Contents

1 Introduction ........................................ 2
   1.1 Installation instructions ................. 2
   1.2 Theory ........................................ 2

2 How to run AutoxX .................................. 5
   2.1 Prerequisites .................................. 5
   2.2 AutoDock ....................................... 5
   2.3 FlexX ........................................... 5
   2.4 AutoxX ........................................... 6
   2.5 Output .......................................... 6
   2.6 Visualization ................................ 7

3 Example ............................................. 8
Chapter 1

Introduction

AutoxX is a docking workflow combining two of the most popular docking tools: AutoDock [1] and FlexX [2]. But AutoxX is more than a simple consensus scoring technique. Our new combined docking workflow – AutoxX – unifies the interaction models of AutoDock and FlexX rather than combining the scores afterwards. It uses the docking algorithm of AutoDock but incorporates parts of the interaction scheme of FlexX.

1.1 Installation instructions

1. untar the file autoxx2.0.tar.gz:
   \texttt{tar -xzf autoxx2.0.tar.gz}

2. Move \texttt{iapoints.bat} to the bat directory of your FlexX installation. Be sure that this directory is given in your config.dat, so that FlexX is able to find the script.

3. Include the autoxx2.0 directory in your PATH or copy the scripts in your working directory, so that you can execute the scripts.

1.2 Theory

The idea of the AutoxX approach is to unify the different interaction schemes of the docking tools AutoDock and FlexX. To achieve this we had to make the different interaction models comparable. In AutoDock the separation of the calculation of the molecular affinity grids from the docking simulation provides a modularity to the procedure. This facilitates the introduction of modified affinity grids. Therefore and because it is algorithmically easier to transform single points into a grid than the other way around we decided to integrate FlexX into AutoDock and not AutoDock into FlexX. Figure 1.1 shows a schematical depiction of the transformation process. For a more detailed description of AutoxX see the original publication [3].

We transformed the FlexX interaction points into grid maps similar to the ones produced by AutoDock. To increase variability in the FlexX grids Gaussians were applied.
for the transformation:

\[ f(x) = h \exp^{-b\|x-c\|^2} \]  

(1.1)

where height \( h \) and center \( c \) are the FlexX energy contribution of this interaction type and the coordinates of the interaction point, respectively. The width \( b \) was chosen that way that two adjoining Gaussians of a common interaction surface intersect at \( h/2 \).

Figure 1.2 depicts the workflow of the AutoxX approach:

1. In the first step the targets have to be prepared for docking in a tool-specific manner.

2. Secondly, the AutoDock grid maps and the FlexX interaction points are computed. AutoDock already produces separate map files, the FlexX interaction points have to be extracted with a special script in the FlexX-own scripting language.

3. That follows the transformation of the FlexX interaction points into energy grids and the union of the AutoDock and FlexX maps as described above.

4. Finally, AutoxX carries out a combined docking with the modified grid maps. During the docking these new modified maps are used instead of the ones created by AutoDock by changing the docking parameter file which gives instructions on the docking procedure.

With this unification approach the source code of the tools does not have to be modified.
Figure 1.2: AutoxX workflow.
Chapter 2

How to run AutoxX

2.1 Prerequisites

AutoDock 4.0 as well as FlexX 2.0 have to be properly installed on your system to run AutoxX.

2.2 AutoDock

Create all the files you would also need for an ordinary AutoDock run:

- PDBQT for the target;
- PDBQT for the ligand;
- gpf (grid parameter file)
- dpf (docking parameter file)

These files will be needed by AutoxX.

2.3 FlexX

Create a rdf (receptor description file) of your target protein. Create a file which contains the name(s) of the rdf (without extension).

Run `flexx -c config.dat -b iapoints -a '$(proteinlist)=yourfile.txt'`. This will produce the interaction points in two separate gdfs (graphic description files) called `yourprotein_iapoints0.gdf` and `yourprotein_iapoints1.gdf`. Move these two files to the directory where the files produced by AutoDock (pdbqt, gpf and dpf) are.
2.4 AutoxX

autogrid4 -p protein.gpf -l protein.glg

If you have not already done so, create the affinity grid maps with AutoGrid.

gaussian4.py protein protein.gpf

Transforms the extracted FlexX interaction points into affinity grids of the same di-
mension like AutoDock (as specified in protein.gpf).

addGrids4.py protein

Adds the AutoDock and FlexX grids, produces protein.NA+.map, protein.OA+.map and protein.HD+.map (if these atom types are present in your ligand).

adjustDPF4.sh ligand protein > ligand_protein_x.dpf

Adjusts the dpf, so that the docking is performed on the combined maps instead of the original AutoDock maps. adjustDPF4.sh requires that a normal dpf has been created ligand_protein.dpf.

autodock4 -p ligand_protein_x.dpf -l ligand_protein_x.dlg

Runs AutoDock with instructions from the modified dpf.

All these single steps are included in a shell script called autoxxdock4.sh. So instead of performing all steps by yourself, you can also simply run

autoxxdock4.sh -p protein -l ligand

Be sure that the following files are present in the directory where you run the script: protein.pdbqt, ligand.pdbqt, protein.gpf, ligand_protein.dpf, protein_iapoints0.gdf and protein_iapoints1.gdf.

2.5 Output

The output of AutoxX consists of a normal AutoDock dlg file. Poses can be extracted with the Python scripts provided by ADT like write_lowest_energy_ligand.py. Scores can be looked at with

grep RANKING ligand_protein_x.dlg

See the AutoDock user’s guide and website for more information on result analysis.
2.6 Visualization

The docked poses are saved in pdb format and can be visualized with a variety of different molecular visualization tools like Rasmol. Sometimes it is interesting to have a look at the docked ligand conformations in combination with the affinity maps. This is possible with the free visualization software gOpenMol [4]. After converting the AutoxX or AutoDock grid maps into the gOpenMol-specific plt format with AutoDock2Plt, they can be displayed as isocontours in differing colors and transparencies. gOpenMol has an intern command line interpreter based on Tcl/Tk [5]. A sample Tcl script for a visualization of AutoxX grid maps and docked poses is included in the example.
Chapter 3

Example

An example is included in the distribution of AutoX 2.0. It is α-thrombin in complex with the inhibitor benzamidine, PDB-ID 1dwb.

1. Create a pdbqt file for the protein:
   Remove water molecules and ligands from the pdb. Have a look at the AutoDock user's guide for more information. You can use ADT or a program of your choice for preparation. Be sure that you add polar hydrogens, Gasteiger charges and solvation parameters.

2. Create a pdbqt file for the ligand ben.mol2:
   You can either use ADT for preparation or the script prepare_ligand4.py.

3. Create a grid parameter file (gpf). Use ADT.

4. Create a docking parameter file (dpf). You can use prepare_dpf4.py.

5. Compute the FlexX interaction points for the receptor:
   Create a file with the name(s) of your protein(s) (without extension).
   Move the pdb of your protein into the pdb directory of FlexX.
   Create a rdf (receptor description file) for the protein. Refer to the FlexX user's guide.
   Run flexx -c config.dat -b iapoints -a '$(proteinlist)=yourfile'
   The gdfs will be written in the predict directory which you have specified in your config.dat. Move them to the 1dwb directory where the other prepared files are.


7. Visualize your results:
   Run e.g. the ADT script write_lowest_energy_ligand.py -f ben_1dwb_x.dlg.
   This extracts the conformation with the lowest predicted binding energy. Convert the resulting pdbqt file into pdb with pdbqt_to_pdb.py. Now start gOpenMol and convert the maps you want to visualize into plt files with Run → AutoDock2plt. Now run 1dwb_v4.tcl which you can find in the example/1dwb/gopenmol directory with File → Import → Tcl script. This should produce something similar to figure 3.1.
Figure 3.1: 1dwb. Affinity grid maps of aromatic carbon (green) and polar hydrogen combined with FlexX acceptor interactions (white) for alpha-thrombin with the best ranked benzamidine conformation docked by AutoX. The crystal conformation is displayed in wireframe.
Bibliography


