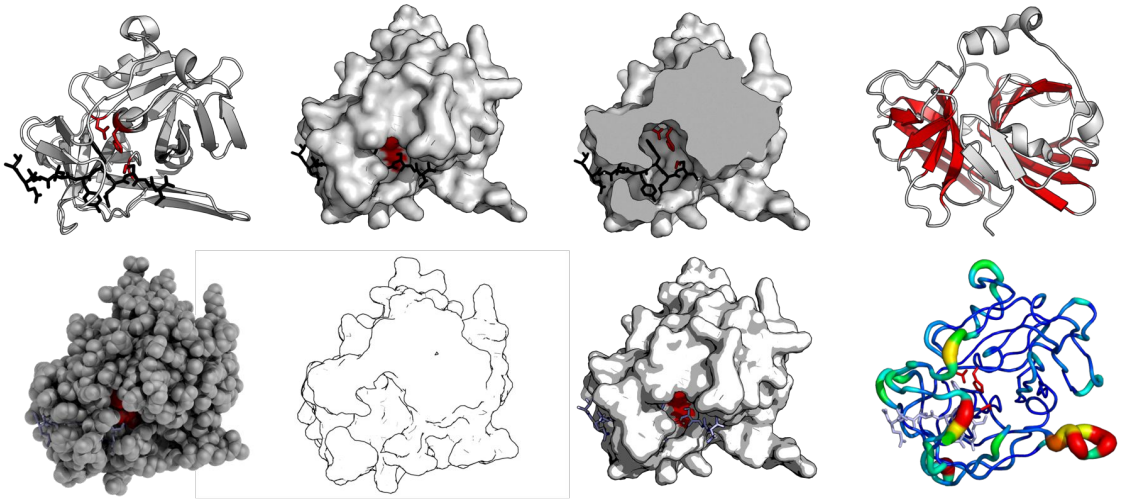
Structure visualization with Pymol

Thomas Hamelryck
thamelry@binf.ku.dk
BIO/DIKU, University of Copenhagen
November 2019

Getting Pymol

- http://www.pymol.org
- Educational version: https://pymol.org/edu/?q=educational/
- On Linux: https://pymolwiki.org/index.php/Linux_Install



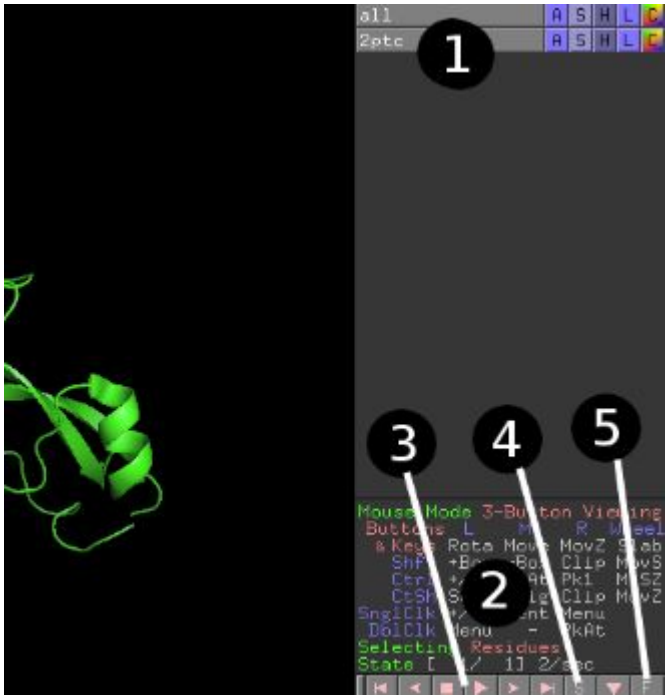
Pymol documentation

- Community Wiki
  - http://www.pymolwiki.org/
- Tutorial on publication quality figures with Pymol
  - http://www.bch.cuhk.edu.hk/kbwong/pymol/pymol_tutorial.html
- Google!

Navigating GUI

Download from PDB: `fetch 2ptc`

1. Objects and selections
   a. Menu: action, show, hide, label, color
2. Mouse actions
3. State navigator for multiple models
   a. Try it on 1adz, for example
4. Toggle show sequence
5. Toggle full screen



Different representations

- Control using H(ide) and S(how)
  - Lines, Ribbon, Cartoon, etc.
- Cartoon
  - Color cartoon
  - Cartoon options (Setting:Cartoon:helices as cylinders, etc.)
  - Ray trace cartoon
- Surfaces
  - Mesh, Surface
  - Color by element
  - Color by chainbows

Exercise

- Download 1FAT from the PDB
- Orient automatically
- Color by chain (1FAT has 4 chains)
- Hide all representations
- Show surface
- Ray trace and save image

Exercise

- Download 1FAT from the PDB: **fetch 1FAT**
- Orient automatically: **A→Orient**
- Color by chain (1FAT has 4 chains): **C→by chain**
- Hide all representations: **H→everything**
- Show surface: **S→surface**
- Ray trace and save image: **Draw/Ray** (upper right)

Pymol Command line

- <command> <argument>, <selection(s)>
- Commands:
  - `show, hide, as` (arg is representation, e.g. cartoon)
  - `select` (arg is name, e.g. my_selection)
  - `color` (arg is color, e.g. red)
- `show dots, resi 8`
- `select myres8, resi 8`
- `color red, resi 8`
- `hide sticks, resi 2`
- `as sticks, * # as will turn everything else off`

Pymol command line

- Ray trace: ray
- Save image: save <filename>
- help <command>
- <command> ?
- Commands are in lower case
- Try tabbing

Selections

- Central and powerful tool in Pymol
- Identifier (PDB file format)
  - `name <atom name>  # e.g. ca`
  - `resn <residue name> # e.g. ala or hoh`
  - `resi <residue number> # e.g. 5 or 4-6`
  - `chain <chain identifier> # e.g. a`
- Combine with logical operators: `and, or, not`
- Example:
  - `select my_domain, chain e and resi 45-65`
- `help selections`

More selections

- Select neighborhood: around, within
  - `select env, (resi 195 and chain e) around 10`
  - `select interface, enzyme within 3.5 of inhibitor`
- Shorthands
  - `select env, (i. 195 & c. e) a. 10`
- Presets
  - `all, visible, hetatm`
- Many more
  - see `help selection`

More selections

- Selections using “/”
  - `select test1,/2ptc//I/25/CA`
  - `select test2,/2ptc//E`
  - `select test3,/2ptc//I/*/CA`
  - `select test4 /2ptc//E/57+102+195`
- `/molecule/segment/chain/residue/atom`

Distances and labels

- Measure distances
  - `dist <name>, <selection1>, <selection2>, <cutoff>`
  - `dist interface_dist, chain e, chain i, 3.5`
- Text label atoms
  - `label <selection>, <label(s)>`
  - `label name ca, resi`
  - `label (i. 102 & c. e), "%s%s-%s" % (resn,resi,name)`

Command examples

- `load 2ptc.pdb # load local file, otherwise “fetch”`
- `show spheres, resn hoh`
- `color red, elem S`
- `orient 2ptc`
- `pseudoatom pcenter, pos=[10.42, 65.13, 19.75]`
- `delete 2ptc`
- `delete *`

Options

- `set / unset <flag>, <value>`
- `set sphere_scale, 0.3`
- `set stick_radius, 0.3`
- `set label_color, black`
- Many, many more...

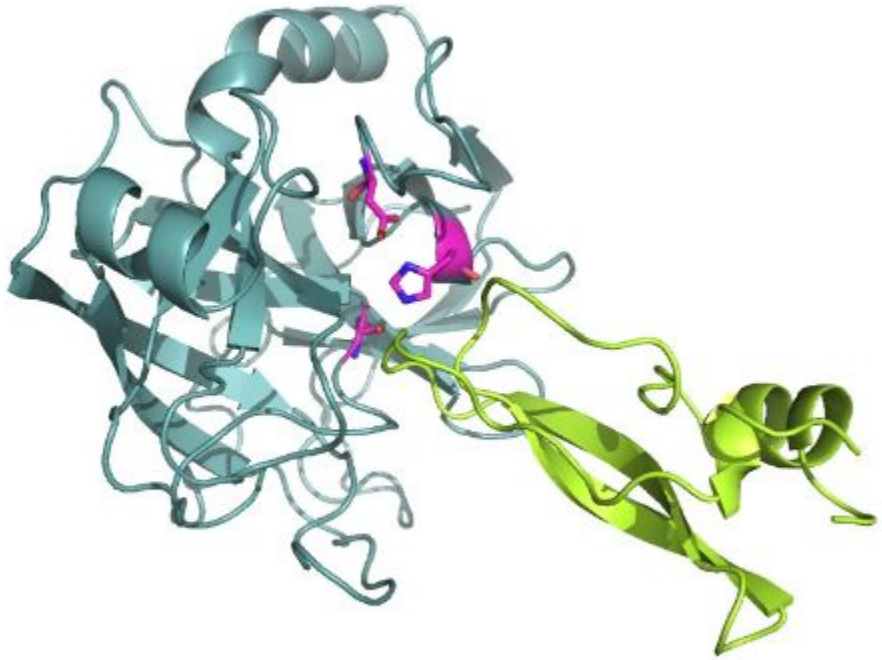
Exercise - use commands and selections

- Delete all and download 1ENH
- Show ribbon representation
- Make yellow
- Select side chain atoms and ca atoms of all aromatic amino acids (F,Y,W,H):
- Visualize aromatics as sticks on ribbon
- Color aromatics red
- Ray trace and save image

Exercise - use commands and selections

- Delete all and download 1ENH: **delete *; fetch 1enh**
- Show ribbon representation: **hide all; show ribbon**
- Make yellow: **color yellow, all**
- Select side chain atoms and ca atoms of all aromatic amino acids (F,Y,W,H):
  - **select ar, (sidechain or name ca) and (resn his or resn tyr or resn trp or resn phe)**
- Visualize aromatics as sticks on ribbon
  - **show sticks, ar**
- Color aromatics red: **color red, ar**
- Ray trace and save image: **ray; save exercise2.png**

2PTC - Trypsin and its inhibitor



Superposition

- `fetch 1trn`
- `fetch 2ptc`
- `align 1trn, 2ptc, object=alignment`
- If you need more control, use pair_fit
  - `pair_fit (selection 1), (selection 2)`
  - Will fit molecule 1 on molecule 2 using the selections

Exercise - superposition

- Using pair_fit, superimpose the catalytic triad of 1trn (chain A) on the catalytic triad of 2ptc (chain E)
    - Catalytic triad: His 57, Asp 102, Ser 195

Exercise - superposition

- Using pair_fit, superimpose the catalytic triad of 1trn (in chain A) on the catalytic triad of 2ptc (in chain E)
    - Catalytic triad: His 57, Asp 102, Ser 195
- **fetch 1trn; fetch 2ptc**
- **select 1trn_triad, 1trn and chain A and (resi 57 or resi 102 or resi 195)**
- **select 2ptc_triad, 2ptc and chain E and (resi 57 or resi 102 or resi 195)**
- **pair_fit 1trn_triad, 2ptc_triad**

NMR structures

- `fetch 1adz`
- NMR structures often consist of several models
  - Step through the models using the state navigator
  - (bottom right)
- Turn models into seperate objects
  - `split_states 1ADZ`
  - **`Try it out. What is happening?`**

Pymol in python

Example in pymol...

```
load 2PTC.pdb
color red, chain E
color green, chain I
show surface, 2PTC
```

...becomes in python:

```
from pymol import cmd
cmd.load("2PTC.pdb")
cmd.color('red', 'chain E')
cmd.color('green', 'chain I')
cmd.show('surface', '2PTC')
```

Execution :

- File -> run
- `run <script name>`

Python example: loop over atoms

```
molecule=cmd.get_model("2ptc")

#iterate over atoms and print information
for a in molecule.atom:
    print(a.coord) # xyz coordinates
    print(a.b) # temperature factor
    print(a.chain) # chain id
    print(a.name) # atom name
    print(a.resi) # residue number+chain id
    print(a.resn) # residue name, eg. ALA
    print(a.hetatm) # 1 if amino acid, 0 otherwise
```

Extending Pymol with new commands

- File -> run
- `run <script name>`

```
from pymol import cmd

def surfwat(obj_id):
    cmd.color('green', obj_id)
    cmd.color('blue', obj_id+' and resname HOH')
    cmd.show('surface', obj_id)
    cmd.show('spheres', obj_id+' and resname HOH')

cmd.extend('surfwat', surfwat)
```

Extending Pymol with new commands

- Run the command in Pymol
  - `surfwat 2ptc`
- You can add a command to PyMol on startup
  - Put “`run script.py`” in your `.pymolrc` file in your home directory

Exercise

- Make your own command called **surf_it** that
  - Downloads a PDB file from the PDB using its identifier
  - Visualizes the protein surface in white
  - Visualizes the waters as blue spheres
  - Usage example: surfwat 2PTC

Script examples: alternating colors

```
from pymol import cmd

#Give residues alternating colors
for res_id in range(1, 59):
    sel='chain I and residue '
    sel=sel+str(res_id)
    if res_id % 2:
        color='red'
    else:
        color='blue'
    cmd.color(color, sel)
```

Compiled graphics objects (CGOs)

- Based on OpenGL
- Simple comma separated list of floats
- Few graphics primitives:
  - `LINES, TRIANGLES, SPHERE, CYLINDER`
- Housekeeping primitives:
  - `BEGIN`
  - `COLOR`
  - `VERTEX`
  - `END`

Compiled graphics objects (CGOs)

```
from pymol.cgo import *
from pymol import cmd
obj = [ BEGIN, LINES, COLOR, 1.0, 1.0, 1.0,
    VERTEX, 0.0, 0.0, 0.0,
    VERTEX, 1.0, 0.0, 0.0,
    VERTEX, 0.0, 0.0, 0.0,
    VERTEX, 0.0, 2.0, 0.0,
    VERTEX, 0.0, 0.0, 0.0,
    VERTEX, 00, 0.0, 3.0,
    END]
cmd.load_cgo(obj,'cgo01')
```

Compiled graphics objects (CGOs)

```
from pymol.cgo import *
from pymol import cmd
# Build up a CGO object
obj=[BEGIN]
# Lines
obj.append(LINES)
# first line
obj.extend([COLOR, 1.0, 0.0, 0.0])
obj.extend([VERTEX, 0, 0, 0])
obj.extend([VERTEX, 10, 0, 0])
```

Compiled graphics objects (CGOs)

```
# second line
obj.extend([COLOR, 0.0, 1.0, 0.0])
obj.extend([VERTEX, 0, 10, 0])
obj.extend([VERTEX, 0, 0, 10])

# Sphere
obj.extend([COLOR, 0.0, 0.0, 1.0])
obj.extend([SPHERE, 0, 0, 0, 40])
```

Compiled graphics objects (CGOs)

```
# Cylinder
obj.extend(
[CYLINDER,
    40, 0, 0,    # xyz 1
    80, 80, 80, # xyz 2
    10,          # radius
    1,0,0,       # color 1
    0,1,0])      # color 2
# Close
obj.append(END)
# Build cgo
cmd.load_cgo(obj,'CGO Object')
```

Assignment

2PTC is a PDB file containing trypsin (chain E) and its inhibitor (chain I). The active site of trypsin includes residues 57, 102 and 195.

Download the PDB file and visualize the enzyme/inhibitor complex, showing the active site and the inhibitor residues (within 4 Å) around it. Write a fully automated pymol script to make the figure. Pymol scripts have the extension **.pml** and can be executed using **"File -> Run Script"**. Make the picture as visually appealing, striking and clear as you can using pymol's functionality Hand in a single PDF file with

- the image
- a caption (description) for the image
- the pymol script