Tutorials for Use of LigandMapper.py

Table of Contents

Setting up the environment

This program is designed to be easy to use and intuitive. So the commands are very easy to comprehend and understand.

First check if you have the python3 and java on your Linux/Mac machine:
https://github.com/niqolla/LigandMapper.py#requirements

Then, follow the instructions on github to install the LigandMapper.py package:
https://github.com/niqolla/LigandMapper.py#installation-of-lignadmapperpy

In case you run into an issue with the installation, you can mannually install the package:
https://github.com/niqolla/LigandMapper.py#manual-installation--in-case-instalmepy-fails-

Next, download the tutorial folder (LigandMapper.py_tutorial_folder) and unzip it.
https://formacio.bq.ub.edu/~u217733/LigandMapper/tutorials/LigandMapper.py_tutorial_folder.zip

Now, we're ready to begin.

Tutorial 1: Analysis of a single PDB file

In this tutorial we will be going over a simple pocket prediction for a local PDB file (already downloaded to your system) and a method of obtaining the PDB file and analyzing it all at once.

1. Change directory to tut_1/

Here we only have 1 pdb file of the entry 1gln.

2. Run: *LigandMapper.py -l 1gln.pdb*

The program should finnish in less then a minute, creating a new subfolder in the tut_1/ dir called predict_1gln/ and printing the following to the standard error:

File found 1gln.pdb
Analysing 1gln.pdb structure...
Removing temporary directories
Creating visualization files
Creating TSV file
Output stored in: /home/nikola/Desktop/LigandMapper.py_tutorial_folder/tut_1/predict_1gln

3. Open the folder and open the 1gln.pdb_predictions.tsv file in Excel/LibreOffice Calc.

You should see that the program predicted 2 pockets sorted by their rank. In addition you will also see score and probability of these pockets. For more information about these 3 attributes, please see the theoretical background pdf file (https://formacio.bq.ub.edu/~u217733/LigandMapper/tutorials/theory.pdf).

Moreover, the rest of the variables are the following, with their corresponding explanation:

- sas_points - (int) number of solvent accessible surface points
- surf_atoms - (int) integer of the number of surface atoms
- center_x - (float) the predicted pockets x center
- center_y - (float) the predicted pockets y center
- center_z - (float) the predicted pockets z center
- residue_ids - (py dict) the residue sequence numbers that create the pocket { Chain : [residue sequence numbers] }
- residue_names - (py dict) the residue names that create the pocket { Chain : [residue names] }
- residue_types - (py dict) the character of the residues that create the pocket { Chain : [characters] }
    - 'N' represents non-polar amino acids
    - 'P' represents polar amino acids
    - '+' represents positively charged amino acids

- '-' represents negatively charged amino acids
- '0' a specific residue for which there is no info in the program

- surf_atom_ids - (py list) the atom serial number of all the atoms that are on the surface of the pocket

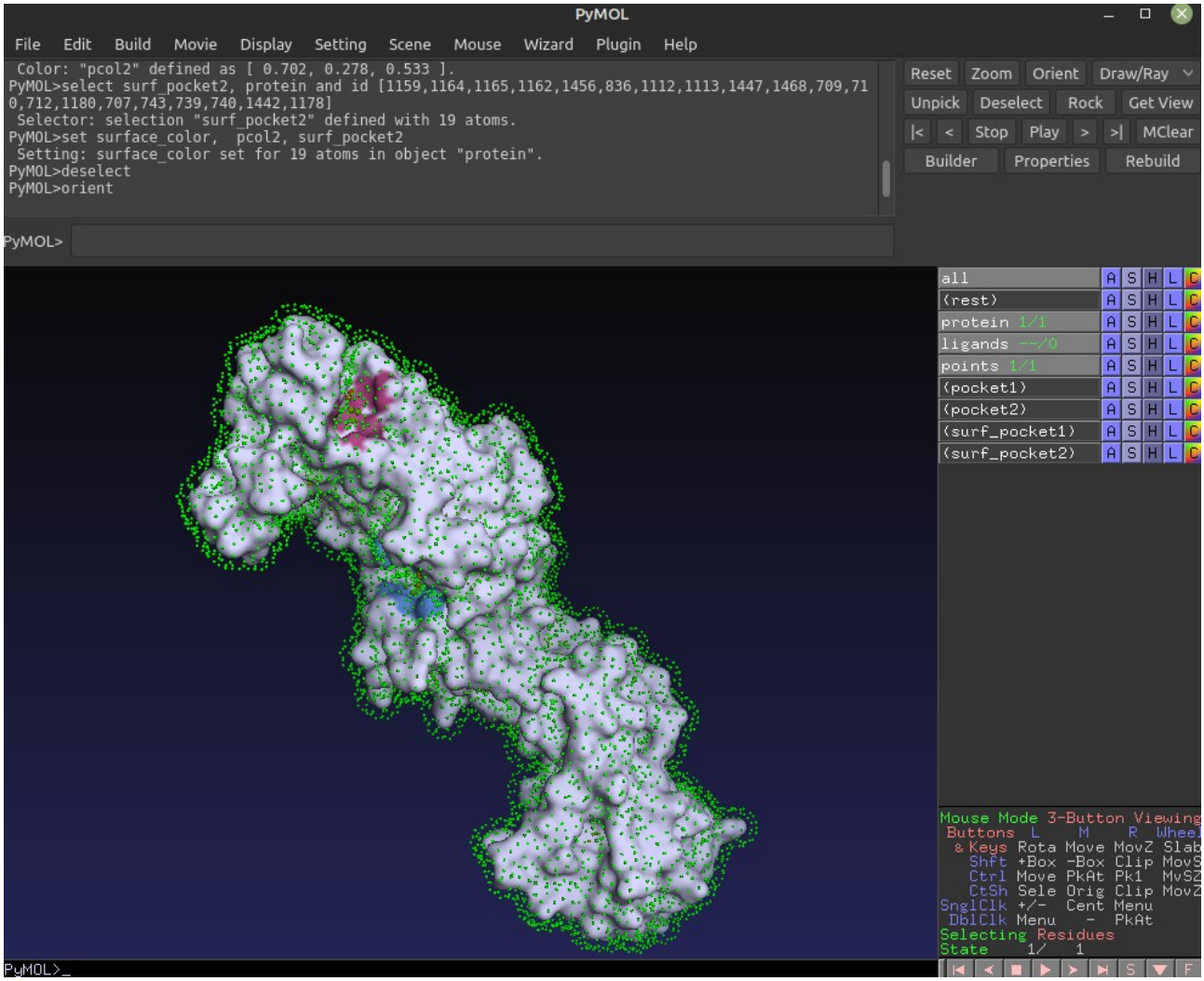
4. Open the visualizations/ directory.

Here you will see :

- one file for visualization in PyMol – 1gln.pdb.pml,
- one file for visualization in Chimera – chimera_1gln.cmd,
- and a folder named data/. This folder has necessary data for the PyMol visualization.

5. Open 1gln.pdb.pml with PyMol.

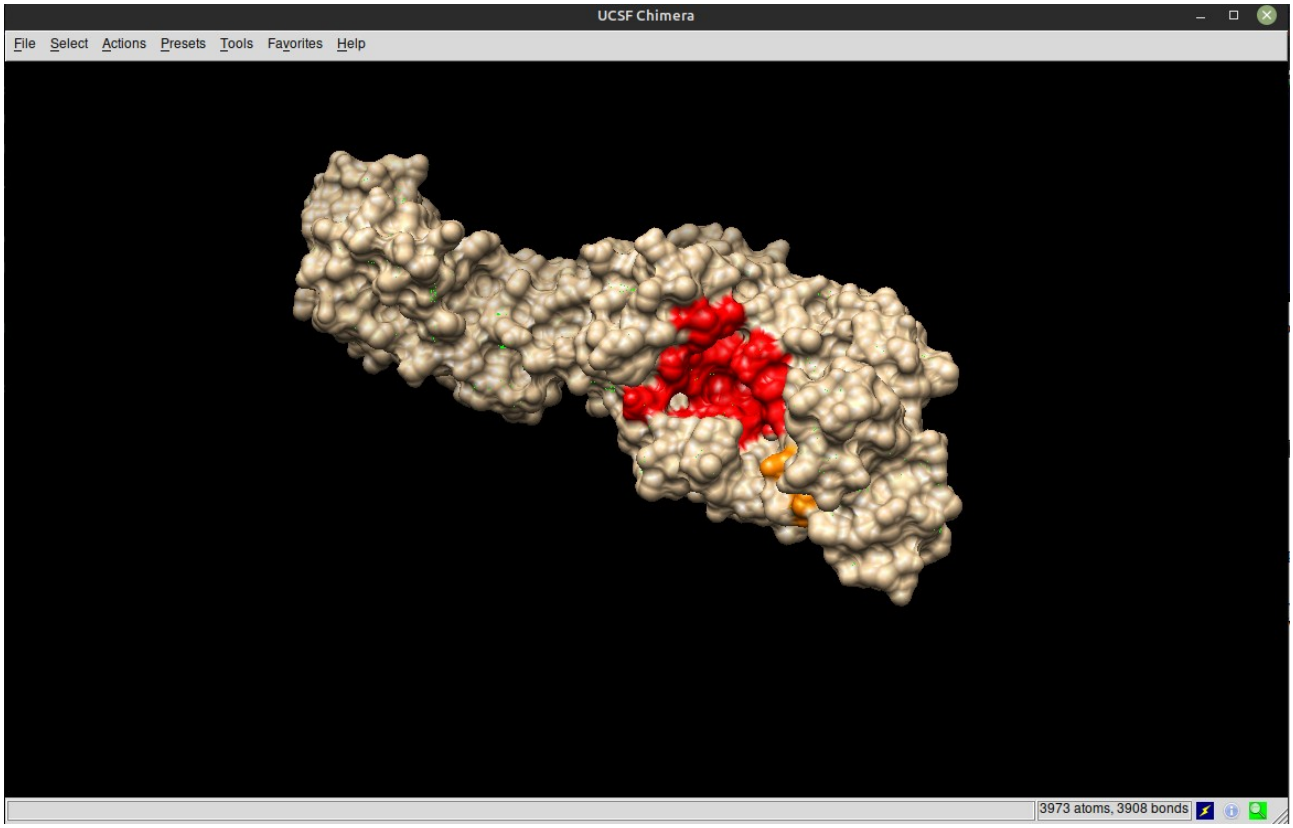
Note:
It's better to first open PyMol, then select File -> Open -> and then open the file from here.
Or, it's the simplest to just run it on the terminal:
pymol predict_1gln/visualizations/data/1gln.pdb.pml



The two pockets can be easily visualized and certain elements can be (not) displayed with the sidebar.

6. Close PyMol and open chimera_1gln.cmd with Chimera.

Note:
Simplest to use the terminal:
chimera predict_1gln/visualizations/data/chimera_1gln.cmd



A similar view appears.

It’s important to note that the colors (Chimera ONLY) have a meaning in the rank sense for easier visualization.

The following are the colors and the rank the correspond to:

Color	Rank
red	1
orange	2
yellow	3
green	4
cyan	5
blue	6
medium blue	7
purple	8
hot pink	9
magenta	10

Color	Rank
white	11
gray	12
black	13
tan	14
slate gray	15
dark khaki	16
plum	17
rosy brown	18

Click on Select -> Named Selections -> you will see the two pockets.

7. Now go back to the LigandMapper.py_tutorial_folder/ and enter the tut_1_online/ directory. This is an empty directory.

7.1 Open a terminal in this directory.

7.2 Run: *LigandMapper.py -o 1gln -ch*

You will notice that first the pdb file is downloaded. Then the file is directly analyzed. And finally, chimera opens with the visualization of this file.

If you want to want to visualize the file in PyMol you can use -pm instead of -ch. Note, these -ch and -pm switches work only for single files (-l or -o). They do not work for the directory method and the many methods (local_many, online_many).

If you try to use *LigandMapper.py -o 1gln -ch* again or *LigandMapper.py -o 1gln -pm*, you will get an error because the program will not allow you to overwrite the predict_1gln/ directory.

So, it’s a good practice to use a new directory for all analyses.

Tutorial 2: Analysis of a multiple PDB files

With LigandMapper.py you can analyze multiple pdb files all at once with 3 different methods:
(1) one is a for a whole directory of files,
(2) another is for selected pdb files of directory and subdirectories and finaly
(3) you can list many PDB entries, get the files and analyze them all at once.

1. Open a terminal in the tut_2_D/ and run: *LigandMapper.py -d ./*

The txt file will be skipped.

The program outputs:

Directory: /home/nikola/Desktop/LigandMapper.py_tutorial_folder/tut_2_D
Found files: ['4nd2.pdb', '1gln.pdb', '2ace.pdb', '2efr.pdb']
Analysing 4nd2.pdb structure...
Removing temporary directories
Creating visualization files
Creating TSV file
Output stored in: /home/nikola/Desktop/LigandMapper.py_tutorial_folder/tut_2_D/predict_4nd2
Analysing 1gln.pdb structure...
Removing temporary directories
Creating visualization files
Creating TSV file
Output stored in: /home/nikola/Desktop/LigandMapper.py_tutorial_folder/tut_2_D/predict_1gln
Analysing 2ace.pdb structure...
Removing temporary directories
Creating visualization files
Creating TSV file
Output stored in: /home/nikola/Desktop/LigandMapper.py_tutorial_folder/tut_2_D/predict_2ace
Analysing 2efr.pdb structure...
Removing temporary directories
Creating visualization files
NO POCKET WERE PREDICTED, REMOVING ADDITIONAL FILES.

The same predict_{pdb} folders are created for 4nd2.pdb, 1gln.pdb, and 2ace.pdb. For 2efr.pdb the predict_2efr directory is created and then removed because the algorithm hasn't detected any pockets. In such a case, the tsv file has no info, and the visualization files are all just command files that don't add any color/selections to the pdb file, so keeping them would just be a waste of memory. That's why we remove them.

2. Open a terminal in the tut_2_L/ folder and run:
LigandMapper.py --local_many 4nd2.pdb not_existant.pdb subfol_tut_2_L/2ace.pdb

Output:
Files found: ['4nd2.pdb', 'subfol_tut_2_L/2ace.pdb']
Files NOT found: ['not_existant.pdb']
Analysing 4nd2.pdb structure...
Removing temporary directories
Creating visualization files

Creating TSV file
Output stored in: /home/nikola/Desktop/LigandMapper.py_tutorial_folder/tut_2_L/predict_4nd2
Analysing subfol_tut_2_L/2ace.pdb structure...
Removing temporary directories
Creating visualization files
Creating TSV file
Output stored in: /home/nikola/Desktop/LigandMapper.py_tutorial_folder/tut_2_L/predict_2ace

All the output will be stored in the current directory which can be convenient if you want to analyze many files all at once. Furthermore, the nonexistent pdb file doesn't break the program.

3. Open a directory in the tut_2_O/ folder. This is an empty directory.
Run:
LigandMapper.py --online_many 4nd2 not_existant 1gno

Output:
The supplied pdb name (not_existant) is not valid.
PDB entry names consist of four alphanumeric characters, where the first character is a number and the following three characters are alphanumeric.

Entry found. Saving as file 4nd2.pdb
File saved successfully as 4nd2.pdb in the current directory

Entry found. Saving as file 1gno.pdb
File saved successfully as 1gno.pdb in the current directory

Succeded in obtaining: ['4nd2', '1gno']
Faled to get: ['not_existant']
Analysing 4nd2.pdb structure...
Removing temporary directories
Creating visualization files
Creating TSV file
Output stored in: /home/nikola/Desktop/LigandMapper.py_tutorial_folder/tut_2_O/predict_4nd2
Analysing 1gno.pdb structure...
Removing temporary directories
Creating visualization files
Creating TSV file
Output stored in: /home/nikola/Desktop/LigandMapper.py_tutorial_folder/tut_2_O/predict_1gno

The existing files are downloaded and analyzed and the non existent one is not even asked to the server because it doesn't have a pdb entry pattern.

Key Take-Aways

This package is made to be intuitive and easy to use.

Remember the following structure of the code: