pyvol Documentation

Release 1.2.27

Ryan Smith

October 28, 2019

CONTENTS

PyVOL is a python library packaged into a *PyMOL* GUI for identifying protein binding pockets, partitioning them into sub-pockets, and calculating their volumes. PyVOL can be run as a PyMOL plugin through its GUI or the PyMOL prompt, as an imported python library, or as a commandline program. Visualization of results is exclusively supported through PyMOL though exported surfaces are compatible with standard 3D geometry visualization programs.

CHAPTER

ONE

INSTALLATION

PyVOL can be installed into any python environment; however, for most users direct installation into PyMOL will be easiest.

1.1 Installation into PyMOL

PyVOL is distributed as a GUI and a backend. Installation into PyMOL uses PyMOL's plugin manager to install the GUI and then the GUI to install the backend. The GUI is installed through the plugin manager through loading the zipped GUI file:

```
https://github.com/rhs2132/pyvol/blob/master/pyvolgui.zip
```

This creates a new `PyVOL` menu entry under plugins. The third tab of the GUI allows installation of PyVOL from PyPI along with all available dependencies. For more information, especially on Windows, see the installation page.

CHAPTER

TWO

RUNNING PYVOL

2.1 Quick Start

From within PyMOL, the simplest binding pocket calculation is simply run at the PyMOL prompt with:

```
pocket protein_selection
```

The two parameters that most dramatically affect calculations are the maximum and minimum radii used to respectively define the exterior surface of the protein and the boundary of the binding pocket itself. In practice, the minimum radius does not need to be changed as its default (1.4) is broadly useful. The maximum radius does often need to be adjusted to find a suitable value using the max_rad parameter:

```
pocket protein_selection, min_rad=1.4, max_rad=3.4
```

2.2 Further Examples

For extensive explanations and documentations of the GUI and command line interfaces, see the examples page.

CHAPTER

THREE

MODULE DOCUMENTATION

For full documentation of the code, see the modules page.

3.1 Installation

PyVOL distribution is hosted by PyPI and accessed through pip. PyVOL can consequently be installed into any python environment. For convenience, the PyMOL GUI contains an installer.

3.1.1 Detailed Installation into PyMOL

PyVOL is distributed as a GUI and a backend. Installation into PyMOL uses PyMOL's plugin manager to install the GUI and then the GUI to install the backend. The GUI is installed through the plugin manager through loading the zipped GUI file:

```
https://github.com/rhs2132/pyvol/blob/master/pyvolgui.zip
```

This creates a new `PyVOL` menu entry under plugins. The third tab of the GUI allows installation of PyVOL from PyPI along with all available dependencies. On Linux and MacOS, MSMS is automatically installed from the platform-limited bioconda channel. MSMS installation instructions are otherwise below.

3.1.2 Manual Installation

PyVOL minimally requires biopython, MSMS, numpy, pandas, scipy, scikit-learn, and trimesh in order to run. PyVOL is available for manual installation from github or through PyPI. Most conveniently:

```
pip install bio-pyvol
```

3.1.3 MSMS Installation

MSMS can be installed on MacOS and Linux using the bioconda channel:

```
conda install -c bioconda msms
```

Otherwise MSMS must be installed manually by downloading it from MGLTools and adding it to the path. PyMOL distributions from Schrodinger have MSMS included; however, it must still be added to the path manually. The executable is located at:

```
<pymol_root_dir>/pkgs/msms-2.6.1-2/bin/msms
```

3.1.4 Updating

PyVOL can be updated via the command line:

```
pip update bio-pyvol
```

If using the PyMOL GUI, the third tab has a button labeled `Check for Updates` that will query PyPI to detect whether an update is available. If one is available, that button changes to `Update PyVOL` and permits updating with a single click.

3.1.5 Uninstallation

PyVOL can be uninstalled via the command line:

```
pip uninstall bio-pyvol
```

If using the PyMOL GUI, the third tab has a button labeled `Uninstall PyVOL` that will remove the PyVOL backend. Afterwards, selecting uninstall on the plugin within the PyMOL plugin manager will the GUI.

3.2 General Usage

The GUI and shell command line prompts recapitulate the PyMOL prompt interface with few exceptions. General usage of PyVOL is shown here with specific examples and modifications for the other interfaces in the following pages. Programmatic invocation of internal functions is supported and covered through the module documentation.

3.2.1 Pocket Specification

PyVOL by default recognizes the largest pocket and returns the volume and geometry for it. However, manual identification of the pocket of interest is generally preferable. This can be done through specification of a ligand, a residue, or a coordinate. If a specification is given, the mode is changed to specific by default.

Default Behavior (Largest Mode):

```
pocket protein_selection, mode=largest
```

Ligand Specification:

```
pocket protein_selection, mode=specific, ligand=ligand_selection
pocket protein_selection, ligand=ligand_selection
```

Residue Specification:

```
pocket protein_selection, mode=specific, resid=A15
pocket protein_selection, resid=A15
pocket protein_selection, mode=specific, residue=residue_selection
pocket protein_selection, residue=residue_selection
```

where the `resid` is written as **:raw-html-m2r:‘<Chain>‘:raw-html-m2r:‘<Residue number>‘**. If there is only one chain in the selection, the chain ID can be excluded.

Coordinate Specification:

```
pocket protein_selection, mode=specific, pocket_coordinate="5.0 10.0 15.0"
pocket protein_selection, pocket_coordinate="5.0 10.0 15.0"
```

where the coordinate is provided as three floats separated by spaces and bounded by quotation marks.

Calculation of All Pockets

Alternatively, PyVOL can return the surfaces and volumes for all pockets above a minimum volume that are identified. By default, this volume cutoff is set at 200 A^3.

```
pocket protein_selection, mode=all, minimum_volume=200
```

3.2.2 Extra Ligand Options

When a ligand is provided, the atoms of the ligand can be used to identify both minimum and maximum extents of the calculated binding pocket.

Ligand Volume Inclusion

To include the volume of the ligand in the pocket volume (useful for when the ligand extends into bulk solvent), use the `lig_incl_rad` parameter:

```
pocket protein_selection, ligand=ligand_selection, lig_incl_rad=0.0
```

where the value of `lig_incl_rad` is added to the Van der Waals radii of each atom in the ligand selection when calculating the exterior surface of the protein.

Ligand-defined Maximum Volume

The atoms of the ligand can also be used to define a maximum boundary to the calculated pocket by specifying the `lig_excl_rad` parameter:

```
pocket protein_selection, ligand=ligand_selection, lig_excl_rad=2.0
```

where the value of 11lig_excl_rad11 is added to the Van der Waals radii of each atom in the ligand selection when calculating the exterior surface of the protein.

3.2.3 Sub-pocket Partitioning

Sub-partitioning is enabled by setting the `subdivide` parameter to `True`:

```
pocket protein_selection, subdivide=True
```

Parameters controlling the number of sub-pockets identified generally perform well using defaults; however, they can be easily adjusted as needed. The two most important parameters are the minimum radius of the largest sphere in each sub-pocket (this excludes small sub-pockets) and the maximum number of clusters:

```
pocket protein_selection, subdivide=True, min_subpocket_rad=1.7, max_clusters=10
```

If the number of clusters must be reduced, sub-pockets are merged on the basis of connectivity between the defining sets of tangent spheres. Practically, sub-pockets with a greater surface area boundary are merged first.

3.2.4 Display and Output Options

By default, PyVOL simply outputs a log containing volumes and, when invoked through PyMOL, displays pocket boundaries as semi-translucent surfaces. This behavior can be extensively customized.

The output name for all computed PyMOL objects and the base filename for any output files can be specified using the prefix option:

```
pocket protein_selection, prefix=favprot
```

PyVOL can also write the input and output files to a directory if given an output directory. In this case it writes out the input protein and ligand structures, a csv report of all calcuated volumes, and paired csv/obj files containing tangent sphere collections and 3D triangulated mesh files respectively.

```
pocket protein_selection, output_dir=chosen_out_dir
```

Calculated surfaces can be visualized in three different ways by setting the `display_mode` parameter. The following three commands set the output as a solid surface with transparency, a wireframe mesh, and a collection of spheres. Color is set with the `color` parameter and transparency (when applicable) with the `alpha` parameter:

```
pocket protein_selection, display_mode=solid, alpha=0.85, color=skyblue
pocket protein_selection, display_mode=mesh, color=red
pocket protein_selection, display_mode=spheres, color=firebrick
```

where `alpha` is [0, 1.0] and the color is any color defined within PyMOL. The presets should generally be sufficient, but custom colors can be chosen using the commands given on the PyMOL wiki.

3.3 GUI Interface

The GUI is divided into three tabs that respectively: 1) run new PyVOL calculations 2) load prior PyVOL calculations from disk and 3) install, update, and uninstall the backend.

3.3.1 Run Tab

test run

3.3.2 Load Tab

test load

3.3.3 Install Tab

test install

3.4 Shell Interface

PyVOL can also be run from the system command line using bash or any standard shell. If installed using `pip`, a `pyvol` entry point should be automatically installed and made available on the path. Otherwise, manual invocation of `pyvol/__main__.py` should work.

3.4.1 Running from the Shell

From the command-line, PyVOL is run exclusively using a configuration file.

```
python -m pyvol <input_parameters.cfg>
```

3.4.2 Template Configuration File Generation

A template configuration file with default values supplied can be generated using:

```
python -m pyvol -t <output_template.cfg>
```

3.4.3 Notes on Output

Currently, PyVOL only reports standard log output to stdout when run this way. So if an output directory is not provided, there is no easy way to retrieve the results.

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search