OpenDiscovery

www.opendiscovery.org.uk

Version: 1.0

Contacts: gareth.price@warwick.ac.uk; a.marsh@warwick.ac.uk

Introduction

Open Discovery is a suite of programs that use Open Source or freely available tools to dock a library of chemical compounds against a receptor protein. In a (currently submitted) paper in the Journal of Chemical Education, we outline the usefulness of having an uncomplicated, free-to-use protocol to accomplish a task that has been the subject of academic and commercial interest for decades. We also highlight the gaps in open source tools around preparing protein - ligand complexes for molecular simulation, an area we expect to develop in the future.

Dependencies

- Python (you probably have this, install from http://www.python.org/getit/ - version developed with: 2.7)
- Open Babel (install from http://openbabel.org/ - version developed with: 2.3.1)
- AutoDock Vina binary (provided in lib folder, but downloaded from http://vina.scripps.edu - version developed with: 1.1.2)

Terminology

- Protocol Folder = the folder containing this readme, the odscreen.py and odparam.py files etc.
- Ligand Folder = the folder with the config file, the folder with the ligands and the folder with the receptors in (PDB and PDBQT).

ODCheck

`odcheck.py` is a useful preliminary check that can be run by typing `python /protocolFolder/odcheck.py`. This ensures that all the dependencies are installed properly. Pymol that is accessible via the command line is not compulsory, but a warning will appear if it isn't installed. You can ignore this if need be. Links are given to the websites of the relevant uninstalled software.

Walkthrough

A simple walkthrough can be seen in the examples folder. Also accessible at http://walkthrough.opendiscovery.org.uk/.

ODScreen

`odscreen.py` can be used to dock chemical compounds against a receptor protein. Compounds of interest can be generated in many ways, but we recommend ChemNProp (found at http://chemnprop.irbbarcelona.org) for fast similarity searching.

To quickly see what is required to run the protocol, in terminal (while in the protocol folder) type:

```
python odscreen.py -help
```

To call the protocol, the path to the ligand folder must be specified with `-d`, within which is a folder of compounds and a folder of receptors. In the ligand folder must be a configuration file. This file is the file AutoDock Vina uses- and therefore needs:

```
center_x = x
center_y = x
center_z = x

size_x = x
size_y = x
size_z = x
```

The compounds to be screened must be in individual files of the same type, in a folder of the same name as their extension. The *type* of the starting files must be specified with `-i`. Input file types can be:

- mol (e.g. mol/compound.mol)
- mol2 (e.g. mol2/compound.mol2)
- cdxml (e.g. cdxml/compound.cdxml)
- pdb (e.g. pdb/compound.pdb)
- smiles (with file extension **.txt**, e.g. smiles/compound.txt)

The *type/file extension* of the starting files must be specified with `-i`.

Exhaustiveness of Vina Docking can be set with `-e`. The default for this is 20, so this is an optional argument. If a job file is required for cluster use, specify the `-g` parameter and a job.pbs file will be generated and the program terminated. It will use the same exhaustiveness value. If screening has already occurred (and the results are in the results/ folder), then `-s` can skip the docking. If the preparatory files are already provided (i.e. the converted files in pdb, pdbqt, mol2 etc), then one can specify the `-a` argument which only creates files in folders that are not present. For example, if we had run a protocol, created a job file (with `-g`), ran the docking on a cluster, and then downloaded the results folder to the original ligand folder, we can carry on from the docking by using `-a`, as the folders from the original run will be there.

The receptor protein must be in **.pdb** and **.pdbqt** format and prepared manually beforehand and placed in the **receptor** folder in the ligand folder. For a good tutorial on how to prepare the receptor, please follow the tutorial at http://vina.scripps.edu/manual.html#tutorial. The filename for the receptor can be specified by `-r`, however if the receptor name is `receptor.pdb` then this can be omitted from the arguments.

If you have PyMol installed and is callable on the command line like `pymol ARGS`, then the protocol will automatically create a folder in the results folder named complexes, within which each ligand and the receptor will be merged into the same configuration file. If PyMol is not installed in this way, this will not occur (it will fail silently).

So, to conclude, the protocol can be performed from a clean slate of a config file and a folder of cdxml files by:

```
python odscreen.py -d ~/Path/To/Ligand_Folder -i cdxml -r aReceptor -e 100
```

With an initial structure like:

```
/Ligand_Folder
|-- conf.txt
```

```
    |_____ /cdxml
           |-- ligand1.cdxml
           |-- ligand2.cdxml
    |_____ /receptor
           |-- aReceptor.pdb
           |-- aReceptor.pdbqt
```

...will generate:

```
    /Ligand_Folder
    |-- combined.mol
    |-- smiles.txt
    |-- conf.txt
    |_____ /cdxml
    |_____ /images
    |_____ /mol
    |_____ /mol2
    |_____ /pdb
    |_____ /pdb-minimised
    |_____ /pdbqt
    |_____ /receptor
    |_____ /results
           |-- summary.csv
           |-- summary-sorted.csv
           |____ /complexes
    |_____ /results-mol2
    |_____ /smiles
```

Tips (for the brave)

If you add `alias odscreen="python /path/to/protocol_folder/odscreen.py "` to your `.bashrc` or `.bash_profile`, or put the protocol folder in a folder already indexed by these files, you can call the script as:

```
odscreen -ARGS
```