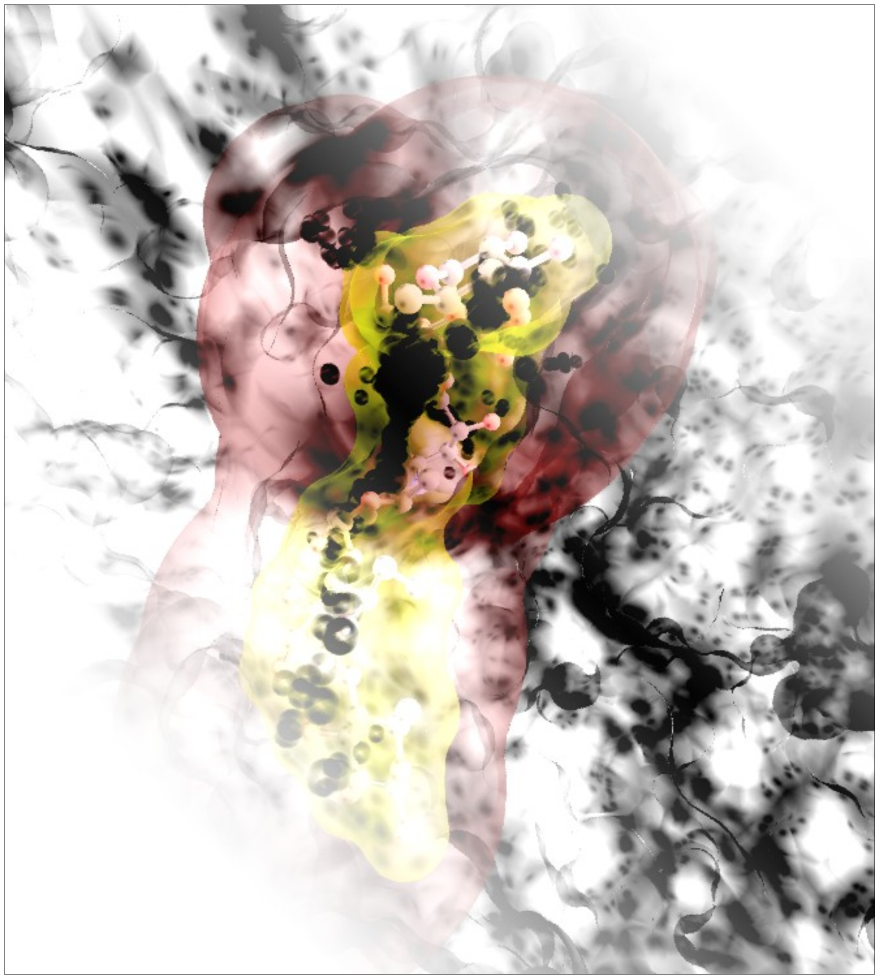
fpocket Users' Manual

version 1.0 November 27, 2008

authors : Vincent Le Guilloux[1] & Peter Schmidtke[2], supervisor : Pierre Tufféry[3]

fpocket is a protein pocket prediction algorithm. Given a PDB protein structure it enables the user to identify potent binding sites. Based on Voronoi tessellation, this algorithm is very fast and particularly well suited for large scale protein binding pocket screenings and development of scoring functions for binding pocket characterization.



acarbose binding site on alpha amylase (7taa).picture generated using VMD and tachyon rendering and GIMP post-processing.

1 ICOA - UFR Sciences – Université d'Orleans
2 MMB - Dept. Physical Chemistry – University of Barcelona
3 MTI - Inserm U973 - Université Paris Diderot

Notes

1. *This program uses output coming from Qhull. Qhull is currently not shipped with fpocket and has to be installed seperately. More information about Qhull can be found in the paper : Barber, C.B., Dobkin, D.P., and Huhdanpaa, H.T., "The Quickhull algorithm for convex hulls," ACM Trans. on Mathematical Software, 22(4):469-483, Dec 1996, http://www.qhull.org*

2. *This software includes code developed by the Theoretical and Computational Biophysics Group in the Beckman Institute for Advanced Science and Technology at the University of Illinois at Urbana-Champaign. The PDB parser of the Molfile Plugin of VMD were modified for the purposes of fpocket's PDB parsing.*

3. *Within the whole documentation code and output from computer programs are represented and formatted in the following way :* **`ls -1 > out.txt`**

Contents

Introduction

Thanks for taking the time to read this official users' guide of *fpocket*. In this guide are presented general functionalities of the *fpocket* program and its derivatives, *dpocket* and *tpocket*. Yes, indeed fpocket is a package of three distinct programs, mentioned here before. fpocket is an acronym for “free” pocket, as fpocket is distributed under the GNU GPL; dpocket is an acronym for “describing” pockets as it is for extraction of physico-chemical descriptors of pockets; tpocket is an acronym for “testing” pockets, as it is for testing on a large scale scoring functions for ranking protein cavities among each other.

This is not a usual guide. You can find here elements you can find in usual user guides, but we included several examples in the getting started section, which should enhance fast understanding of how to work with fpocket. The getting started guide can be understood like a mini tutorial of basic functionalities of this software.

License & Copyright

Contributions

This software was developed, validated, documented and distributed by Vincent Le Guilloux & Peter Schmidtke. Both, contributed equally to this project. The work was supervised by Pierre Tufféry.

Publication

This software was submitted for publication as Software paper in BMC Bioinformatics. Upon publication, the paper explaining methodological details will be freely available on the BMC Bioinformatics website.

Installation

Prerequisites

Currently fpocket proposes two different ways for visualization of binding pockets. Both are based on commonly used molecular visualization tools : VMD[REF] and PyMol[REF]. In order to use visualization you need to install at least one of both softwares. Currently, visualization using VMD has better rendering and performances and visualization using PyMol better handling of binding pockets. You can download VMD for free from http://www.ks.uiuc.edu/Research/vmd/. PyMol can be freely downloaded from http://delsci.com/rel/099/.

Dependencies

fpocket relies on Qhull[REF]. This software computes the convex hull, Delaunay triangulation, Voronoi diagram, halfspace intersection about a point, furthest-site Delaunay triangulation, and furthest-site Voronoi diagram. For more information and for downloading Qhull refer to http://www.qhull.org. Qhull is free software. Unfortunately, qhull is currently not distributed with fpocket. So you have to install it separately on your desktop and enable the *qvoronoi* program to be executable in a normal shell (bash, csh...).

Please test first if qhull works properly using the following procedure :

1. open a new shell
2. type the following : **`rbox c P0 D2 | qvoronoi Fo`**

If you obtain the following result, your installation of qhull was correct :

```
4
5 1 2     -1      0   -0.5
5 1 3      1     -0     -0
5 2 4      1     -0     -0
5 3 4     -1      0    0.5
```

If you obtain a segmentation fault, please refer to known bugs section of this manual (page 5).

Qhull is only required upon execution of fpocket, tpocket and dpocket, but not necessary for compilation of fpocket, dpocket and tpocket. The version Qhull 2003.1 was tested during the development of this release and should perform well for fpocket's purposes. The GNU C compiler version 3.4.6 was used to compile Qhull. See known bugs for possible compilation/execution issues with Qhull.

System Requirements

fpocket is available for Linux/Unix type systems only. Although it hasn't been tested on Mac OS X for now, it should also compile and work nicely on this system type. fpocket is currently not available for Windows systems.

In order to run fpocket, you should have at minimum a Pentium III 500 Mhz with 512Mb of RAM. This program was co-developed and tested under the following Linux distributions : openSuse 10.3, Centos 5.2, Fedora Core 7, Ubuntu 7.1.

How to install fpocket

j'attends le ./configure etc...pour voir commenton pourra faire...

Known Bugs

A known bug exists for Qhull using some newer versions of gcc. Qhull under some conditions returns a segmentation fault during execution, which would render fpocket unusable. If you issue this kind of bug, please do the following modification in the Makefile of Qhull.

Replace the occurrence of **`CFLAGS = -g -O2`** in this file by **`CFLAGS = -g -O`**

Recompile Qhull using the **`./configure ; make; make install`** commands and normally Qhull should work properly. For more Qhull secific questions and bugs, please refer to Qhull directly at qhull@qhull.org or on the website http://www.qhull.org.

Getting Started

fpocket

Example

Here is shown a very simple and straightforward example of how to run fpocket on a single PDB file downloaded from the RCSB PDB[REF]. The following line will execute fpocket on the **`3LKF.pdb`** file in the **`path_to_file`** directory.

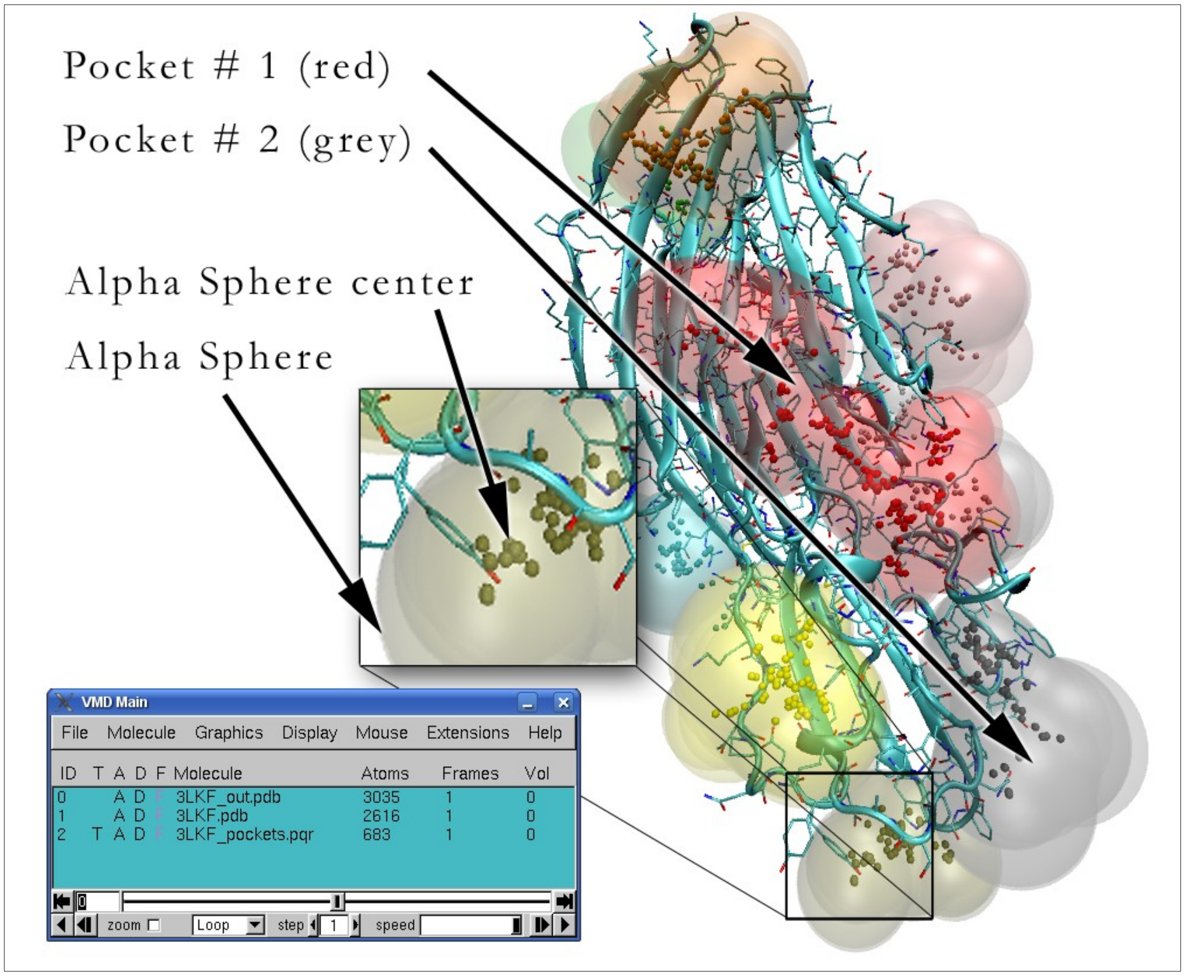
```
fpocket -f path_to_file/3LKF.pdb
```

It is mandatory to give a PDB input file using the **`-f`** flag in command line. If nothing is given, fpocket prints the fpocket usage/help to the screen. fpocket will use standard parameters for the detection of cavities. Fore more information about these parameters see the Advanded features chapter – fpocket section (page 12).

If fpocket works properly the output on the screen should look like this :

```
=========== Pocket hunting begins ==========
> Freeing remainnig memory...
=========== Pocket hunting ends ==========
```

If you have a look now in the **`path_to_file`** directory, you will notice that fpocket created a folder named **`3LKF_out/`**. This folder contains all the output from fpocket, so what you are actually interested in. If you just want to see rapidly the results, go to the **`3LKF_out`** directory and launch the **`3LKF_VMD.sh`** script. This script will launch the VMD molecular visualizer and charge the protein with binding site information coming from fpocket.



```
ID  T A D F Molecule            Atoms   Frames   Vol
0     A D F 3LKF_out.pdb        3035    1        0
1     A D F 3LKF.pdb            2616    1        0
2   T A D F 3LKF_pockets.pqr    683     1        0
```

Illustration 1: Explanation of the fpocket VMD output

The illustration above is somehow what you will see if you launch the VMD script. Well, you will see this in less beautiful, but let us oversee the eye candy we have prepared here for you. VMD is well suited for representing both information, the volume of alpha spheres and their respective centers. Usually the visual volume information is not of primordial importance, as the larger alpha spheres tend to reach far out of the protein and smaller alpha spheres are not visible because they are recovered by larger ones. As it can be seen within the Main VMD window, the visualization script charges 3 structures, all of them are explained in more detail in the output section of this chapter.

If you had a closer look before on the methodological aspects of this algorithm (we invite you to read the paper) a natural question would be how to represent apolar and polar alpha spheres. Currently the color code represents only the residue ID (rank of the cavity). If you want to see characteristics of alpha spheres we invite you to change te representation of alpha spheres. This can be found by clicking Graphics -> Representations. Another window will show up. There you select the first molecule (3LKF_out.pdb), like represented on the figure .

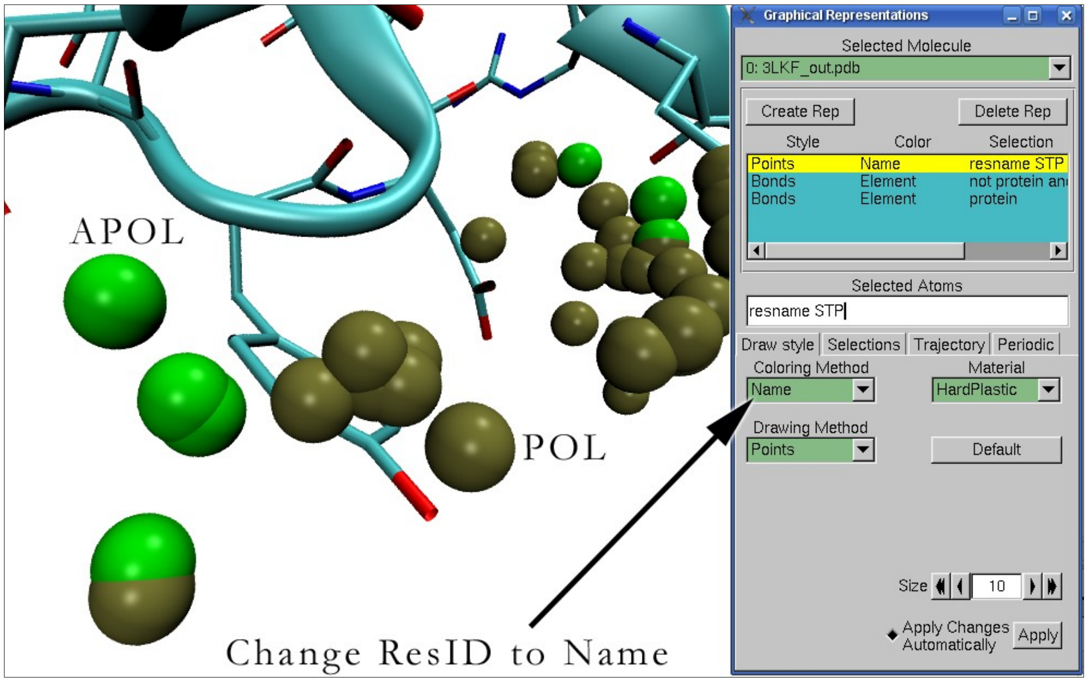


Illustration 2: Showing alpha sphere characteristics using VMD

Input

Mandatory:

1: flag -f : one standard PDB file

Optional:

For this see Advanced features chapter – fpocket section (page 12).

Output

fpocket yields output directly in the directory of the data file. The **`ll 3LKF_out`** output of the current sample run would look something like this:

```
total 332
-rw-r--r-- 1 peter users    769 Nov 29 00:14 3LKF.pml
-rw-r--r-- 1 peter users    698 Nov 29 00:14 3LKF.tcl
-rwxr-xr-x 1 peter users     30 Nov 29 00:14 3LKF_PYMOL.sh
-rwxr-xr-x 1 peter users     41 Nov 29 00:14 3LKF_VMD.sh
-rw-r--r-- 1 peter users 245835 Nov 29 00:14 3LKF_out.pdb
-rw-r--r-- 1 peter users   6725 Nov 29 00:14 3LKF_pockets.info
-rw-r--r-- 1 peter users  49355 Nov 29 00:14 3LKF_pockets.pqr
drwxr-xr-x 2 peter users   4096 Nov 29 00:14 pockets
```

As you can see, fpocket provides a lot of files and another subdirectory. However, majority of these files are necessary for easy visualization of binding pockets. Lets explain the content and utility of each file :

- **`3LKF.pml`** : this is a PyMOL script for visualization of binding pockets using PyMOL
- **`3LKF.tcl`** : this a tcl script for visualization of binding pockets using VMD
- **`3LKF_PYMOL.sh`** : this is the executable script to launch fast visualization using PYMOL
- **`3LKF_VMD.sh`** : this is the executable script to launch fast visualization using VMD
- **`3LKF_out.pdb`** : this is the most important file, it contains the initial PDB structure given as input. Non cofactor HETATM occurrences will be stripped of in this file compared to the original PDB input file. The PDB file contains centers of alpha spheres using the HETATM definition as dummy atoms. These alpha sphere centers are attached in the end of the PDB file, using the STP residue name (for site point). Apolar alpha spheres carry the atom name APOL, polar alpha spheres the atom name POL. Pockets are sets of alpha spheres. They can be distinguished by residue number. Thus residue STP 1 would be the first binding pocket according to fpocket. To show this more clearly here is an extract of the **`3LKF_out.pdb`** :

```
ATOM   2349   CD LYS A 299       9.679  16.827 105.636  1.00 19.91           C
ATOM   2350   CE LYS A 299      10.371  16.314 104.370  1.00 25.17           C
ATOM   2351   NZ LYS A 299      11.749  15.794 104.597  1.00 32.36           N
ATOM   2352  OXT LYS A 299       5.240  20.009 107.670  1.00 16.06           O
HETATM 2736  POL STP C   1      18.291  37.420  83.622  0.00  0.00          Ve
HETATM 2756  POL STP C   1      18.445  37.638  83.606  0.00  0.00          Ve
HETATM 3208  POL STP C   1      18.325  37.403  83.631  0.00  0.00          Ve
HETATM 3208  POL STP C   1      18.450  37.618  83.610  0.00  0.00          Ve
```

- **`3LKF_pockets.info`** : this file contains more human readable information about results after pocket detection. It contains the score associated at each pocket found, as well as all the descriptors of the pocket. The sample gives the following output for the first ranked binding pocket :

```
        ## FPOCKET RESULTS ##

  ## POCKET 1 ##
          0  - Pocket Score:                 86.1249
          1  - Number of Voronoi vertices:  163
          2  - Mean alpha-sphere radius:    3.786007
          3  - Mean alpha-sphere solvent accessibility: 0.495167
          4  - Flexibility:                 0.277080
          5  - Hydrophobicity Score:        24.393940
          6  - Polarity Score:              16
          7  - Volume Score:                4.030303
          8  - Real volume (approximation): 4128.224121
          9  - Charge Score:                2
          10 - Local hydrophobic density Score: 19.434782
          11 - Number of apolar alpha sphere: 46
          12 - Amino acid composition:
          12 -   0   0   0   3   3   1   2   3   2   3   2   5   2   2
                 0   2   1   0   0   2
```

- **3LKF_pockets.pqr** : This file contains all alpha sphere centers, as the 3LKF_out.pdb file, but contains no information about the protein structure. Furthermore using the pqr format enables writing of the van der Waals radius of atoms explicitely in this file. Here this possibility was used to output the radii of alpha spheres of a pocket. Charging this pqr file, one can analyze more precisely the volume recognized by fpocket. Note that, currently only VMD supports reading this format correctly. PyMOL is able to read pqr file, but does not interpret van der Waals radii.
- **pockets/** : Well, again a subdirectory. But I promise, it's the last one. For development purposes or easy analysis, fpocket proposes this directory which contains according to the current example :

```
pocket0_atm.pdb   pocket2_vert.pqr  pocket5_atm.pdb   pocket7_vert.pqr
pocket0_vert.pqr  pocket3_atm.pdb   pocket5_vert.pqr  pocket8_atm.pdb
pocket1_atm.pdb   pocket3_vert.pqr  pocket6_atm.pdb   pocket8_vert.pqr
pocket1_vert.pqr  pocket4_atm.pdb   pocket6_vert.pqr  pocket9_atm.pdb
pocket2_atm.pdb   pocket4_vert.pqr  pocket7_atm.pdb   pocket9_vert.pqr
```

The ***_atm.pdb** files contain only the atoms contacted by alpha spheres in the given pocket. Complementary to this information, ***_vert.pqr** files contain only the centers and radii of alpha spheres within the respective pocket. As extensions mention, atoms are output in the PDB file format and alpha sphere centers in the PQR file format.

Is there something else? No, you have done. Congratulations, you have successfully performed your first cavity prediction with fpocket...without any accidents we hope. As you might have seen, usage of fpocket is rather simple, although it is command line based software (for now). Furthermore you should have seen that fpocket is very fast, well, lets say if you do not run a P1 100Mhz.

As mentioned before, fpocket provides much more possibilities especially for filtering out unwanted pockets, clustering of alpha spheres. For all these issues and usage of these more advanced features, refer to chapter Advanced features, section fpocket (page 12) of this manual.

dpocket

Until now you have seen what the majority of cavity detection algorithms can do. So a part from speed and hopefully prediction results, nothing distinguishes fpocket from other algorithms like ligsite, sitemap, sitefinder, pocketpicker, pass .... etc...

This is just partially true, because the fpocket package contains dpocket. D is an acronym for describing. One purpose a cavity detection algorithm can be used for is for extraction of descriptors of the physico-chemical environment of the cavity. dpocket allows to do this in a very simple and straightforward way. As extracting binding pocket descriptors on only one protein would be somehow meaningless for studying pocket characteristics, dpocket enables analysis of multiple structures. So now, no longer scripting and automation is necessary to do these kind of things. But lets have a closer look using again a very simple example you can try on your workstation.

Example

Here we go. dpocket requests one single input file. This input file must be a text file containing the following information : 1 – the PDB file of the protein you want to analyze and 2 – the ID of the ligand you would like to have as reference in order to define an explicitly defined binding pocket. Your file, let us call it **`test_dpocket.txt`** should look something like this :

```
data/3LKF.pdb   pc
data/1ATP.pdb   atp
data/7TAA.pdb   abc
```

Here we analyze three pdb files. Note that the ligand name should be separated by a tab from the pdb file name. You can launch dpocket on this sample file using the following command :

```
dpocket -f test_dpocket.txt
```

dpocket will yield 3 results files in the current directory. These files will be by default :

```
dpout_explicitp.txt
dpout_fpocketnp.txt
dpout_fpocketp.txt
```

If you want to change naming of these files, use the -o flag in command line to define a new prefix for the fpocket output files, for example **`my_test`** as prefix would yield **`my_test_explicitp.txt`**. The three output files contain the in fpocket implemented pocket descriptors for each binding pocket found by fpocket : **`*_fpocketp.txt`**, for all binding pockets (that is to say that do not contain the reference ligand here) : **`*_fpocketnp.txt`**, and the binding pocket defined using explicitly the ligand for binding pocket identification : **`*_explicitp.txt`**.

The ouput files are tab separated ASCII text files that are easy to parse using statistical software such as R. Thus statistical analysis of pocket descriptors becomes a very straightforward and easy process. For more details of the output refer to the output section below, or to dpockets Advanced features section (page 12).

Input

Mandatory:

1: flag -f : a dpocket input file, this file has to contain the path to the PDB file, as well as the residuename of the reference ligand, separated by tabulation.

Optional:

1 : flag -o : the prefix you want to give to dpocket output files

dpocket offers much more optional parameters in order to guide the pocket detection. For this see Advanced features chapter – dpocket section (page 12).

Output

As shown in the example, dpocket creates 3 output files. Lets describe them a bit more in detail here :

- **`dpout_explicitp.txt`** : This file contains all pocket descriptors implemented in fpocket of the explicitly defined binding pocket. What does this mean, explicitly? In the input you have associated a ligand identification to each PDB file. This ligand is used by fpocket in order to identify the actual binding pocket. If you want to know more about this process, refer to the Advanced features section of dpocket (page 12). Now let us have a look was is actually in this file.

```
pdb ligand overlap lig_vol pocket_vol nb_alpha_spheres mean_asph_ray
data/3LKF.pdb PC 100.00   132.90  1678.64    29  3.94
data/1ATP.pdb ATP 100.00   322.62  2127.53    65  3.59
data/7TAA.pdb ABC 100.00   608.66  4977.48    97  4.20
```

  Note that this is only an extract of this file. It contains a lot of columns (descriptors) that are not represented here. The first line describes the nature of the entry. The next line recapitulates the pdb structure analyzed (**`data/3LKF.pdb`**), the ligand used as reference (**`PC`**). Next the overlap between the actual and found binding pocket is shown, here 100% as this is an explicitly defined binding pocket. The next entries can be used as descriptors, like the ligand volume, the pocket volume, the number of alpha spheres in the binding pocket, the mean alpha sphere radius ... For a complete list of all implemented descriptors in fpocket, refer to the Advanced features – Pocket descriptors section (page 13).

  The volumes calculated here are not accurate at all. If you want to calculate accurate volumes you have to change parameters for volume calculation. As volume calculations are generally over-estimated using alpha sphere approaches, especially for open binding pockets, this calculation is made available, but uses the minimum sampling for the calculation. For more accurate calculation significantly more calculation time would be necessary.

- **`dpout_fpocketnp.txt`** : This file contains the same kind of descriptors as the preceding one, but this time for pockets identified by fpocket, that are "non binding pockets". Non binding pockets means here, that the pockets do not correspond to the pocket where the reference ligand binds.

- **`dpout_fpocketp.txt`** : The last file is also formated the same way as the preceding both. This file contains the binding pocket, this time identified by fpocket and not explicitly by the ligand.

In conclusion of this first very easy dpocket run, you can see that you have a very fast and reliable tool to extract pocket descriptors, of binding pockets and "non binding pockets" on a large scale level. These descriptor files provide an excellent tool for further statistical analysis and model building, which leads immediately to your wish to write a new scoring function for ranking cavities using the different descriptors. Well, fpocket, dpocket and tpocket are very useful tools to do exactly this! So go ahead. Lets suppose you have passed several thousands of PDB files and analyzed statistically the significance of all descriptors. You have set up a new scoring function. Now you have an external test set of PDB files you haven't tested. How can you evaluate your scoring function? This is actually also a very easy task, using tpocket.

tpocket

As already mentioned in the preceding paragraph, tpocket can be used in order to evaluate rapidly cavity scoring functions. If you are for example in the pharmaceutical industry and you want to set up the ultimate drugability prediction score, you might be able to do this with fpocket and dpocket. Afterwards you can actually test your method using tpocket. T is an acronym for testing, here.

Something fancy we did not tell you about before is, that you also can test your scoring function on apo structures using tpocket. The only requirement is that they have to have a holo structure and must be structurally aligned to this holo structure in order to get meaningful results. But lets explain this with an example.

Example

Input

Output

Advanced features

fpocket

dpocket

tpocket

Pocket descriptors

Cofactor definition

Writing your own scoring function

Writing your own descriptor

Samples

Sample Tpocket input

Sample Dpocket input