STEVE'S PYMOL CHEAT SHEET

INTRODUCTION

Part of what we do within the Protein & Structure department at Sygnature Discovery is to generate crystal structures of proteins, and this generates an estimate of the coordinates for all the atoms in that protein. However, X-ray crystallography is not the only method to determine protein structures, with electron microscopy and nuclear magnetic resonance also contributing to the experimentally determined structures. Furthermore, with the recent advent of DeepMind's AlphaFold2 for the accurate prediction of structure from protein sequence, and the release of a massive database of predicted protein structures from several species in collaboration with EBI, there are now even more structural models for proteins available. These models are important to us, not only because we help generate them for our clients, but also because structures of proteins can assist in understanding the biochemical function of a protein and help us design better constructs for e.g. purification.

Protein structures have complex spatial and chemical properties that need to be represented in a visually digestible manner. To better understand what the structure tells us and to make publication quality figures for reports, we would often like to focus on one particular part of the structure in detail e.g. a compound binding site, but remove the rest of the complexity of the electron density and the polypeptide chain in the protein. For this purpose, we use the software programme PyMol (https://pymol.org/2/ ; available as an open source version (https://github.com/schrodinger/pymol-open-source), which can display the protein in a shorthand representation for alpha-helices and beta-sheets that we call a 'cartoon' representation. With this simplified view of the protein, complexity can then be added back sparingly to highlight only the most important parts. This is the most common way to see protein structures depicted in scientific publications. Even though viewing models with PyMol is very easy, and the default settings will already provide a good view, I find that with some extra tweaks to the settings, the images can be made to look far more visually appealing. Please read on to find out my recipe for creating great images with PyMol.

THE BASICS

1. Navigating the GUI and typing commands to the console

The software can initially be quite confusing as there are numerous different control panels, all with lots of different settings to play with (Fig. 1), some more useful than others. There are many things that you can do by directly clicking, dragging or selecting settings through menus. Particularly for new users, the model and object properties panel will be the most useful, as this will allow the showing and hiding of certain visual representations, and recolouring of the different models to suit. However,

PyMol also provides the powerful feature of the command line, that when used by experienced users can unlock a whole host of settings and shortcuts to enhance the images that are displayed on the screen. There is a comprehensive users guide available online that documents all the commands available, and much of what follows is a parred down set of controls from that document (https://pymolwiki.org/index.php/Main_Page)

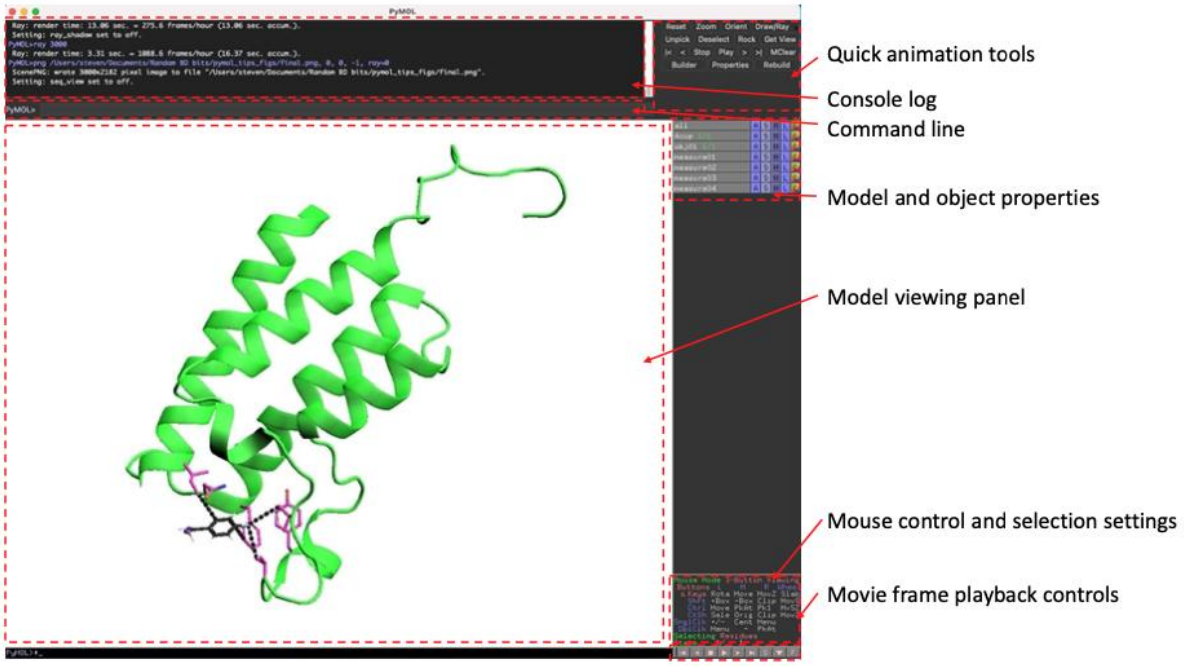

Figure 1 – The anatomy of the PyMol GUI

2. Background colour

Depending on whether you are viewing a structure on a screen or making a figure for a report you may want the background coloured differently. For creating figures, I always set my background to white to provide high contrast. The background colour can be changed by navigating to Display -> Background -> White or simply typing the following magic command into the command line: `cmd.bg_color('white')`. This is a clear example of where knowing the command line shortcut can speed-up using PyMol.

3. Shadows

By default, the models in PyMol act as though they are being shined on by a light source, and parts of the protein that are in front will cast a shadow on those parts behind. For viewing the model and understanding the depth and distance this can be quite useful. However, in a published figure, the shadows can add complexity and confuse the image, obscuring the important detail you wanted to highlight. Therefore, my preference is to turn the shadows off for making figures.

REPRESENTATIONS

1. Cartoon representation

The cartoon representation is a visually appealing way to represent the alpha-helices, beta-sheets and loops of a protein. The alpha-helices look like spiralled ribbons, the beta-sheet are flat arrows pointing in the direction from N- to C-terminus and the loops are strands of spaghetti. My personal preference for the cartoon settings is to make the ribbon even narrower, thinner and flatter than default, almost as though the structure is made of a paper or fabric ribbon (Fig 2). The settings required for this are the cartoon_oval_length and cartoon_oval_width.

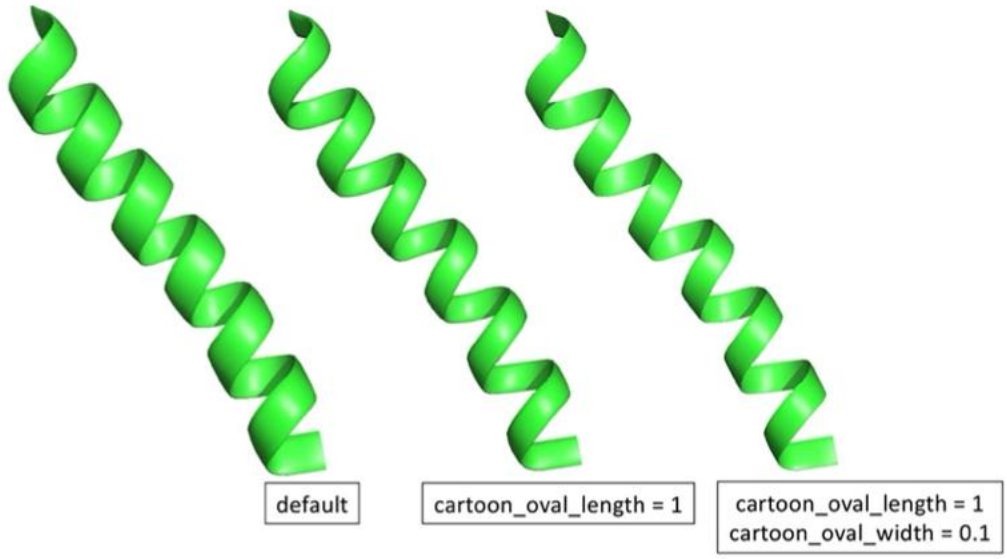

Figure 2 – Cartoon representation settings

2. Stick representation

Another important representation that allows more chemical detail to be displayed is the 'stick' representation. In this setting the covalent bonds between atoms are displayed as thin cylinders. This setting reveals the chemical detail of the structure and is useful for highlighting important side-chain functional groups, or small-molecules bound to proteins. My preference is to make the sticks even thinner than default while creating a 'bulge' where the atoms sit (Fig. 3). These settings are stick_radius, stick_ball and stick_ball_ratio.

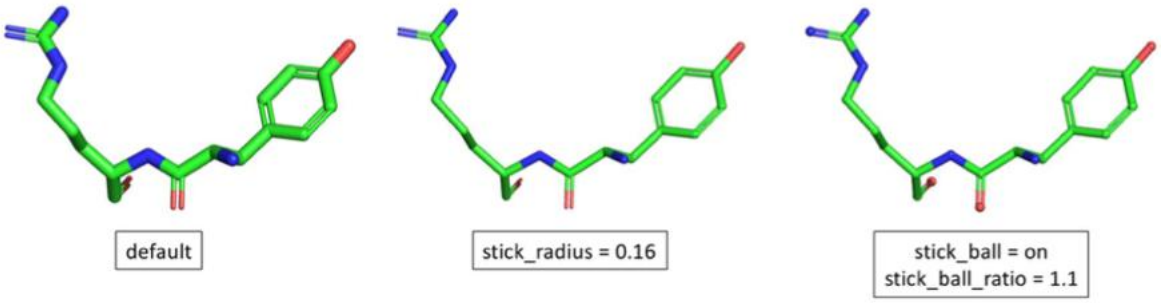

Figure 3 – Stick representation settings

3. Distance representation

The final representation that I will highlight, and like to modify from the default settings is the distance representation. The distance representation can be used in a number of different ways, sometimes highlighting the distance between two particular atoms of the model, but often used to represent important bonding properties between two atoms. I find that by default the dashed line that is rendered between the atoms is too thin and can be difficult to spot in complicated figures. My preference is to increase the weight of this dashed line using the settings dash_radius, dash_gap and dash_width (Fig. 4).

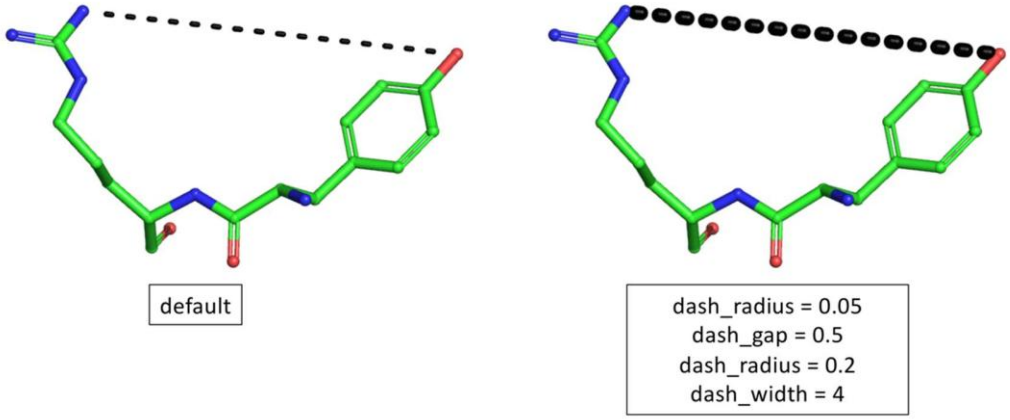

Figure 4 – Distance representation settings

SCENE LIGHTING AND RENDERING

1. Lighting

As mentioned, the scene in which PyMol renders objects acts as though there is a light source illuminating the model and casting shadows. There are many settings that control different aspects of how the light enters the scene and how the object reacts to the light. Personally, I find the default settings PyMol uses looks almost like the model is made of shiny plastic. I prefer to make the material of the model look more like it is made of matt plasticine or fabric, removing all light reflection from the model (Fig. 5).

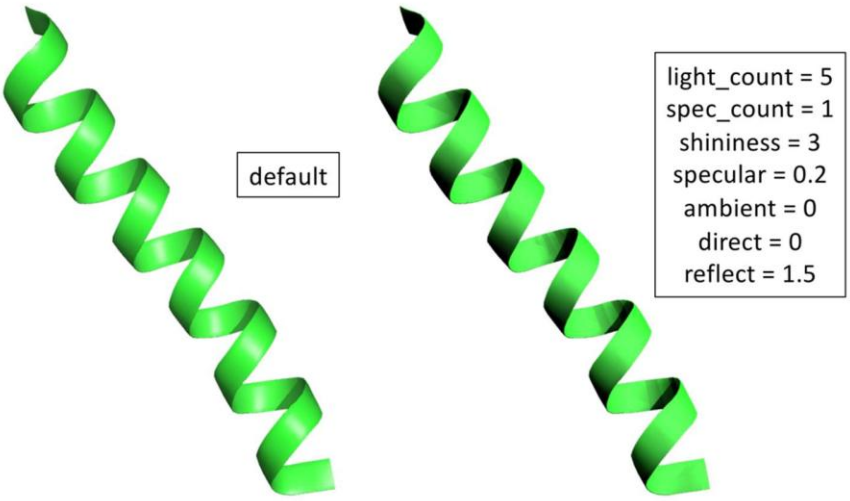

Figure 5 – Lighting settings

2. Rendering

Once all settings have been modified and you have a view of the model you would like to make a figure from, we're not quite done yet. The view provided in the viewport of the GUI looks pretty good, but the effect of the light is simulated in a very simple manner and the resolution is only that of the screen. In order to include the best lighting effects and to specify the quality of the image to be saved, we need to activate ray casting (literally casting rays of light from the light source to the camera). The required command is 'ray', and by typing a number after this, the number of pixels along the longest edge of the image will be set. For medium quality but faster processing, I use ray 1000. For high quality figures I use ray 3000. However, be aware that for very complicated scenes with lots of model or one in which transparency is active this can take a very long time depending on the specification of the computer you are using.

PUTTING IT ALL TOGETHER

Once you have the final image composition, and have calculated the ray casting, you can save the image to a .png. It is then usual to add annotation to the figure so that readers can make sense of the information you are showing them (Fig. 6). Any image editing software could be used for this, but personally I find Powerpoint works great!

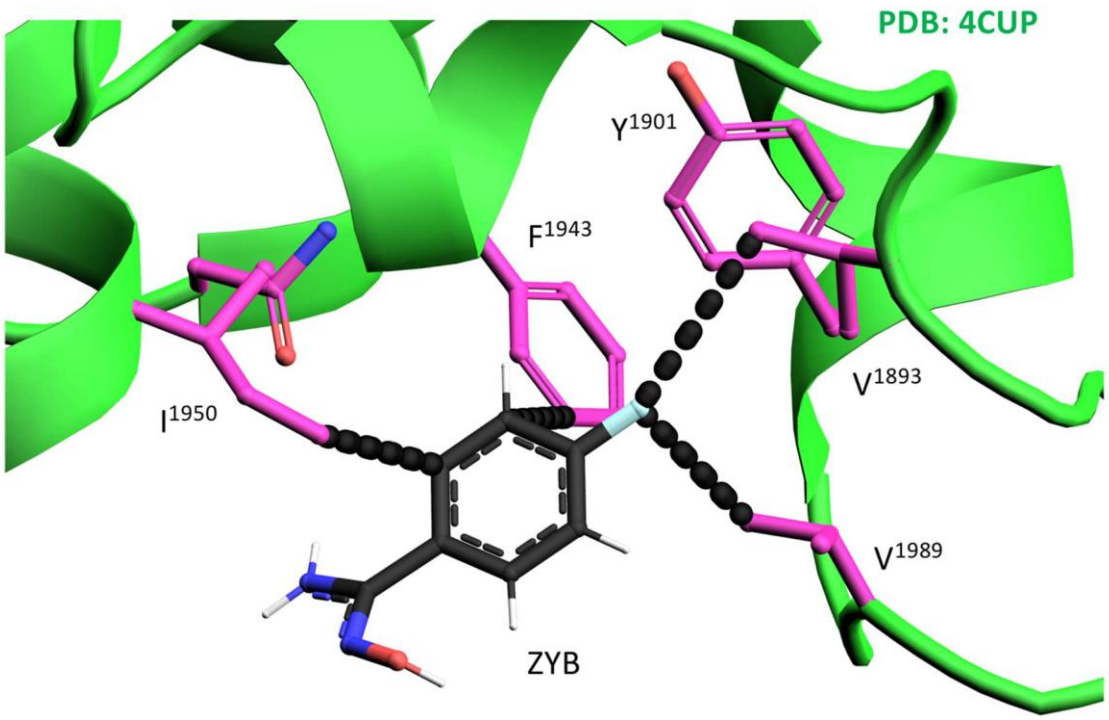

Figure 6 – Final product

CHEAT SHEET

Here are all the settings I altered to make this particular figure as outlined above. These settings can all be applied in one go by copying and pasting them into the command line box.

```
cmd.bg_color('white')
set cartoon_oval_length, 1
set cartoon_oval_width, 0.1
set stick_radius, 0.16
set stick_ball, on
set stick_ball_ratio, 1.1
set dash_gap, 0.50000
set dash_radius, 0.20000
set dash_width, 4
set light_count,5
set spec_count,1
set shininess, 3
set specular, 0.2
set ambient,0
set direct,0
set reflect,1.5
set ray_shadow_decay_factor, 0.1
set ray_shadow_decay_range, 2
set orthoscopic, on
cartoon automatic
set depth_cue, 0
```