MODPIN

Modeling Protein-Protein Interactions
User Guide
Structural Bioinfromatics Group
(GRIB-IMIM)

Lluis Dominguez, Jaume Bonet, Oriol Fornes, Patricia M. Bota, Baldo Oliva

May 8, 2018

Contents

1 Modpin

MODPIN is a set of python scripts to model and analyze protein-protein interactions.

1.1 Installation

- Clone the main source from GitHub and install it:

```
git clone
    https://github.com/structuralbioinformatics/modpin.git
cd modpin
```

If you prefer you can install Modpin inside the folder you have cloned from github:

```
pip install -e . --user
```

Otherwise pip will install Modpin in a default directory:

```
pip install . --user
```

- Or download Modpin from Pypi (Python Package Index):

```
 pip install modpin --user
```

Since is preferable to install the package for your local user and don't write to the system directories, we set the the *"–user"* flag, this will also allow you to install Modpin even if you don't have root privileges.

If you want to know where Modpin is installed run the comand below:

```
pip show modpin
```

If you want to uninstall Modpin run the comand below:

```
pip uninstall modpin
```

- Or download Modpin from GitHub:
  https://github.com/structuralbioinformatics/modpin
  You can work directly in Modpin folder without installing it but then you will have to specify the path to the python scripts you want to use.

1.2 Requirements

MODPIN works with Python 2.7 version and to perform modeling also some external software is required.

- **BLAST**
  You need to download blast version 2.2.26 . BLAST+ executables can be found here:
  ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/

- **CD-HIT**
  Is a software for clustering and comparing protein or nucleotide sequences and you can download it from:
  http://weizhongli-lab.org/cd-hit/download.php

- **Clustalw**
  Is a software for multiple alignment of nucleic acid and protein sequences, you need to download clustalw-2.0.12 version that can be found here:
  http://www.clustal.org/download/

- **HBPLUS**
  Is a hydrogen bond calculation program, you need to download hbplus-3.2 version, instructions on how to obtain it can be found here:
  http://www.ebi.ac.uk/thornton-srv/software/HBPLUS/

- **Reduce**
  Reduce is a program for adding hydrogens to a Protein DataBank (PDB) molecular structure file. You can find reduce-3.23 version here:
  http://kinemage.biochem.duke.edu/software/reduce.php

- **ZRANK** Is an algorithm for quickly and effectively reranking rigid-body docking predictions and it can be found here:
  http://zdock.umassmed.edu/software/

- **MODELLER**
  Is a software used for homology or comparative modeling of protein three-dimensional structures. You can download Modeller version 9 from here:
  https://salilab.org/modeller/download_installation.html

- **Rosetta**
  The Rosetta software suite includes algorithms for computational modeling and analysis of protein structures. You can download it from here:
  https://www.rosettacommons.org/software/license-and-download

1.3 Configuration

For the program to work you need to setup properly the configuration file (config.ini) that can be found in the "scripts" directory of Modpin.
In order to do so, if you have installed correctly Modpin, you can run from terminal the comand below:

```
configuration
```

If you have not installed Modpin you can run the script from terminal in Modpin folder:

```
python config_setup.py
```

Below you can see an example on how the "config.ini" file should look after the configuration.

Paths
In this section will be specified the paths of the software installation folders required by Modpin in order to work.

- Installation path of Python 2.7:
  python_path = */usr/bin/python*
- Installation path of Modpin:
  modppi_path = */home/user/modpin/modpin*
- Path to SBI library (already included in Modpin):
  sbi_library_path =*./src*
- Path to functions (already included in Modpin):
  functions_path = *./src/functions*
- Path to "data" directory where data from PDB and 3did will be installed:
  data_path = *./data*

- Path to "data" directory where data from PDB will be installed:
  pdb_path = *./data/pdbsource*
- Path to "data" directory where data from 3did will be installed:
  3did_path = *./data/3did*
- Path to archdb (already included in Modpin):
  archdb_path = */home/user/modpin/modpin/src*
- Installation path of Blast:
  blast_path = */home/user/blast-2.2.26/bin/*
- Installation path of Clustalw:
  clustal_path = */home/user/clustalw-2.0.12/bin/*
- Installation path of Modeller:
  modeller_path = */home/user/modeller9.17/bin*
- Path to "blast" folder where the FASTA sequences database will be stored:
  blast_dir = *./blast*
- Installation path of Zrank:
  zrank_path =*/home/user/modeller9.17/zrank/zrank*
- Installation path of hbplus:
  hbplus_path = */home/user/hbplus-3.2/hbplus*
- Installation path of Rosetta:
  rosetta_path = */home/user/Rosetta/rosetta_ 2014.35.57232_ bundle/-main*
- Installation path of the Fixed backbone design application (fixbb) of Rosetta:
  relax_exe =*/home/user/Rosetta/rosetta_ 2014.35.57232_ bundle/main/-source/bin/fixbb.linuxgccrelease*
- Installation path of the Interaface analyzer application of Rosetta:
  interface_analyzer =*/home/user/Rosetta/rosetta_ 2014.35.57232_ bundle/-main/source/bin/InterfaceAnalyzer.linuxgccrelease*

- Installation path of Reduce:
  reduce_path = */home/user/reduce-3.23/reduce.3.23.130521.linuxi386*
- Path to PDB HET dictionary file (that can be downloaded from http://kinemage.biochem.duke.edu/software/reduce.php):
  reduce_db_path = */home/user/reduce-3.23/reduce_ wwPDB_ het_ dict.txt*

Files

This section contains the paths to folders and files generated during the installation of data. We use the folder "data" under the installation of MODPIN, but this can be selected somewhere else.

- Path of the file with FASTA sequences:
  fasta_list_file = *./data/nr.fa*
- Path of the file with non-redundant set of sequences:
  nr90_list_file = *./data/merged_nr90.fa*
- Path of the protein-protein interaction file that will be generated during the installation of data:
  ppi_files = *./data/pdb*
- Path of the domain-domain interaction file that will be generated during the installation of data:
  ddi_files = *./data/3did*
- Path of the list file with PDB entries:
  pdb_list_file = *./data/pdb.list*
- Path of the list file with 3did entries:
  3did_list_file = *./data/3did.list*
- Path of the file with the database of FASTA sequences generated after running modppi:
  database_file = *./database.fasta*

Parameters

This section contains the parameters used to generate the models of protein-protein interactions:

- *PPI distance threshold and PPI threshold type*
  A pair of amino acids are in contact if the distance between their specific atoms is less than a distance threshold (usually 8 Å). The choice of distance threshold and sequence separation threshold also defines the number of contacts in a protein. At lower distance thresholds, a protein has fewer number of contacts and, at a smaller sequence separation threshold, the protein has many local contacts.
  When generating the data we can change the parameters of the configuration file in order to set different distance thresholds:
  1) Use contact by minimum distance of 6 Å:
  PPI_threshold_type = min PPI_distance_threshold = 6
  Store the files as pdb_Min6.ppi and pdb_Min6.dat by specifing the name in the section Files of configuration file: ppi_files = ./data/pdb_Min6

  2) Recalculate by Cβ-Cβ distance of 12 Å:
  PPI_threshold_type = cb PPI_distance_threshold = 12
  Store them as pdb_CB12.ppi and pdb_CB12.dat by specifing the name in the section Files of configuration file: ppi_files = ./data/pdb_CB12

  3) Finally use a more stringent cut-off 4.5 Åand minimum distance:
  PPI_threshold_type = min PPI_distance_threshold = 4.5 Å
  Save it as pdb_Min4.ppi and pdb_Min4.dat by specifing the name in the section Files of configuration file: ppi_files = ./data/ pdb_Min4
  Now we can select either pdb_Min4, pdb_Min6 or pdb_CB12 and use it as pdb.ppi. Due to some errors in modelling, we suggest the use of pdb_Min4.ppi to generate the database, but then use Cβ-Cβ distance of 8.0 for modelling.

- *PPI distance threshold shell and overlap interface*
  To analyze mutants we define a shell for clustering mutants with the similar interface larger than 10 Å(ie. from 12 to 20 Å), and a minimum overlap of the same residue-residue interactions of 5%:
  overlap_interface = 5

- *Minimum ratio common poses*
  Also for the analyses a minimum number of poses with the same interface is required to cluster them in the same group of study:
  min_ratio_common_poses = 0.49

- *Hydrogen and relax*
  In order to evaluate the models we also decide whether to relax or not the model and if adding hydrogens (full, polar or none):
  hydrogens = full
  relax = yes

- *Twilight zone*
  Models will be done if they are acceptable according to the Rost curves strength parameter:
  n_parameter_twilight_zone = 0

2 Installation of data

First of all we need to generate the data that MODPIN will use to create the models.
Beware, for running the comands of "Installation of data" step we are supposing you are in the **location of your Modpin installation**. To find out where Modpin is installed run:

```
pip show modpin
```

Then open the terminal in that location. Below there is an example, but the path may change depending on where you chose to install Modpin:

```
cd /home/user/.local/lib/python2.7/site-packages
cd modpin
```

2.1 Data from PDB

We need to import PDB sources and create the FASTA files.
In order to do this run the comand below:

```
python src/functions/PDB2SQL.py -d data/pdbsource -q
    data/pdbseq -s data/pdbsql -v
```

This comand will run PDB2SQL.py script with the " -d " option in order to create the destination directory for PDB database where the compressed files of PDB entries will be stored. The "-q" option for destination directory of PDBseq database. Finally the "-s" flag to create the output directiory for SQL.

Then we will create a list with path and file name of all the entries contained in pdbsource folder, obtained in the above step.

Run this comand from terminal in MODPIN folder:

```
\ls `pwd`/data/pdbsource/*/pdb* > data/pdb.list
```

Now we are ready to initialize and write the set of protein-protein interactions from PDB.

If you have installed Modpin run the comand below:

```
modppi -init_pdb -v
```

Otherwise if you have downloaded but not installed it open the terminal in Modpin folder (or specify the full path to "modppi.py" script) and run:

```
python scripts/modppi.py -init_pdb -v
```

We run modppi with "-init_ pdb" flag to force initialization of dataset from PDB. In this way "PDB2PIN.py" script will use the "pdb.list" file to create a network of PPI's extracted from PDB complexes.

2.2 Data from 3did

The 3did, database of three-dimensional interacting domains, is a collection of protein interactions for which high-resolution three-dimensional structures are known. You need to download the "3did_ flat" file (you can find it here: https://3did.irbbarcelona.org/download.php) that contains interacting domain pairs (ID) and instances of these interactions in PDB structures along with InterPreTS scores (Aloy and Russell, Bioinformatics 2003) and details on the contacts.

If you have installed Modpin run the comand below:

```
3did -i path/to/3did_flat -ppi 3did.ppi -seq 3did.fasta -v
   --dummy=dummy
```

Otherwise if you have downloaded but not installed it, open the terminal in MODPIN folder (or specify the full path to "3did.py" script) and run:

```
python src/functions/3did.py -i path/to/3did_flat -ppi
    3did.ppi -seq 3did.fasta -v --dummy=dummy
```

This will create two output files, one with a list of interactions from 3did and one with FASTA sequences of the proteins from 3DiD.

And now we can create a list with the obtained 3did files by running this comand:

```
ls data/3did/*brk > data/3did.list
```

2.3 Whole dataset

Now we need to retrieve the universe of sequences obtained from PDB and from 3did and store them in a file, named nr.fa (non redundant), already defined in the configuration file. In order to to this, run the comand below:

```
cat data/pdbseq/PDBseq.fa >  data/nr.fa
cat data/3did.fasta >> data/nr.fa
```

Avoid redundancies: Create a non-redundant set of sequences combining PDB and 3DiD .

We can reduce the overall size of the database by removing redundant sequences thanks to CD-HIT (Cluster Database at High Identity with Tolerance).The program takes a fasta format sequence database as input and produces a set of 'non-redundant' (nr) representative sequences as output and a text file of list of clusters.
Essentially, cd-hit produces a set of closely related protein families from a given fasta sequence database http://weizhongli-lab.org/cd-hit/ .

You can set an alias for cd-hit, if you want to avoid writing the whole path of the program when using it, like for example:

```
alias cd-hit =
    '/path/to/ch-hit/folder/CD-HIT/4.6.4/bin/cd-hit'
```

Then from temrinal in MODPIN folder run:

```
cd-hit -i data/nr.fa -o data/nr90.fa  -n 5  -c 0.9
```

The input is the whole dataset (nr.fa), located under the data folder and the output will be a file named "nr90.fa" since "-c 0.9" option means 90% identity in the clustering threshold, while "-n 5" is the word size (for thresholds between 0.7 ˜1.0).

We can create a non-redundant set of sequences joining PDB and 3DiD non-redundant. The from temrinal in MODPIN folder run:

```
cd-hit -i data/pdbseq/PDBseq.fa -o data/pdb90.fa -n 5 -c 0.9
cd-hit -i data/3did.fasta -o data/3did90.fa  -n 5 -c 0.9
cat data/pdb90.fa > data/merged_nr90.fa
cat data/3did90.fa >>  data/merged_nr90.fa
cat data/pdb90.fa.clstr > data/merged_nr90.fa.clstr
cat data/3did90.fa.clstr >> data/merged_nr90.fa.clstr
```

2.4 Initialize dataset of sequences

If you have installed Modpin then run the comand below:

```
modppi -init_blast -3did -v
```

Otherwise open the terminal in Modpin folder (or specify the full path to "modppi.py" script) and run:

```
python scripts/modppi.py -init_blast -3did -v
```

We set "- 3did" flag to include domain-domain interactions from 3DiD and "- init_blast" flag to format the sequence dataset for BLAST.

3 Modeling

This method performs automatic modelling of protein-protein interaction by reading the input data, creating the database, blasting all the requested interactions and modeling them.

3.1 Script overview: modppi.py

This script reads the input file with a list of FASTA sequences of the proteins to test (flag *"-seq"*) and the input file with a list of the interactions to test (flag *"-ppi"*).
Then looks in the configuration file for installation paths of the required external software and also for the parameters specified for generating the models.
In the initialization step we need to store and check the dataset sequences but once this is done we can skip the check of dataset sequences to fasten up the modeling (flag "-skip").
A network of PPI's extracted from PDB complexes is created using the PDB2PIN.py script, that takes in input a list with the PDB entires. This step is skipped if the file already exists unless forcing initialization (flag "-init_pdb").
Domain-domain interactions from 3did databse can be also included (flag *"-3did"*). If not existing the database of sequences is created and then formatted and indexed.
After this the input file of protein-protein interactions is read and groups A and B are created in order to store the pair of interacting proteins. FASTA sequences are then retrieved from the input query sequences file and also stored in the corresponding query group.
Then for both query sequences group A and query sequences group B we need to do blast, so a FASTA file is created for every protein name, and stored in the blast subdirectory of the dummy folder, where also the blast output results will be stored. So *blastpgp* that performs gapped blastp searches is executed using the previously created FASTA database and the hits are filtered by Rost's sequence identity curve. For each obtained hit checks if it's equal to the query and then store to used them in the modeling process.
Finally modeling can be done, so an ID is assigned to the dummy modeling in order to avoid overwriting files.

In this step PDB file will be created and in order to do this it is necessary to look for a template in the PDB source folder or among the 3did files. If it is not found it jumps to the next interaction for which a template can be found, then only the chains present in the alignment are added and the dummy PDB file is created.
Then contacts are checked and if the proteins don't form a complex the modeling it will be avoided otherwise the interaction between template A and template B is accepted. If needed, remaining residues or gaps are added at the beginning or at the end of the template sequences.
The PIR alignment and the folders for models of each type of interaction are created and if not existing the models are added to the list of models.
The set of models is then completed using MODELLER, hydrogen are added if specified by the corresponding option, the model is relaxed (Rosetta output in relax.log and score.sc files) and residues are renumerated as original sequence.

If you want more information on the usage of this script run it with the help option *"-h"*:

```
modppi -h
```

3.2 Model building

We need to provide two input files:

1. a list of FASTA sequences of the proteins to test, (flag *-seq*).
2. a list of the interactions to test, (flag *-ppi*).

If Modpin is installed run the comand below:

```
modppi -seq example/baxbid.fa -ppi example/baxbid.ppi -o
   example/BAX_BID -d ./dummy -v --hydrogens -skip -force
   -3did -n 5 --renumerate
```

Otherwise open the terminal in Modpin folder (or specify the full path to "modppi.py script" and run:

```
python scripts/modppi.py -seq example/baxbid.fa -ppi
    example/baxbid.ppi -o example/BAX_BID -d ./dummy -v
    --hydrogens -skip -force -3did -n 5 --renumerate
```

With this comand we specify the directory in which we want to store the results (flag *"-o"*) and the dummy folder with the temporary files (flag *"-d"*). So we need to clean the dummy files and force the modeling even if the models already exist (flag *"-force"*),then we can add hydrogen atoms on the models (flag *–hydrogens*) and include domain-domain interactions from 3did ("-3did"). To speed things up we can skip checking the database of known interactions and their sequences (-skip).

4 Analysis

4.1 Script overview: analysis.py

This script creates a list of models storing the name of each pair of interactors and a list with the cluster name(pose) and the name of the file with the list of PDB files.
To do this it selects the active models and if requested by the corrsesponding option it will add hydrogens and it will renumerate PDBs.
Then files with similar interfaces will be grouped according to the criteria specifed in the configuration file.
So it will try to get the name of the wild-type and if there is no wild-type it will select the pair with the biggest cluster to use as basic core in substitution of the wild-type.
Then the clusters of Wild Type are ranked by size and a lists with grouped structures is created. It writes a list of non redundant interfaces for wild-type and pose 0 is reserved for all structures whithout clustering, next steps pose will increase by 1. Later it compares and group similar poses to the wilde type for the other pairs and the remaining poses are ranked at the end by the size of the clusters.
Then it selects the poses of wilde type that corresponds to each cluster of similar interfaces.
After the creation of the list of models it checks the sequences of the pairs and the differences of the interface for each cluster-pose, in this case will store the cluster-pose name and the pairs of interactors and also the ratios

of structures where a difference in sequence with respect to its wild-type is involved in the interface.
Finally, if the option *"-zrank"* is specified Zrank will run and after obtaining its results the changes in binding free energies upon mutation are calculated using Rosetta.
Interface Analyzer will be used since it combines a set of tools to analyze protein-protein interfaces.
It calculates binding energies, buried interface surface areas, and other useful interface metrics.

If you want more information on the usage of this script run it with the help option *"-h"*:

```
analysis -h
```

4.2 Analysis of the models

After the models are generated we can analyse them.
If Modpin is installed, to do the analysis, run the following comand:

```
analysis -o example/BAX_BID/ -l BAX_BID -zrank -v -d dummy
    -boxplot -ppi example/BAX_BID/interactions_done.list -seq
    example/baxbid.fa
```

Otherwise open the terminal in the folder of Modpin (or specify the full path to "analysis.py" script) and run:

```
python scripts/analysis.py -o example/BAX_BID/ -l BAX_BID
    -zrank -v -d dummy -boxplot -ppi
    example/BAX_BID/interactions_done.list -seq
    example/baxbid.fa
```

5 Modelist

This method is called Modelist and it will run in a cluster several sequences with mutant forms.
For the method to work, it requieres that mutant forms are specified in the name of the FASTA file. It reads the sequences from the tinput FASTA file and the also the pairs of proteins to group and test from the input query list.

- >NATIVE and >NATIVE_G84D
  It refers to a wild type protein named NATIVE and its mutant form G84D.

- >sp|P55957_R84W|BID_HUMAN_R84W and >sp|P55957|BID_HUMAN
  It refers to a wild type protein with accession P55957 and entry BID_HUMAN and its mutant form R84W.

5.1 Models creation

Create models of two large sets of mutations involving rewiring (hydrogens and relaxation won't be added in order to fasten up the obtantion of the models).

For the **rewiring** case, if Modpin is installed run the comand below:

```
modelist -i example/rewiring_mutants.dat  -l REWIRED -seq
    example/uniprot_Marc_Vidal.fasta -o example/rewired -n 3
    -d dummy -3did -v --parallel -opt
```

Otherwise open the terminal in Modpin folder (or specify the full path of "modelist.py" script), and run:

```
python scripts/modelist.py -i example/rewiring_mutants.dat
    -l REWIRED -seq example/uniprot_Marc_Vidal.fasta -o
    example/rewired -n 3 -d dummy -3did -v --parallel -opt
```

For the **unrewiring** case, if Modpin is installed run the comand below:

```
modelist -i example/unrewiring_mutants.dat  -l UNREWIRED -seq
    example/uniprot_Marc_Vidal.fasta -o example/unrewired -n
    3 -d dummy -3did -v --parallel -opt
```

Otherwise open the terminal in Modpin folder (or specify the full path of "modelist.py" script), and run:

```
python scripts/modelist.py -i example/unrewiring_mutants.dat
    -l UNREWIRED -seq example/uniprot_Marc_Vidal.fasta -o
    example/unrewired -n 3 -d dummy -3did -v --parallel -opt
```

5.2 Models analysis

Then after the models are create we can continue the analysis, we have to add hydrogens and optimize structure and the file with hydrogens will be the same as the PDB input with extension ".h" instead of ".pdb".

If Modpin is installed run the following comand:

```
modelist -i example/rewiring_mutants.dat  -l REWIRED -seq
    example/uniprot_Marc_Vidal.fasta -o example/rewired -n 3
    -d dummy -3did -v --parallel -opt -a --continue
    --hydrogens --renumerate
```

Otherwise open the terminal in the downloaded folder of Modpin, or specify the full path to the "modelist.py" script, and run:

```
python scripts/modelist.py -i example/unrewiring_mutants.dat
    -l UNREWIRED -seq example/uniprot_Marc_Vidal.fasta -o
    example/unrewired -n 3 -d dummy -3did -v --parallel -opt
    -a --continue  --hydrogens --renumerate
```

5.3 Cluster selection

To further analyze and compare the whole set of mutations we run the script "select_cluster".
This produces a selection of **clustered interfaces** for each complex with mutant forms compared with the wild type according to two statistics (Mann-Whitney and Kolmnorov-Smirnov). Plots are collected in the output folders *example/rewired* and *example/unrewired*.

Open the terminal in modpin folder and run:

```
python src/functions/select_cluster.py example/rewired
python src/functions/select_cluster.py example/unrewired
```

6 Extra utilities

If you forgot to renumearte a file you can fix it when the model is done. The program requires the original sequences in FASTA format and an input file to indicate the sequence name of each chain in the complex.

If Modpin is installed run the comand below:

```
Renumerate -f example/sequences_Marc_Vidal.fasta -p
    example/test4renumber.pdb -d dummy -l
    example/renumerate_P02792_H133P_P02794.dat -o
    example/renumbered.pdb -v -hydro
```

Otherwise, open the terminal in the downloaded folder of Modpin, or specify the full path to the script, and run:

```
python scripts/Renumerate.py -f
    example/sequences_Marc_Vidal.fasta -p
    example/test4renumber.pdb -d dummy -l
    example/renumerate_P02792_H133P_P02794.dat -o
    example/renumbered.pdb -v -hydro
```

The file "renumerate_P02792_H133P_P02794.dat" (in the "example" folder)

contains the sequences of each chain P02792_H133P (FRIL_HUMAN_H133P in chain A) and P02794 (FRIH_HUMAN in chain B) and the PDB file to renumber is test4renumber.pdb.