A Beginner's Guide to Molecular Visualization Using PyMOL

By Nicholas Fitzkee
Mississippi State University

In this lab, we will be using the program PyMOL to visualize and analyze protein structures. PyMOL is a powerful utility for studying proteins, DNA, and other biological molecules. The software itself is well written and easy to use, and in the past 10 years it has become very popular with structural biologists.

Many of the concepts we will learn are explored in greater detail in the *PyMOL User's Guide*. Although somewhat dated, the *User's Guide* has very useful information and is definitely worth reading. Several of the images from the *User's Guide* have been reproduced in this document. You can download the guide at http://pymol.sourceforge.net/newman/userman.pdf.

Throughout this document, you will be asked to answer questions about proteins and protein structures. To differentiate questions from the rest of the text, the questions are placed against a background of grey, like this. In some of the questions, you will be making molecular graphics, and while you can print this and submit them in class, you are welcome to submit your answers digitally via email if it is more convenient. You can place your pictures into a Word document using the "Insert Picture" feature.

Obtaining PyMOL

PyMOL was originally written by Warren Delano as an updated molecular viewer. Back in the early 2000's, many viewer programs existed, but all of them were aging, and none took advantage of the recent advances in video card technology. Additionally, no one program was sufficiently polished to do many things well. RasMol was great for structural analysis, but it had dated graphics. Molscript produced fabulous illustrations, but it was cumbersome to use and was not designed for analyzing structures. MolMol was a great tool for analysis, but it was no longer being supported. Insight2 (now Discovery Studio) could do many things well, but it was expensive and was eventually bought out by Accelrys, and it remains very expensive. Other viewers, like SwissPDB Viewer and Cn3D functioned well, too, but all of them had limitations of one sort or another. PyMOL is not perfect, but had several unique advantages for the time:

- Unlike most scientific software, PyMOL is highly polished; it won't unexpectedly crash while you're using it.
- PyMOL can produce high-quality graphics, on par with Molscript, without needing to manually edit text files.
- PyMOL has an extensive help system, and documentation can be found by typing `help command` for many commands.
- Measurement of bond distances and angles is straightforward in PyMOL. Structures can be analyzed in a semi-automated way with scripting support.
- PyMOL is optimized to use high-end graphics hardware, and it can support 3-D graphics (the same 3-D that modern TVs are just now starting to use).

Warren implemented PyMOL in the Python programming language, which made it easy for end users to extend its functionality with plugins and scripts. He also released PyMOL as a completely open-source project, which encouraged other users to download the source code (for free) and experiment with the program. Warren's payment model was based on the honor system: if you were a student, you could use PyMOL for free, but academic labs were encouraged to support PyMOL by paying a yearly subscription based on the size of the lab. Accordingly, subscribing labs could get help and support (often direct from Warren himself), and they would have access to newer versions than what was made available for free. Since PyMOL was open source software, savvy users could always download and compile the latest version and compile it themselves, but this required a certain level of expertise and time commitment that many academic users did not have.

Unfortunately for all of us, Warren passed away in 2009, and the fate of PyMOL was uncertain for a time. Eventually, the software company Schrödinger took over the project, and since 2009 they have maintained Warren's vision (more or less) and kept the project going.

PyMOL is still freely available for academic use, with two main limitations: (1) the version you use as an academic may lag somewhat behind the most recent version that Schrödinger maintains, and (2) no official (unpaid) support is offered. Fortunately, there is a strong user community, and it's easy to find answers to questions on the web.

To obtain PyMOL, visit the PyMOL website (http://www.pymol.org/educational), read the notice, and then click on the "register here" link at the bottom of the page. You'll need to fill out the form, and the automated system will eventually send you a link with a username and password. This allows you to download the software for your Mac or PC system.

Installation is straightforward, and PyMOL can be installed like any other PC or Macintosh software. During the installation process on a PC, you may be presented with several dialogs regarding initial configuration of PyMOL. You may safely leave these set at the default values.

Alternatively, you can obtain an older version of PyMOL directly (version 0.99 rc6) from the following site: http://sourceforge.net/projects/pymol/files/Legacy/. This version is fully functional and is sufficient for this tutorial; however, it does not appear to work with Windows 7 systems. You may have better luck than me, so it's worth trying.

Running PyMOL

Running PyMOL is like running nearly any other program on your computer. When you run PyMOL (on Windows, run "PyMOL + Tcl-Tk GUI"), you will be presented with the main display (Figure 1).

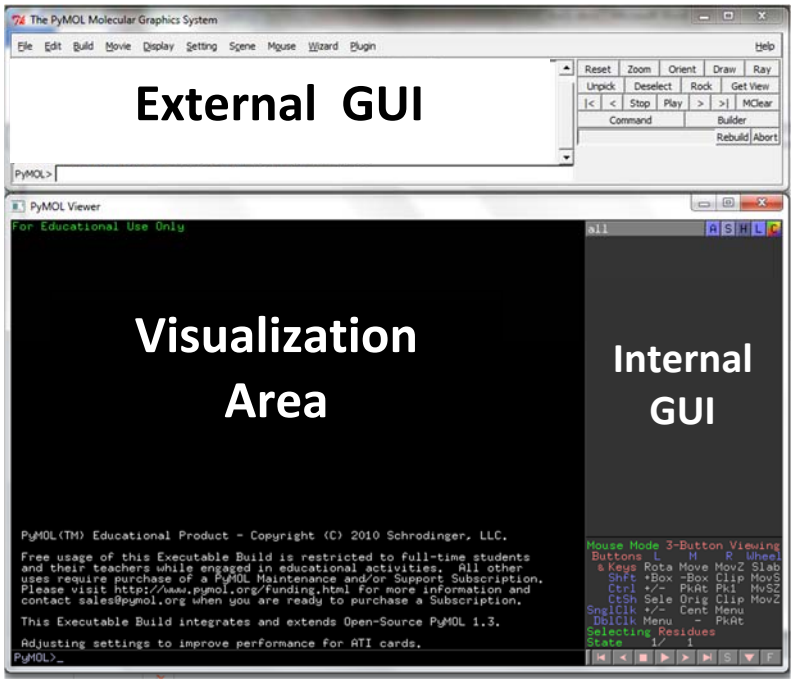


Figure 1. The PyMOL main display.

In Windows, this display is set up across two windows. The top window constitutes the "External GUI," and contains the menu options as well as buttons for advanced visualization. It contains a large text area as well, which logs the commands you have used in the viewer.

The bottom window contains the "Visualization Area," which is the main area where molecules will be displayed. The visualization area can also display text, like help text. When in text mode, the visualization area displays similar information to what is displayed in the external GUI text box.

The bottom window also contains another "Internal GUI." This GUI will contain a list of molecular objects once you have loaded a protein structure. The bottom of this GUI has a matrix displaying the current mouse configuration, namely what mouse button combinations control which functions. It also contains additional buttons for making molecular movies.

On Macintosh systems, all three of these regions are merged into the same window, but the regions are all there, and the behavior between Windows and Mac is otherwise identical.

Opening Your First PDB File

High-resolution molecular structures are determined by one of two methods, namely X-ray crystallography or NMR spectroscopy. Unfortunately, time doesn't permit us to discuss these techniques in depth; suffice it to say that once the three-dimensional atomic coordinates are determined, they can be formatted into a text file that programs like PyMOL can read. These files are called "PDB" files, short for the "Protein Data Bank."

As scientists determine new molecular structures, they submit the coordinates to the Research Collaboratory for Structural Bioinformatics (RCSB). This organization maintains the PDB, and it ensures that all PDB files have the proper format and supporting data. They also offer outreach and implement new approaches to understanding macromolecular structure. The PDB website is available at http://www.pdb.org/, and you can browse this site to learn more about what the RCSB does.

Database entries in the PDB are given a characteristic four-character code that is used to identify the structure. For example, 1SNC is an entry for the protein staphylococcal nuclease. Staphylococcal nuclease is an enzyme that hydrolyzes (cleaves) DNA and RNA. It is used by *Staph. aureus* to destroy foreign genetic material from bacteria and other sources. Nuclease has been extensively studied, and many of its properties were established by Chris Anfinsen in the 1960's. The following paper describes the properties of staphylococcal nuclease in detail, including the sedimentation and diffusion coefficients:

Heins, James N., *et al.* (1967) *J. Biol. Chem.* **242** (5): 1015-1020.

The crystal structure of nuclease has been determined, and you can access this entry by searching through the PDB website for 1SNC. The web page for 1SNC contains much information about how the structure was obtained. It is possible to download the entry directly, and this file is called a PDB file. The normal extension for these files is PDB, e.g. the file would be named `1SNC.pdb`.

Visit the PDB website page for 1SNC and download the file. At the right hand side of the screen is an option to "Download Files." When you click this link, you'll be presented with the option to download the PDB file as text. Save this file to a convenient location – you will shortly open the file in PyMOL.

1. Several critical pieces of information are given on the 1SNC web page. What is the length of this protein (the number of residues)? What is the resolution of this structure (in Angstroms)? Who are the scientists responsible for this structure?

To open the PDB file, select "File → Open" in the external GUI window, and select the 1SNC PDB file that you downloaded. The PDB file will load, and you will see the "lines" representation of the protein (Figure 2). In this representation, each chemical bond is drawn as a line, and atom nuclei exist where the bonds intersect. In the default representation, Carbon atoms are green, nitrogen is blue, oxygen is red, sulfur is yellow, and phosphorus is orange. Hydrogen atoms are rendered white, but they aren't typically visible in a crystal structure.

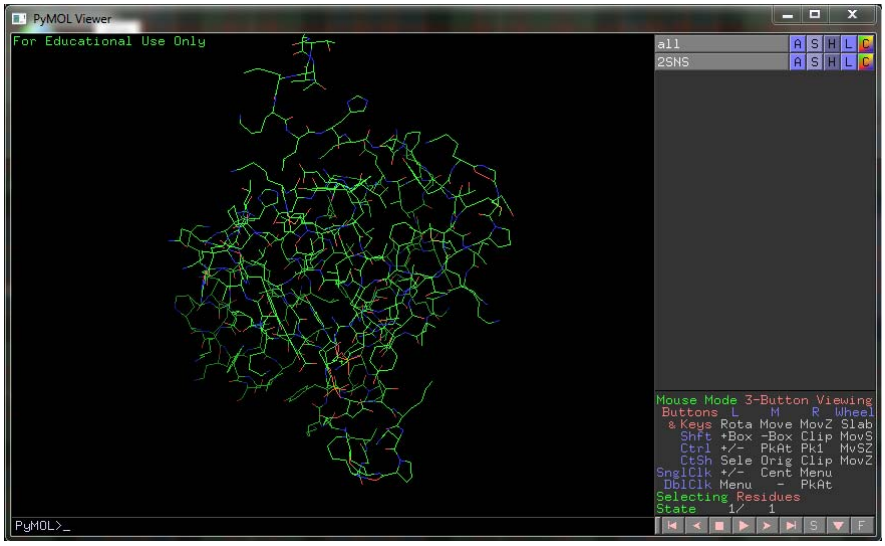


Figure 2. Staphylococcal nuclease rendered as lines.

Basic Viewing Functions and Navigation

Within the viewing window, you can click and drag with the left mouse button to rotate the molecule. Dragging with the right mouse button will allow you to zoom in and out. Finally, dragging with the middle mouse button will translate the structure in the X-Y plane of your monitor. Using a combination of rotations, translations, and zoom operations, it's possible to position yourself anywhere within the molecular frame, although it does take some getting used to.

Another useful visualization tool is called "slab." As you look at the protein, the viewing axis coming out of the monitor is the Z-axis. Sometimes, the region of interest is in the center of the protein, occluded by the atoms on the surface. The slab setting allows you to adjust the viewing "slab" to eliminate the extra atoms from the display (Figure 3).

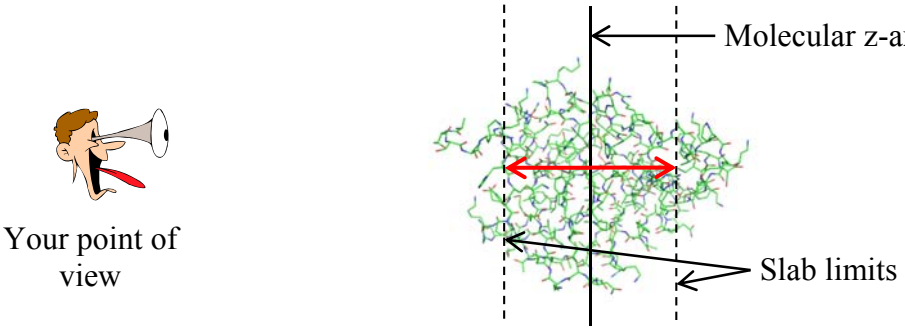


Figure 3. The concept of slab.

In the figure, anything outside of the slab limits is hidden, and only the region between the dotted lines is displayed. As you adjust the slab, the slab limits change: the length of the red arrows can

be very large, allowing you to view the entire molecular frame. Alternatively, you can make the slab very small, focusing in on a particular region of the protein. In PyMOL, rolling the mouse wheel toward you decreases the size of the slab, and rolling it away from you increases the slab.

PyMOL also allows you to interact with the molecule itself, selecting individual residues (or atoms) by clicking on them. When you click on the protein, the atoms in the selected residue are highlighted with pink boxes. You can see the selection in the text box of the external GUI window:

```
You clicked /1SNC//A/LYS`16/CD
 Selector: selection "sele" defined with 9 atoms.
```

From this syntax, I know that I clicked on the delta carbon (CD) of 1SNC, chain A, Lysine 16. Since multiple atoms were defined in my selection, I know that the whole residue was selected. You can select multiple residues with the mouse by clicking on additional atoms, or you can unselect residues by clicking the same residue again (not a double click; two single clicks). Whenever you make or modify a selection, you can see the number of atoms in the external GUI window. To unselect all residues, click on an area of the viewer window with no atoms.

A summary of all this is displayed in the lower right hand corner of the viewer window. It will tell you that you are in "3-Button Viewing" mode, and that you are selecting "Residues." A summary of the mouse commands are displayed in a convenient matrix. By clicking on the region, it is possible to change the mouse mode (from "3-Button Viewing" to "3-Button Editing"), and you can also change the selection mode (possible options are: Objects, Segments, Chains, Molecules, Residues, Atoms, and C-alpha atoms). For our purposes, we will operate mostly in "3-Button Viewing" mode, selecting residues.

An alternative way to select residues is by directly using the protein sequence. In the external GUI window, select "Display → Sequence." You'll notice that at the top of the viewer window you can now see the sequence of residues in Staphylococcal nuclease (starting at residue 7, "LHKEP…," or "Leu, His, Lys, Glu, Pro"). The sequence starts at the N-terminus (Ala 7) and ends at the C-terminus (Ser 141). By using the scroll bar and clicking on the residues, you can select residues by number without having to find them in the structure. This is a convenient way to locate a residue if you aren't sure of its location.

The observant among you will notice that the last residue in the PDB file (Ser 141), doesn't correspond to the number of residues you found for question 1. Additionally, there are a lot of "O" residues that in red that go past the end of the protein's natural sequence. Both of these observations have to do with how the protein structure was determined experimentally. In a crystal structure like this one, disordered atoms frequently do not appear. Thus, residues 1-6 and the residues past 141 are missing. Additionally, ordered solvent atoms which are associated with the protein *can* appear in the PDB file. In this case, several water molecules can be seen in the structure, and those appear as "Os."

Directly above the mouse mode matrix is a region in the viewing window which displays a list of visible objects available in PyMOL. At the top of this list is "all," and clicking this will allow you to quickly show or hide all visible objects. Below this, you will see "1SNC," which is the PDB file we are currently viewing. And, depending on whether you have atoms selected, you will see

a "(sele)" below that, denoting the selection you have currently created. (Remember, since they have pink dots, selections are "visible" objects, too!)

Next to each object name, you will see five letters: A (actions), S (show), H (hide), L (label), and C (color). Each of these buttons brings up a window with additional options for this object. For example, under the action menu (A) for 1SNC, you can select "zoom" to center the molecule in the viewer window and zoom so that the entire molecule fits in the window. We will discuss other options later on.

We mentioned that water molecules are often associated with protein structures ("crystallographic waters"), and we saw those waters in the sequence display. Let's practice using the action menu to remove those waters, since we aren't really interested in them for this tutorial. Select (A) → "remove waters," and you should see the "cloud" of crosses in your structure disappear. What's left is just the protein and the substrate atoms.

Before we move on, remember that the graphical viewer window can also be toggled with a text display. If you select the viewer window and press ESC, you will see the text associated with all of the commands you have performed so far. Unlike the text in the external GUI, this text does not have a scroll bar, but it is helpful for seeing a log of what you've been doing. Pressing ESC again will switch you back to graphics mode.

2. What is the three-letter amino acid sequence for residues 100-105 in 1SNC?

Selection Commands

In the previous section, we demonstrated how molecules could be selected using the mouse or sequence display. However, often times it's necessary to select atoms more precisely. To facilitate this, PyMOL offers a command-line for fine control of its functionality. Commands in PyMOL can be entered in two places: the `PyMOL>` prompt at the bottom of the external GUI window, or the same prompt in the viewer.

As an example of atom selection, type the following command into either PyMOL prompt:

```
select loopca, resi 42-52 and name CA
```

If you zoom in on the selected region, you'll notice that the C-alpha (CA) atoms have been selected in the loop between residues 42 and 52. You'll also notice that a new selection object has been created in your list of objects called "loopca" (selection objects are enclosed in parentheses). The external GUI once again notes the number of selected atoms. You can refer to this selection object in other PyMOL commands, as we'll see below.

Breaking up this particular command, we can identify its distinct parts:

```
select loopca,
```

> This tells PyMOL to define a new selection named "loopca." The name of the selection is the first "argument" to the selection command. The comma following this command tells

PyMOL's parser that we're going to move on to another argument. The second argument of the select command is the selection itself.

```
resi 42-52 and name CA
```

This syntax tells PyMOL how to define the selection "loopca." The entire statement is the second argument (arguments in PyMOL are separated by commas). The selection syntax is straightforward:

- The first selection statement (the text before the `and`) tells PyMOL to select residues by index (that's the *i* in `resi`), from 42-52.
- The second selection (after the `and`) tells PyMOL to select all atoms with name CA (the C-alpha atoms).
- Finally, the `and` operator tells PyMOL to take the intersection of the two sets: only those atoms that are both named CA and are in the loop from residues 42-52.

Obviously, we could have dropped the second half of the selection statement to select all atoms in residues 42-52. Similarly, we could have reversed the order of the residues: the intersection does not depend on the order of operations.

Some other useful selection statements are below. They can all be combined with the operators `and`, `or`, or `not`. You can also use parentheses to group statements if you aren't sure how PyMOL will order them – just like in math.

- `resn <name>`

  This statement will select all residues with a given 3-letter name `<name>`. For example, `select ala, resn ala` will select all alanines in the protein. Multiple residue names can be selected with the "+" sign, e.g. `select negative, resn asp+glu`.

- `elem <name>`

  This statement allows you to select elements by their atomic symbol, e.g. "He" for helium, "C" for carbon, etc. It's useful for changing the default color scheme, since you can easily select all carbon atoms (if you don't like green carbons.)

- `<selection 1> within <distance> of <selection 2>`

  This statement allows you to select things by distance, where `<distance>` is in Angstroms. The command

  ```
  select site, name CA within 10 of resi 25
  ```

  will select all C-alpha atoms within 10 Å of any atom in residue 25. Note that this involves some calculation: some CA atoms may be within 10 Å of parts of residue 25, but they may

be farther from other atoms. If the distance cutoff applies for any atom pair from `<selection 2>` and `<selection 1>`, it will be included.

The selections `all` and `visible` can also be useful sometimes, too. Respectively, they select all atoms or only those that are already visible in the viewer window. You can get more help on selection syntax by typing "`help selection`" into the viewer window prompt. Remember to press ESC so you can view the text!

3. How many carbon atoms are there in all the Alanine residues between residues 15-60? What command did you use to determine this?

Molecular Representations

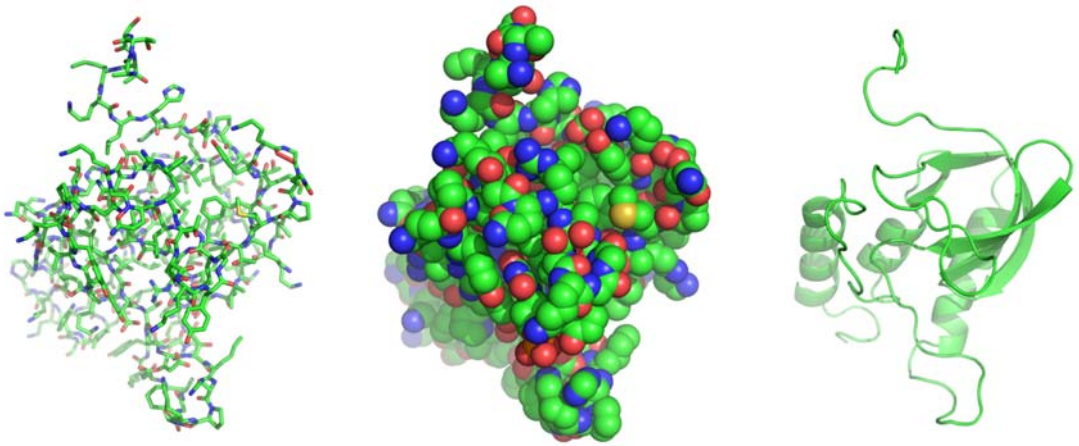
As you have probably noticed by now, viewing only bonds has its disadvantages. For one, there's no concept of how much space the atoms occupy. Secondly, the sheer number of atoms in a protein can be cumbersome. It would be nice to simplify the view a bit, and then highlight areas of interest as needed.

To accomplish this, scientists have developed multiple schemes for visualizing proteins and nucleic acids. Some examples are

- Sticks: These are similar to the lines we have been working with so far, but they are thicker, like the molecular models used in organic chemistry.
- Spheres: In this representation, all atoms are drawn as spheres, with radii that are characteristic of their (s) electron orbitals. This is also called CPK representation, after Corey, Pauling, and Branson, the originators of such models.
- Cartoons: In this representation, the side chain atoms are ignored, and smooth line is drawn through the backbone alone. Alpha helices and beta strands are drawn as coils and arrows, respectively.

As you might expect, it's possible to draw these representations in PyMOL (Figure 4, next page).

Each representation has its strengths and weaknesses. Cartoons, for example, work great for simplifying the structure, but it's hard to get information about the chemistry involved in the enzyme. Sticks, on the other hand, reveal the chemical structure but are hard to interpret for large systems. Spheres make it easy to understand packing and steric hindrance, but they occlude the interior of the protein. Published figures often include some combination of these three representations.



Sticks Spheres (CPK) Cartoons

Figure 4. Alternative representations of protein structure.

PyMOL supports all of these representations through the `show` and `hide` commands. So, typing `show spheres` will show the CPK model of the protein. Typing `hide spheres` will hide it. There are many representations to choose from, but for our purposes we will only work with: `lines`, `sticks`, `spheres`, and `cartoon`.

Multiple models can be displayed simultaneously. For example when you type `show cartoon` when the program first loads the 1SNC file, you will still see that the lines representation is displayed until you type `hide lines`. Additionally, the `show` and `hide` commands can accept an optional second argument. If you place a comma after the representation type, you can specify a selection of atoms to show, like so:

```
show <representation>, <selection>
hide <representation>, <selection>
```

Here, `<selection>` is either a selection string (e.g., `resi 40-52`), or a named selection that you have defined with the `select` command (e.g. "loopca"). This allows you to mix and match representations. Areas where interesting chemistry occurs (i.e. the active site) can be shown in atomic-level detail, while the rest of the protein can be drawn as a cartoon model. If you ever reach a point where you are frustrated and want to start over, you can type `hide everything` to hide all of the representations.

When a scientist prepares a molecular figure for publication, he or she must make reasonable decisions about how to create that figure. It is not always easy to know what level of detail to include in a picture, and every picture represents an individual's *interpretation* of what's important. This interpretation can be better or worse depending on the chemical reality. Importantly, **by choosing to hide some atoms in the cartoon representation, a scientist can intentionally or inadvertently leave out important structural details.**

Now that you know how to change the molecular representation, there are only two more commands you need to know to produce professional-quality molecular images. The first is the `color` command. It works exactly like the show command, except that its first argument is a color. Most colors are okay to use, for example, this command:

```
color orange, resn ser+thr and elem C
```

will color the carbons of all serine and threonine residues orange. This is useful for when you want to highlight something distinct from the rest of the protein.

Another useful command is the `ray` command. This command performs ray-tracing on the molecule to produce a photorealistic picture. Although ray-tracing is far beyond the scope of our discussion of macromolecules, it is simply a simulation where calculations are made to determine how light will reflect off of an object (our protein) and be visible from a viewport (our screen). Because this calculation involves simulating the paths of many photons, it takes a while and it's impractical to do all the time. However, ray-tracing produces the best molecular graphics possible, and once you have your display configured, it's well worth the time.

To ray trace the image in PyMOL, simply type `ray` at the command line. After a few seconds, the ray-traced image will appear. We will talk about how to save the image in the next section, but note that if you click on the viewer window after ray tracing the image will be lost; thus, ***it's important to save the image right after ray tracing so you don't lose the calculation.*** There are other commands to change the resolution of the final image, but for the purposes of this lab the default resolution is sufficient (640 by 480 pixels).

Finally, for publications, it's good to use a white background instead of black. You can change this by selecting "Display → Background → White" in the external GUI window. Since black is easier to view on a screen, you can change back to black by selecting "Black" from the menu.

Saving Your Results

Once you have an appealing image, it's a good idea to save your results. In PyMOL, the state of the molecule along with the coordinates and object listing are saved as a session file. A session contains all the needed information to reproduce the view window, and you should save your session frequently to avoid losing work. To do this, select "File → Save Session As…" from the external GUI window. When you restart PyMOL, you can load this session and all of the settings you used will be recalled. Given that some molecular images can take a significant amount of time to design, this feature is very useful.

While PyMOL sessions contain program information about your molecule, they do not contain graphical information for use in other programs (like MS Word). Since ray tracing can take a long time, PyMOL provides a method for exporting images, too. Saving an image is like saving a session; select "File → Save Image As → PNG" to save your file in PNG format, which is readable by nearly every word processing and presentation application available today. If you do this immediately after ray-tracing, the ray-traced image will be saved. You can save images at any time, and it will capture the current display in the viewer window; however, if the images are not ray traced they will be of much lower quality.

4. The 1SNC file you are using contains several parts: The protein itself spans residues 1-141. Additionally, there is a nucleic acid analog present in the active site (Thymidine 3',5'-diphosphate, resn = THP). There is also a calcium ion in the active site.

   Construct a **ray-traced** image where all protein carbon atoms are colored grey. Start with a cartoon model on a white background (no lines), and then add the following details: The carbon atoms of the THP should be yellow, and all non-protein atoms should be represented as spheres. Then, display as sticks all protein atoms with 10 angstroms of the THP and calcium atoms. Select an orientation that highlights the active site, and submit this image to your instructor with your completed assignment.

   For this figure, use a white background (Display → Background → White).

Structural Analysis

Our lab is almost complete, but there is one more important feature of PyMOL that bears mention. In addition to aiding in the visualization of proteins, it can also be used to analyze proteins. It is possible to measure interatomic distances, scalar angles, and torsion angles using PyMOL. These measurements can be used in interpreting other biophysical experiments. For example, one would expect tight binding to result in close contact between ligands and proteins, and one could also expect a correspondence between measured hydrodynamic data and the observed shape of the protein (i.e. how spherical it is).

To measure distances, select "Wizard → Measurement" from the external GUI window. You'll see some options appear in the object list (Figure 5).

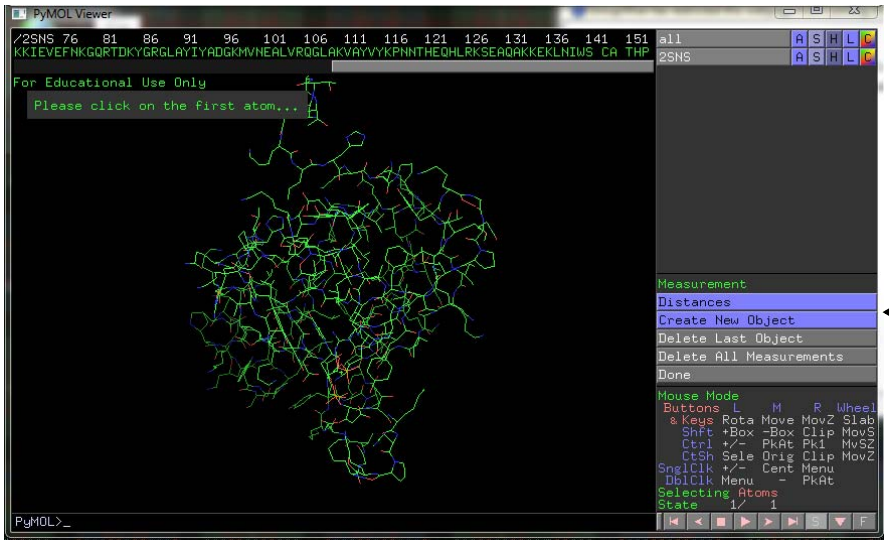


Figure 5. Distance measurement options in PyMOL.

Once the measurement tool is enabled, you will be asked to pick atoms for use in the measurement. Since we are currently measuring distances, PyMOL will ask you for two atoms. If you are

measuring scalar angles, three atoms will be needed, and torsion angles require four atoms. Go ahead and pick two atoms to measure the distance between them. A yellow dotted line is displayed indicating your distance, and the distance itself is displayed by the line (in Angstroms). A new measurement object appears in the object list as well, allowing you to hide it if you prefer. Eager to proceed, PyMOL asks you for another atom when the first measurement is complete.

The measurement window has several options. By clicking on "Distances" you can select other measurements, including scalar and dihedral (torsion) angles. You can also control how PyMOL deals with new distance objects. The default is to create a new object with each measurement, but by clicking on the "Create New Object" button, you can control this behavior as well. Finally, there are buttons to delete the last measurement object or delete all objects. When you are done with measurement, click the "Done" button and you will be returned to the normal mouse mode.

5. What is the longest dimension you can find in staphylococcal nuclease? If you had to estimate the volume of a prolate spheroid of the same size, what would it be? (Recall that $V = \frac{4\pi}{3}ab^2$, where a is the radius of the major axis and b is the radius of the minor axis.)

A Goldmine for the Observant

Currently, there are over 90,000 structures of biological macromolecules in the PDB. Back in 2001, the number was less than 20,000. This increase in structural data has been very useful for scientists, but analyzing that amount of data is a challenge. Structural trends and organizing principles undoubtedly exist in the PDB, but it takes time and a keen eye to identify them. Programs like PyMOL are useful for structural analysis, not only because you can measure atomic geometries, but also because you can spot more qualitative trends in the structures themselves.

As an example, recall our discussion of alpha helices. Each carbonyl oxygen in an alpha helix at residue *i* makes a hydrogen bond to the amino proton at residue *i*+4. But this leaves several residues on each end of the helix without hydrogen bond partners. George Rose and Leonard Presta first observed that side chains will occasionally snake around so that they can satisfy these hydrogen bonds, a phenomenon called "helix capping."

In staphylococcal nuclease, many of the helices are solvent exposed, and water can satisfy the hydrogen bonds. Thus, no side chains are involved in helix capping. However, the N-terminus of helix 2 (residues 99-106) is buried in the core of the protein, and for this helix the capping problem is solved in an interesting way.

To visualize helix capping, we will need to add hydrogen atoms to the structure. In the object window, by the 1SNC object, click the action button (A). Then select "Hydrogens → Add." While not perfect, this routine creates hydrogen atoms based on the known bonding geometries from proteins. The backbone hydrogen atoms are of interest here, and you can select them by their atom name (H02 or *h-zero-two*). A helpful command to select the backbone atoms in a protein is:

```
select bb, name N+CA+C+O+H02
```

This command makes it much easier to visualizing backbone hydrogen bonds. The selection can be used in combination with the show command to examine hydrogen bonding patterns, e.g.

```
show sticks, bb and resi 99-106
```

6. Submit an image depicting the capping strategy for the N-terminus of helix 2 in staphylococcal nuclease, and briefly describe how the hydrogen bonds are satisfied. *Hint:* What residue acts as a hydrogen bond acceptor for the amino proton of residue Val 99?

Closing Thoughts

Much of our course will focus on thermodynamic and kinetic methods for studying how proteins function. For many years, our understanding of proteins was limited to these methods, and scientists were forced to speculate on the relationship between structure and function. It is a testimony to the efforts of many physical chemists that, by and large, scientists could make accurate predictions about protein function *before* the structures of enzymes were widely available.

While the wealth of structural information in the PDB has made it easier for us in a way, in some ways it complicates matters. The old adage "seeing is believing" often becomes "seeing is deceiving" when it comes to protein structures. Thus, while having the structure is useful, you must remember that the structures in the PDB are also models – typically very good models – but they are certainly not infallible. The experiments that biochemists perform complement the protein structures: kinetics experiments can help us to hypothesize which mechanism is favored by an enzyme, and structural analysis can be used to confirm or disprove those hypotheses. When the models are in conflict with one another, there is often something interesting to be learned.

PyMOL is a powerful tool to explore structure as it relates to binding, kinetics, folding, and all the other topics pertaining to protein and DNA chemistry. Even if you never perform a kinetics experiment yourself, you will be able to take the skills you have learned in this tutorial and examine the structural claims made by structural biologists and biochemists. Good luck, and happy hunting!