pyvol Documentation

Release 1.2.27

Ryan Smith

October 28, 2019

CONTENTS

1 Installation **3**
- 1.1 PyMOL Installation . . . 3
- 1.2 Basic Manual Installation . . . 3
- 1.3 Manual MSMS Installation . . . 3
- 1.4 Quick Start . . . 3
- 1.5 Basic Usage . . . 4

2 Indices and tables **7**

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 PyMOL Installation

PyVOL can be installed by using PyMOL's plugin manager to load the zipped GUI file. This installation creates a `PyVOL` menu item under plugins. Opening this menu launches the PyVOL GUI. The third tab manages the installation of the PyVOL backend. Simply clicking `Install PyVOL` installs the PyVOL backend and, on Linux and MacOS, all dependencies. For dependency installation on Windows or anything else related to installation, see the installation page.

1.2 Basic 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 from PyPI.

```
pip install bio-pyvol
```

1.3 Manual 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
```

1.4 Quick Start

From within PyMOL, the simplest binding pocket calculation is simply run either with the provided GUI or 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 adjust to find a suitable value using the max_rad parameter:

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

1.5 Basic Usage

1.5.1 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:

```
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.

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
```

1.5.2 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. 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.

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

1.5.3 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.

1.5.4 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=best_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.

1.5.5 Command-line Interface

PyVOL can also be run from the command-line. 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. From the command-line, PyVOL is run with a standard configuration file.

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

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

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

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.

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.

Detailed PyMOL Installation

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:

Manual Installation

test

Manual MSMS Installation

test

Updating

test

Uninstallation

test

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search