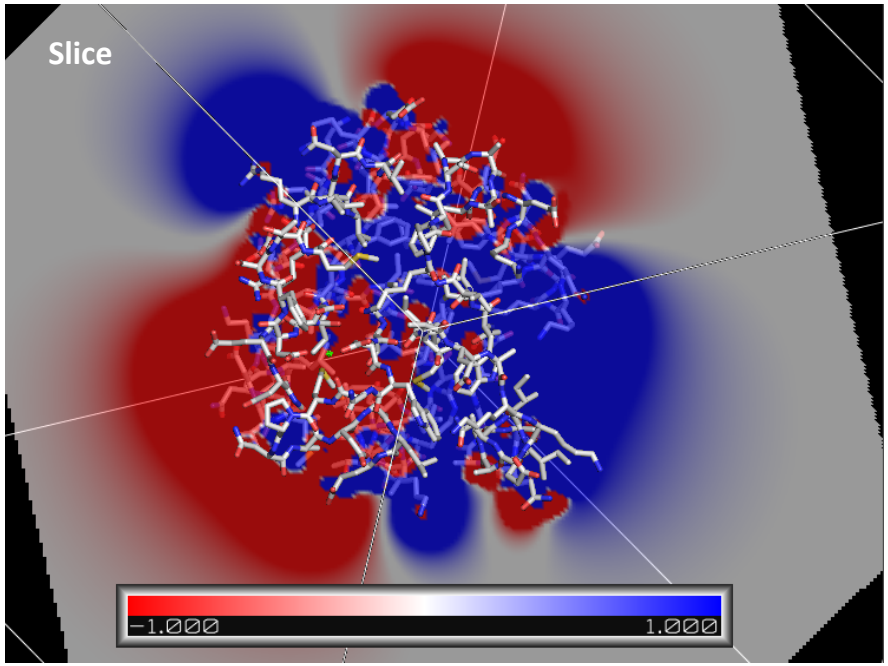


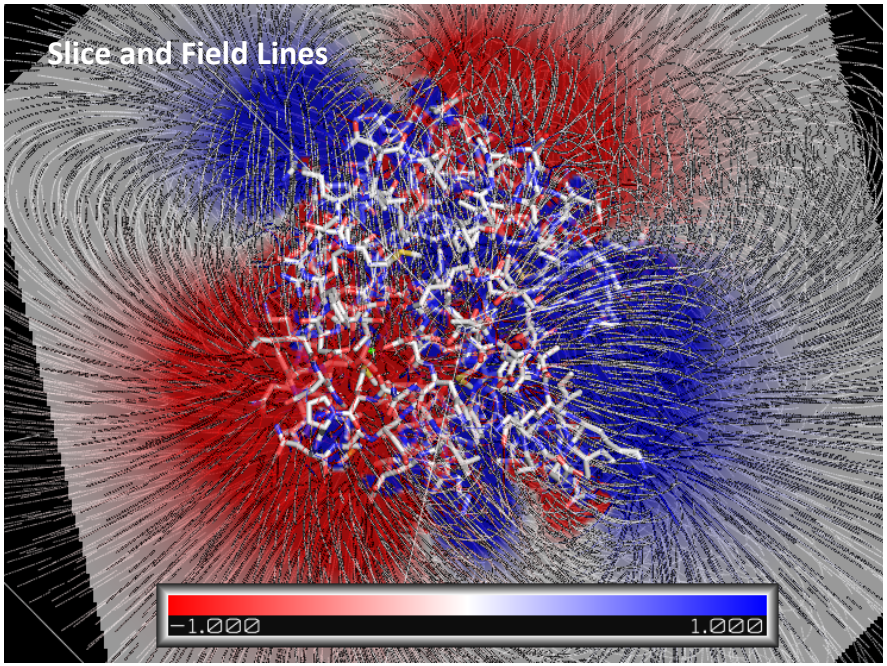
- Click on the **Visualization** tab and hit **Update**
- In the Molecular Surface area, click on **Show**











APBS tutorial at

http://web.chem.ucsb.edu/~kalju/CSUPERB/public/prot_prot_A1.html
goes on to investigate electrostatic interactions between CheA and CheY

- Try it out!

CastP http://sts.bioe.uic.edu/castp/pymol.php

CAVER http://www.caver.cz/index.php?sid=199

Autodock http://wwwuser.gwdg.de/~dseelig/adplugin.html

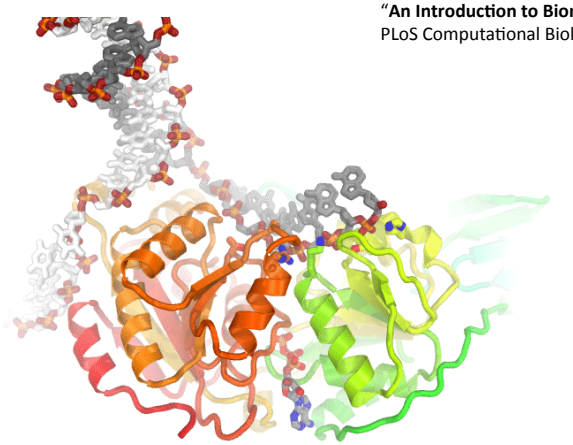
Gromacs http://www.pymolwiki.org/index.php/GROMACS_Plugin

DSSP and Stride http://www.biotec.tu-dresden.de/~hongboz/dssp_pymol/dssp_pymol.html

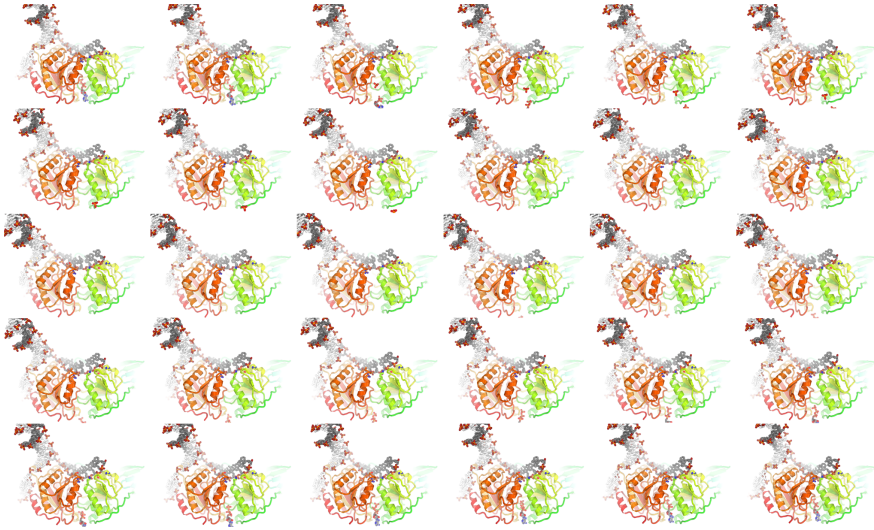
Xpyder http://linux.btbs.unimib.it/xpyder/

MLP Tools http://mlptools.altervista.org/index.html

An Animation is a Series of Images

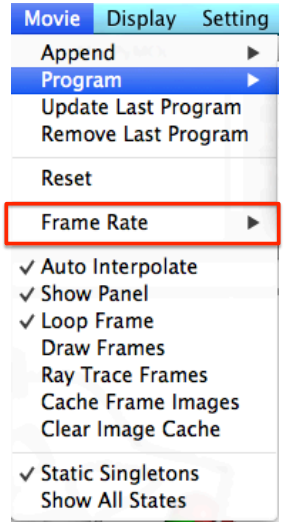


C.Mura, C.M. McCrimmon, J.Vertrees, M.R. Sawaya
"**An Introduction to Biomolecular Graphics**"
PLoS Computational Biology 8 (2010) Vol. 6, pp 1-11



http://pymolwiki.org/index.php/PLoS
Movie of a DNA helicase unwinding dsDNA (animated GIF format)

Setting up the Movie



Set up movie, apply standard motions, undo last addition to movie

Up to 10-12 pictures per second can still be perceived as individual pictures, the animation appears jerky
Standard for 35mm sound movies: 24 Fps.

PyMOL default: 30 Fps, the animated gif was 15 Fps

http://en.wikipedia.org/wiki/Frame_rate

Database of Macromolecular Movements with Associated Tools for Flexibility and Geometric Analysis

http://molmovdb.mbb.yale.edu/molmovdb/

Types of Animations

Camera stationary
Molecule immobile
Stationary figure
Stationary phase in movie

Camera mobile **Camera loop**
Molecule immobile **Scene loop**
Animation achieved by changing the transformation matrix (get_view)
Rotate, Rock, Nutate, Zoom, Clip
with or without change in representation

Camera stationary **State loop**
Molecule mobile **State sweep**
- Multiple conformations of an object provided as multiple states in an object, e.g. MD Trajectories, NMR, Morphs
Moving rigid objects relative to each other

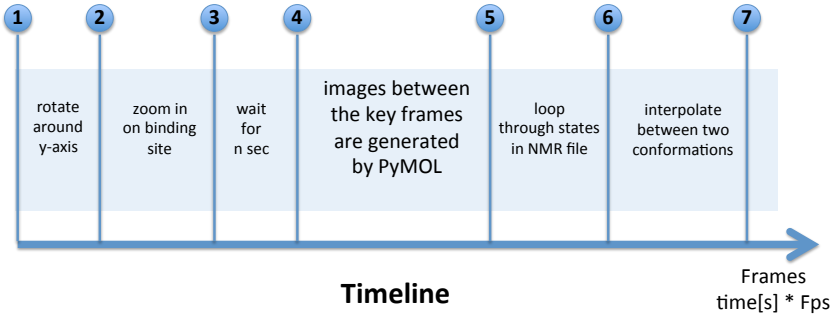
Camera mobile
Molecule mobile
Transformation matrix and object coordinates (displayed state) both change during the animation

Smooth Transitions between representations
Fading in and out of surfaces, side chains etc.

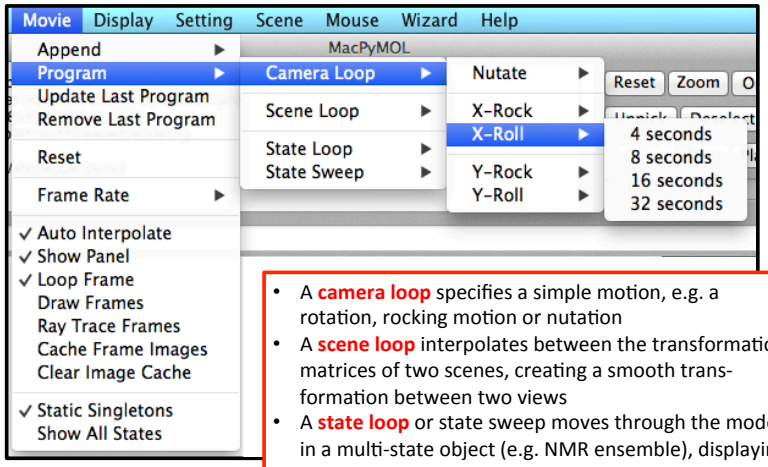
The Movie Timeline

scenes
key frames
defined by user

The user defines the key frames (individual images saved as scenes)
PyMOL interpolates between the scenes according to the actions and timing specified by the user



Simple Camera, Scene and State Loops by GUI

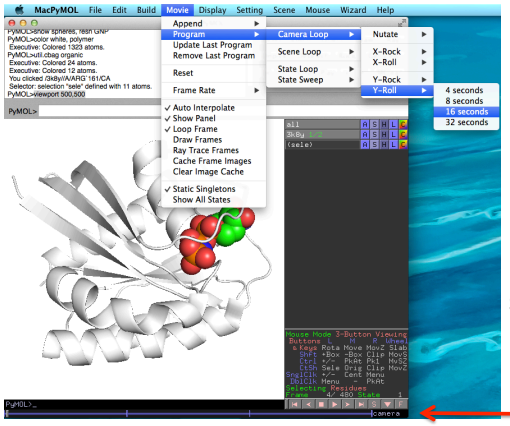


- A **camera loop** specifies a simple motion, e.g. a rotation, rocking motion or nutation
- A **scene loop** interpolates between the transformation matrices of two scenes, creating a smooth trans-formation between two views
- A **state loop** or state sweep moves through the models in a multi-state object (e.g. NMR ensemble), displaying them one after the other

More complex animations require the command line or manipulation of the coordinates outside PyMOL

Camera Loop

Demo, Play along



Write the script to set up the image, copy-paste into command line:

```
viewport 500,500
bg_color white
load pdb/3K8Y.pdb
select lig, resn GNP
select prot, polymer
hide all
show cartoon
show spheres, lig
color white, polymer
util.cbag lig
```

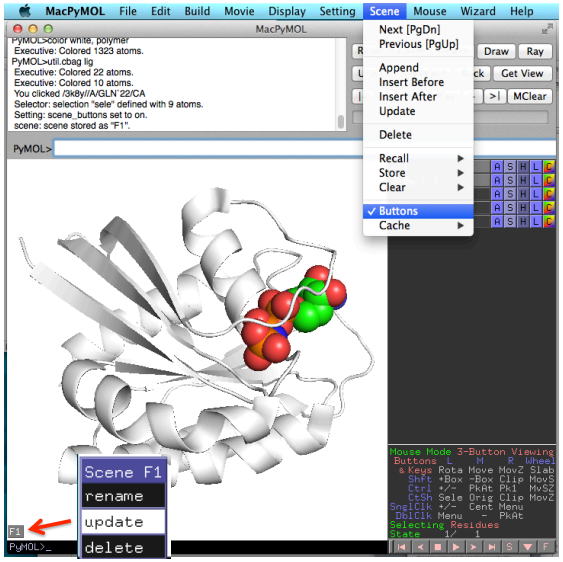
Select from pull-down menu:
Movie:Program:Camera Loop:Y-Roll:16 sec

Movie Timeline
(shown only if "Show Panel" is selected)

use movie control panel to preview the movie:

Try out the effects of other "Camera Loop" options: Nutate, Rock, Roll around X and Y axis

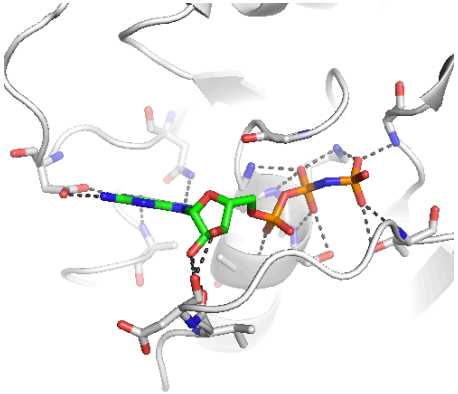
Store Scene for Future Use



Scene buttons can be renamed and reordered
The first 12 scenes are mapped to function keys F1-F12

Right-click for scene button context menu

Camera Loop



add to the script and paste into command line:

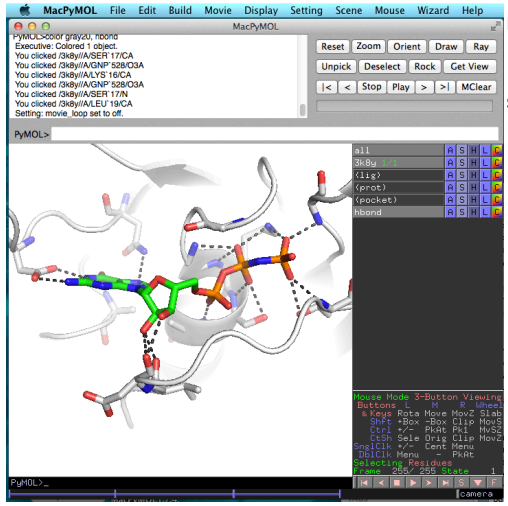
```
hide spheres, lig
show sticks, lig
select pocket, br. (prot within 3.2 of lig)
show sticks, pocket
util.cbaw pocket
distance hbond, lig, pocket, 3.2, 2
hide label, hbond
color gray20, hbond
orient lig
rot y, 180
```

correct zoom and orientation to nicely shows the ligand

Store to scene F2

select from pull-down menu:
Movie: Reset
Movie: Camera Loop: Nutate: 30 deg. over 12 sec

Scene Loop



view F1 transition view F2

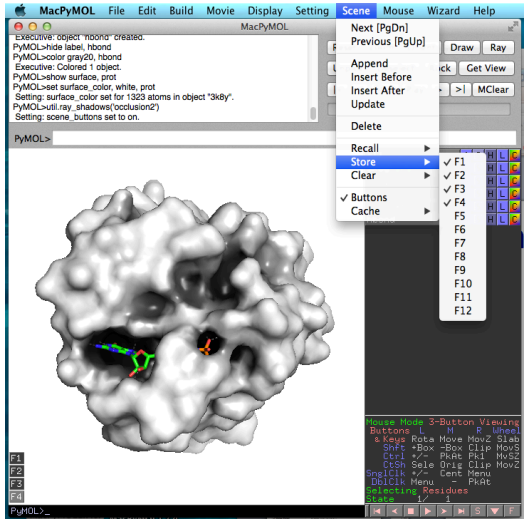
Putting the two scenes together:

select from pull-down menu:
Movie: Reset
Menu: Scene:Recall:F1
Movie:Program:Scene Loop:Steady:4 sec

The speed of the transition is set by:
"set scene_animation_duration"
default is 2.25s

try out different values!

Scenes are your Story Board




```
viewport 500,500
bg_color white
load pdb/3k8y.pdb
select lig, resn GNP
select prot, polymer
hide all
show cartoon
show spheres, lig
color white, polymer
util.cbag lig
### save scene (F1) ###
orient lig
hide spheres, lig
show sticks, lig
select pocket, br. (prot within 3.2 of lig)
show sticks, pocket
util.cbaw pocket
distance hbond, lig, pocket, 3.2, 2
hide label, hbond
color gray20, hbond
### optimize zoom and orientation ###
### save scene (F2) ###
show surface, prot
set surface_color, white, prot
util.ray_shadows('occlusion2')
### save scene (F3) ###
zoom out
### save scene (F4) ###
```

add and apply

Save script as: MyMovie1.pml

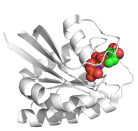
PyMOL Show

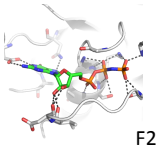
- Scenes are simple PyMOL memory snapshots, in which PyMOL saves the current colors, representations and camera position.
- Scenes can be given informative names an reordered by dragging the scene buttons.
- Settings are not saved, changing settings can affect all previously saved scenes in the show
- A PyMOL show is a series of Scenes that the user can navigate through by pressing the pgUp and pgDown keys
- The transition between two scenes is made by a smooth camera animation
- A scene loop sweeps through all stored scenes, the speed is determined by “set scene_animation_duration”, default 2.25 s
- For better control of the movie, use scenes as Key frames and control the timing through the timeline

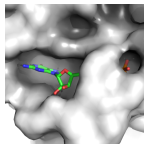
Planning your Animation – The Storyboard

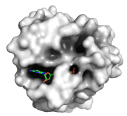
Plan your story: what message do you want to convey to the viewer?

Set up the key images for your movie: what do you want to show?
Save the key images as scenes


F1


F2


F3

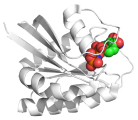

F4

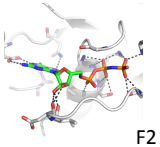
How do you want to transition from one view to the next?

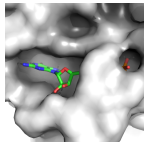
360° rotation around y-axis	zoom in on ligand	rock x	show surface	rock x	zoom out to full protein	360° rotation around y-axis
6 sec	6 sec	12 sec		12 sec	6 sec	6 sec

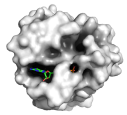
Plan the timing – is there a narration?

Putting the Scenes together


F1

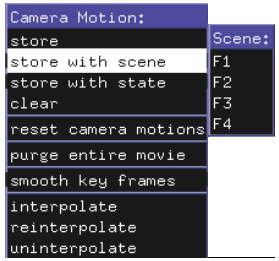

F2


F3


F4

360° rotation around y-axis	zoom in on ligand	nutate	show surface	nutate	zoom out to full protein	360° rotation around y-axis

Scene:Recall:F1
Movie:Program:Camera Loop:Y-Roll:16s
Movie:Append:12s
right-click on end of timeline and select: Store with scene F2
Movie:Program:Camera Loop:Nutate:30 degrees over 4s
Movie:Program:Camera Loop:Nutate:30 degrees over 8s
right-click on end of timeline and select: Store with scene F3
Movie:Program:Camera Loop:Nutate:30 degrees over 8s
Movie:Program:Camera Loop:Nutate:30 degrees over 4s
Movie:Append:12s
right-click on end of timeline and select: Store with scene F4
Movie:Program:Camera Loop:Y-Roll:16s



play movie

Changing the Timing of a Transition

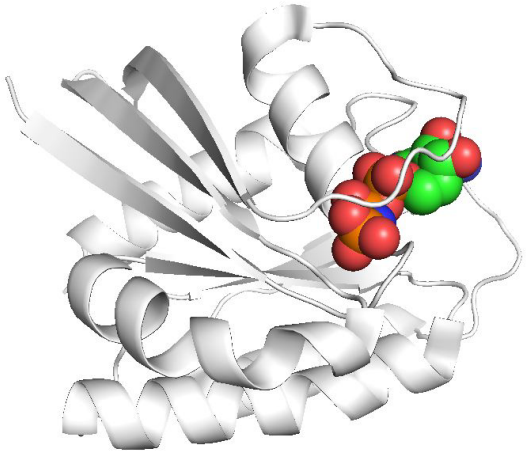
<ctrl> left drag mouse
to the right (green):
These frames get added
to the timeline, this phase
of the movie is lengthened

<ctrl> left drag mouse
to the left (red):
These frames get deleted
from the timeline,
this phase of the movie
is shortened

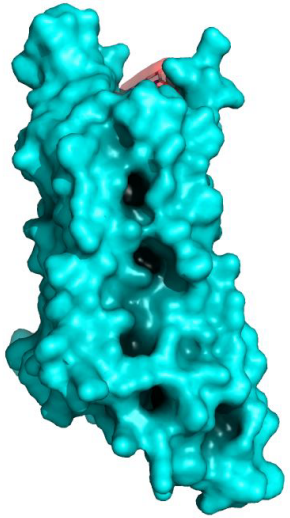
Play the movie: altering the timing of nutation does not work, motion gets jerky – redo the movie with different time settings, or using rock instead of nutation

Movie: Cache Frame Images
Movie: Ray Trace Frames
File: Save Movie as: Quicktime Movie

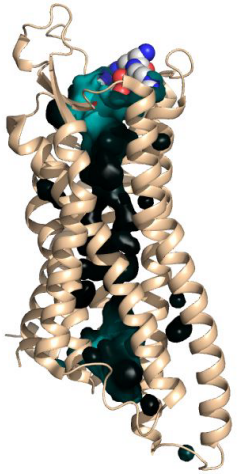
Save session as: MyMovie1.pse



Animating the Clipping Plane



Cavities



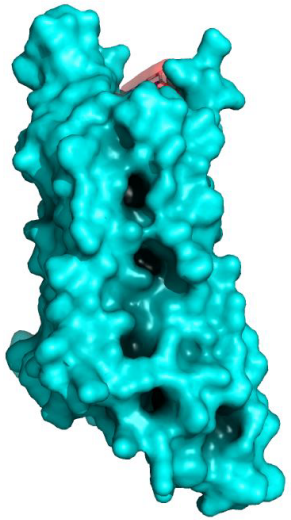
Putting the Movie together in Quicktime

- Changing settings such as transparency or surface cavity mode cannot be changed in the course of a Pymol movie, since they affect all states. Produce separate movies and put them together in Quicktime or an other movie editing program.

- Ray-tracing movie sequences can be quite time-consuming, if you want e.g. several rotations of the molecule without changing anything else, do a single rotation and insert it several times into the final movie

DEMO

The two Movies Combined



Types of Animations

Camera stationary
Molecule immobile
Stationary figure
Stationary phase in movie

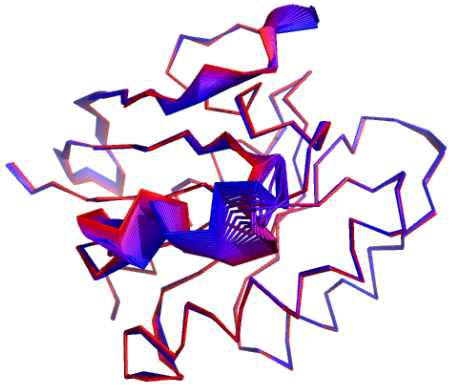
Camera stationary **State loop**
Molecule mobile **State sweep**
- **Multiple conformations of an object provided as multiple states in an object, e.g. MD Trajectories, NMR, Morphs**
- Moving rigid objects relative to each other

Camera mobile **Camera loop**
Molecule immobile **Scene loop**
Animation achieved by changing the transformation matrix (get_view)
Rotate, Rock, Nutate, Zoom, Clip
with or without change in representation

Camera mobile
Molecule mobile
Transformation matrix and object coordinates (displayed state) both change during the animation

Smooth Transitions between representations
Fading in and out of surfaces, side chains etc.

Morph between the GDP and GTP bound Ras



```
viewport 500,500
bg_color white
load pdb/1Q21.pdb
load pdb/1QRA.pdb
align 1Q21, 1QRA, cycles=0, object=aln
morph QQQ, 1Q21, 1QRA
```

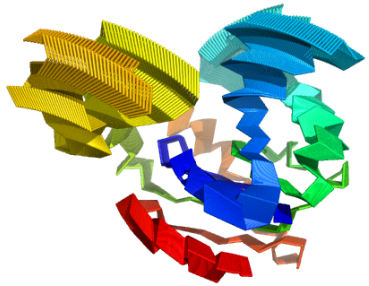
```
hide all
show ribbon
run py_scripts/spectrum_states.py
spectrum_states QQQ, ribbon, red purpleblue blue
```

@pml_scripts/rasMorph1.pml
requires py_scripts/spectrum_states.py

Movie: Program: State Sweep: ¼ Speed: No Pause

morph name, sele1 [, sele2 [, state1 [, state2 [, refinement [, steps [, method [, match]]]]]]]

Interpolating Between two Conformations



1ake, 4ake: two conformations of adenylate kinase from *E. coli*

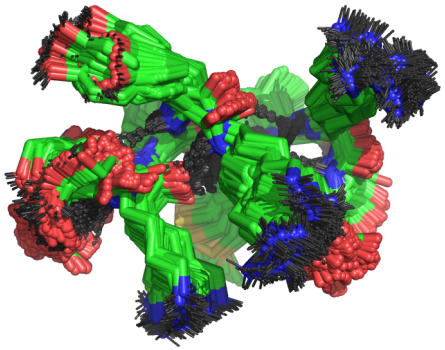
@pml_scripts/MakeMorph.pml

```
viewport 500,500
bg_color white
load pdb/1ake.pdb
load pdb/4ake.pdb
create 1akeA, 1ake and chain A and polymer
create 4akeA, 4ake and chain A and polymer
disable 1ake
disable 4ake
align 1akeA, 4akeA, cycles=0, object=aln
morph Make, 1akeA , 4akeA

hide all
cmd.spectrum("count",selection="Make",byres=1)
set all_states, on
ray
png figures/all_states.png

set all_states, off
```

Movie: Program: State Sweep: ¼ Speed: No Pause
Try other representations, e.g. cartoon, sticks, spheres

http://www2.molmovdb.org

Animating a MD Trajectory



@pml_scripts/Trajectory.pml

load pdb/SampleTrajectory.gz, ST
show sticks, ST and not h.
color gray25, hydrogen
show sticks, ST and not hydrogen

Action:Find:Polar Contacts: Within Selection
(distance hbond, all, all, 3.2, 2; hide label)
set dash_color, gray25

you can already run the trajectory as movie.
To see all states at once, choose:
Movie: Show all States.

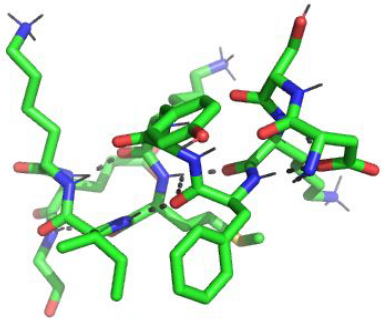
To supress high frequency
motions, type:
smooth

to keep peptide from wandering type:
intra_fit ST

Movie:Program:State Loop:Full Speed: No Pause

smooth [selection [, passes [, window [, first [, last [, ends]]]]]]

Animate a Trajectory



Types of Animations

Camera stationary
Molecule immobile
Stationary figure
Stationary phase in movie

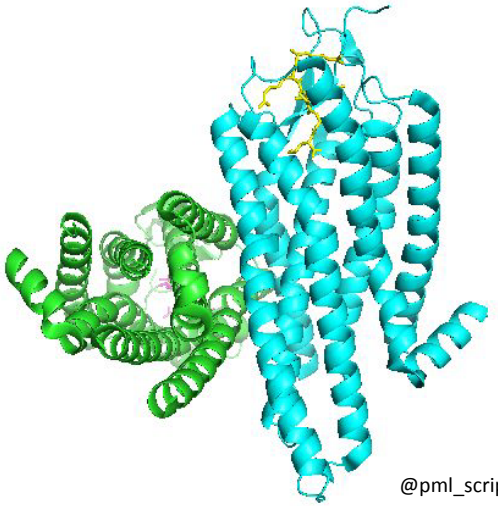
Camera stationary **State loop**
Molecule mobile **State sweep**
- Multiple conformations of an object provided as multiple states in an object, e.g. MD Trajectories, NMR, Morphs
- **Moving rigid objects relative to each other**

Camera mobile **Camera loop**
Molecule immobile **Scene loop**
Animation achieved by changing the transformation matrix (get_view)
Rotate, Rock, Nutate, Zoom, Clip
with or without change in representation

Camera mobile
Molecule mobile
Transformation matrix and object coordinates (displayed state) both change during the animation

Smooth Transitions between representations
Fading in and out of surfaces, side chains etc.

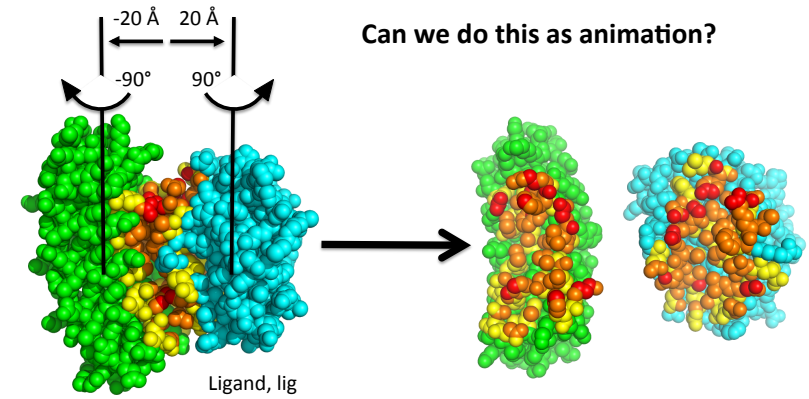
Animating a 3D-Superposition



@pml_scripts/Alignment1.pml

Cannot be done with a scene animation, should theoretically work by timeline editing, it works for dragging objects around, but not for alignment=> Do it by scripting

Example: Looking at a Binding Interface



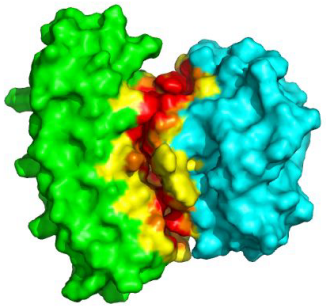
Can we do this as animation?

rotate y, -90, darp
translate [-20,0,0], darp
rotate y, 90, lig
translate [20,0,0], lig

Moving two Objects Relative to Each Other

Animation in object coordinates, not camera coordinates

DARPin-BCL-W complex



@pml_scripts/darpins.pml

translate vector [,selection [,state [,camera [,object]]]]
rotate axis, angle [,selection [,state [,camera [,object [,origin]]]]]

Doing it all in a script

open DARPins.pml in text editor

 # load the coordinates

```
load pdb/S4K5B_B-C.pdb, Cmplx
extract darp, Cmplx and chain B
extract lig, Cmplx and chain C
delete Cmplx
```

change this to load a different molecule

 # representation and coloring

```
viewport 500, 500
bg_color white
color green, darp
color cyan, lig
select C1, br. ((darp within 5.0 of lig) or (lig within 5.0 of darp))
color yellow, C1
select C2, ((darp within 5.0 of lig) or (lig within 5.0 of darp))
color orange, C2
select C3, ((darp within 3.6 of lig) or (lig within 3.6 of darp))
color red, C3
hide all
show surface
util.ray_shadows('occlusion1')
select none
```

change this for a different representation

 # setup PyMOL for movies

```
set matrix_mode, 1
set movie_panel, 1
set scene_buttons, 1
set cache_frames, 1
set move_auto_interpolate, 1
config_mouse three_button_motions, 1
```

Settings affecting movie production and manual editing

 # initialize the movie

```
mset 1 x840
```

The entire movie will be 840 frames long (28 s)

 # nothing happens in frames 1-60

```
frame 1
mview store, object=darp
mview store, object=lig
frame 60
mview store, object=darp
mview store, object=lig
```

darp and lig do not get changed between being stored in frame 1 and stored in frame 60

 # move darp and lig apart in frame 60-120

```
frame 120
rotate y, -90, object=darp
translate [-20,0,0], object=darp
mview store, object=darp
rotate y, 90, object=lig
translate  [20,0,0], object=lig
mview store, object=lig
```

Darp and lig are rotated and translated in different directions before stored in frame 120. These transformations are evenly spread out over frames 61 to 120

 # nothing happens in frames 120-300

```
frame 300
mview store, object=darp
mview store, object=lig
```

Darp and lig do not get changed between being stored in frame 120 and stored in frame 300

 # move darp and lig together in frame 330-360

```
frame 360
translate [20,0,0], object=darp
rotate y, 90, object=darp
mview store, object=darp
translate  [-20,0,0], object=lig
rotate y, -90, object=lig
mview store, object=lig
```

Note that the order of “rotate” and “translate” have to be reversed to undo the trans-formation that occurred between frames 60 and 120

 # nothing happens in frames 540-660

```
frame 660
mview store, object=darp
mview store, object=lig
```

 # move darp and lig together in frame 660-720

```
frame 720
translate [20,0,0], object=darp
rotate z, -90, object=darp
mview store, object=lig
translate  [-20,0,0], object=lig
rotate z, 90, object=lig
mview store, object=lig
```

 # nothing happens in frames 720-780

```
frame 780
mview store, object=darp
mview store, object=lig
```

 # frames 360-420
 # rotate complex around x-axis

```
frame 420
rotate x,-90,object=darp
mview store, object=darp
rotate x,-90,object=lig
mview store, object=lig
```

 # nothing happens in frames 420-480

```
frame 480
mview store, object=darp
mview store, object=lig
```

 # move darp and lig apart in frame 480-540

```
frame 540
rotate z, 90, object=darp
translate [-20,0,0], object=darp
mview store, object=darp
rotate z, -90, object=lig
translate  [20,0,0], object=lig
mview store, object=lig
```

 # frames 780-840
 # rotate complex back into initial orientation

```
frame 840
rotate x, 90,object=darp
mview store, object=darp
rotate x, 90,object=lig
mview store, object=lig
```

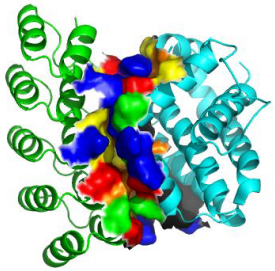
 # play movie

```
mplay
```

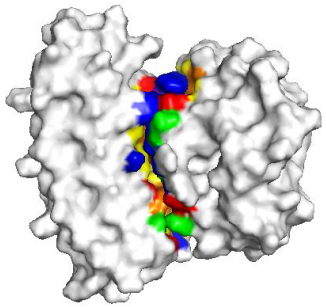
Changing the Representation of the DARPins

 # define representation

```
viewport 500, 500
bg_color white
color white, all
color orange, resn Tyr+Phe+Trp
color yellow, resn Leu+Ile+Val+Pro+Ala+Met+Cys
color green, resn Ser+Thr+Asn+Gln
color red, resn Asp+Glu
color cyan, resn His
color blue, resn Arg+Lys
color limon, resn Gly
select C1, br. ((darp within 5.0 of lig) or (lig within 5.0 of darp))
color white, not C1
hide all
show surface, C1
show cartoon
set cartoon_color, green, darp
set cartoon_color, cyan, lig
util.ray_shadows('occlusion1')
select none
```



@pml_scripts/darpins1.pml



@pml_scripts/darpins2.pml

Showing Movies in Powerpoint

Movies do not become part of the Powerpoint file!!!

- Create a folder for your ppt file
- In this folder, create a subfolder for the movie files
- Now insert your movies into your ppt file
- When copying your ppt presentation to a memory stick or a different computer, always copy the entire folder containing both your presentation and your movies!

The safest method is to create your presentation
on the labtop you intend to use for the presentation!

Avoid switching presentations from Win to Mac or vice versa !!!

Introduction to PyMOL

Presentation · February 2017

DOI: 10.13140/RG.2.2.19625.80483

CITATION
1

READS
5,055

1 author:

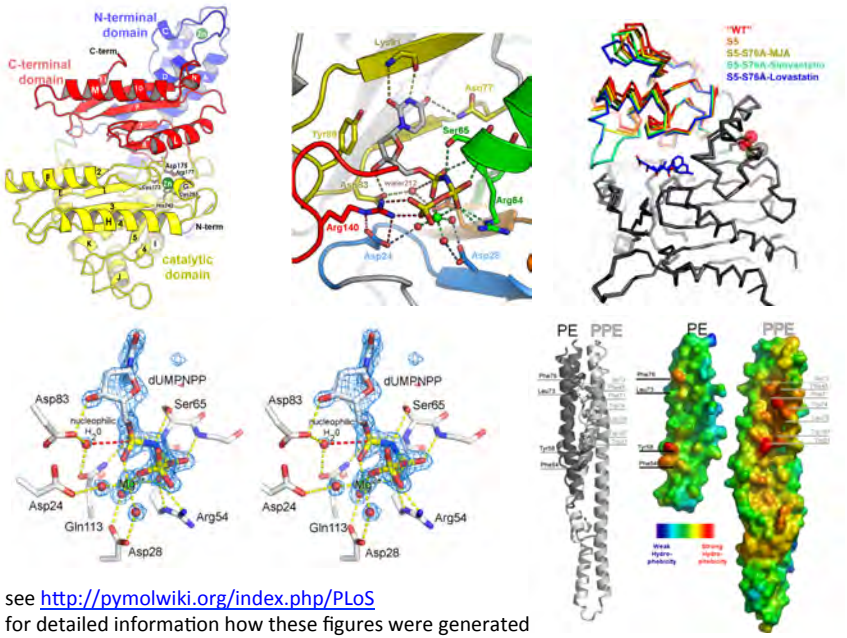
Annemarie Honegger
University of Zurich

123 PUBLICATIONS **10,844** CITATIONS

SEE PROFILE

Molecular Visualization with PyMOL

Creating publication-quality figures,
presentations and videos



see http://pymolwiki.org/index.php/PLoS
for detailed information how these figures were generated

Before you start, read:

C.Mura, C.M. McCrimmon, J.Vertrees, M.R. Sawaya
"**An Introduction to Biomolecular Graphics**"
PLoS Computational Biology 8 (2010) Vol. 6, pp 1-11

including Supporting Information!

http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1000918
The scripts and data used to generate the figures in the paper can be found here:
http://pymolwiki.org/index.php/PLoS

Overview

1 - Getting started:

Point-and-Click: Working with the graphical user interface
Searching the PDB
Working with multiple models, simple alignments

2 – Intermediate:

Using the command line, Command syntax, getting help
Writing a startup script for customized default settings
Selections and selection algebra, Fine-tuning display settings
Keeping a Log, .pml scripts
Homework: creating

3 – Advanced: Movies, Scripting

Python scripts, PlugIns and Wizards
Making movies

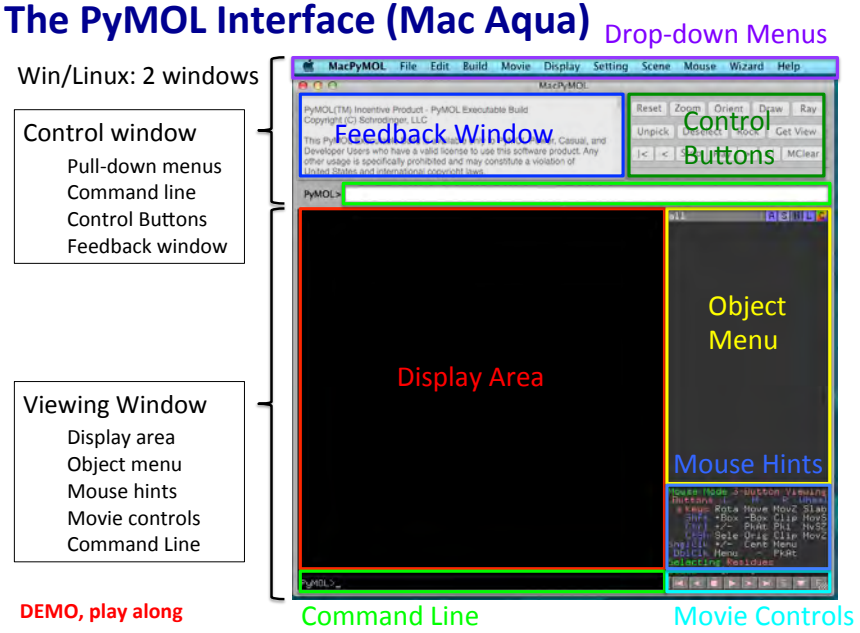
Where can I get PyMOL

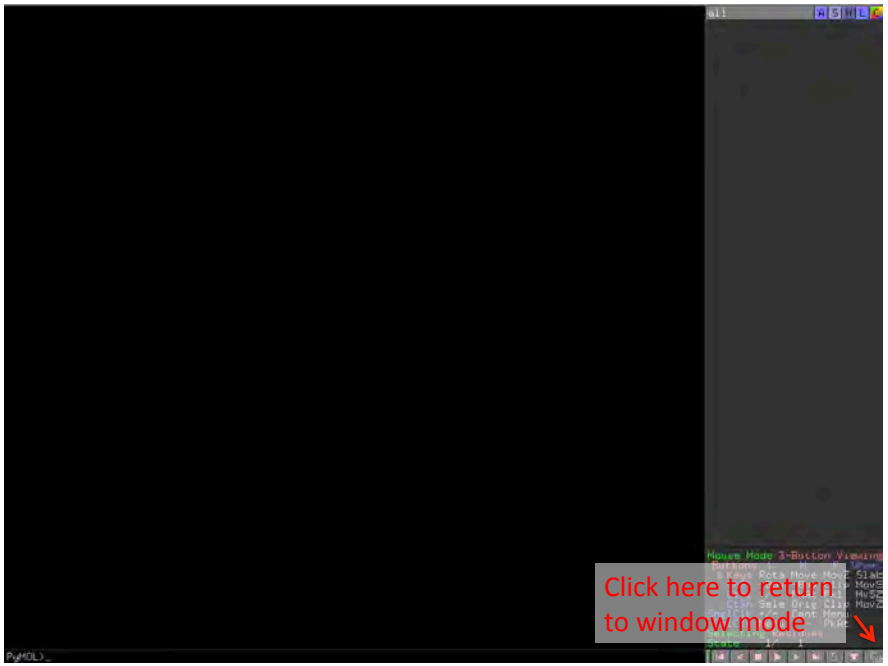
- **Educational version of PyMOL** are available free to students and teachers, you need to register to download: (http://pymol.org/edu/index.php (does not support ray tracing)
- The **source code** of the full version is available through source-forge http://sourceforge.net/projects/pymol/ (you have to compile it)
- **For-Pay full version** from Schrödinger: https://www.pymol.org ($99)
- **BIOC intranet**: Linux/Mac pre-compiled **full version from SBgrid**: https://sbgrid.org (NFS-mount volume "programs" from linfs) **UZH BCH306 students can download this version from OLAT**
- **Unofficial precompiled windows binaries:** http://www.lfd.uci.edu/~gohlke/pythonlibs/
- **Dokumentation:** PyMOL wiki http://pymolwiki.org/index.php/ use **Google to search**, search through the wiki search field is useless
- **Support:** http://sourceforge.net/p/pymol/mailman/pymol-users/
- **Current Version**: 1.8.2 (May 2016)

PyMOL: Introduction

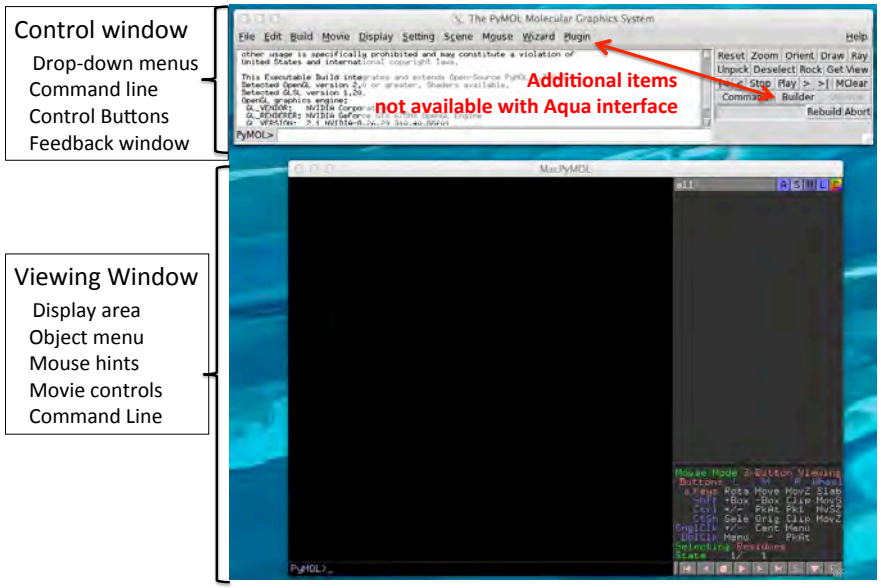
Working with the graphical user interface
Mouse modes, context menu, object menus, pull-down menus
Working with multiple models, sequence display and simple alignments
Working with Wizards

The PyMOL Interface (Mac Aqua)



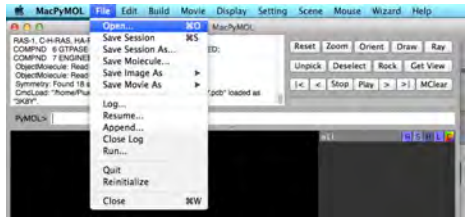


The PyMOL Interface (Mac X11 Hybrid)

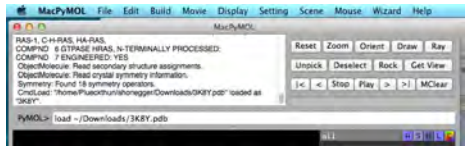


Loading a Model into PyMOL

- Using the **Drop-down menu** to select a local file: ~/pymol/pdb/3K8Y.pdb

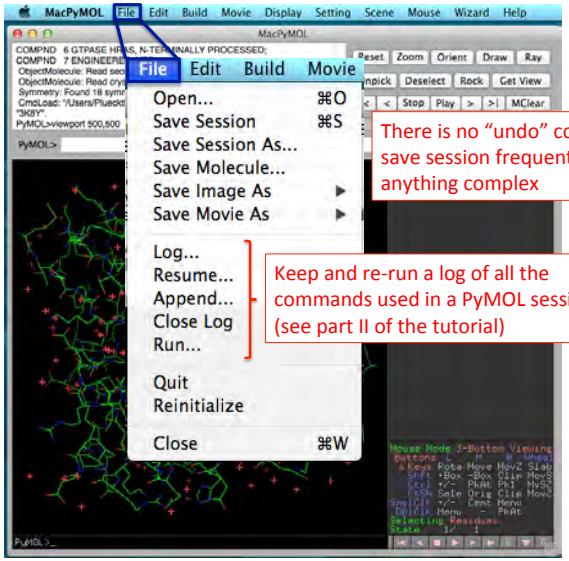


- Using the **command line** to select a local file: load ~/pymol/pdb/3K8Y.pdb



- Using the **command line** to fetch a file from the PDB: fetch 3k8y — **Blocked by the departmental firewall!**
- PlugIn: **PDB Loader Service** — **Blocked by the departmental firewall!**
- load http://www.rcsb.org/pdb/files/3k8y.pdb — **This works despite the firewall !**
- Mac only?: **Drag-and-drop** pdb file onto the program icon or into the PyMOL display area. If PyMOL is already open, multiple pdb files dropped onto the program icon will open in separate windows. If the program is closed, they will open as separate objects in the same window. **DEMO, play along**

Drop-Down Menu: File



There is no "undo" command save session frequently if you do anything complex

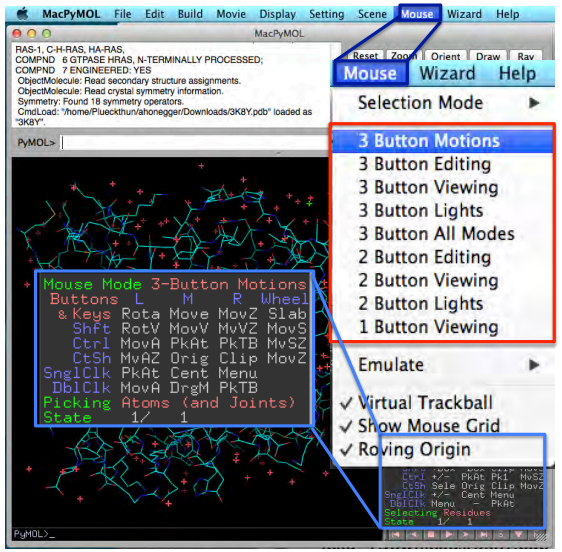
Keep and re-run a log of all the commands used in a PyMOL session (see part II of the tutorial)

Using the Mouse

Mouse behavior can be altered using the "Mouse" drop-down menu

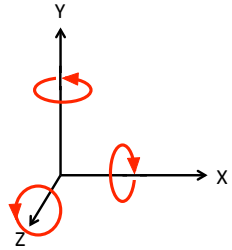
M
L ↕ R
wheel

Two buttons and a scroll wheel that can also act as a button

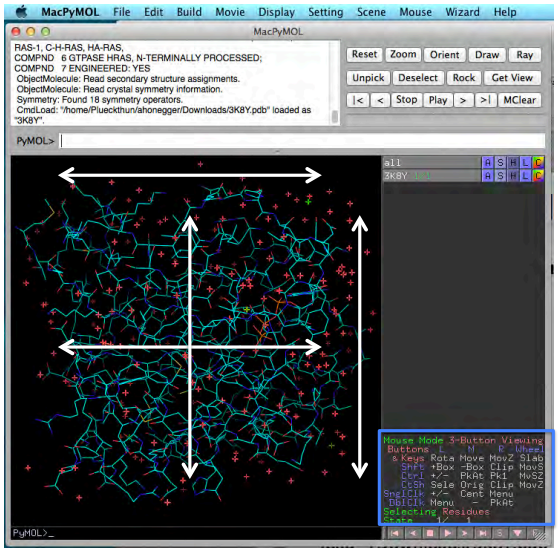


click on mouse hints after altering mouse mode to update panel

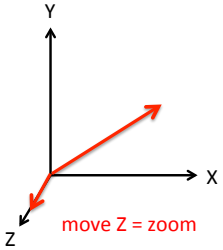
Using the Mouse: Rotations



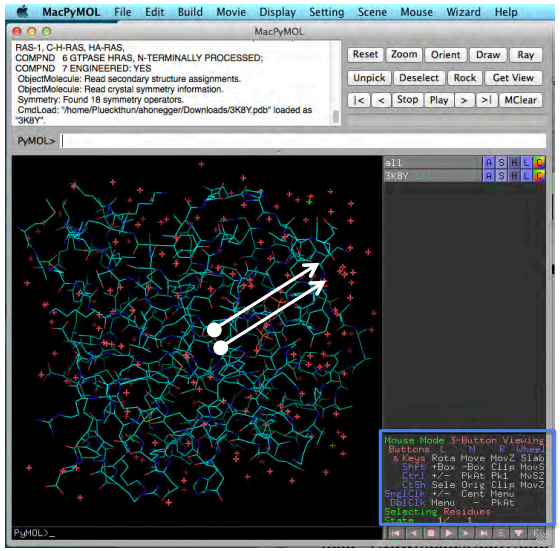
M
L R
wheel



Using the Mouse: Translation



M
L R
wheel



Selecting Atoms and Residues by Mouse-Click

Atoms
Residues
Chains
Segments
Objects

Molecules

C-alphas

Copying or Moving a Selection to a new Object

by cut-and-paste:

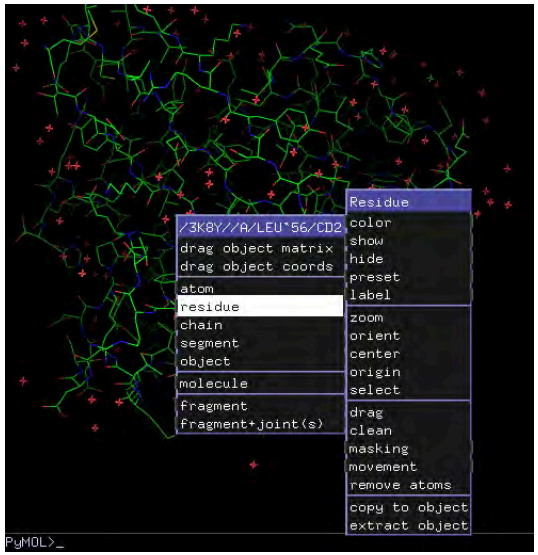
Special Key	Text on the Command Line	Empty Command Line
TAB	Auto-complete text	Display list of all commands
CTRL-A	Position cursor at beginning of line	Select all atoms
CTRL-C		Copy currently selected atoms
CTRL-E	Position cursor at end of line	
CTRL-I		Invert selection
CTRL-K	Delete all text to the right of the cursor	
CTRL-V	Paste into the command line	Paste copied atoms into new object
CTRL-X		Cut atoms
CTRL-Y		Undo
CTRL-Z		Redo

Does not yet work reliably

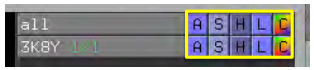
Context Menu

Normally opened by a single left click on an atom (dependent on mouse mode, see mouse hints “Menu”)

Apply selected action to individual atom, to the residue, the whole chain, segment, object or molecule

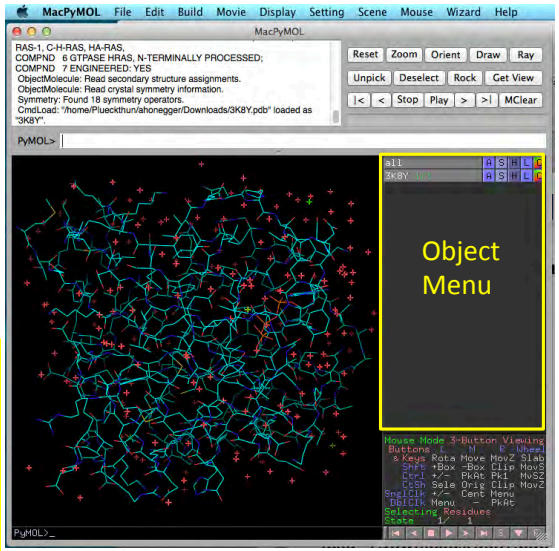


PyMOL Object Menu

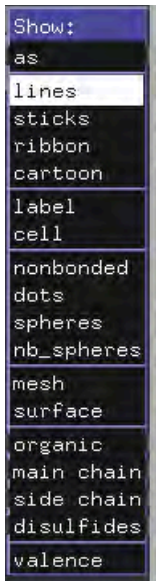


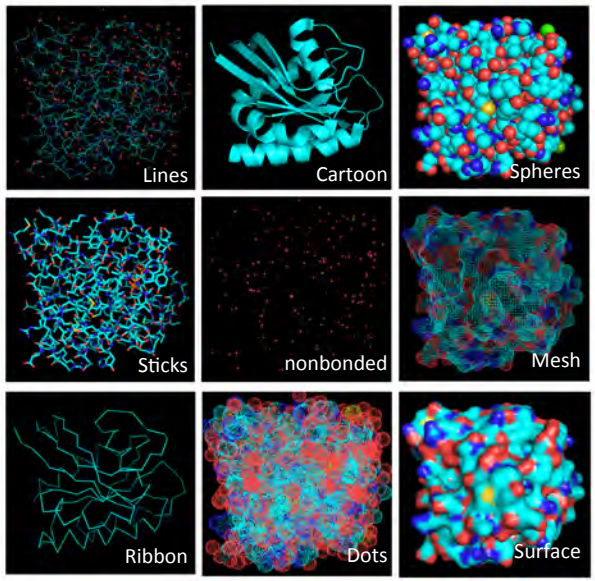
Contains an entry for each object loaded into PyMOL and for each selections. **Display** of an object can be switched off and on by clicking on its name without altering the representation.

ASHLC(M)
- **A**ction
- **S**how
- **H**ide
- **L**abel
- **C**olor
- (**M**ovie)

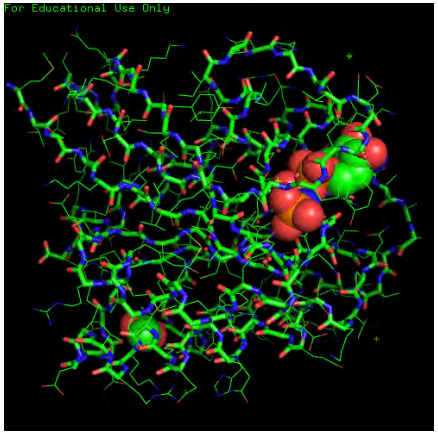


“S” stands for “Show”, “H” stands for “Hide”





Various Representations can be combined



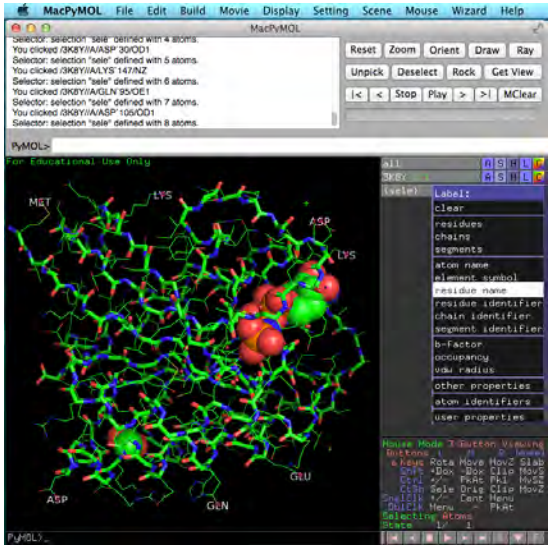
Waters are hidden, side chains shown as lines, the main chain as sticks and the organic molecules as spheres

Use “L” to Label

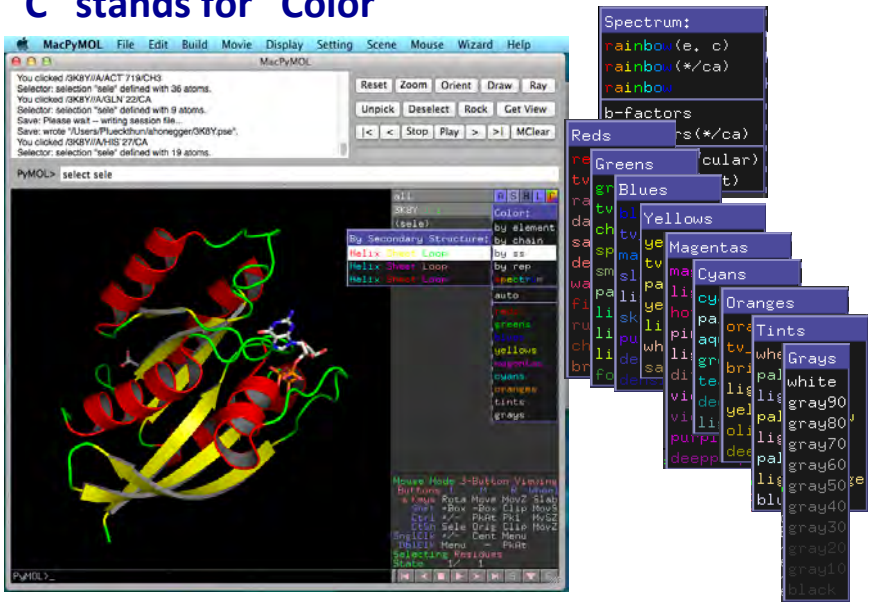
If you label 3K8Y, every single atom will be labelled

Whenever you select a subset of the atoms, item (sele) containing this selection appears on the Object Menu

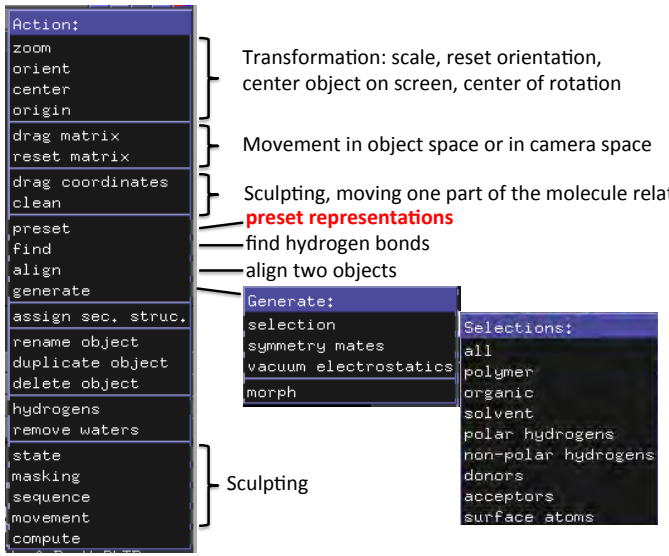
- Mouse: Selection Mode: Atoms
- Pick individual atoms by left clicking then
- (sele): Label: Residue name



“C” stands for “Color”



“A” for “Action”: a mixed bag of Options



Transformation: scale, reset orientation, center object on screen, center of rotation

Movement in object space or in camera space

Sculpting, moving one part of the molecule relative to the rest

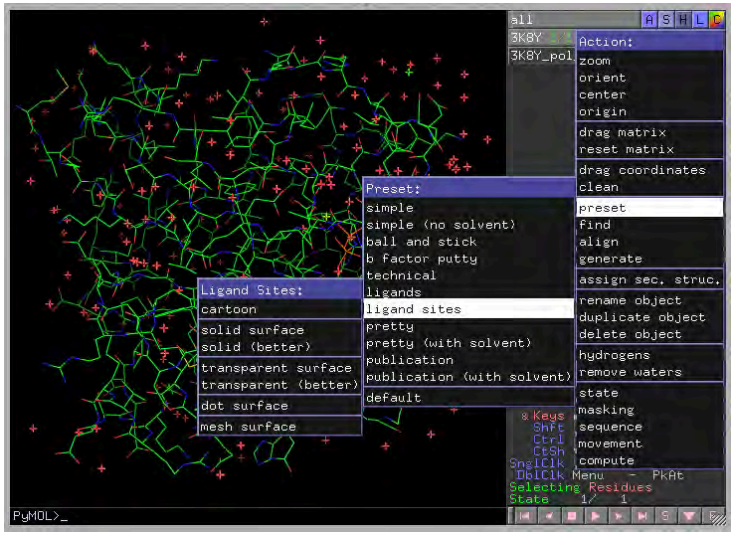
preset representations

find hydrogen bonds

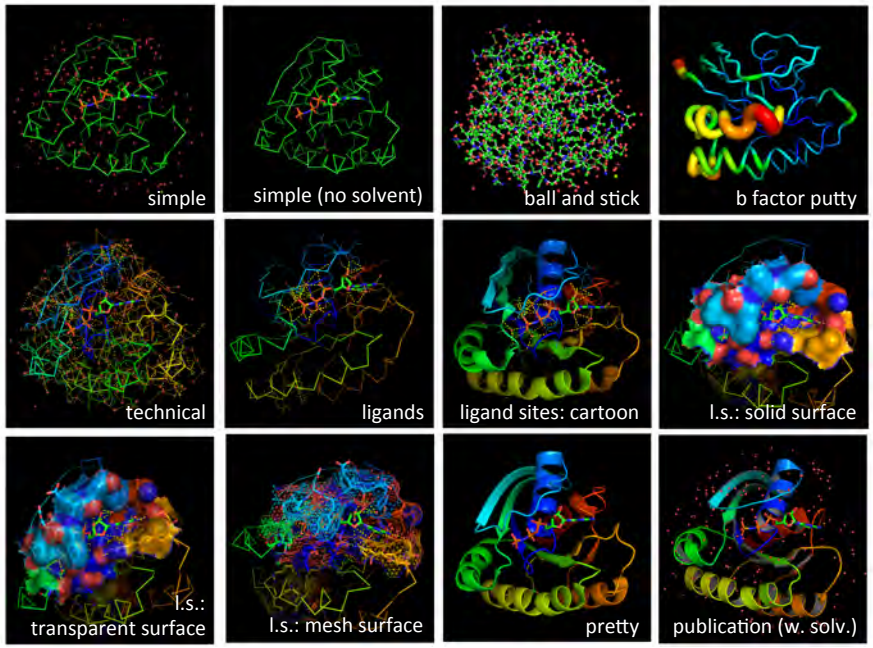
align two objects

Sculpting

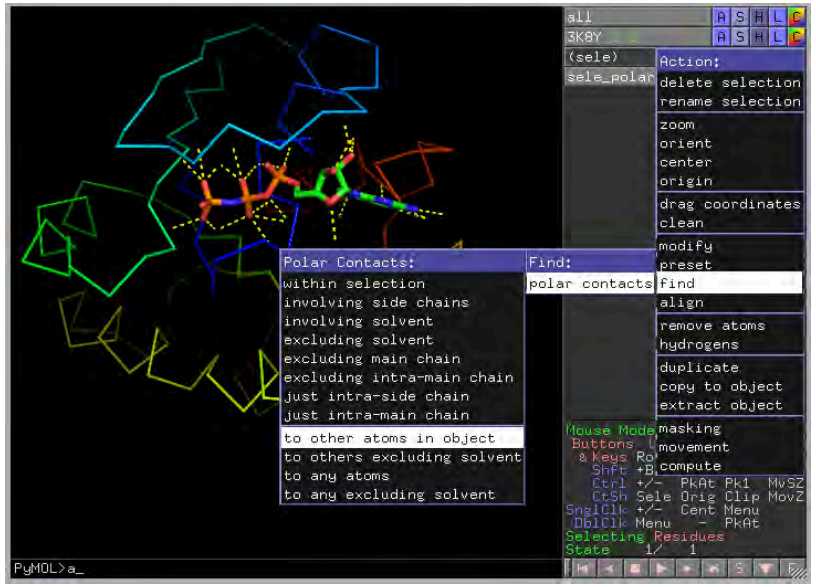
Choose “A” for “Action”: Presets



Preset representations resent a quick shortcut for various representations



Action: find: polar contacts (~hydrogen bonds)



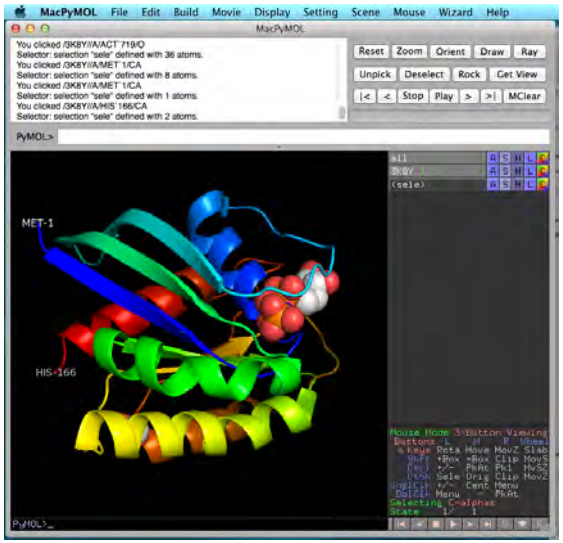
Combining Different Representations

Download 3K8Y from PDB
http://www.rcsb.org and
open 3K8Y with PyMOL

3K8Y: Hide: everything
3K8Y: Show: cartoon
3K8Y: C: spectrum: rainbow */CA
3K8Y: Show: organic : spheres
select these by left click,
sele: C: by element : CHNOS
select C-alpha atoms at either
end of the protein chain
sele: L: residues

rotate so that both ends of the
chain are clearly visible

For printing:
Display: Background: white
click on the “ray” button
File: save image: png

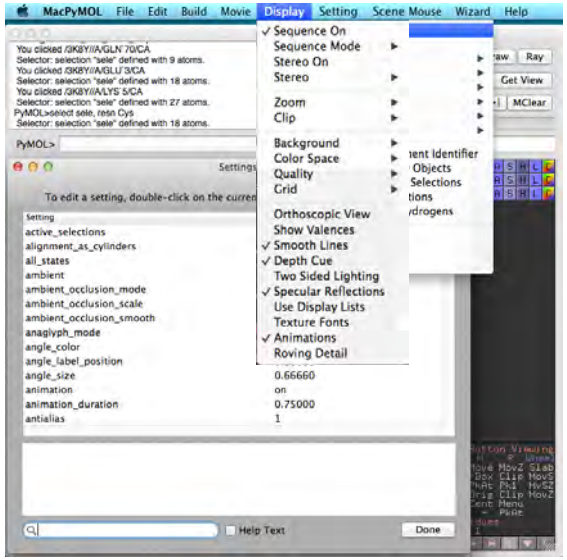


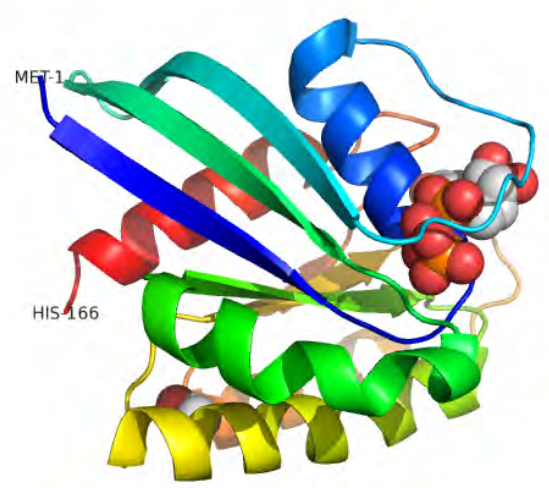
DEMO, play along

Fine-tuning the Figure: Display & Settings

Style and quality of
PyMOL representations
are controlled by more
than 600 different
settings.

A few of them can be
accessed through the
“Settings” and “Display”
pull-down menus,
others only through the
“Edit all” interface or
the command line.



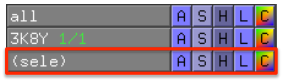


Labels are difficult to place precisely – It is often easier to add labels to the image later in a graphics program such as gimp or photoshop.

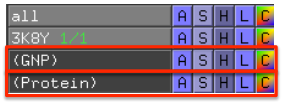
Saving Images to a File, Image Size

- By default, PyMOL images are produced at the size and resolution of the PyMOL display area. You can set the display area to a precise size with the "viewport" command (command line):
  viewport 1200, 900
- This default is overridden if you specify the desired number of pixels when calling the draw or ray command (in the command line):
  draw 1200, 900 (fast rendering, lower quality)
  or
  ray 1200, 900 (slow rendering, high quality, shadow effects)
- The resulting image can be saved through the "File" pull-down menu or with the command line, this can also be used to specify size and resolution:
  png filename[, width[, height[, dpi[, ray[, quiet]]]]]
  png my_image.png, 1200, 900, 300, 1

Using Named Selections

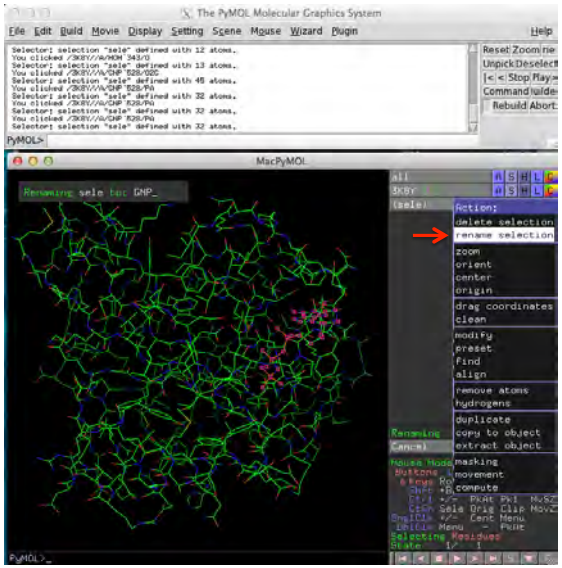


temporary selection changes whenever you select something new

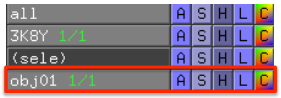


permanent shortcut to selections of ligand and protein

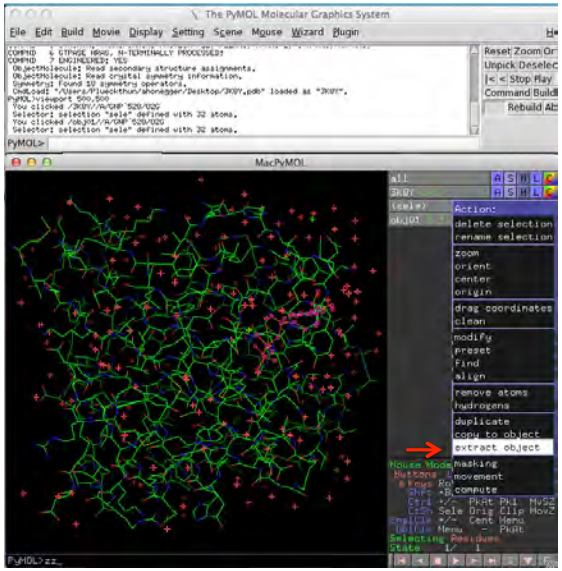
(GNP) and (Protein) are still part of object 3K8Y



Extracting Selection to a Separate Object



The coordinates of the ligand have been made into an independent object, they are no longer part of object 3K8Y



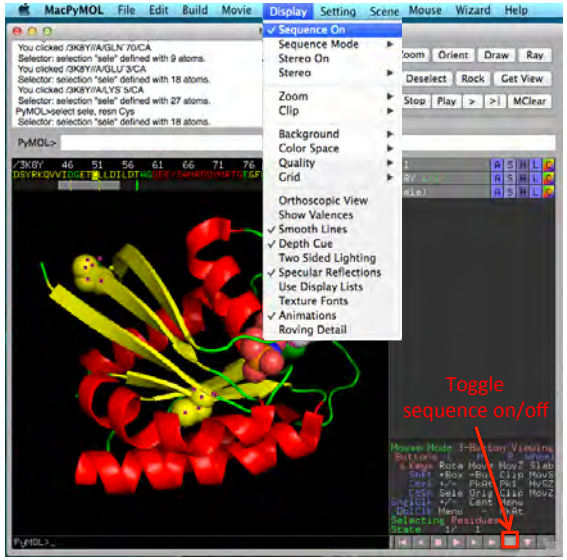
Sequence Display

Coloring of the model is reflected in the sequence

Residues can be selected by left-click in the sequence

Residues selected in the structure are highlighted in the sequence

Vertical bars in the sequence slider indicate the location of selected residues in the sequence



Working with Multiple Models

Load structures **1CTQ.pdb, 1Q21.pdb, 1QRA.pdb, 3K8Y.pdb, 3TGP.pdb** into PyMOL

Click the **"zoom" button** to fit all structures on the screen

For the 3K8Y object, hit **Action:align:all** for a sequence-based structural alignment of the five structures

Click the "**zoom" button** to fit all structures on the screen

Show all structures as **ribbon** only (all – hide:everything, all:show ribbon)

By clicking on the name of each object, you can enable or disable the display of that object without losing its representation

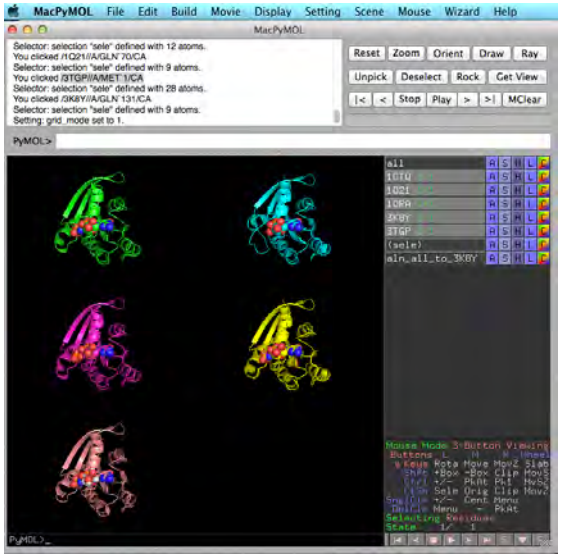
Which of the objects differs most from the others?
If you click on one of the atoms, its full name is listed in the feedback window.
e.g. /3TGP//A/MET`1/CA is the Cα in residue 1 (methionine) in chain A of object 3TGP

Now compare the ligands of the different structures (all – show:organic-sticks)

Do you find any difference correlating with the difference in main chain structures?

DEMO, play along

Side-by-side comparison: Arrange Objects in Grid



Demo and Exercise:

- **Find and download the structure of neurotensin receptor type 1 Seven structures are available, which one should you use?**
  http://www.rcsb.org
- What are the key differences between these structures? Align and compare the structures.
- Isolate the structure of one receptor/ligand complex
- Prepare a figure that illustrates the ligand in its binding pocket.
- Prepare a figure that illustrates the locations of the cysteine residues within the receptor. Are any involved in disulfide bonds? Which ones are located in a hydrophobic environment, which ones are exposed to the solvent?
- Mutate the free Cys to Ala or Val

Information on the PDB website

- Species / Variant / Link to UniProt sequence annotations
- Type of construct
  (domain fusions, stabilizing mutations, loop deletions)
- Resolution, method, refinement statistics, validation report
- Crystallization conditions
- Presence of ligands or inhibitors, proteinacious binding partners
- Biological sequence vs. theoretical sequence vs. residues visible
  (missing loops, terminal sequences)
- Biological assembly vs. asymmetric unit
- Related PDB entries
- Literature links

Inspecting a PDB file in a Text Editor

- PDB files are **plain text files** that can be viewed and edited in a text editor.
- They contain a **Header** that contains general information about the molecule: type of molecule, literature reference, resolution and refinement statistics, theoretical sequence (SEQRES), non-standard chemical groups (HETATM), secondary structure, SS-bonds, active sites, crystallographic symmetry parameters.
- **ATOM** (standard protein and nucleic acids) and HETATM (all other chemical compounds) records contain the actual coordinates for each atom of the molecule:

ATOM 1 N ASP L 1 29.392 -8.290 26.733 1.00 42.11 N

- For very high resolution structures, ANISOU records may specify anisotropic temperature factors.

ANISOU 1 N ASP L 1 4820 5485 5696 878 -962 -1205 N

- The most important of these informations are listed on the RCSB page for the structure

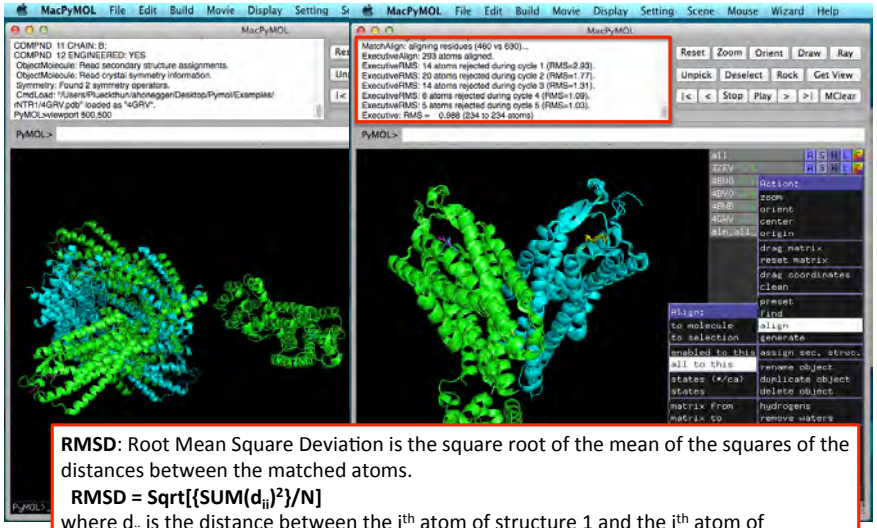
Inspecting a PDB file in PyMOL

- Completeness – entire molecule of interest or isolated domain?
- Additional fused domains to improve production, crystallization
- One or more copies of the molecule in the asymmetric unit?
- One or more chains in the biological unit?
- Overall topography, secondary structure?
- Chain breaks, missing loops and ends?
- Non-protein components: Ligands, Ions, etc.?
- Crystal contacts, b-factors?
- Structural divergence between related structures

Demo and Exercise:

- Find and download the structure of neurotensin receptor type 1
  Five structures are available, which one should you use?
  http://www.rcsb.org
- **What are the key differences between these structures?**
  Align the structures in PyMOL and compare.
- Isolate the structure of one receptor/ligand complex
- Prepare a figure that illustrates the ligand in its binding pocket.
- Prepare a figure that illustrates the locations of the cysteine residues within the receptor. Are any involved in disulfide bonds? Which ones are located in a hydrophobic environment, which ones are exposed to the solvent?
- Mutate the free Cys to Ala, Ser or Val

Sequence-based Structural Superposition



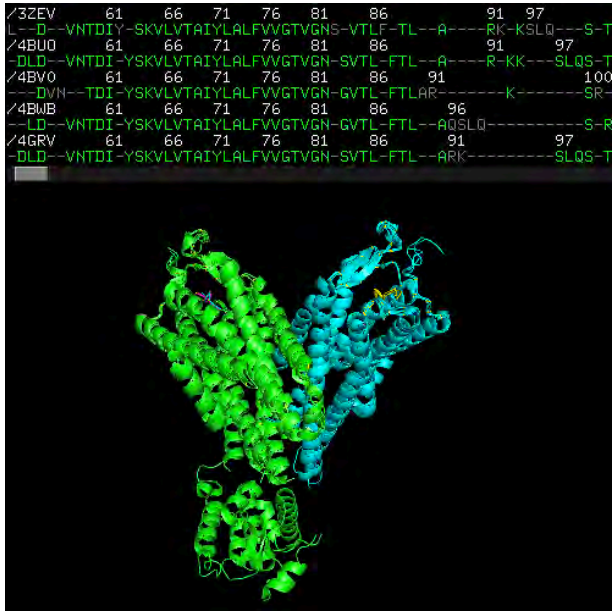
RMSD: Root Mean Square Deviation is the square root of the mean of the squares of the distances between the matched atoms.

RMSD = Sqrt[{SUM(d_{ii})^2}/N]

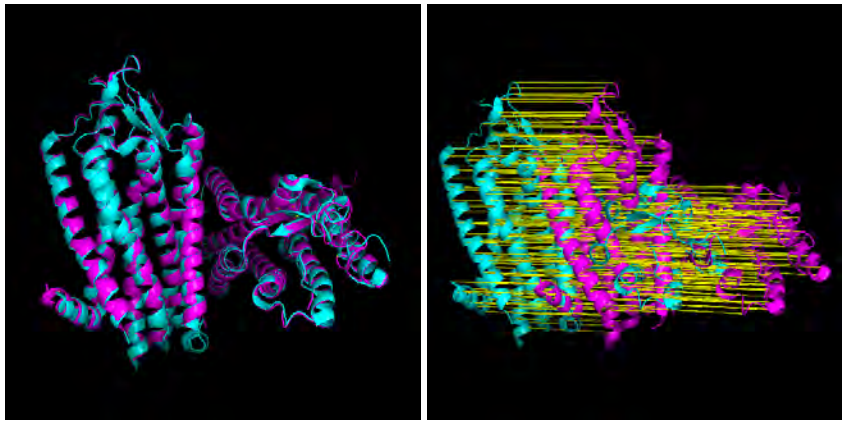
where d_{ii} is the distance between the ith atom of structure 1 and the ith atom of structure 2 and N is the number of atoms matched in each structure.

3D-superposition based on sequence alignment

Works well for close homologs, not so well for more distant relatives



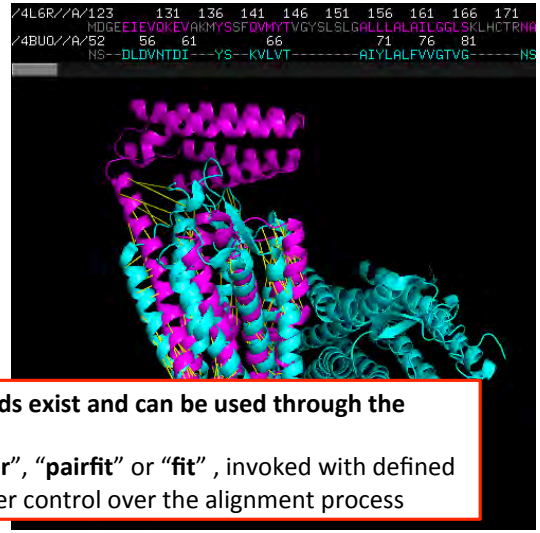
Alignment of structures 4BUO-4BV0



In a good alignment, the lines indicating which atom was superimposed to which only become visible when one moves the aligned structures apart and re-draws the alignment object

If align does not give reasonable results

In this alignment of the glucagon receptor (4L6R) to the rat neuro-tensin receptor 1 (4BUO) the sequence similarity was too low for a good sequence alignment, resulting in a bad residue pairing for the structural alignment



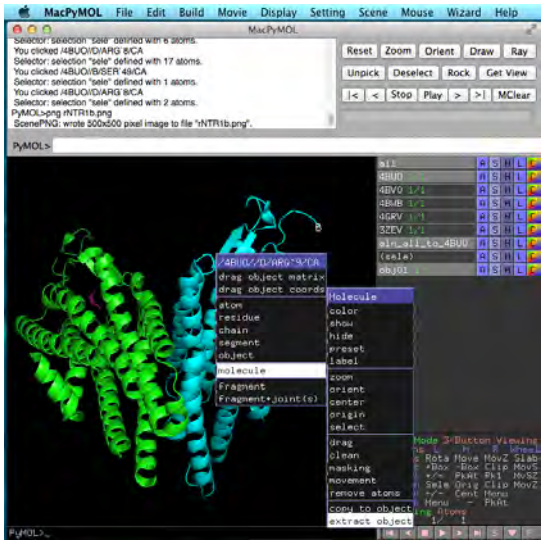
Other alignment methods exist and can be used through the command line:

"**cealign**", "**align**", "**super**", "**pairfit**" or "**fit**" , invoked with defined atom selections for better control over the alignment process

Demo and Exercise:

- Find and download the structure of neurotensin receptor type 1
  Five structures are available, which one should you use?
  http://www.rcsb.org
- What are the key differences between these structures?
  Align and compare the structures.
- **Isolate the structure of a single receptor/ligand complex**
- Prepare a figure that illustrates the ligand in its binding pocket.
- Prepare a figure that illustrates the locations of the cysteine residues within the receptor. Are any involved in disulfide bonds? Which ones are located in a hydrophobic environment, which ones are exposed to the solvent?
- Mutate the free Cys to Ala, Ser or Val

Isolating chain B and D of 4BUO



If you separate out chain B and D in this way, you get chain B in two objects, because there are unresolved residues in the first intracellular loop.

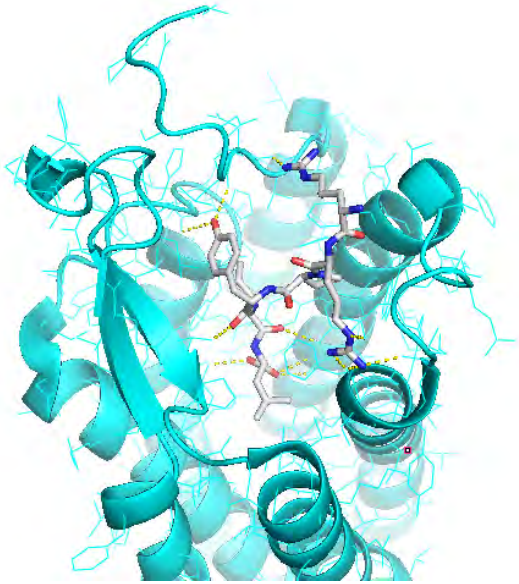
Alternatively, you can delete chains A and C from object 4BUO, then cut-and-paste chain C to a new object

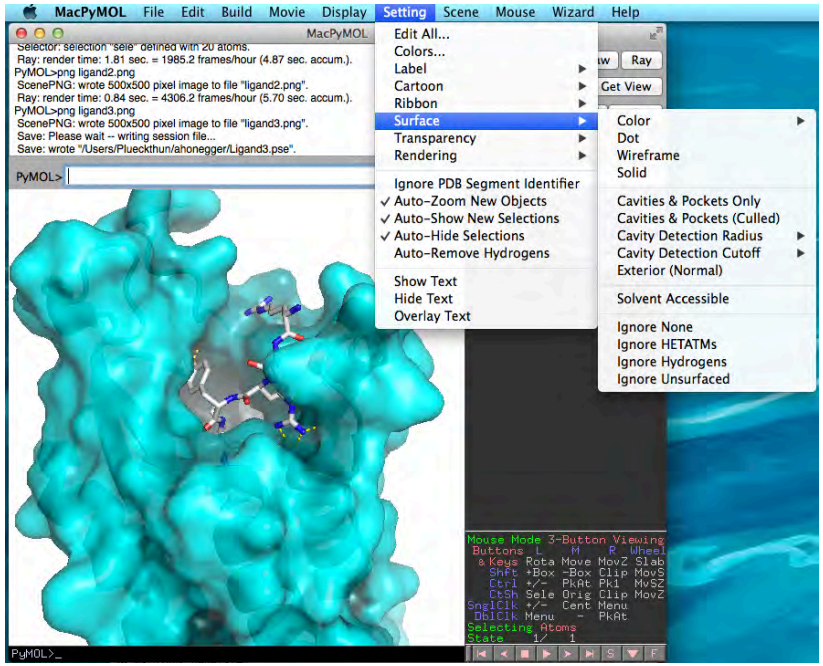
Or you can use a text editor to delete chain A and B from the PDB file

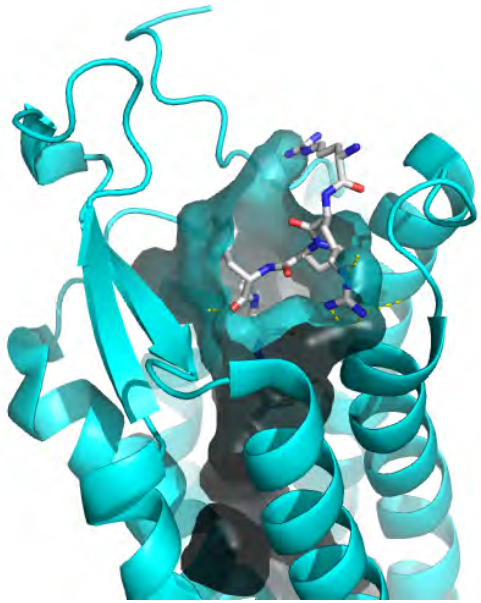
Or you can set selection to chain, select in sequence view, then copy-paste or cut-and-paste to a new object

Demo and Exercise:

- Find and download the structure of neurotensin receptor type 1
  Five structures are available, which one should you use?
  http://www.rcsb.org
- What are the key differences between these structures?
  Align and compare the structures.
- Isolate the structure of a single receptor/ligand complex
- **Prepare a figure that illustrates the ligand in its binding pocket**.
- Prepare a figure that illustrates the locations of the cysteine residues within the receptor. Are any involved in disulfide bonds? Which ones are located in a hydrophobic environment, which ones are exposed to the solvent?
- Mutate the free Cys to Ala, Ser or Val

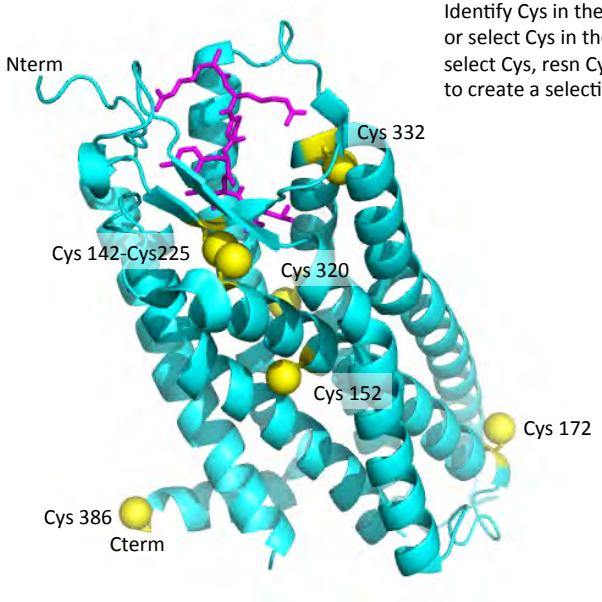






Demo and Exercise:

- Find and download the structure of neurotensin receptor type 1
  Five structures are available, which one should you use?
  http://www.rcsb.org
- What are the key differences between these structures?
  Align and compare the structures.
- Isolate the structure of one receptor/ligand complex
- Prepare a figure that illustrates the ligand in its binding pocket.
- **Prepare a figure that illustrates the locations of the cysteine residues within the structure. Are any involved in disulfide bonds? Which ones are located in a hydrophobic environment, which ones are exposed to the solvent?**
- Mutate the free Cys to Ala, Ser or Val



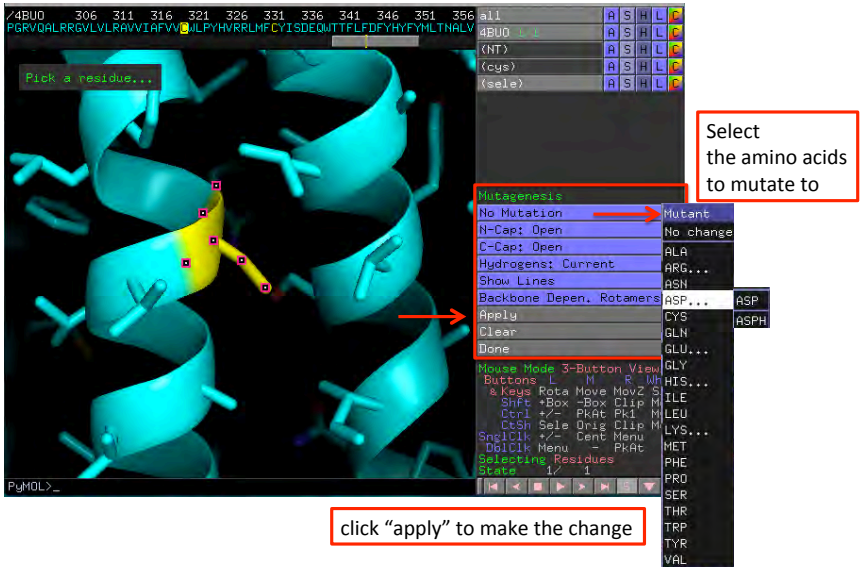
Identify Cys in the sequence display
or select Cys in the command line:
select Cys, resn Cys
to create a selection object called Cys

Demo and Exercise:

- Find and download the structure of neurotensin receptor type 1
  Five structures are available, which one should you use?
  http://www.rcsb.org
- What are the key differences between these structures?
  Align and compare the structures.
- Isolate the structure of one receptor/ligand complex
- **Prepare a figure that illustrates the ligand in its binding pocket and the membrane exposed surfaces of the receptor.**
- Prepare a figure that illustrates the locations of the cysteine residues within the structure. Are any involved in disulfide bonds? Which ones are located in a hydrophobic environment, which ones are exposed to the solvent?
- **Mutate the free Cys to Ala, Ser or Val**

Working with Wizards: Mutagenesis

Working with Wizards: Mutagenesis



Working with Wizards: Mutagenesis

Clean up the Neighborhood (local minimization)

select mutated residue

(for the selection) Action:Modify:Expand:by 5 Å

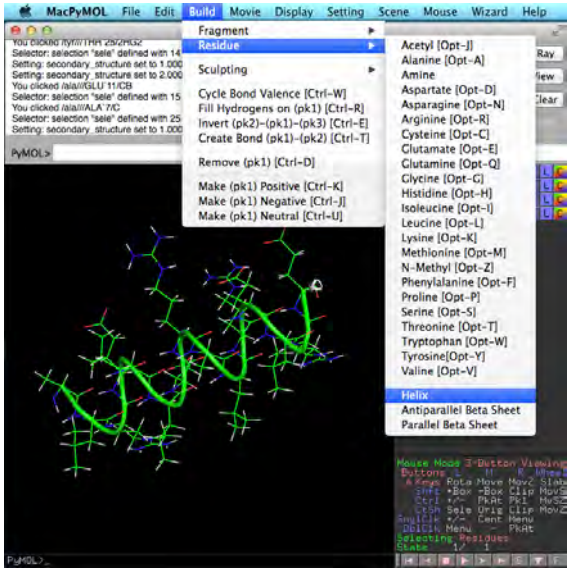
(for the selection) Action:Clean

Uses the Merck Molecular Force Field (MMFF)

The SIB (Swiss Institute of Bioinformatics) offers a PyMOL plugin for download that enables you to mutate any residue of a PDB structure into one of the non-natural L- or D-sidechains of the SwissSidechain database.

http://www.swisssidechain.ch/visualization/pymol.php

Modelling a Peptide from Scratch



Select the desired secondary structure
e.g. Build:Residue:Helix

Type the desired sequence while keeping the <alt> key pressed

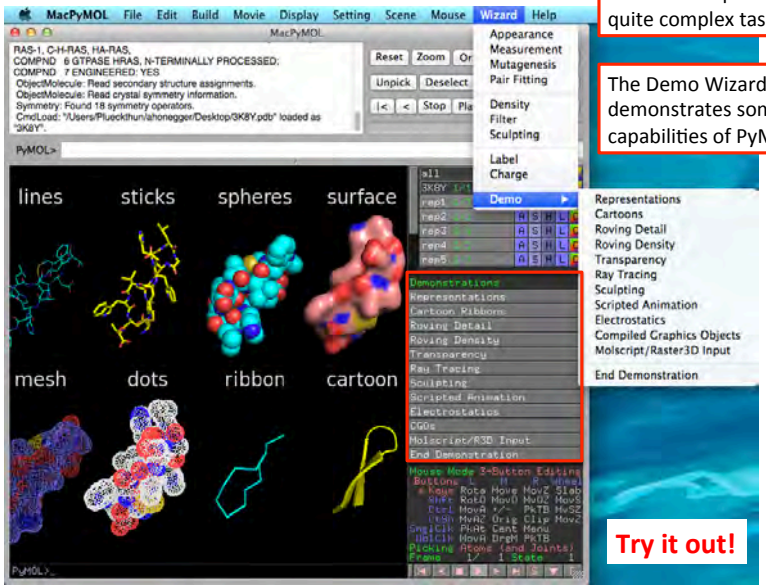
Program bug on Mac:
(CH) <alt>G results in an "@" added to the command line, not in a Gly added to the peptide. Similar problem for <alt>L on German keyboard

Work-around:
Model peptide with Ala, use mutagenesis wizard to substitute Ala by Gly

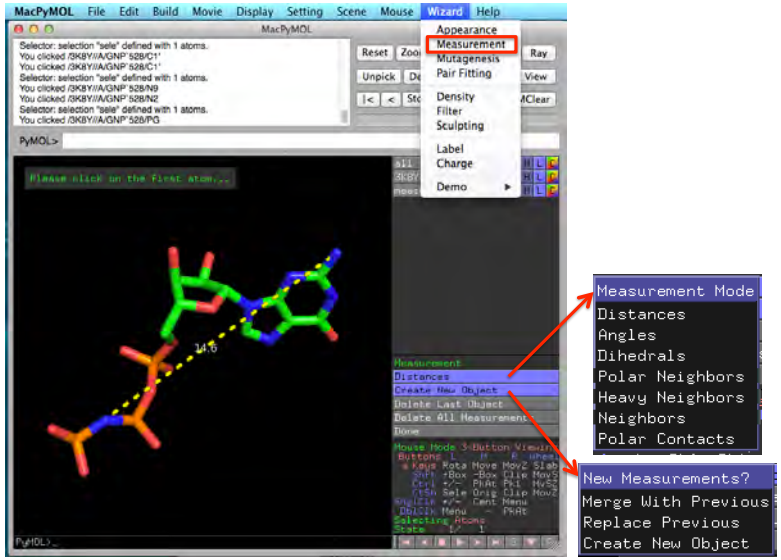
Working with Wizards

Wizards are PyMOL add-ons that can perform quite complex tasks

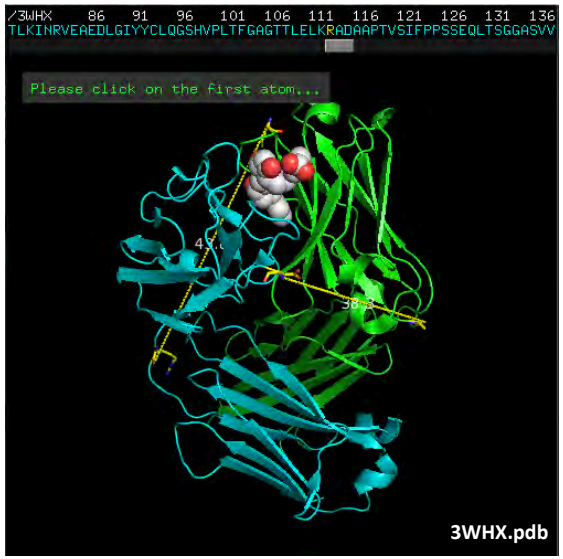
The Demo Wizard demonstrates some of the capabilities of PyMOL



Working with Wizards: Measurements



Application: Linker Length



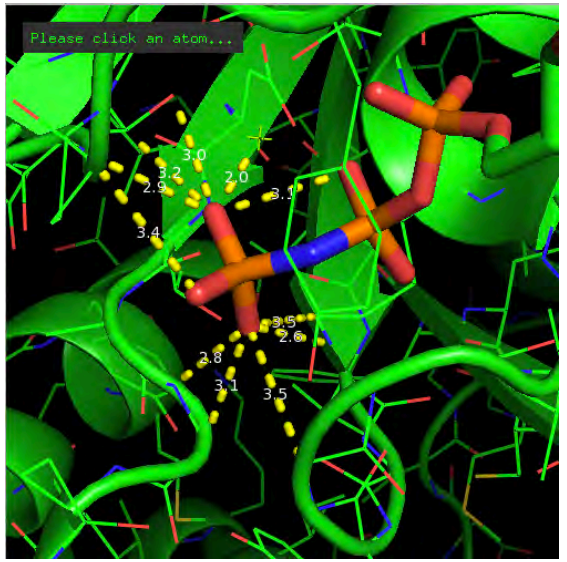
Example:
You want to know how long a peptidic linker you need to either connect the C-terminal end of the VL domain in an antibody Fab fragment to the N-terminal end of VH or vice versa.
Highlight the residues want to connect and measure the **Distance**:
Cterm VL – Nterm VH 43.8 Å
Cterm VH – Nterm VL 38.3 Å

repeat for several structures

Rule of thumb: Linker needs to be long enough to connect ends in a semi-circle of AA in beta-strand conformation:
n AA = (d*π/2)/3.5
n AA = 0.45 *d

17-20 Aminosäuren

Working with Wizards: Measurements



Example:
You want to know what atoms make contacts to the gamma phosphate group of GPPNP in the ras structure 3K8W

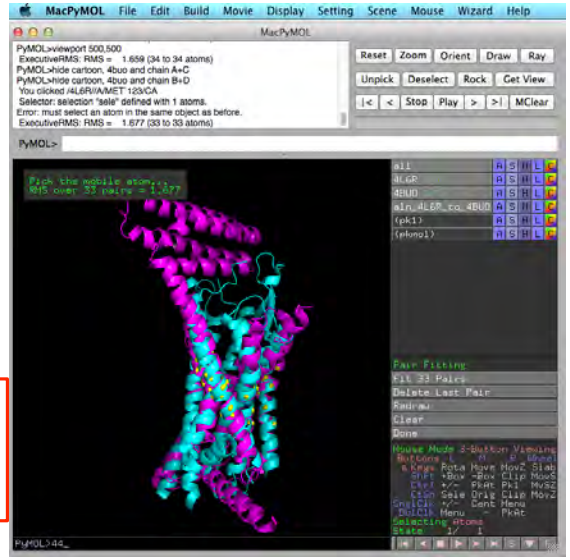
Open measurement wizard
select Neighbors: in same obj.
click on the atoms comprising the terminal phosphate group

Working with Wizards: Pair fitting

In pair_fit, the subset of atoms pairs to be used for the fit are picked by the user.

This way, an iterative improvement of the initial fit is possible, however since the user picks a small subset of atoms to fit, the alignment is more subjectiv.

Pairfit is particularely useful to align small molecules or for a rough initial alignment of key residues in a protein structure



PyMOL is Not Yet Ready for serious Modelling

Molecular sculpting works like a real-time energy minimizer, except that it isn't minimizing the energy. Instead, its just trying to return local atomic geometries (bonds, angles, chirality, planarity) to the configuration the molecules possess when they were first loaded into PyMOL.

Optimize provides a PyMOL graphical interface to some of the molecular mechanics features available in openbabel, allowing the user to optimize (minimize) the energy of any molecule uploaded on PyMOL.

Afternoon:

- **1 - Getting started:**
  - Point-and-Click: Working with the graphical user interface
  - Working with multiple models, simple alignments
- **2 – Intermediate:**
  - Using the command line, Command syntax
  - Writing a startup script for customized default settings
  - Selections and selection algebra
  - Fine-tuning display settings
  - Keeping a Log
  - .pml scripts
- **3 – Advanced: Scripting, Movies**
  - Making movies
  - About Python scripts, installing PlugIns and optional Wizards
  - Working with Wizards: Xray and Electrostatics

Intermediate PyMOL

Presentation · February 2017

DOI: 10.13140/RG.2.2.33047.57764

1 author:

Annemarie Honegger
University of Zurich
123 PUBLICATIONS **10,844** CITATIONS

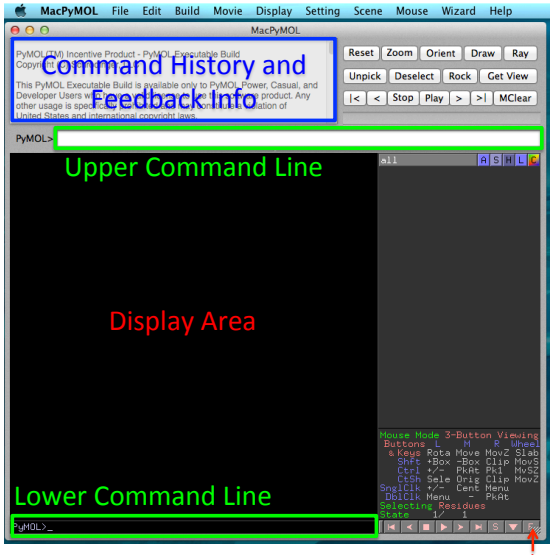
PyMOL: Intermediate

Using the command line
Keeping a command log
Commands, Selections and Settings
Defining default settings
Writing and executing simple scripts

The PyMOL Interface: Command Line (CL)

Use **up and down arrow** to scroll through command history

<ESC> toggles display of feedback text in the display area (useful when working in full-screen mode)



General Command Syntax

command parameter1[, parameter2[, parameter3]]

parameter1 is always required

square brackets denote optional Parameters

<TAB>
In the empty command line list of all commands recognized by the current version of PyMOL

c<TAB>
list of all commands that start with c

command ? (e.g. show ?)
Usage: show [representation [, selection]]

help command (e.g. help show)
DESCRIPTION
"show" turns on representations for objects and selections.
... With no arguments, "show" alone turns on lines for all bonds and nonbonded for all atoms in all molecular objects.

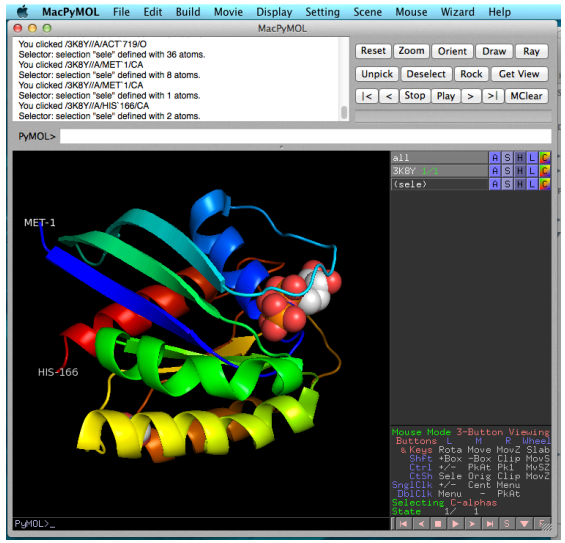
Combining Different Representations: GUI

open file 3K8Y.pdb with PyMOL

Object Menu:
3K8Y: Hide: everything
3K8Y: Show: cartoon
3K8Y: C: spectrum: rainbow */CA
3K8Y: Show: organic : spheres
select these by left click,
sele: C: by element : CHNOS
select C-alpha atoms at either end of the protein chain
sele: L: residues

rotate so that both ends of the chain are clearly visible

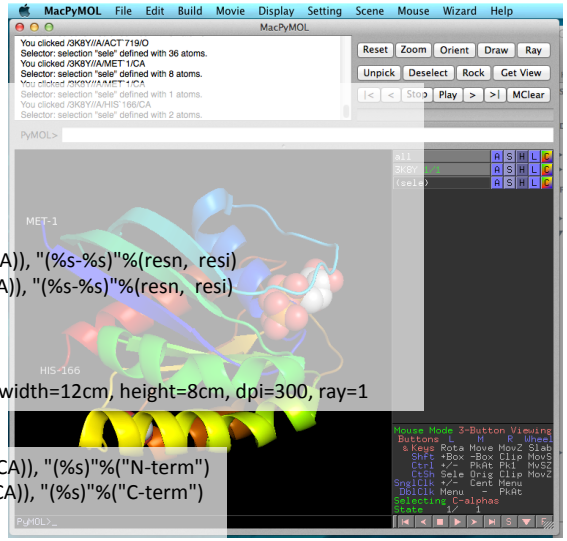
For printing:
Display: Background: white
click on the "ray" button
File: save image: png: fig1a.png



Combining Different Representations:

Command-line script to do the same:

```
load ~/pymol/pdb/3K8Y.pdb
hide all
show cartoon
util.chainbow polymer
show spheres, organic
util.cbaw organic
label (first (polymer and name CA)), "(%s-%s)"%(resn,  resi)
label (last (polymer and name CA)), "(%s-%s)"%(resn,  resi)
#rotate so that both ends of the
#chain are clearly visible
bg_color white
png ~/pymol/figures/fig1b.png, width=12cm, height=8cm, dpi=300, ray=1
```




```
label (first (polymer and name CA)), "(%s)"%("N-term")
label (last (polymer and name CA)), "(%s)"%("C-term")
set label_size, 30
set label_position,(-1.5,1,1)
```

DEMO, play along

Two Types of Scripts:

PyMOL scripts (extension .pml)**:**

Use only PyMOL commands

Commands are either in PyMOL syntax:
cartoon type, (selection)
or PyMOL API syntax:
cmd.cartoon(string type, string selection)

Can either be copy-pasted into the command line (whole or in segments) or by
@ path/scriptname.pml
and are executed immediately

This course deals only with .pml scripts

Python scripts (extension .py)

Use the **Python programming language** and can access functionalities of Python libraries (NumPy, SciPy, ChemPy, cctbx, OpenBabel …) and interact with external command-line driven programs, e.g. APBS, Caver, etc..

Always contain at least this line near top:
from pymol import cmd
additional similar lines indicate other dependencies, e.g.
from cctbx import sgtbx, uctbx
Commands only in the pymol API syntax:
cmd.cartoon(string type, string selection)

PlugIns are installed through the **PlugIn manager** or are imported by
run path/scriptname.pml
They introduce new commands defined by **cmd.extend**("cmd_name", python_function) that can be accessed through the command line or in some cases from the GUI.

PyMOL command line

More than 300 different commands in PyMOL
Scripts and PlugIns further expand the repertoire
More than ~~700~~ **1400** different setting variables modify the effects of these commands

Only a fraction of these can be accessed through the GUI

Use of the command line allows:
- much better **control of atom selections**
- access to **all commands** and their **parameters**
- **keeping a log** of applied commands and parameters
- **command sequences can be prepared as a text file** (script) and copied to the command line or called by other scripts
- adaptation and **re-use of scripts**
- automation

Nobody knows all these commands by heart

Always keep the PyMOL Wiki at hand!

http://pymolwiki.org/index.php/Main_Page

Google PyMOL and <command> to find things

The PyMOL Interface:

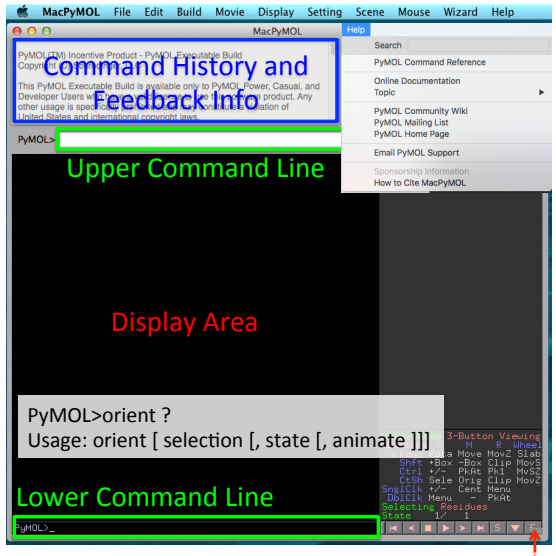
Enter **<TAB> in empty CL** for a list of all commands

<TAB> in non-empty CL: autocompletiton
e.g. “ori”<TAB> orient, origin

help <command> lists full information on that command

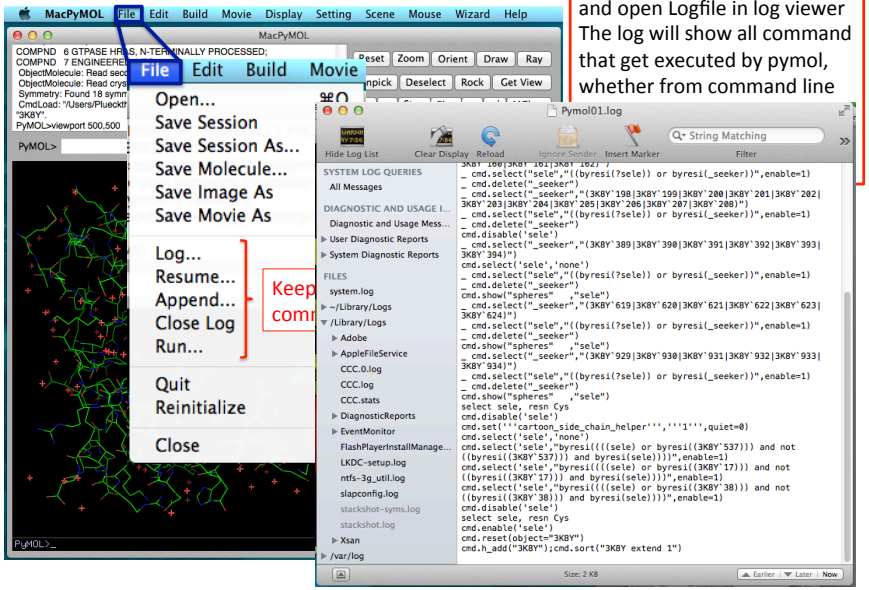
<command> ? to get a list of parameters

<ESC> toggles display of feedback text in the display area (useful when working in full-screen mode)



Keep a Log

Pymol Tricks
After opening Pymol, save log and open Logfile in log viewer
The log will show all command that get executed by pymol, whether from command line



Example

DEMO, play along

load filename [, object]
hide [representation [, selection]]
show [representation [, selection]]

Command:	Effect:
load ~/pymol/pdb/3K8Y.pdb	loads structure 3K8Y.pdb from folder pdb
	4K8Y is shown in “lines” representation
hide	nothing displayed
show cartoon, 3K8Y	4K8Y is shown in cartoon representation
show spheres, organic	ligands GNP and ACT are shown as spheres
hide spheres, resn ACT	ligand ACT is no longer visible
show surface, polymer	protein is shown in surface representation
#now use the mouse to orient the molecule into a pleasing view	

color color [, selection]
png filename [, width [, height [, dpi [, ray]]]]
save filename [, selection [, state [, format]]]

Command:	Effect:
color white, polymer	the color of the protein is changed to white
png ~/pymol/figures/fig1a.png	save image to folder “figures”
hide surface, polymer	the surface is no longer shown, we can see that the cartoon representation also changed color, although it was not visible.
png ~/pymol/figures/fig1b.png	save image to folder “figures” size and resolution are as in the display window
save ~/pymol/fig1a.pse	save the PyMOL session to pymol folder

Advanced Coloring: util.cba

color atoms by atom type: Oxygen red, nitrogen blue, sulfur yellow, hydrogen white, … the color of the carbon atom can be varied

util.cbax selection
util.cnc selection

util.cba colors atoms by atom type, carbon atoms by the color defined by the last letter in the command

util.cnc colors atoms by atom type, but does not alter the color of the carbon atoms

command	carbon color
util.cba**g**	green
util.cba**c**	cyan
util.cba**m**	light magenta
util.cba**y**	yellow
util.cba**s**	salmon
util.cba**w**	white/grey
util.cba**b**	slate
util.cba**o**	bright orange
util.cba**p**	purple
util.cba**k**	pink

util.chainbow object

colors each chain in the object in a rainbow of colors, from Nterm:blue to Cterm: red

util.cbc [object]

colors each chain in a different color

util.cbss("object", "helixcolor", "sheetcolor", "coilcolor")

colors object by secondary structure, if the secondary structure of the object is poorly defined, use command **dss selection** to re-assign secondary structure

Defining your own colors

The color names used by pymol are documented here:
http://www.pymolwiki.org/index.php/Color_Values

You can list the colors used in a selection by this command:

iterate all, print color

You can define your own color names and associated colors by their RGB values

set_color dblue, [0.05 , 0.19 , 0.57] values between 0 and 1
set_color dblue, [13 ,48 , 146] values between 0 and 255

(Or: select menu settings/color, enter a new color name in the name field and adjust the colors with the sliders. This can also be used to adjust the colors used for different elements. However, if you write a script, you can reproduce your color scheme across different PyMol sessions)

Read the PyMOL wiki on the "spectrum" command to see how you can generate and apply color gradients

Write the Command Sequence to a Text File

On the Mac, TextWrangler is a good free text editor
http://www.barebones.com/products/textwrangler/

```
load ~/pymol/pdb/3K8Y.pdb
hide all
show cartoon, 3K8Y
show spheres, resn GNP
show surface, polymer
color white, polymer
bg_color white
ray
png ~/pymol/figures/fig1c.png
hide surface, polymer
ray
png ~/pymol/figures/fig1d.png
save ~/pymol/fig1b.pse
```

Save as plain text file as
~/pymol/pml_scripts/MyScript1.pml

Running your script

Re-initialize or restart PyMOL and type:

@~/pymol/pml_scripts/MyScript1.pml

Were the two figures and the PyMOL file saved to your pymol folder?

Congratulations!
You've successfully generated and executed a functioning PyMOL script.

Setting the File Path

To keep PyMOL-related data together, we have placed the "pymol" data folder in the home directory, and within this folder, subfolders called "pdb", "figures", "movies" and "pml_scripts".

Find the file path to your home directory (~/)

- this is where saved files are stored by default
- this is where PyMOL is looking for files to load

On my MacBook, this would be /Users/ahonegger
Under Windows, this would be something like …

To set the PyMOL default path to the pymol folder, type in the command line:

cd ~/pymol

.pymolrc

If a PyMOL command file named "pymolrc" (visible) or ".pymolrc" (invisible) exists in your home directory, PyMOL will excute this file on start-up.

This is a convenient way to have Pymol always open with your preferred settings, e.g. default path, viewport size, background color, parameters modifying the representation, illumination etc.

Files ending on **.pml** or without suffix will be parsed as **PyMOL command files**.

Files ending on **.py** (or .pym) will be parsed as **python command files**.

If neither extension is used, PyMOL will judge based on the content of the file.

Now we only need to specify the local path:

```
load pdb/3K8Y.pdb
hide all
show cartoon, 3K8Y
show spheres, resn GNP
show surface, polymer
color white, polymer
bg_color white
ray
png figures/fig1c.png
hide surface, polymer
ray
png figures/fig1d.png
save fig1b.pse
```

Delete all instances of ~/pymol/ from the script file and run the script in PyMOL

reinitialize
@pml_scripts/MyScript1.pml

Selections

Greatly expand on the selection capabilities of the GUI

explicit selections:

select (expression)
produces a **temporary selection object** named "**sele**"

select sele, (expression)
produces a **temporary selection object** named "**sele**"

select name, (expression)
produces a **named selection object** for further use

implicit selections:

color red, (expression)
colors the residues specified by the expression red without creating a selection object

show cartoon, (expression)
displays the specified residues as cartoon, no selection object

Single word selectors

Single-Word Selector	Abbrev. Selector	Description
all	*	All atoms currently loaded into PyMOL
none		No atoms (empty selection)
hydro	h.	All hydrogen atoms currently loaded into PyMOL
hetatm	het	All atoms loaded from Protein Data Bank HETATM records
polymer	pol.	All atoms on the polymer (not het). Protein, DNA or RNA
visible	v.	All atoms in enabled objects with at least one visible representation
enabled		All atoms in enabled objects
backbone	bb.	Polymer backbone atoms
sidechain	sc.	Polymer non-backbone atoms
donors	don.	All potential hydrogen bond donors
acceptors	acc.	All potential hydrogen bond acceptors
solvent	sol.	All water molecules
organic	org.	All atoms in non-polymer organic compounds (e.g. ligands, buffers).
inorganic	ino.	All non-polymer inorganic atoms/ions.
bonded		All atoms making at least one bond
metals		All metal atoms/cations
guide		All protein CA and nucleic acid C4*/C4'
present	pr.	All atoms with defined coordinates in the current state (used e.g in creating movies)

Propery Selectors: Selecting Atoms, Residues, Chains

One word Selector	Abbrev. Selector	Description
element	e.	**chemical element**, e.g. C, N, O, H, …, Ca, Fe, Mg
name	n.	**atom name**, eg. select mainchain, n. N+CA+C+O
resn	r.	**residue name**, e.g. select neg, r. Glu+Asp
resi	i.	**residue number** e.g. select domain1, i. 33-126 if you have a negative residue number, a "\" is needed, e.g. select Ntag, i. \-5+\-4+\-3+\-2+\-1
alt	alt	**alternative conformation**, e.g. "", a, b, c
chain	c.	**chain identifier**
ss	ss	**secondary structure**, e.g. select allSTR, h+s+l+""
id	id	**atom number**
b	b	**b-factor value**, e.g. select fuzzy, b > 10
q	q	**occupancy**, e.g. select lowOccupancy, q < 0.5

Selection Macros

Shorthand for selecting specific parts of a protein or nucleic acid chain
Instead of
select name, pept1 and segi a1 and chain b and resi 142 and name ca
you can type
select name, /pept1/lig/b/142/ca
also for wildcards, ranges, multiple selections
select name, /pept1//b/142-163/n+ca+c+o
select name, /pept1//A/L* selects Leu and Lys

If you click on an atom in PyMol, the feedback window shows the selection in this form:

You clicked /3K8Y//A/GLU`3/CA
Selector: selection "sele" defined with 9 atoms.

Selection Macros

You need only the relevant part of the chain
e.g.
show spheres, CYS/CA
shows the Calpha atoms of all cysteins as spheres,
show spheres, CYS/
shows all atoms in Cystein residues as spheres

beginning with a slash:
/object-name/segi-identifier/chain-identifier/resi-identifier/name-identifier
/object-name/segi-identifier/chain-identifier/resi-identifier
/object-name/segi-identifier/chain-identifier
/object-name/segi-identifier
/object-name

or not beginning with a slash:
resi-identifier/name-identifier
chain-identifier/resi-identifier/name-identifier
segi-identifier/chain-identifier/resi-identifier/name-identifier
object-name/segi-identifier/chain-identifier/resi-identifier/name-identifier

Selection-Algebra

Selections can be combined by logical operators:

not s1	!s1	all atoms except those in selection s1
s1 and s2	s1 & s2	intersection, atoms that are both in s1 and in s2
s1 or s2	s1 | s2	union, atoms that are either part of s1 or s2

Expansion of selections

byres s1	br. s1	expands sel. from atoms to residues
bymolecule s1	bm. s1	expands sel. to molecule
bychain s1	bc. s1	expands sel. to chain
byobject s1	bo. s1	expands sel. to object
bycell s1		expands sel. to unit cell
neighbor s1	nbr. s1	directly bonded to s1, excl. s1
bound_to s1	bto. s1	directly bonded to s1, incl. s1
first, last		first or last atom in selection

Selection: Distance operators

select s3, s1 within 5.0 of s2
all atoms in s1 that are no farther than 5.0 Å from atoms in s2

select s3, s1 near_to 5.0 of s2 ((s1 within 5.0 of s2) and not s2))
all atoms in s1 that are no farther than 5.0 Å from a atoms in s2, excludes s2

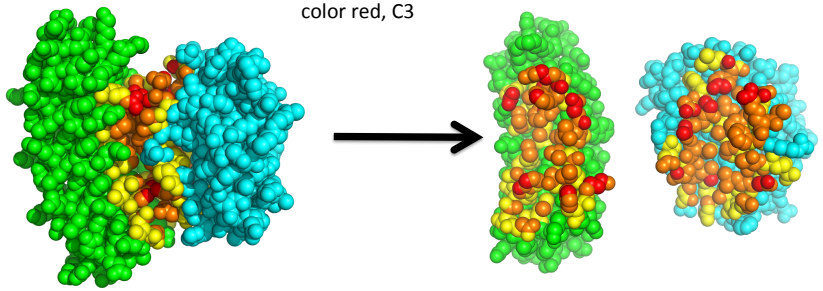
select s3, s1 beyond 5.0 of s2 ((s1 and not (s1 within 5.0 of s2))
all atoms in s1 that are farther than 5.0 Å away from atoms in s2

select s3, s2 around 5.0 ((all within 5.0 of s2) and not s2))
all atoms within 5.0 Å of s2, excluding s2

select s3, s2 expand 5.0 (all within 5.0 of s2)
all atoms in s2 or within 5.0 Å of s2

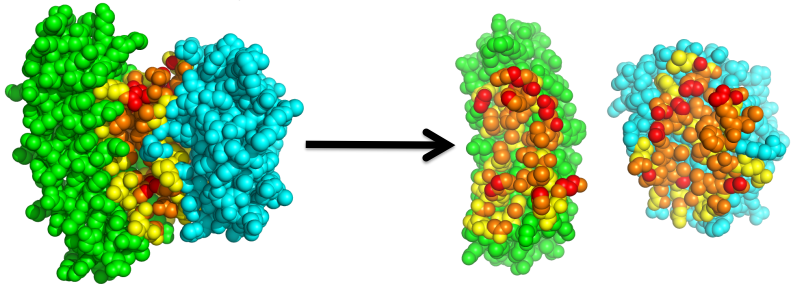
Example: Determine Contact Residues

```
# load complex and separate its components
load pdb/S4K5B_B-C.pdb, Cplx
extract darp, Cplx and chain B
extract lig, Cplx and chain C
delete Cplx

color green, darp
color cyan, lig
select C3, (darp within 3.6 of lig) or (lig within 3.6 of darp)
select C2, (darp within 5.0 of lig) or (lig within 5.0 of darp)
select C1, br. C2
color yellow, C1
color orange, C2
color red, C3
```



Listing Contact Residues

```
select C3, (darp within 3.6 of lig) or (lig within 3.6 of darp)
select C2, (darp within 5.0 of lig) or (lig within 5.0 of darp)
select C1, br. C2
```



list contact atoms

```
list=[]
iterate (C2),list.append((chain,resi,resn,name))
print list
```

list contact residues

```
list=[]
iterate (C1 and name CA),list.append((chain,resi,resn))
print list
```

Setting the Orientation of a Molecule in a Script

get_view [output [, quiet]]
set_view view

Open fig1a.pse, rotate structure into a nice orientation and type:

get_view

from the feedback window, copy-paste this to your script:

```
### cut below here and paste into script ###
set_view (\
     0.2857,   -0.0944,    0.9536,\
     0.0625,    0.9948,    0.0797,\
    -0.9562,    0.0368,    0.2902,\
     0.0000,    0.0000, -128.3235,\
   -16.5101,   63.8420,  -76.8616,\
   103.0045, 153.6425,  -20.0000 )
### cut above here and paste into script ###
```

Rotation Matrix

Origin in Camera Space

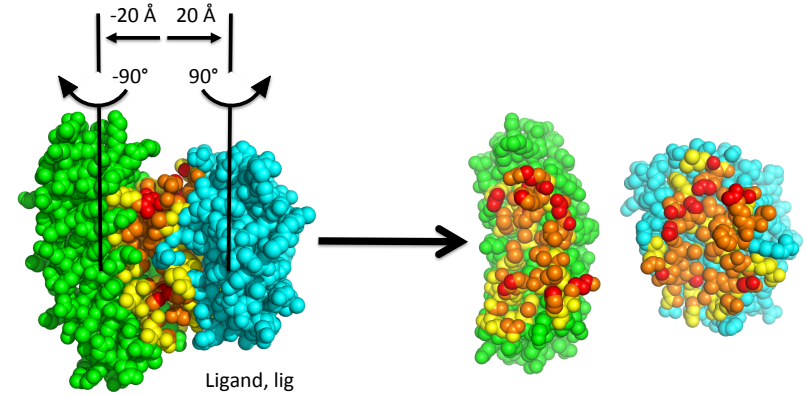
Origin in Coordinate Space

Back & Front Clipping Planes, Perspective

Defined Transformations

reset [object]
turn axis, angle
move axis, distance
orient object-or-selection [, state]
center [selection [, state [, origin [, animate]]]]
zoom [selection [,buffer [, state [, complete]]]]
clip mode, distance [, selection [, state]]
origin selection [, object [,position, [, state]]]
translate vector [,selection [,state [,camera [,object]]]]
rotate axis, angle [,selection [,state [,camera [,object [,origin]]]]]

Example: Looking at a Binding Interface



rotate y, -90, darp
translate [-20,0,0], darp
rotate y, 90, lig
translate [20,0,0], lig

Measuring Distances

distance [name [, selection1 [, selection2 [, cutoff [, mode]]]]]

name	string: name of the distance object to create
selection1	string: first atom selection
selection2	string: second atom selection
cutoff	float: longest distance to show
mode	0: all interatomic distances
	1: only bond distances
	2: only show polar contact distances
	3: like mode=0, but use distance_exclusion setting
	4: distance between centroids (*new in 1.8.2*)

Simple H-bond detection:

dist name, sele1, sele2, mode=2

dependent on parameters:

```
set h_bond_cutoff_center, 3.6
set h_bond_cutoff_edge, 3.2
```

More sophisticated H-Bond detection

load target.pdb,prot
load docked_ligs.sdf,lig

 # add hydrogens to protein

h_add prot

select don, (elem n,o **and** (neighbor hydro))
select acc, (elem o **or** (elem n **and not** (neighbor hydro)))
dist HBA, (lig **and** acc),(prot **and** don), 3.2
dist HBD, (lig **and** don),(prot **and** acc), 3.2
delete don
delete acc
hide (hydro)

hide labels,HBA
hide labels,HBD

Get information

get	get_extent	get_title
get_angle	get_position	get_type
get_area	get_property	get_version
get_bond	get_property_list	get_view
get_chains	get_renderer	get_viewport
get_dihedral	get_sasa_relative	
get_distance	get_symmetry	

get_angle [atom1 [, atom2 [, atom3 [, state [, quiet]]]]]

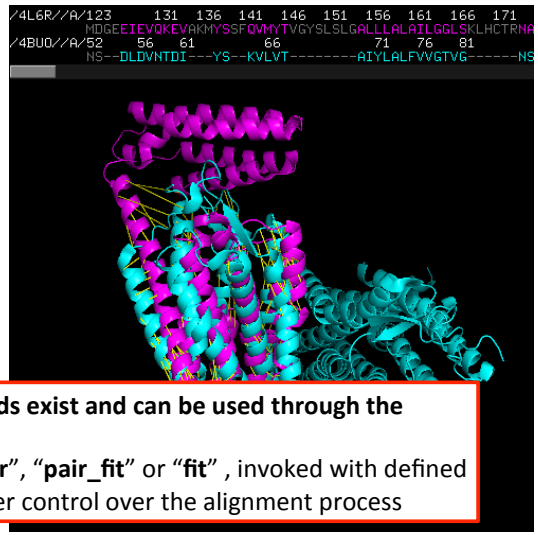
get_dihedral [atom1 [, atom2 [, atom3 [, atom4 [, state [, quiet]]]]]]
also: **phi_psi [selection [, quiet]]** to get main chain torsion angles

get_area [selection [, state [, load_b [, quiet]]]] to get the surface area of an selection

get_sasa_relative [selection [, state [, vis [, var]]]] to get per-residue relative accessibility

If align does not give reasonable results

In this alignment of the glucagon receptor (4L6R) to the rat neuro-tensin receptor 1 (4BUO) the sequence similarity was too low for a good sequence alignment, resulting in a bad residue pairing for the structural alignment



Other alignment methods exist and can be used through the command line:
"**cealign**", "**align**", "**super**", "**pair_fit**" or "**fit**" , invoked with defined atom selections for better control over the alignment process

Least Squares Superposition of two Structures

PyMOL offers several different commands for sequence-based and sequence independent structural alignments:

fit, intra_fit, pair_fit, super, align, cealign, rms, rms_cur, intra_rms, intra_rms_cur, extra_fit.py, super_all.py, align_all.py, tmalign.py

They differ in how they determine the atom pairs included in the fit, and how they treat outliers (parts of the molecule that do not fit well).

Each method will return an rmsd value (root mean squares deviation)
However, the values you get depend on the method used!

rmsd values are meaningless
if you do not indicate exactly what method and what parameters you used!!!

fit mobile, target [, mobile_state [, target_state [, quiet [, matchmaker [, cutoff [, cycles [, object]]]]]]]

Fit superimposes the model in the first selection on to the model in the second selection. Only matching atoms in both selections will be used for the fit.

- mobile = string: atom selection
- target = string: atom selection
- mobile_state = integer: object state {default=0, all states)
- target_state = integer: object state {default=0, all states)
- matchmaker = integer: how to match atom pairs {default: 0}
- -1: assume that atoms are stored in the identical order
- 0/1: match based on all atom identifiers (segi,chain,resn,resi,name,alt)
- 2: match based on ID
- 3: match based on rank
- 4: match based on index (same as -1 ?)
- cutoff = float: outlier rejection cutoff (only if cycles>0) {default: 2.0}
- cycles = integer: number of cycles in outlier rejection refinement {default: 0}
- object = string: name of alignment object to create {default: None}

Fit, Rms, Rms_Cur are finicky and only work when all atom identifiers match: segi, chain, resn, name, alt. If they don't, you'll need to use the alter command to change the identifiers to make them match -- typically that means clearing out the SEGI field, renaming chains, and sometimes renumbering.

pair_fit (selection), (selection), [(selection), (selection) [...]]

Pair_Fit fits **a set of atom pairs between two models**. Each atom in each pair must be specified individually, which can be tedious to enter manually. Script files are recommended when using this command. So long as the atoms are stored in PyMOL with the same order internally, you can provide just two selections. Otherwise, you may need to specify each pair of atoms separately, two by two, as additional arguments to pair_fit.

Useful if you want to fit, e.g. the ring systems of ligands.

Examples:
 # superimpose protA residues 10-25 and 33-46 to protB residues 22-37 and 41-54:
pair_fit protA///10-25+33-46/CA, protB///22-37+41-54/CA

 # superimpose ligA atoms C1, C2, and C4 to ligB atoms C8, C4, and C10, respectively:
pair_fit ligA////C1, ligB////C8, ligA////C2, ligB////C4, ligA////C3, ligB////C10

intra_fit (selection),state

intra_fit fits **all states of an object** (e.g. NMR) to an atom selection in the specified state. It returns the rms values to python as an array.

extra_fit [selection [, reference [, method]]]

extra_fit aligns **multiple objects** to one reference object. It can use any of PyMOL's pairwise alignment methods (align, super, cealign, fit...). More precisely it can use any function which takes arguments mobile and target, so it will for example also work with tmalign.
Additional keyword arguments are passed to the used method, so you can for example adjust outlier cutoff or create an alignment object.

rms, rms_cur, intra_rms, Intra_rms_cur
compute a RMS fit between two atom selections, but **do not transform the models after performing the fit.**

align mobile, target [, cutoff [, cycles [, gap [, extend [, max_gap [, object [, matrix [, mobile_state [, target_state [, quiet [, max_skip [, transform [, reset]]]]]]]]]]]]

align performs a **sequence alignment followed by a structural superposition**, and then carries out zero or more cycles of refinement in order to reject structural outliers found during the fit. align does a good job on proteins with decent sequence similarity (identity >30%). For comparing proteins with lower sequence identity, the super and cealign commands perform better.

cealign target, mobile [, target_state [, mobile_state [, quiet [, guide [, d0 [, d1 [, window [, gap_max [, transform [, object]]]]]]]]]]

cealign aligns two proteins using the CE algorithm. It is **very robust for proteins with little to no sequence similarity** (twilight zone). For proteins with decent structural similarity, the super command is preferred and with decent sequence similarity, the align command is preferred, because these commands are much faster than cealign.

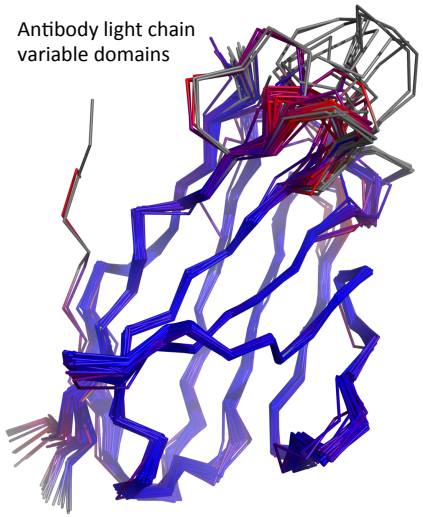
super mobile, target [, cutoff [, cycles [, gap [, extend [, max_gap [, object [, matrix [, mobile_state [, target_state [, quiet [, max_skip [, transform [, reset [, seq [, radius [, scale [, base [, coord [, expect [, window [, ante]]]]]]]]]]]]]]]]]]]]]]

super aligns two selections. It does a **sequence-independent** (unlike align) **structure-based dynamic programming alignment** followed by a series of refinement cycles intended to improve the fit by eliminating pairing with high relative variability (just like align). super is more robust than align for proteins with low sequence similarity.

Python Scripts offer additional Functionalities:

run py_scripts/colorbyrmsd.py

colorbyrmsd 4d3c, 2x7l
colorbyrmsd 4ht1, 2x7l
colorbyrmsd 5ds8, 2x7l
colorbyrmsd 5dtf, 2x7l
colorbyrmsd 5dub, 2x7l
colorbyrmsd 4ma3, 2x7l
colorbyrmsd 4o4y, 2x7l
colorbyrmsd 4jo3, 2x7l
colorbyrmsd 4jo4, 2x7l
colorbyrmsd 4o51, 2x7l
colorbyrmsd 4hbc, 2x7l
colorbyrmsd 5i8o, 2x7l
colorbyrmsd 4jo1, 2x7l
…
hide all
show ribbon



PyMOL Settings

Style and quality of PyMOL representations are controlled by more than ~~600~~ 1400 different settings …

General Syntax for Settings

set name [, value [, selection [, state [, updates [, log [, quiet]]]]]]

"set" is a command

the command "set" assigns a value to named variable

dependent on the setting, one or more additional parameters are required
for boolean variables (on/off or 0/1), no parameter means "on"
some settings are global (default), others can be applied to a selection.

set<TAB>
parser: matching commands:

set	set_dihedral	set_property
set_atom_property	set_geometry	set_symmetry
set_bond	set_key	set_title
set_color	set_name	set_view

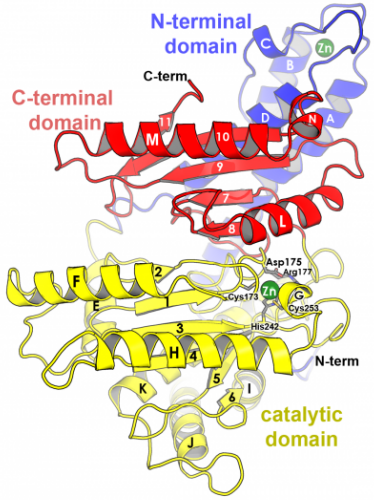
set <TAB>
list of all settings set by "set" recognized by the current version of PyMOL

set ?
Usage: set name [, value [, selection [, state [, updates [, log [, quiet]]]]]]

Settings define the Style of the Figure

```
set antialias = 1

set ambient=0.3
set direct=1.0

set ray_trace_mode=1

set stick_radius = 0.2
set cartoon_tube_radius, 0.2
set cartoon_fancy_helices=1
set cartoon_cylindrical_helices=0
set cartoon_flat_sheets = 1.0
set cartoon_smooth_loops = 0
set cartoon_highlight_color =grey50

bg_color white

set_color maarine=  [0.3, 0.8, 1.0]
set_color graay=[0.8,0.8,0.8]
set_color greeen=[0.0,0.5,0.0]
```

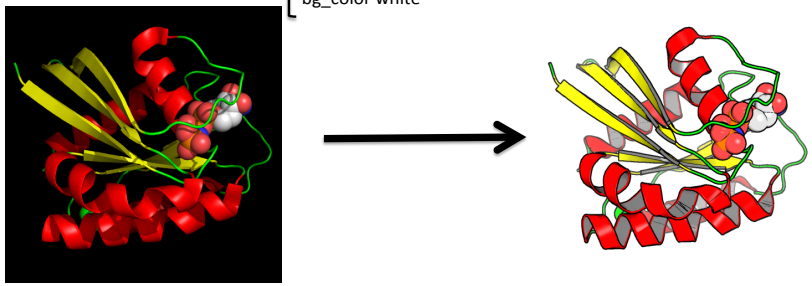
copy



PLoS_script1.pml

paste

```
set antialias = 1
set ambient=0.3
set direct=1.0

set ray_trace_mode=1

set stick_radius = 0.2
set cartoon_tube_radius, 0.2
set cartoon_fancy_helices=1
set cartoon_cylindrical_helices=0
set cartoon_flat_sheets = 1.0
set cartoon_smooth_loops = 0
set cartoon_highlight_color =grey50

bg_color white
```

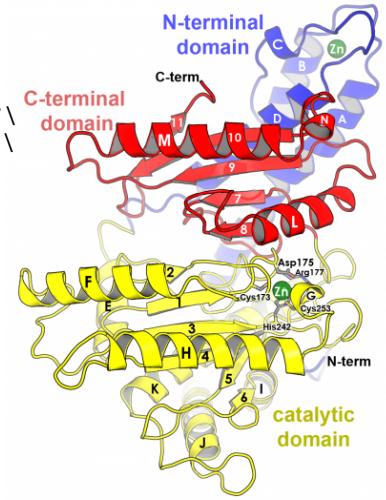


These commands define what's shown

```
load csos3_18o_nobreak.pdb, csos3
hide all
show cartoon
show sticks, (resid 173 or resid 175 or resid 177 \
      or resid 242 or resid 253) and not (name n \
      or name c or name o)
show spheres,  elem ZN
```

and how it's colored

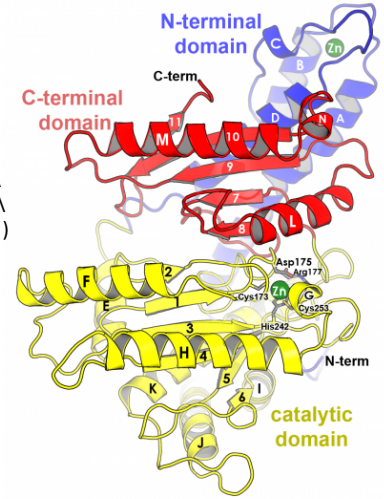
```
color gray50, elem C
color greeen, elem ZN
color blue, resid 38:147 and name ca
color yellow, resid 148:397 and name ca
color red, resid 398:514 and name ca
```



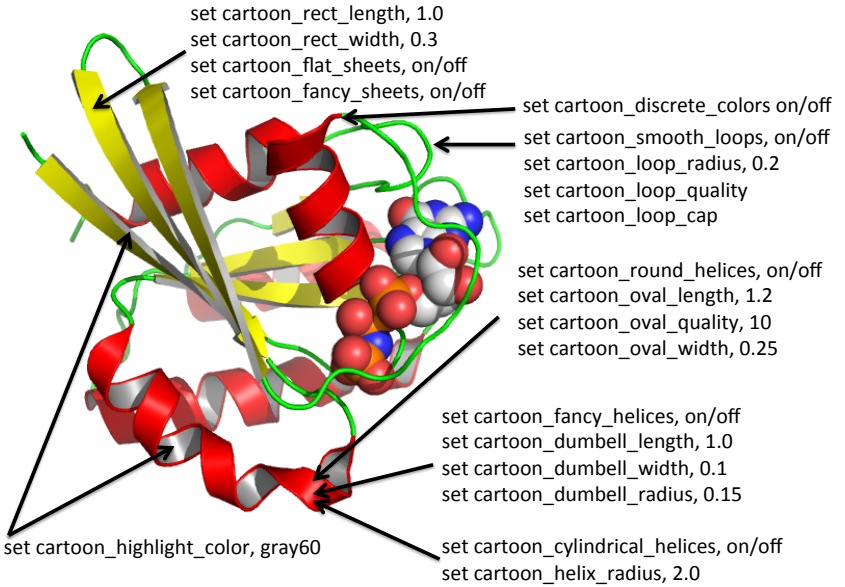
Carbonic anhydrase, CsoS3,
from Halothiobacillus neapolitanus.
(PDB ID 2G13)

These commands define the orientation

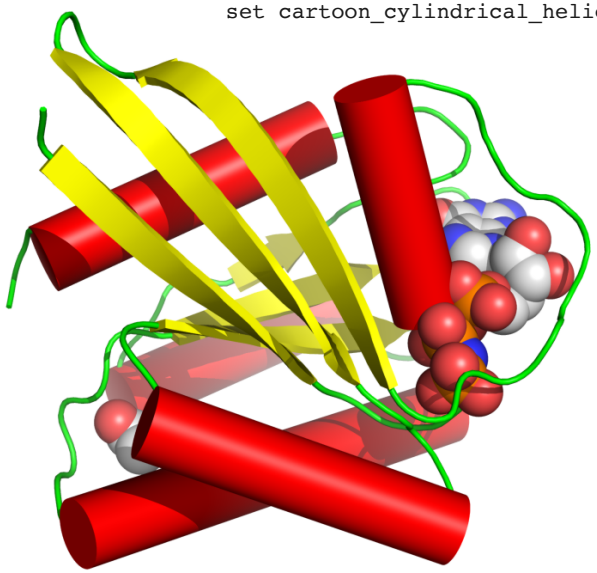
```
set_view (\
    0.091340274,   -0.606698275,    0.789650559,\
   -0.991202235,   -0.131515890,    0.013612081,\
    0.095602803,   -0.783963323,   -0.613382638,\
    0.001799395,    0.001679182, -246.492980957,\
   12.976243019,   41.245639801,   62.928291321,\
  187.538497925,  249.492980957,    0.000000000 )
turn y, 3.5


# and generate the figure
viewport 1200,1500
ray
png csos3-left.png

# labels were added in a generic graphics program
# e.g. Photoshop, Illustrator ...
```



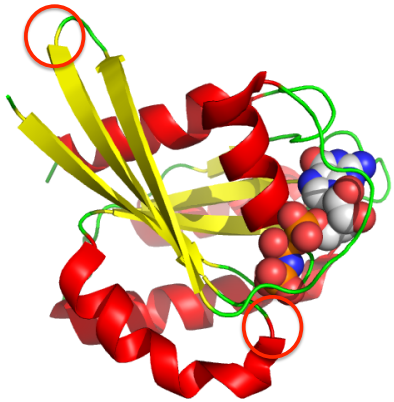
Settings: Cartoon representation



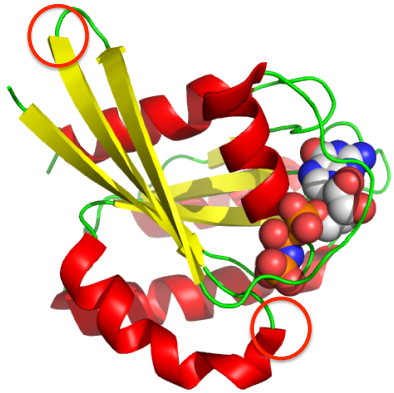
```
set cartoon_cylindrical_helices, on
```



Settings: cartoon_discrete_colors



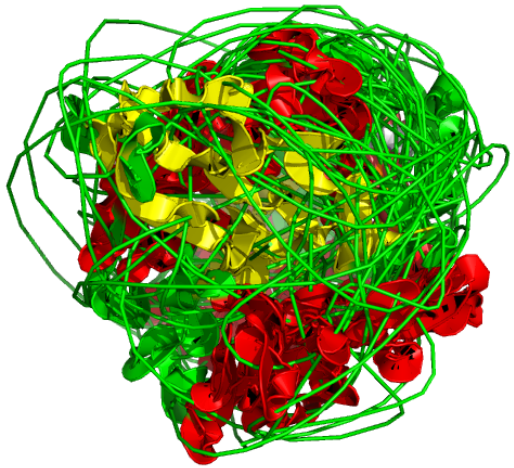
set cartoon_discrete_colors , off



set cartoon_discrete_colors , on

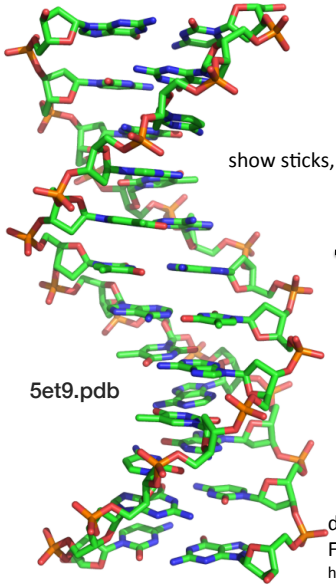
Stops secondary structure colors from bleeding into the coil areas

When things going haywire



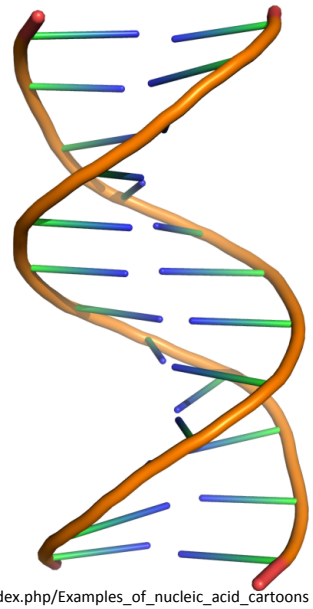
"set cartoon_trace, 1" seems to confuse PyMOL if the structure contains more than just Calpha atoms

Nucleic Acid Cartoons



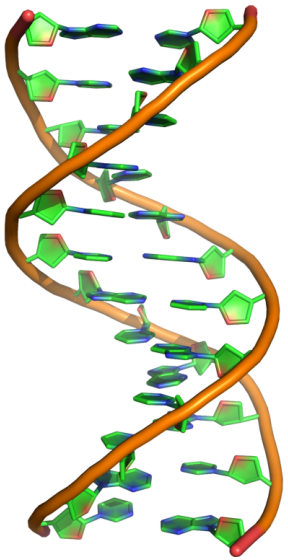
show sticks, 5et9

5et9.pdb

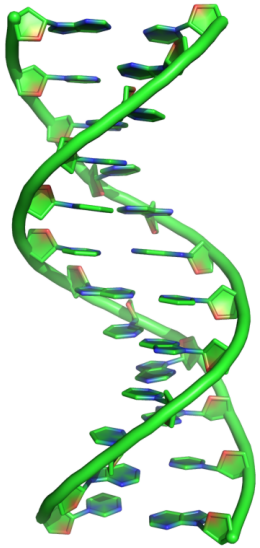


default cartoon view
For other options, see:
http://www.pymolwiki.org/index.php/Examples_of_nucleic_acid_cartoons

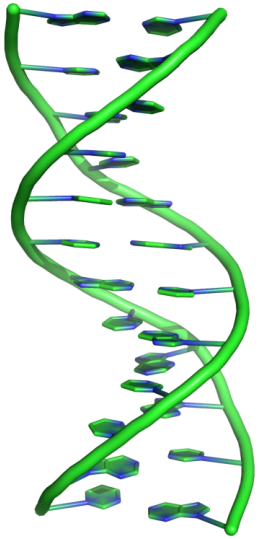
Nucleic Acid Cartoons



set cartoon_ring_mode, 3



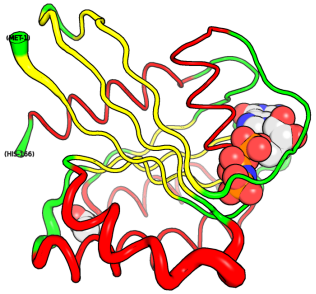
set cartoon_nucleic_acid_mode, 3



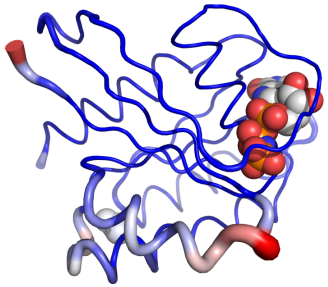
set cartoon_ring_finder,2

cartoon putty

Show which parts of the structure are more flexible in the crystal (b-factor)

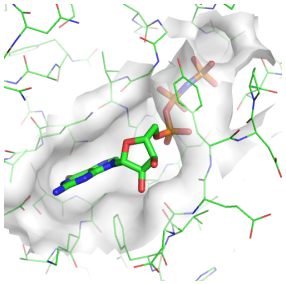


show cartoon
cartoon putty
unset cartoon_smooth_loops
unset cartoon_flat_sheets



spectrum b, blue_white_red, minimum=20, maximum=40
as cartoon
cartoon putty

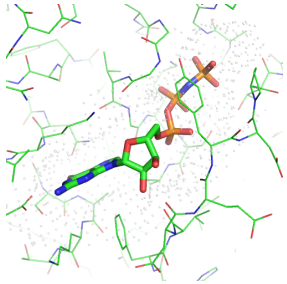
Surface Settings



set surface_type, 0

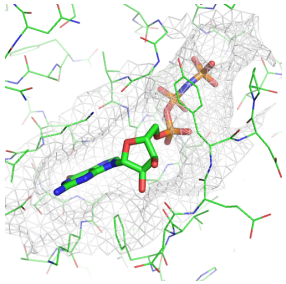
smooth surface, made transparent by

set transparency, 0.5



set surface_type, 1

dot surface

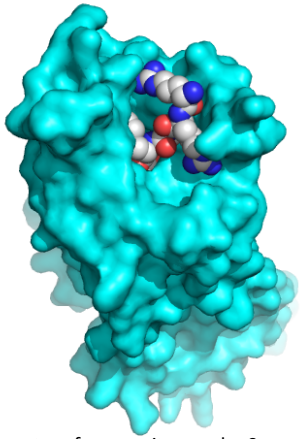


set surface_type, 2

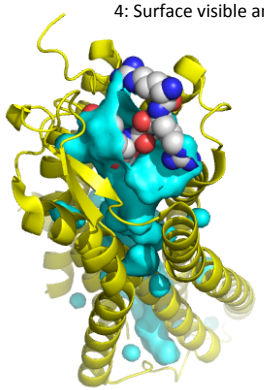
mesh surface

Surface settings

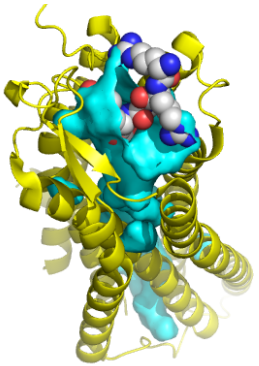
set surface_mode, **int**
0: Default mode, surfacing with respect to flags.
1: Surface everything, including HET and hydrogens
2: Surface only heavy atoms
3: Surface only visible
4: Surface visible and heavy



set surface_cavity_mode, 0

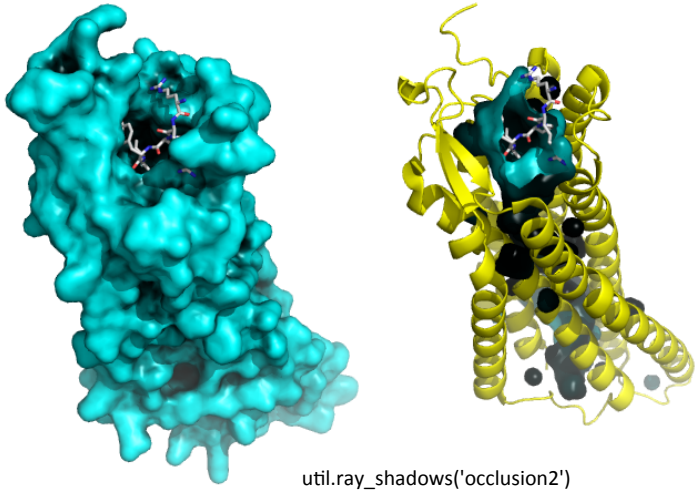


set surface_cavity_mode, 1



set surface_cavity_mode, 2
set surface_cavity_radius, -3
set surface_cavity_cutoff, 7

Shading the Surface



util.ray_shadows('occlusion2')

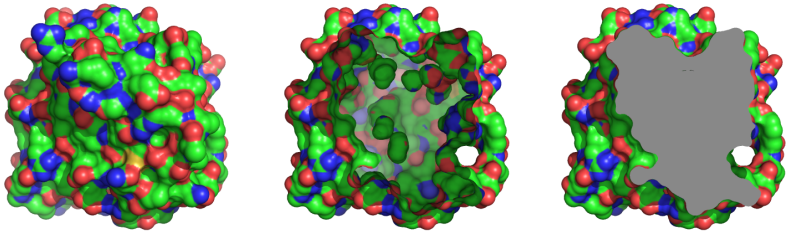
highlights pockets and cavities by depth-dependent shadowing

Showing a solid clipping plane

Pymol Tricks

Normally, if the near clipping plane cuts a surface, the surface is shown as an open shell.

By turning interior lighting off and assigning a fixed color to the interior, in the ray-traced image, the cut appears closed by the clipping plane.

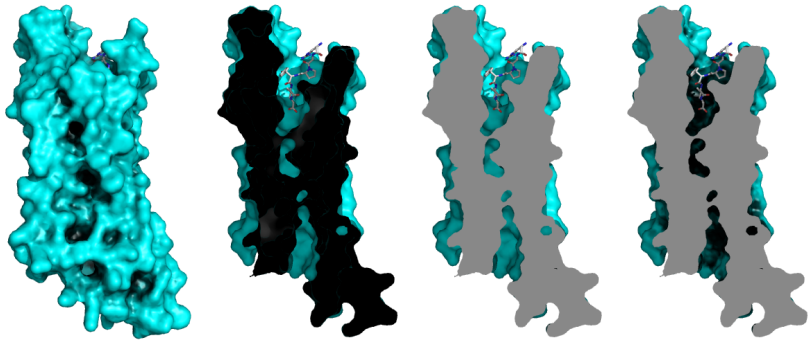


hide all
show surface

clip near, -20

set two_sided_lighting, off
set ray_interior_color, grey70

Solid Clipping Plane



clip near, -30

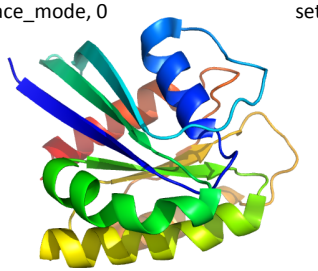
set two_sided_lighting, off
set ray_interior_color, grey70

util.ray_shadows('occlusion2')

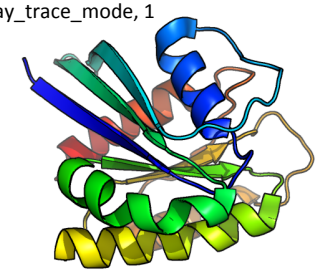
util.ray_shadows('occlusion2')

Ray_trace_mode

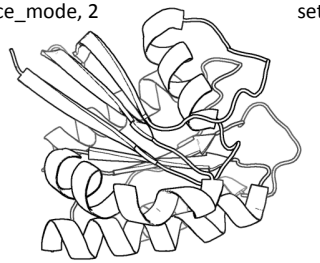
set ray_trace_mode, 0



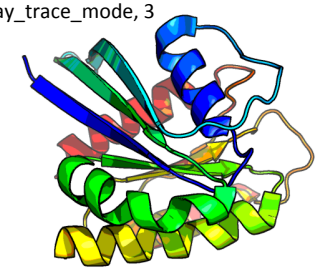
set ray_trace_mode, 1



set ray_trace_mode, 2



set ray_trace_mode, 3



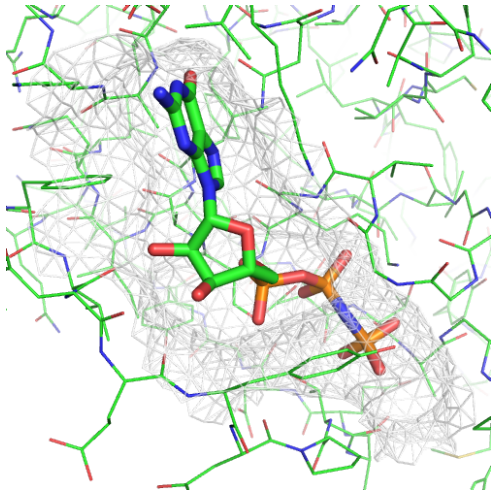
set ray_trace_gain, 0.0 - sets thickness of outline, set ray_trace_disco_factor, 1 to clean up

Examples from the PyMOL Gallery

adapted to 3K8Y

http://www.pymolwiki.org/index.php/Gallery

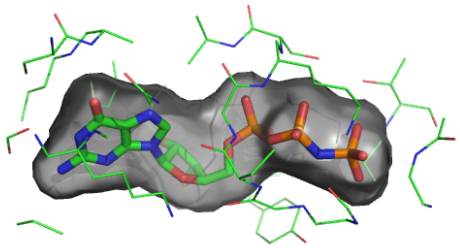
Binding Pocket



@pml_scripts/AHo_BindingPocket.pml

```
#load your molecule, extract prot, lig
load pdb/3K8Y.pdb, tmp
extract lig, resn GNP
extract prot, polymer
delete tmp
#representation
hide all
show lines, prot
show surface, prot within 8 of lig
show sticks, lig
#coloring, prot & lig in default color
bg_color white
set surface_color, white
#orientation (correct as needed)
orient lig
# special settings for this representation
set surface_carve_cutoff, 4.5
set surface_carve_selection, lig
set surface_carve_normal_cutoff, -0.1
set two_sided_lighting
set transparency, 0.5
set surface_type, 2
unset ray_shadows
#render image and save
ray
png figures/BindingPocket.png
save examples/AHo_BindingPocket.pse
```

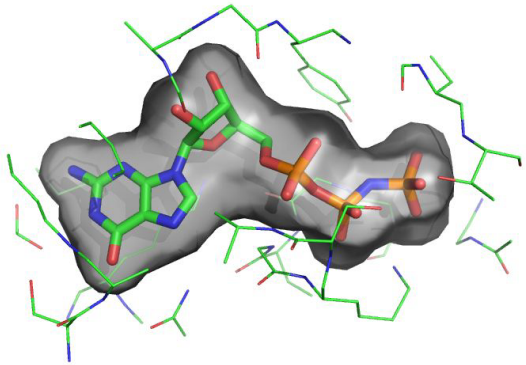
Ray-Normal-Based Transparency



@pml_scripts/AHo_RayNormal.pml

```
#load your molecule, extract prot, lig
load pdb/3K8Y.pdb, tmp
extract lig, resn GNP
extract prot, polymer
delete tmp
#representation
hide all
show surface, lig
show sticks, lig
show lines, prot within 5 of lig
#coloring
bg_color white
set surface_color, grey
# set the view (correct as needed)
orient lig
# special settings for this representation
set surface_mode, 3
set transparency_mode, 1
set transparency, 0.5
set ray_transparency_oblique
set ray_transparency_oblique_power, 8
set ray_transparency_contrast, 7
#render image and save
ray
png figures/RayNormal.png
save examples/AHo_RayNormal.pse
```

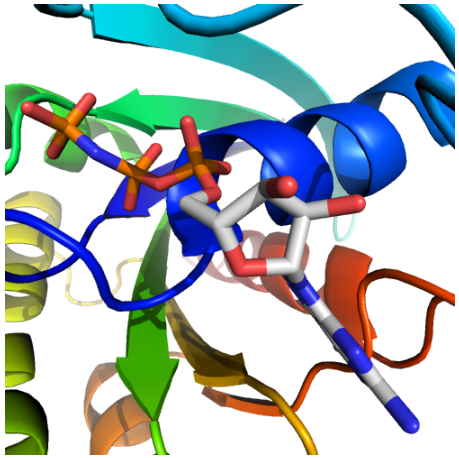
Make a Movie



```
#continued from last slide

# animate
set cache_frames, 1
set ray_trace_frames, 1
mset 1x120
movie.roll 1, 120, 1, x
mplay

save movie as:
as image sequence
or Quicktime movie
```

Get Quicktime Pro 7 from:
http://www.id.uzh.ch/dl/sw/angebote/grafik/QuickTimePro.html

Cool Perspective

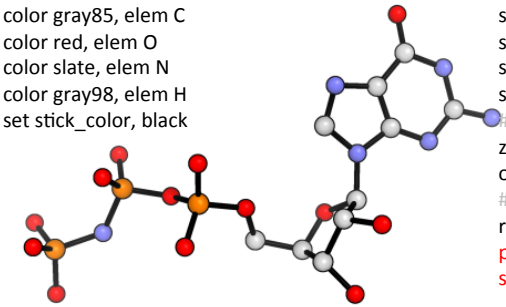


@pml_scripts/AHo_CoolPerspective.pml

```
#load your molecule, extract protein and ligand
load pdb/3K8Y.pdb, tmp
extract lig, resn GNP
extract prot, polymer
delete tmp
#representation
hide all
show cartoon, prot
show sticks, lig
#coloring
bg_color white
spectrum count, rainbow, prot, byres=1
util.cbaw lig
#correct orientation and zoom factor as needed
zoom lig
# special settings for this representation
set field_of_view, 60
#render image and save
ray
png figures/CoolPerspective.png
save examples/AHo_CoolPerspective.pse
```

Stylized Ball-and-Stick

```
load pdb/3K8Y.pdb, tmp
extract lig, resn GNP
extract prot, polymer
delete tmp
#representation
hide everything
show sticks, Lig
show spheres, Lig
#coloring
color gray85, elem C
color red, elem O
color slate, elem N
color gray98, elem H
set stick_color, black
# special settings
set stick_radius, .07
set sphere_scale, .18
set sphere_scale, .13, elem H
set bg_rgb=[1, 1, 1]
set stick_quality, 50
set sphere_quality, 4
set ray_trace_mode, 1
set ray_texture, 2
set antialias, 3
set ambient, 0.5
set spec_count, 5
set shininess, 50
set specular, 1
set reflect, .1
set dash_gap, 0
set dash_color, black
set dash_gap, .15
set dash_length, .05
set dash_round_ends, 0
set dash_radius, .05
#orientation
zoom lig
orient lig
#render image and save
ray
png figures/Ball-and-Sticks.png
save examples/AHo_Ball-and-Sticks.pse
```

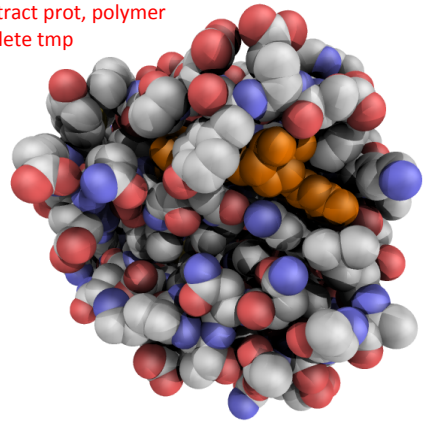


@pml_scripts/AHo_Ball-and-Sticks.pml

QuteMol Style

http://qutemol.sourceforge.net

```
#load your molecule, extract prot, lig
load 3K8Y.pdb, tmp
extract lig, resn GNP
extract prot, polymer
delete tmp
```

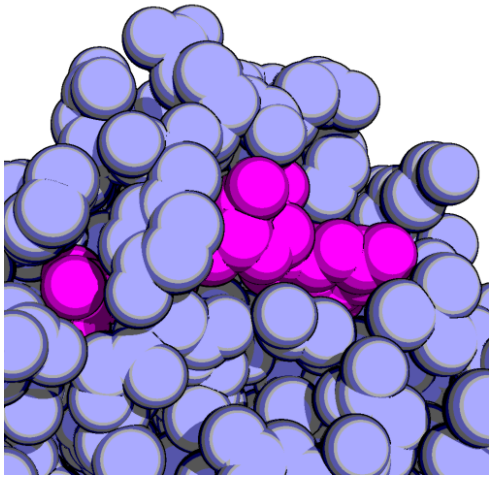


@pml_scripts/AHo_QuteMol.pml

```
# representation
hide all
as spheres
#coloring
bg_color white
set_color oxygen, [1.0,0.4,0.4]
set_color nitrogen, [0.5,0.5,1.0]
util.cbaw
color orange, resn GNP
# special settings for this representation
set light_count, 8
set spec_count, 1
set shininess, 10
set specular, 0.25
set ambient, 0
set direct, 0
set reflect, 1.5
set ray_shadow_decay_factor, 0.1
set ray_shadow_decay_range, 2
unset depth_cue
set field_of_view, 60
#render image and save
ray
png figures/QuteMol.png
save examples/AHo_QuteMol.pse
```

Goodsell-like Representation

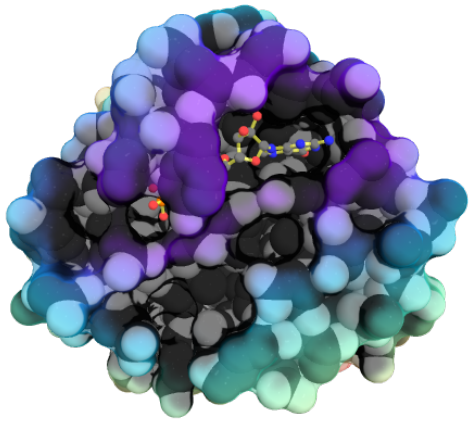
http://www.rcsb.org/pdb/101/motm.do?momID=184



@pml_scripts/AHo_GoodsellLike.pml

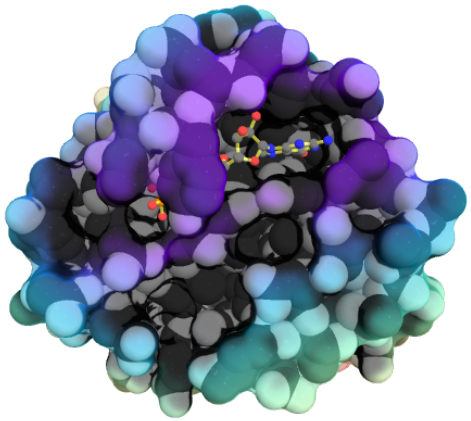
```
#load your molecule, extract prot, lig
load pdb/3K8Y.pdb, tmp
extract lig, resn GNP
extract prot, polymer
delete tmp
#representation
as spheres
#coloring
bg_color white
color lightblue, prot
color magenta, lig
# set the view (correct as needed)
orient all within 8 of lig
# special settings for this representation
unset specular
set ray_trace_gain, 0
set ray_trace_mode, 3
set ray_trace_color, black
unset depth_cue
# render image and save
ray
png figures/GoodsellLike.png
save examples/AHo_GoodsellLike.pse
```

Complex Stylized Protein



```
...
# representation
hide all
preset.ball_and_stick("lig")
show spheres, prot
set sphere_scale, 0.99, prot
show surface, prot
# coloring
bg_color white
set_bond stick_color, 0xffff44, lig
set_bond stick_transparency, 0.35, lig
color grey, lig and e. C
ramp_new pRamp, lig, selection=prot, \
      range=[5,30], color=rainbow
set surface_color, pRamp, prot
color white, prot
color grey30, prot and e. C
disable pRamp

# continued on next slide
```

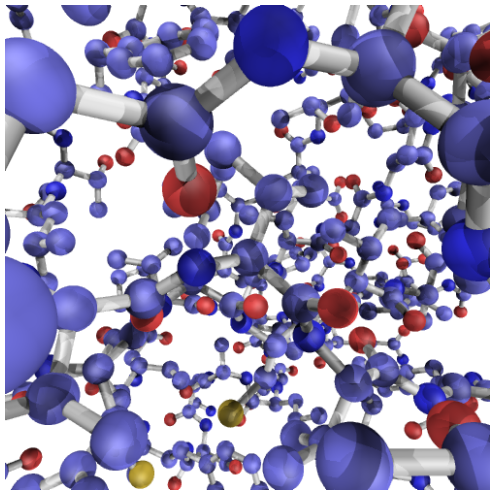
Complex Stylized Protein



@pml_scripts/AHo_StylizedProtein.pml

```
# special settings for this representation
set ray_transparency_contrast, 0.20
set ray_transparency_oblique, 1.0
set ray_transparency_oblique_power, 20
set surface_quality, 2
set light_count, 5
set ambient_occlusion_mode, 1
set ambient_occlusion_scale, 50
set ambient, 0.40
set transparency, 0.50
set spec_power, 1200
set spec_reflect, 0.20
set ray_opaque_background, 0
set ray_shadow, 0
set field_of_view, 50
# orientation
zoom complete=1
# render image and save
ray
png figures/StylizedProtein.png
save examples/AHo_StylizedProtein.pse
```

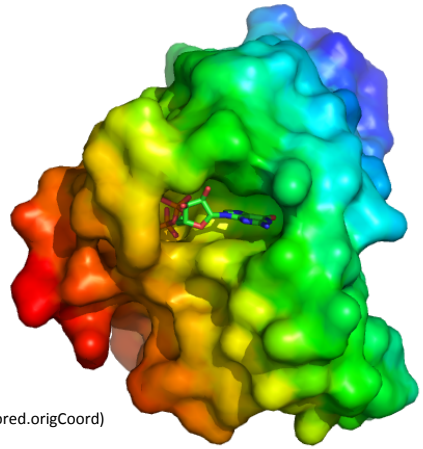
Ball-and-Stick



@pml_scripts/AHo_Ball_and_Stick2.pml

```
load pdb/3K8Y.pdb, tmp
extract lig, resn GNP
extract prot, polymer
delete tmp
#representation
hide all
show spheres
show sticks
#coloring
bg_color white
util.cbab
set stick_ball_color, atomic
set_bond stick_color, white, all, all
# special settings for this representation
set stick_radius, 0.4, (all)
set sphere_scale, 0.3, (all)
set_bond stick_radius, -0.14, all, all
set light_count, 8
set spec_count, 1
set shininess, 10
set specular, 0.25
set ambient, 0
set direct, 0
set reflect, 1.5
set ray_shadow_decay_factor, 0.1
set ray_shadow_decay_range, 2
unset depth_cue
set field_of_view, 60
# render and save
ray
png figures/Ball_and_Stick2.png
save examples/AHo_Ball_and_Stick2.pse
```

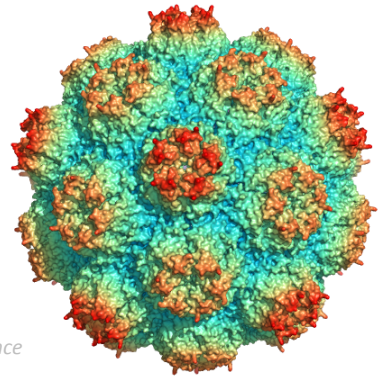
Color by Distance from Origin



@pml_scripts/AHo_ColorDist.pml

```
diff_len = lambda x,y : math.sqrt((x[0]-y[0])*(x[0]-y[0]) + (x[1]-y[1])*(x[1]-y[1]) + (x[2]-y[2])*(x[2]-y[2]))
#load your molecule, extract prot, lig
load pdb/3K8Y.pdb, tmp
extract lig, resn GNP
extract prot, polymer
delete tmp
#representation
as surface, prot
as stick, lig
# create the pseudoatom at the origin
pseudoatom pOrig, pos=(0,0,0), label=origin
# these are special PyMOL variables that will hold the
# coordinates of the atoms and the  pseudoatom
stored.origCoord = []
stored.distCoord = []
# copy the coordinates into those special variables
iterate_state 1, pOrig, stored.origCoord.append((x,y,z))
iterate_state 1, prot, stored.distCoord.append((x,y,z))
# extend origCoord to be the same length as the other
stored.origCoord *= len(stored.distCoord)
# calculate the distances
newB = map(lambda x,y: diff_len(x,y), stored.distCoord, stored.origCoord)
# put them into the b-factor of the protein
alter prot, b=newB.pop(0)
# color by rainbow_rev or any other palette listed in "help spectrum"
spectrum b, rainbow_rev, prot
bg_color white
...
```

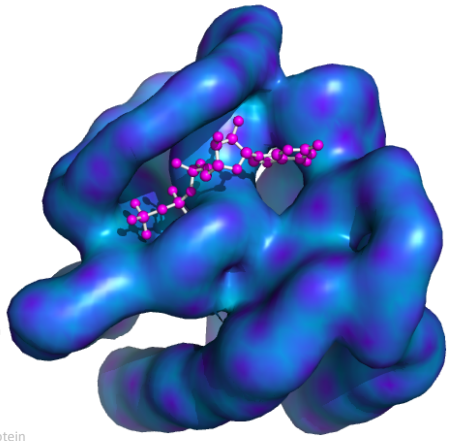
Virus Capsid



```
...
# create a pseudoatom at the origin-- we will
# measure the distance from this point
pseudoatom pOrig, pos=(0,0,0), label=origin
# load and build the capsid
load pdb/2xpj.pdb1.gz
split_states 2xpj
delete 2xpj
# show all 60 subunits it as a surface
# this will take a few minutes to calculate
as surface
# create a new color ramp, measuring the distance
# from pOrig to 1hug, colored as rainbow
ramp_new proximityRamp, pOrig, selection=(2xpj*), range=[110,160], color=rainbow
# set the surface color to the ramp coloring
set surface_color, proximityRamp, (2xpj*)
# some older PyMOLs need this recoloring/rebuilding
recolor
bg_color white
disable proximityRamp
# render image and save
...
```

@pml_scripts/AHo_VirusCapsid.pml

Smooth Pseudo-Surface with Ligand



```
...

# Ligand as ball and stick
bg_color white
hide lines
show sticks, lig
show spheres, lig
color magenta, lig
set_bond stick_radius, 0.13, lig
set sphere_scale, 0.26, lig
set_bond stick_radius, 0.13, lig
set_bond stick_color, white, lig
set sphere_scale, 0.26, lig

#protein pseudo-surface
# set the B-factors nice and high for smoothness
alter all, b=10
alter all, q=1
# 3.5 A map resolution
set gaussian_resolution, 8
# new gaussian map w/resolution=0.5 Ang on just the main chain
map_new map, gaussian, 1, n. C+O+N+CA, 5
# create a surface from the map
isosurface surf, map, 1.5
# color the protein by number
spectrum count, rainbow, prot
# now color the map based on the b-factors of the underlying protein
cmd.ramp_new("ramp", "prot", [0,10,10], "rainbow")
# set the surface color
cmd.set("surface_color", "ramp", "surf")
# hide the ramp and lines
disable ramp
...
```

#pml_scripts//AHo_SmoothSurfwLig.pml