INSPECTOR POCKET

An open source software to detect protein binding sites
Instruction manual & tutorial

Elizaveta Korchevaya and Jordi Martín

MSc in Bioinformatics for Health Sciences - Universitat Pompeu Fabra
Introduction to Python & Structural Bioinformatics joint project
05-04-2024

INSPECTORPOCKET
Instruction manual & tutorial

1. INSTALLATION

InspectorPocket is a relatively simple application designed with potential users in mind. Its interactive standard output redirection allows the user to see the process of the analysis as well as some potential errors that could be encountered during runtime. To install InspectorPocket, run the following commands:

1. Download and clone the git repository for Inspector pocket from https://github.com/Jordi-mp/InspectorPocket.git.

```
$ git clone https://github.com/Jordi-mp/InspectorPocket.git
```

2. Get inside the repository directory, and check if you have setuptools installed:

```
$ pip install setuptools
```

3. Run the following commands:

```
$ python setup.py sdist
$ pip install dist/InspectorPocket-1.0.tar.gz
```

2. ANALYZING A PDB FILE

In this tutorial we will:

- Analyze a PDB file in the local mode, downloading the PDB file manually from the protein data bank (PDB) [1].

- Analyze a PDB file in the online mode, initiate the process by entering the PDB code when prompted in the terminal.

A specific example using the structure for the crystal structure of human B2-adrenergic G protein-coupled receptor will be followed.

Analyzing by a downloaded PDB file (local mode)

When opting to analyze using a downloaded PDB file, the process involves utilizing a locally stored PDB file for analysis.

1. Download from PDB the structure of the protein in PDB format.

Human B2-adrenergic G protein-coupled receptor (PDB ID: 2RH1,https://www.rcsb.org/structure/2RH1).

Click on download files and then download using the output PDB format.

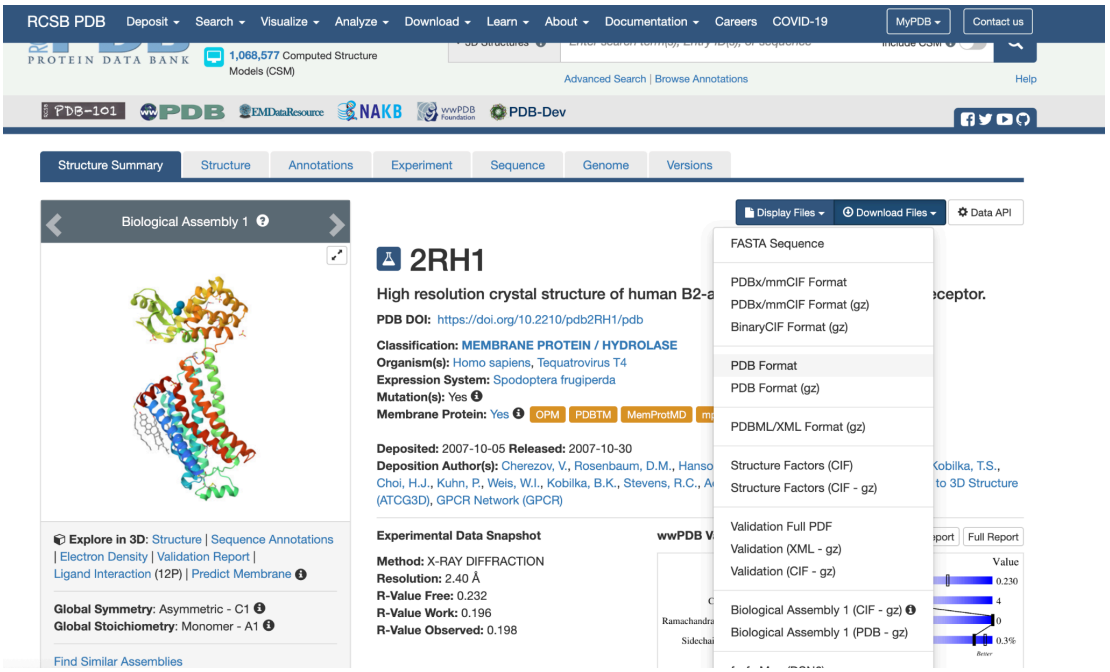


Figure 1. *Screenshot of the PDB web indicating how to download the structure in a PDB format.*

2. Make a new directory and move there the downloaded PDB.

```
$ mkdir Tutorial_InspectorPocket
$ mv 2rh1.pdb Tutorial_InspectorPocket
```

3. Run the script in the local mode.

To run the script in local mode, the -l option (for local) has to be specified before the pdb file.

```
$ InspectorPocket.py -l pdb_file.pdb
$ InspectorPocket.py -l 2rh1.pdb
```

The expected output in the terminal will be:

```
Hello! You are running InspectorPocket on 2rh1.
Identifying pockets...
Pockets identified.
Identifying residues in the vicinity of each pocket.
Pocket residues identified.
File '2rh1.pdb' copied to output directory: 2rh1
Report generated successfully. Thank you for using InspectorPocket!
```

4. Move to the newly created directory.

The newly created directory contains the outputs of InspectorPocket:

- Detected pockets in pdb format (2rh1_pocketX.pdb, being X the number of the pocket).
- A file containing all pockets within a single document (2rh1.pdb_all_pockets.pdb).
- A PyMOL [2] script (visualize_pymol.pml) to directly charge the PDB structure with its corresponding detected pockets represented as surfaces in PyMOL.
- A Chimera [3] script (visualize_chimera.cmd) to directly charge the PDB structure with its corresponding detected pockets represented as surfaces in Chimera.
- A text file including a list of amino acids involved in each pocket (pocket_report.txt).

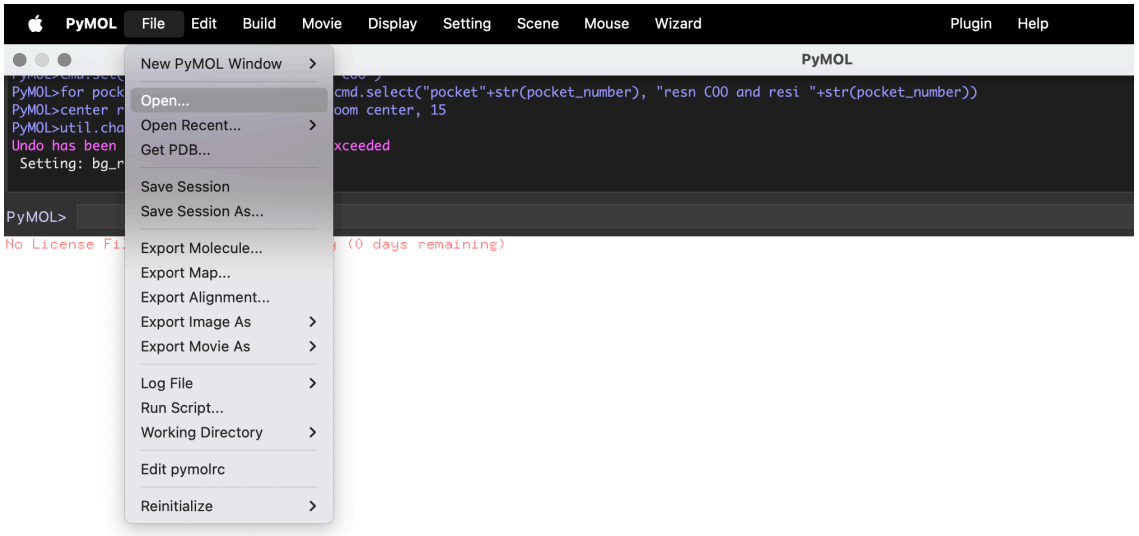
If we list the files in this directory:

```
$ cd 2rh1
$ ls
2rh1.pdb                    visualize_chimera.c     2rh1_pocket2.pdb
2rh1_pocket3.pdb            md                      2rh1_pocket6.pdb
pocket_report.txt           2rh1_pocket1.pdb        2rh1_pocket6.pdb
2rh1.pdb_all_pocket         2rh1_pocket5.pdb
s.pdb                       visualize_pymol.pml
2rh1_pocket4.pdb
```

6. Open the visualization script.

Using your software of preference (PyMOL or Chimera) open the visualization script (visualization_pymol.pml or visualization_chimera.cmd).

First open PyMol or Chimera, select "File" from the menu, then choose "Open," and proceed to open the file from there. Note that each pocket is loaded individually.



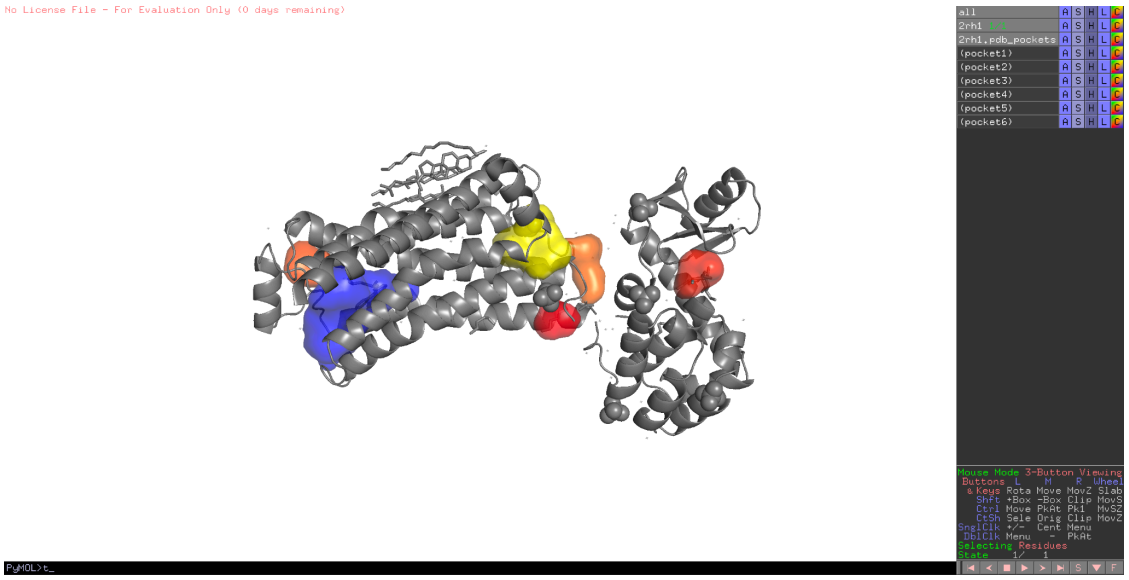


Figure 2. *Screenshots of PyMOL: indicating how to open the script (image in the top) and visualizing the protein with its predicted pockets (image in the bottom).*

7. Open the pocket_report file.

The pocket_report file contains the list of pockets detected and a list of potential residues involved in each one.

```
INSPECTOR POCKET REPORT
A report including a list of amino acids involved in each pocket.
Pocket1: GLY90, VAL114, THR118, ASP192, PHE193, PHE194, THR195, GLN197, ALA200, SER203, SER204, SER207, PHE208, ASN293, HIS296, VAL297, ASN301, LYS305, ILE309, ASN312
Pocket2: LEU64, LYS267, GLU268, LYS270, ALA271, LEU272, THR274, LEU275, TYR326, SER329
Pocket3: LYS140, GLN229, LEU230
Pocket4: GLU107, TYR185, ASP192, PHE193
Pocket5: GLY1030, LEU1032, PHE1104, GLN1105
Pocket6: PHE223, ALA226, LYS227, LYS263
```

Figure 3. *Screenshot of the residues list for each pocket reported by InspectorPocket.*

Analyzing by PDB ID from RCSB-PDB (online mode)

If the preferred approach involves executing InspectorPocket with an externally obtained PDB file, it can be acquired from an online source. Steps 1-3 are modified accordingly to facilitate the understanding of this process.

1. Create a novel directory and navigate to it.

```
$ mkdir Tutorial_InspectorPocket_Online
```

2. Run the script in the online mode.

To run the script in online mode, the -o option (for online) has to be specified. Then, a message in the terminal will appear asking for an input PDB ID.

```
$ InspectorPocket.py -o
```

The expected output in the terminal will be:

```
Enter PDB ID: 2rh1
Requesting PDB entry: 2rh1
Entry found. Saving as file 2rh1.pdb
File saved successfully as 2rh1.pdb in the current directory
```

```
Hello! You are running InspectorPocket on 2rh1.
Identifying pockets...
Pockets identified.
Identifying residues in the vicinity of each pocket.
Pocket residues identified.
File '2rh1.pdb' copied to output directory: 2rh1
Report generated successfully. Thank you for using InspectorPocket!
```

The next steps to follow correspond to 4-7 steps described in the local mode tutorial.

3. IMPORTANT CONSIDERATIONS

This section is reserved to note some important considerations and recommendations when running InspectorPocket:

- After running InspectorPocket for a particular structure, ensure you delete the output directory associated with that protein before rerunning the analysis for the same structure. Failing to do so may cause certain functionalities, like assigning the correct number of pockets, resulting in a fail in the program.

- To avoid unexpected or inaccurate results, it is recommended to run the program from the cloned github repository of InspectorPocket.

4. REFERENCES

[1] H.M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T.N. Bhat, H. Weissig, I.N. Shindyalov, P.E. Bourne, The Protein Data Bank (2000) *Nucleic Acids Research* 28: 235-242 https://doi.org/10.1093/nar/28.1.235.

[2] The PyMOL Molecular Graphics System, Version 1.2r3pre, Schrödinger, LLC.

[3] UCSF Chimera--a visualization system for exploratory research and analysis. Pettersen EF, Goddard TD, Huang CC, Couch GS, Greenblatt DM, Meng EC, Ferrin TE. *J Comput Chem.* 2004 Oct;25(13):1605-12.