Tutorial: Dynamical Perturbation Network Analysis

Gheeraert Aria
Contact: aria.gheeraert[at]univ-savoie.fr
With extensive help from Lorenza Pacini and Rodrigo Dorantes-Gilardi

October 23, 2019

Requirements

The version of the different packages are only indications. Other versions *should* work but haven't been tested.

- Python 3.7.3
- matplotlib 3.1.0
- numpy 1.16.4
- pandas 0.24.2
- tqdm 4.32.1
- BioPython 1.74
- networkx 2.3

1 Installation

1. For git users, via command line:

```
git clone https://github.com/agheeraert/PerturbationNetworkAnalysis.git
```

2. Or download zip and extract in a folder:
https://github.com/agheeraert/PerturbationNetworkAnalysis/archive/master.zip

2 Usage

The aim of this python script is to compute the dynamical perturbation network of a protein. Usually this will be the comparison between MD simulations of a protein in a *perturbed* state and in a *unperturbed* state. To use this script, one must extract PDB frames from one (or more) MD simulation of the *pertubed* protein and the *unperturbed* protein and to put them in two distinct folders. One also needs to create a folder for the output results.

To use the script, you have to launch the main file (*main.py*) in command line with several arguments that corresponds to the different options.

```
python main.py path1 path2 output [-cutoffs [CUTOFFS ...]] [-avg AVG]
              [-drawing_method DRAWING_METHOD] [-pdb_path PDB_PATH]
              [-drawing_colors DRAWING_COLOR_1 DRAWING_COLOR_2]
```

Required arguments:

- path1 Input file/folder of the unperturbed state frame(s)
- path2 Input file/folder of the perturbed state frame(s)
- output Folder where to put the results

Optionnal arguments:

- cutoffs A list of integers that will be used as the different cutoff values (in Å) to build the amino acid networks. Default value, 5 Å.
- avg By default, the script only works comparing two frames. If a string (arbitrary) is specified here, the script will compute the average perturbation network over the frames contained inside the folder.
- drawing_method Method used to draw the networks. Only two methods are available now:
  - default = Uses the spring layout algorithm to draw the network. With a big number of nodes this is usually a bad idea.
  - IGPS = IGPS splitting (should only be used for the IGPS). Should be easily adaptable to other systems.
- drawing_colors Two strings that represent Matplotlib colors. These colors are used to draw the edges network. The first color is the color used to represent edges with a bigger weight in the *unperturbed* state while the second color is the one used to represent edges with a lower weight.
- pdb_path PDB structure file to help draw the network (works only with IGPS drawing method as of now).

Please note that a short help is also accessible through the command line:

```
python main.py -h
```

Examples

```
python main.py /home/aghee/PDB/Apo_frames/Sim1/frame1.pdb
/home/aghee/PDB/Prfar_frames/Sim1/frame1.pdb /home/aghee/results/test_tutorial/
```

Computes the perturbation network at cutoff 5 Å between the two specified frames and outputs the results in the test_tutorial/cutoff5/ folder. The networks are drawn with the spring layout algorithm.

```
python main.py /home/aghee/PDB/Apo_frames/Sim1 /home/aghee/PDB/Prfar_frames/Sim1
/home/aghee/results/test_tutorial/ -avg 1 -drawing_method IGPS
-pdb_path /home/aghee/PDB/base_IGPS.pdb
```

Computes the average perturbation network at cutoff 5 Å between the frames contained in the two specified folders and outputs the results in the test_tutorial/cutoff5/ folder. The networks are drawn with the algorithm specific for IGPS.

```
python main.py /home/aghee/PDB/Apo_frames/Sim1 /home/aghee/PDB/Prfar_frames/Sim1
/home/aghee/results/test_tutorial/ -avg 1 -cutoffs {3..9} -drawing_method IGPS
-drawing_colors red dodgerblue -pdb_path /home/aghee/PDB/base_IGPS.pdb
```

Same but computes the average perturbation network at cutoffs between 3 and 9 Å.

3 Results

Inside the output folder, a folder is created for each cutoff used to draw the perturbation networks. These subfolders contains two kind of files:

- X.p Pickled NetworkX representation of the network at threshold X. These files can be re-opened within a python script with the command nx.read_gpickle(path).
- X.pdf Graphic representation of the network at threshold X.

This script creates perturbations networks for values of threshold between 0 up until the network is empty with a step of 1.

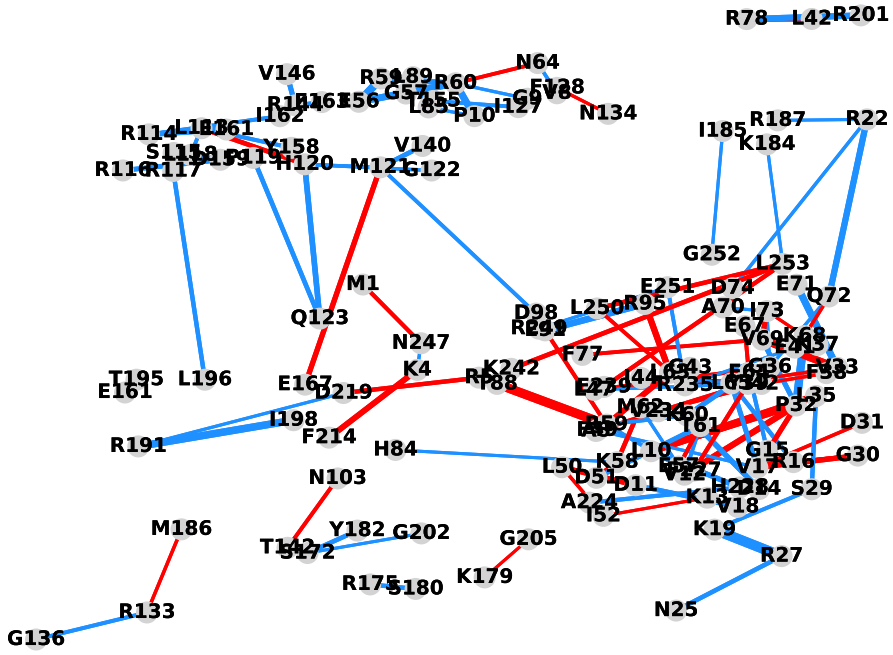

Figure 1: 25.pdf output with IGPS drawing method, red and dodgerblue colors.

4 Vizualizing the network with VMD

The script analysis/network_vmd.py allows to visualize the network with VMD.

This script needs two files: the pdb structure on which to draw the network and the pickled NetworkX representation of the network. Then run the command

`python network\_vmd.py pdb_file networkx_file output -nc [1] -ntodraw [LIST OF NODE LABELS TO DRAW] -norm [NORM]`

Required arguments:

- pdb_file path of the pdb structure to draw the network on
- networkx_file path of the pickled NetworkX representation of the network
- output path of the output (should be in .tcl format)

Optional arguments:

- nc Should be 1 if you want to draw your network without colors on the edges (useful for amino acid graphs)
- ntodraw A list of nodes if you want to draw only these nodes and edges connecting them (useful for zooming)
- norm Changes the normalization factor of the size of the edges (by default it's 1.5)

Then in VMD, you source the .tcl output file in a TkConsole after loading your PDB file.

If you want to change the colors, you can do it manually in the script with the colors that VMD uses. 1 = red, 7 = green and 23 = light blue.

Examples

To draw one network with bash:

```
python network_vmd.py /home/aghee/PDB/base_IGPS.pdb
/home/aghee/results/test_tutorial/cutoff5/25.{p,tcl}
```

To draw multiple networks with bash:

```
for file in $(ls /home/aghee/results/test_tutorial/cutoff5/*.p);
do python network_vmd.py /home/aghee/PDB/base_IGPS.pdb $file
$(echo $file | cut -f 1 -d .).tcl; done
```

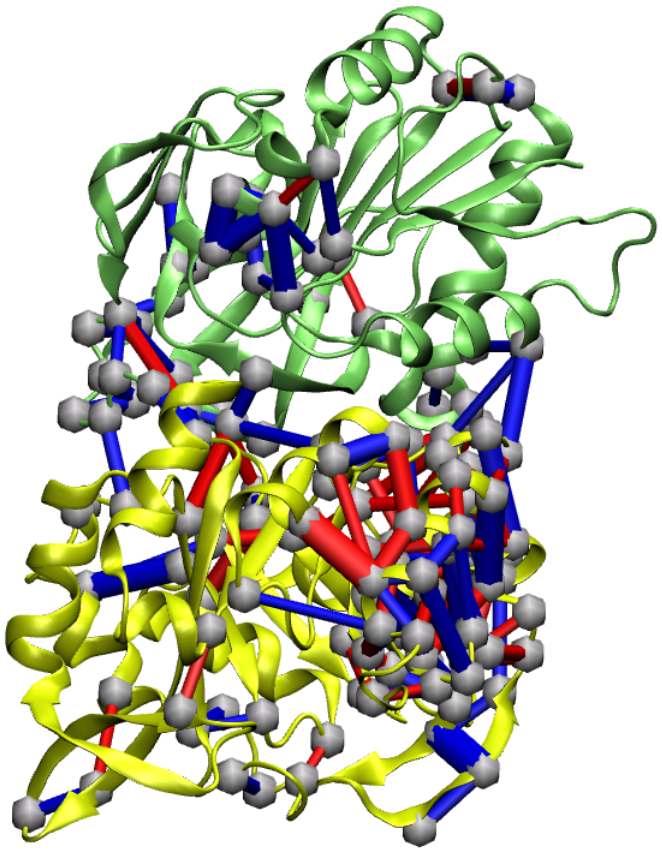

Figure 2: Vizualization of the network built at cutoff 5 and threshold 25 on IGPS.

5 Troubleshooting

5.1 Some residues from my MD simulation aren't recognized

In MD simulations, the 3-letter name of some residues is sometimes changed because of the protonation state. This issue can be solved in two ways: Changing the 3-letter amino acid name in the frame files (not recommended) or adding a line in the __init__ method in the class CreateNetwork in the file CreateNetwork.py

```
self.three2one['3-letter atypical name'] = 1-letter typical name
```

5.2 Something is wrong with the file path I give

Python doesn't handle well the ~ shortcut in file paths, I recommand that you type directly your /home/user/ folder instead.