Tutorial on molecular visualisation

Contents

1 Introduction

1.1 Purpose of this tutorial

This tutorial intends to teach you some very simple techniques of molecular visualisation, skills that can be useful to most people working in a (bio)chemical area. In this way, you will not depend on a Molecular Modelling expert to perform these simple tasks.

The simpler skills you are expected to develop are viewing and manipulating molecular structures on the computer screen, and producing quality pictures for presentation or publication purposes. More sophisticated but often quite useful techniques are also introduced, namely building small molecules and making mutations in proteins. These are more advanced topics, but simple tasks can be easily done by plain users. If more detailed work turns out to be necessary, you can always ask an expert.

Although this tutorial is intended for a hands-on class, these notes are also intended to be used as a short reference guide for self-study, until you move on to the program manuals (if that ever happens). Thus, we hope that you "play" with the software tools yourself and eventually find

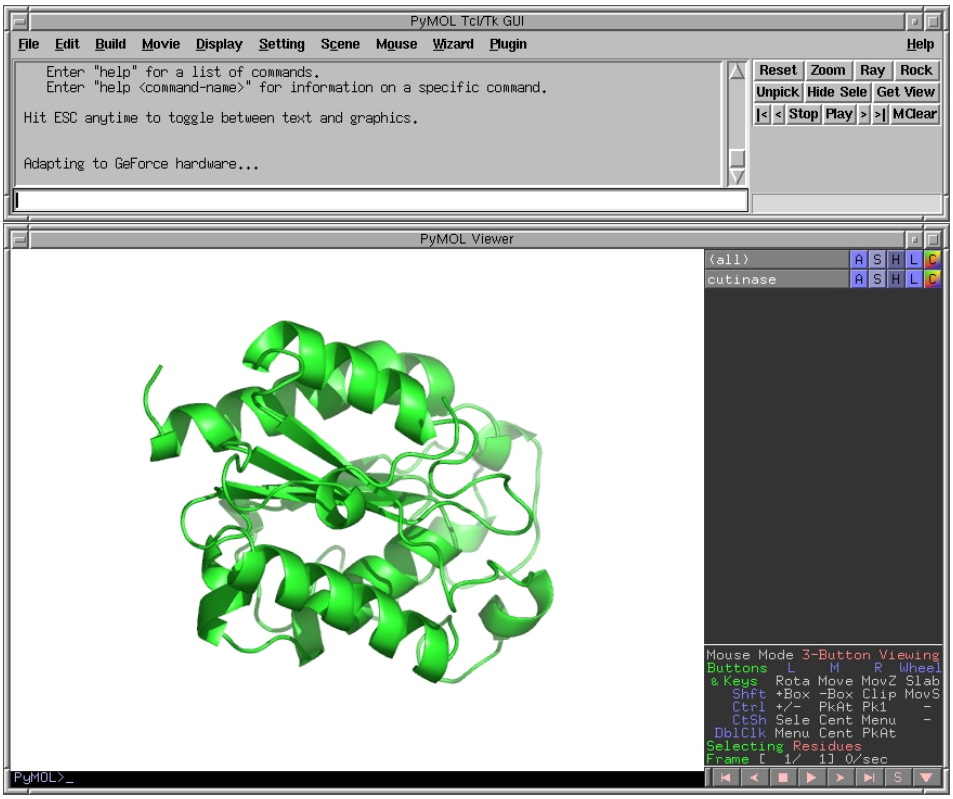

Figure 1: Snapshot of a PyMOL session.

out additional features not discussed here, which can be useful to your work. You can get the material for this tutorial at http://www.itqb.unl.pt/labs/molecular-simulation/education.

1.2 Software

This tutorial uses the molecular graphics program PyMOL (http://www.pymol.org). which is a robust, stable and widely used program that we recommend. The free 0.99 version is the one used in this tutorial (it is designated at PyMOL's website as "obsolete", but it works fine for the present purposes). The tutorial web page has PyMOL installation files for Windows, Mac and Linux (you can also download them from the official website). **Get the one you need for your laptop, install PyMOL and start/run it.**

A snapshot of PyMOL is reproduced in figure 1, showing that the interface consists of two windows:

Main window This window is labelled as PyMOL Viewer and contains a large area for displaying molecules. Below this displaying area there is a line where commands can be written.

There is also an area at the right-hand-side where many commands for displaying and manipulation can be easily accessed.

Top window This window is labelled as PyMOL Tcl/Tk GUI and contains the main program menus. Below the menu bar there is a message-reporting area with a command-line region underneath. At the right-hand-side there are buttons for executing some commands.

In what follows some more-or-less obvious notations will be used to designate keyboard keys and mouse buttons. Thus Ctrl, Shift, Alt, etc., refer to these keyboard keys, while Left-Button, Middle-Button and Right-Button refer to these mouse buttons. A designation like Ctrl + Middle-Button means you should press first the Ctrl key on the keyboard and then the middle mouse button. PyMOL also supports many powerful written commands, but that is not explored in this simple tutorial.

1.3 Structural databases

Although it may be easy to build small molecules (see section 2), quite often you need to obtain the structure of the molecule you are studying, either because your molecule is too big (e.g., a protein) or it is small but not obvious in terms of structure. In the present section we provide some very brief information on structural databases.

The place to look for protein structures is the Protein Data Bank (PDB), which can be freely accessed at http://www.rcsb.org. The PDB is easy to search and you can find there all the structures experimentally obtained so far (mainly by x-ray diffraction and NMR). The PDB contains also model structures obtained with comparative modelling methods (although they are not part of the default database); never forget that these are *not* experimental data, just presumably good guesses. If you know the 4-character PDB code of the structure you want, PyMOL can get it directly from the PDB site: on the PyMOL menu bar select Plugin → PDB Loader Service and then enter the code.

Many other structural databases exist, oriented towards specific subjects (small molecules, nucleic acids, ligand binding, etc) and featuring special search methods. Many of these databases are at least partially dependent on the PDB. If you think you need some specific database, you can start by looking at the PDB site, which has links to many databases.

2 Build and view small molecules

Although many molecular structures can be obtained from databases (see above), it is often necessary to look at a molecule for which there is no structure available, or whose building involves less work than finding it in a database. For these cases it is convenient to know how to build such molecules. The skills you are expected to develop with this building tutorial are very simple and aimed only at small molecules. The building of larger molecules has to deal with the fact that, as size increases, the molecule gains more flexibility (more torsion angles) and a structure build with the simple techniques described here becomes quite arbitrary: for each new torsion you have to make a decision. Much more sophisticated methods exist to model the structure of such large molecules, such as molecular dynamics and comparative modelling. The use of these specialised methods should be left to experts.

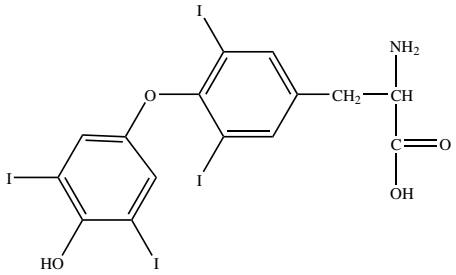

Figure 2: Structural formula of the T4 hormone.

2.1 Building an organic molecule

In this example we will use the hormone $3,5,3',5'$-tetraiodothyronine, usually referred to as T4. This hormone is produced by the thyroid gland in response to the thyroid-stimulating hormone released by the pituitary gland. Thyroid malfunctioning can be detected by measuring the T4 levels in the blood. The structural formula of T4 is shown in figure 2. Note that the right-hand side is similar to a tyrosine amino-acid, and therefore we may use a tyrosine as a starting point to build the T4 molecule. This use of other molecules or fragments usually makes building much easier.

1. Open PyMOL.
2. Start with a tyrosine amino acid residue: on the menu bar select Build → Residue → Tyrosine. A tyrosine residue appears on the screen, using a standard convention for atom colours: green for carbon, white for hydrogen, blue for nitrogen, and red for oxygen. You can select other colour schemes at the "rainbow" squares at the upper right corner.
3. Note that one of the carbon atoms comes with a ring-made sphere around it, meaning it is "picked" (see below). Press the Unpick button on the menu window to "unpick" everything.
4. Try using the mouse to move (Middle-Button), rotate (Left-Button) and zoom (RightButton) the molecule.
5. "Fill" valences in the tyrosine with hydrogen atoms: at the upper right corner, click (any mouse button) on the A in front of tyr and select hydrogens → add. This adds one hydrogen atom to the nitrogen at the N-terminus, and another to the carbon at the C-terminus.
6. Finish building the carboxyl group at the C-terminus.
   a) Pick (Ctrl + Middle-Button) the hydrogen atom just added at the C-terminus.
   b) Replace this hydrogen with an oxygen: on the menu bar select Build → Fragment → Oxygen. Note that the oxygen comes already "filled" with one hydrogen atom.

   This finishes building the tyrosine amino acid.

7. Add a phenyl group at the hydroxyl group attached to the carbon ring.
   a) Pick (Ctrl + Middle-Button) the hydrogen atom of the hydroxyl group.
   b) Add the phenyl group: on the menu bar select Build → Fragment → Phenyl.
8. Note that the atom colour codes are different in the phenyl ring you just added. You can change to a single colour scheme at the rainbow squares at the upper right corner. Each time a new fragment is added or a molecule is loaded (see below) PyMOL uses a different colour for the carbons. This is sometimes annoying, but you can always switch to the colours you want as just described.
9. You can recenter everything using (all) A → centre. This makes easier to control the motion of the molecule. Do this whenever you find it convenient.
10. Add a hydroxyl group at the tip of the new carbon ring.
    a) Pick (Ctrl + Middle-Button) the hydrogen atom at the tip of the ring.
    b) Add the hydroxyl group. Since added fragments come "filled" with hydrogen atoms, we just need to add one oxygen: on the menu bar select Build → Fragment → Oxygen.
11. Add iodine atoms. For each of the four hydrogen atoms to be replaced with iodine (see structural formula) do:
    a) Pick (Ctrl + Middle-Button) the hydrogen atom.
    b) Add iodine group: on the menu bar select Build → Fragment → Iodine.
12. The N- and C-terminal groups are neutral. However, in a solution of neutral pH they would tend both to be ionised (zwitterion). To change them do:
    a) Pick (Ctrl + Middle-Button) the "filled" oxygen in the C-terminal carboxyl group.
    b) Change the ionisation state: on the menu bar select Build → Make (pk1) Negative.
    c) Correct the "filling" of the valences: on the menu bar select Build → Fill Hydrogens on (pk1).
    d) Unpick everything (Unpick button on the menu window).
    e) Pick (Ctrl + Middle-Button) the nitrogen in the N-terminal amino group.
    f) Change the ionisation state: on the menu bar select Build → Make (pk1) Positive.
    g) Correct the "filling" of the valences: on the menu bar select Build → Fill Hydrogens on (pk1).
    h) Unpick everything (Unpick button on the menu window).
13. When building a molecule, some geometric properties may be set wrongly (i.e., the program is not "smart" enough). In the case of T4, the torsion around the final ring and the hydroxyl group is wrong, since the hydrogen atom should lie on the plane of the ring. To fix this do:

a) Switch the mouse mode from 3-Button Viewing to 3-Button Editing: click (with any mouse button) on the lower right panel (e.g., over Mouse Mode).

b) Pick (Ctrl + Right-Button) the C–O bond.

c) Grab the H atom with Ctrl + Left-Button and move the mouse to rotate around the bond, until the hydrogen atom lies on the ring plane. This is easier if you use the angle indication displayed on the screen each time a bond is selected.

d) Unpick everything (Unpick button on the menu window).

14. Following the same procedure, try changing other torsions. For example, you can change the torsion around one of the bonds connecting the two aromatic rings. This may be needed to make sure that the atoms of both rings are not too close together.

15. When you are satisfied with the final structure you can save it:

a) On the menu bar select File → Save Molecule.

b) You will be asked for the object or selection to be saved. Since your T4 molecule is still called tyr, select that one.

c) You will be asked for the file name and file type (the default type is PDB). Choose a name more appropriate than tyr.pdb, perhaps T4.pdb.

16. You are done with this exercise. Clean up things: at the upper right corner, click (any mouse button) on the A in front of (all) and select delete everything.

2.2 Building a peptide

To build peptides in PyMOL is quite simple. You just add the amino-acid residues in the order specified by the peptide sequence, specifying one of three standard secondary structures.

This exercise can be performed with any peptide you choose. An example of a very small peptide is kyotorphin (Tyr-Arg), a neuropeptide. An example of a large peptide is glucagon, a 29-residue hormone which stimulates the increase of glucose concentration in blood. For simplicity in building, you can use a peptide consisting of 10 alanine amino-acid residues. This can be regarded as a fragment of polyalanine, a well-studied example of helix–strand–coil transition, which adopts an α-helical conformation in a nonpolar environment and a β-type conformation in aqueous solution.

1. Build the peptide with uniform secondary structure:

a) On the menu bar select Build → Residue and choose one of the standard secondary structures allowed by PyMOL: Helix, Antiparallel Beta Sheet or Parallel Beta Sheet.

b) On the menu bar select Build → Residue and choose your first residue, Alanine. Repeat this step for all residues. Note that the chain is build with the specified secondary structure.

c) To finish the peptide, fix the N- and C-terminal groups (carboxyl group and ionisation states), as done above.

2. You can delete everything ((all) A → delete everything), select a different secondary structure, and rebuild the peptide (you can discard the final fixing of the terminal groups). Note the different final conformation of the peptide.

3. You can rebuild once more the peptide, this time switching to different secondary structures during the building process. This allows mixing different types of secondary structure in the same peptide chain.

4. Save your peptide in a PDB file (e.g., Ala10.pdb), as done above, and clean up things ((all) A → delete everything).

2.3 Molecular representations

Different molecular representations can be very useful in identifying particular features in a molecule. A good setting can usually be obtained by combining suitable representations for bonds, atoms and surfaces. This is important both for screen visualisation and for production of figures.

1. Load the T4 molecule you have build above: on the menu bar select File → Open, and then choose the corresponding file (T4.pdb).

2. Use a good graphical quality: on the menu bar select Display → Quality → Reasonable/Maximum Quality. (If your computer has not a good graphic card, this may cause move/rotate/zoom actions to become sluggish.)

3. The loaded molecule is drawn in the default way, as lines representing bonds. However, different representations can be used. To show or hide (S or H) each of them, go to the upper right corner, click (any mouse button) on the S or H in front of T4 and select one of the following:

   a) lines or sticks for bonds.

   b) dots or spheres for atoms.

   c) mesh or surface for surfaces.

4. When you are satisfied with the final representation, you can make a realistic "ray-traced" version of it: click the button Ray on the menu window. Note that if we move/rotate/zoom the molecule, you loose the ray-traced image (you have to hit Ray again).

5. Save this image in a file (PNG format): on the menu bar select File → Save Image, and then choose a file name (e.g., T4.png). This image can be included in most standard documents formats.

6. If you think that the black background is not appropriate for your presentation or publication, you can switch to a different one: on the menu bar select Display → Background → White (or one of the other colours). You can then repeat the last steps.

7. Clean up things ((all) A → delete everything).

3 View, mutate and compare proteins

Protein molecules have special needs in terms of visualisation and modelling techniques. Here we discuss how to view some particular structural features and the qualitative analysis of mutation effects.

3.1 Special molecular representations

Due to their size and complexity, protein molecules often require special visual representations in order to highlight the features of interest. These representations will be illustrated with cutinase, a small serine esterase produced by some pathogenic fungi to hydrolise cutin, a constituent of the cuticle that protects aerial plant organs.[1]

1. Load cutinase: on the menu bar select File → Open, and then choose cutinase.pdb.
2. You can use with cutinase the same general representation types used above for T4: lines, sticks, dots, spheres, mesh, and surface.
3. You can show and hide (S and H) the main chain and the side chains of the protein, for any of the representations.
4. You can also use some specific representations that apply only to proteins (best seen after hiding all other representations):
   a) The ribbon representation displays a smooth line following the protein main chain (i.e., the chain formed by the peptide bonds).
   b) After the assignment of the protein secondary structure has been done (cutinase A → assign sec. struc.)[2], the cartoon representation draws a schematic representation of the molecule.
5. If you want to make very fancy images, you may want to explore the options Cartoon, Transparency and Rendering under the Setting item at the menu bar. You can easily spend hours tweaking the different combinations, so don't mess with this unless you really need it. The default settings are perfectly good for most cases.
6. Your final image can be ray-traced and saved, as done above.

3.2 Stereo images

Stereo images are quite helpful in visualising the structure of proteins and other large molecules, and are becoming quite common in the literature. They consist of two slightly different images that your brain may perceive as a three-dimensional object. There are two modes of viewing stereo images:

[1] Suggestion: if you want to try later a really large molecular system get the chaperonin GroEL (PDB code 1AON) or the large ribosomal subunit (PDB code 1JJ2).

[2] Actually, the assignment of the secondary structure is usually done within the PDB file, and rarely needs to be done with PyMOL.

Cross-eye You cross your eyes so as to put the two images on the top of each other. (Hint: if you move a finger or pen from the screen towards you, the focusing will be correct when it is 10–15 cm from your eyes.)

Wall-eye (or Wide-eye or Parallel-eye) You look "through" the image, as if you were focusing a distant point. (Hint: avoid each eye to see the image at the other side by putting a sheet of paper perpendicular to the screen, from your nose to the screen.)

Seeing stereo images without using special glasses or lenses requires some training. If you can see decorative stereograms, you should be able to see in wall-eye mode. Some people can see in both stereo modes, others in just one, and some are never able to see in any mode.

1. To activate stereo viewing go to the menu bar and select Display → Stereo, which displays a cross-eye stereo image of the molecule.

2. To switch to wall-eye stereo, go to the menu bar and select Display → Stereo Mode → Wall-Eye Stereo.

3. As done above, you can ray-trace this stereo view and save it to an image file.

3.3 Making point mutations

It is often important to have an idea of the structural effect of a point mutation in a protein. Specialised modelling methods exist to allow relaxation of the structure around the mutated residue, but these require more advanced methods. Fortunately, useful qualitative information can often be obtained by simple replacement of the residue and a visual analysis of its structural effect. This can be very useful in planing mutation experiments.

This example uses again the structure of cutinase, obtained with an inhibitor at the active site (N-hexylphosphonate-2-ethyl-ester). The purpose is to mutate a tyrosine residue near the active site, so that the protein may accommodate a larger substrate.

1. Load the inhibitor from the file inhibitor.pdb, which allows you to easily locate the active site of cutinase (hint: use different representations and/or colour codes for cutinase and the inhibitor).

2. Identify the tyrosine residue near the active site. If you have trouble finding it, you can try the following:
   a) Hide the main chain in order to more easily visualise the side chains.
   b) Display the protein sequence by selecting Display → Sequence on the menu bar. Now you can select the tyrosine residues by clicking over them in the sequence bar. Remember that Y is the one-letter code for tyrosine.
   c) If you still have trouble deciding which tyrosine seems closer to the inhibitor, measure the distances between them. Select Wizard → Measurement on the menu bar and you will be asked to pick the atoms that you want to measure the distance between. To finish, click over Done.

d) If you still cannot find that tyrosine, click on residue 119 in the sequence bar.

3. On the menu bar select Wizard → Mutagenesis.

4. Pick (Ctrl + Middle-Button) the tyrosine residue. A white copy of the residue appears with a different conformation of the amino-acid side-chain (see footnote 3) and surrounded by red disks; ignore it.

5. Click under the green header Mutagenesis (which shows No Mutation), select an alanine (ALA) and click Apply and Done. This replaces the tyrosine with an alanine (whose carbon atoms are now displayed in white).[3]

6. Save your mutant protein molecule as mutant.pdb, and then delete it (cutinase A → delete object).

7. Load both cutinase.pdb and mutant.pdb, and represent them both as surfaces.

8. If you now click over cutinase at the upper right corner, this will switch off the display of the wild-type cutinase, showing only the mutant you just made. By clicking again over cutinase, it is again displayed. In this way you can effectively switch between the wild-type and the mutant, showing the large cavity created by the mutation.

9. Select an orientation that clearly displays the difference between the wild-type and the mutant. Hide the mutant and save a ray-traced image of the wild-type. Hide the wild-type and show the mutant, and save also a ray-traced image of it (keeping the same orientation). These images could be used to illustrate the (approximate) effect of the mutation.

10. Clean up things ((all) A → delete everything).

3.4 Comparing similar proteins

It is often important to make a visual comparison of proteins that are similar in terms of structure and/or function, because it may help you to rationalize about their structure–function determinants. There are specialised modelling methods to do this, but a simple visualization can be done using PyMOL, at least for proteins sharing a reasonable amount of homology.

This example uses again cutinase. The structure you used in the previous sections belongs to *Nectria haematococca* Mpvi[4] and corresponds to the PDB entry 1XZL. The purpose of this section is that you compare the 1XZL structure with two other cutinase structures, one from *Glomerella cingulata* (PDB code 3DCN) and another from *Aspergillus oryzae* (PDB code 3GBS).

[3] If you choose an amino-acid besides alanine, several side chain orientations (rotamers) are possible. You can use the back-and-forth movie controls (lower right corner) to display (in white) each of the rotamers available for this residue in PyMOL, whose current and total numbers are shown in the (green) Frame info. The rotamers are ordered according to their frequencies of occurrence in proteins, shown as a green percentage at the mutation object, which exists while mutagenesis is being performed. After selecting the rotamer you think better fits your structure, proceed to Apply and Done the mutation, as above.

[4] Note that this organism was previously called *Fusarium solani* pisi.

Since the entry 1XZL has both the protein and inhibitor structures, they were split beforehand for the previous sections of the tutorial. But, to better simulate a real-life situation, here you will load the complete structures (including 1XZL) directly from the PDB.

1. Load the 1XZL cutinase structure from the PDB: on the menu bar select Plugin → PDB Loader Service and then enter the code 1XZL.

2. Load the 3DCN structure from the PDB in a similar way and then enter the following command in one of PyMOL's windows:

   ```
   align (3DCN and name ca), 1XZL
   ```

   This makes a sequence and structural alignment of 3DCN onto 1XZL. The structural alignment is done using the C_α atoms of the peptide chain ("name ca").

3. Repeat the previous step for the 3GBS structure. The three structures are now aligned. To facilitate their comparison you can start by removing all water molecules and hydrogen atoms (see the A options).

4. Compare the structure of the main chain of the three molecules using a main-chain representation (such as cartoons) and find the major structural differences between them. **Are those differences located at regions with well-defined secondary structure or at loops?**

5. Locate the active site (you may use the inhibitor to do that). **Is any of the major main-chain structural differences located near the active site?**

6. To compare the overall shape of the molecules you can switch to an all-atom representation (such as lines or surface). **What do you gain by using these representations?**