GAOptimizer Manual

Introduction

GAOptimizer.py is a script that uses a genetic algorithm to optimize the combination of consensus mutations that stabilize a protein structure. It introduces a consensus mutation into the template protein structure (PDB) by random mutation (specified by mutation_rate, default is 30%) or by recombination mutation (70%). Total 100 of children would be generated per one generation, and selection of the children would be performed as tournament selection based on the scores implemented in PyRosetta. PyRosetta's score function is used as a selection pressure in a tournament style (randomly select 5 children, and choose the PDB with the best score among them. This is repeated 30 times [the selected PDBs are stored under temporary_selected/]. Finally, the PDB with the lowest score in temporary_selected/ is selected as the parent of the next generation. This process is repeated for hundreds to thousands of generations to optimize the combination of consensus mutations.

In the first 50 generations, the parameters are set so that random mutations are easily introduced, and after that, random and recombinant mutations are introduced with the above probability (30% for random mutation and 70% for recombination). For each generation, the target protein (input PDB data) and the sequences saved under the “Library/” directory are randomly selected, aligned by MAFFT, and the frequency of occurrence of each amino acid for each residue of the target protein is calculated. The output file was saved as “tmp_ intmsa.out”. By referring to this file, the consensus residues are identified and introduced into target protein by PyRosetta.

Required software

Python 3.7.6 or higher, PyRosetta-4, Biopython 1.77, MAFFT (should be able to run from command line)

Required files

3D structure of the template protein to introduce the mutation (PDB data)
Sequence data of the template protein homologs used to identify consensus mutations (Fasta format, up to 50 sequences are recommended)

Procedures

Building the program execution environment

1. Download and unzip the source file from the MAFFT website. Follow the instructions to complete the installation (https://mafft.cbrc.jp/alignment/software/).
2. Start a terminal and type "mafft". Make sure there are no errors.

3. Download the source from PyRosetta-4 (https://www.pyrosetta.org). Installation of the software should be completed by following the instructions.
4. Install Biopython (e.g. pip and conda).
5. Launch Python interactive mode (from command line, type "python") and type import pyrosetta. Make sure you don't get any errors (from the terminal, type python. If you have more than one python installed, make sure you know which python you have installed 3 and 4).
6. Next, type import Bio from command line. Make sure there are no errors.
7. Stop the Python interactive mode (shift+D).

Processing the input files and preparing before running the program

0. Make a directory to perform analysis. In the directory, save EM.py, PDB data, and homolog sequence data.
1. (Processing PDB file) Open the PDB file with a text editor and remove all HETATM, HOH and solvent molecules. If you want to reduce the computation time, use only one chain.
2. After completion of the processing PDB file, save the PDB file as ****-clean.pdb.
3. Make a directory (e.g. Library/) to store the sequence data of the template protein homologs, and move the sequence data into it (mv command).
4. (Important) Perform energy minimization on the ***-clean.pdb structure. Use the additional script "EM.py" for this purpose. The PDB name after energy minimization should be ***-clean-EM.pdb. Here is how to run the program (arguments, etc.)

```
python EM.py -PDB ***-clean.pdb -OUTPUT ***-clean-EM.pdb
```

5. Run the program in 4, and check the score output at the end on the terminal. repeat 4 until there is no energy difference between before and after EM.
6. Open the pre- and post-EM structures in PyMOL, and check that no molecules are missing and no major structural changes have occurred.
7. Open the post-EM PDB in a text editor, remove any extraneous text other than the PDB code, and save it (***-clean-EM.pdb).

Preparation of the ligand file (if necessary).

In order to run the calculation considering molecules that do not exist in the parameter file on PyRosetta, it is necessary to create a parameter file for the ligand. The procedure is as follows. (Note): It is recommended to minimize (EM) the ligand molecules using external software. Here, we explain the procedure using the ligand file after EM.

1. Open the ligand file in PyMOL, and select File-Export Molecule.... and save the file in MDL format (,mol) (***.mol. *** is the 3-letter notation of the ligand).

2. Download the molfile_to_param.py source files from the PyRosetta website and unpack them (tar -xvf ***).
3. From the command line, type python molfile_to_params.py ***.mol -n ***. Make sure that ***.params and ***_0001.pdb are output without errors. Use python 2.x here, Python 3 will give you an error.
4. Add the output ****.params to the installed PyRosetta

/minirosetta_database/chemical/residue_type_sets/fa_standard/residue_types
Copy and paste it into the following directory. After that
/minirosetta_database/chemical/residue_type_sets/fa_standard/residue_types.txt
in the same format as the other parameter file definitions.

Perform combinatorial optimization of consensus mutations

Next, introduction of random or recombination mutations into the 3D structure of the template protein was performed using PyRosetta. Here, combinatorial optimization of the mutations was performed using a genetic algorithm. Since it takes a lot of computation time, run at least three calculations in parallel (since only one CPU is used. This will save computation time).

1. Create a directory for analysis from the command line (mkdir -m 755 ***-analysis-n1). (mkdir -m 755 ***-analysis-n1). Label the directory n1 at the end to indicate how many times it has been analyzed.
2. Copy and paste GAOptimizer.py into the analysis directory created in step 1.
3. Save the homologous sequence of the template protein used to identify the consensus mutation in Fasta format. Save it in the “Library/” directory.
4. Prepare the PDB data of which energy was minimized by script “EM.py”. The PDB data was saved as “***-clean-EM.pdb”
5. Run GAOptimizer.py from the command line. Enter the following command.

```
python GAOptimizer.py -PDB ***-clean-EM.pdb -DIRECTORY Library -GENNUM 1000 -
CHAIN A -OUTPUT ***-n1.log
```

EXPLANATION OF ARGUMENTS:

-PDB: Specifies the PDB file after EM.

-DIRECTORY: Specify the sequence library used for consensus mutation identification. No “/” (slash) is required (fasta format, with one space between data).

-GENNUM: Specify the number of generations (1000 is recommended).

-CHAIN: Specify the chain to put the mutation.

-OUTPUT: Specify the log name to output.

NOTE: Using AncLAAO-N5 (M.W. = 75 kDa, monomer), the calculation takes about 12 minutes per generation. 1000 generations would take about 250 hours.

[Additional Function]

To perform mutation optimization with HiSol values, enter -HISOL in the command line of 5. This will design a mutant that introduces as many soluble hotspots as possible into the target protein, while keeping the structural stability unchanged from that of the template.

Analysis of the results

After the calculation is completed, the following information will be written to the log file specified by -OUTPUT.

#parameters: contains sample_num (the number of individuals to be generated per generation [number of pdb]), mutation_rate (the probability of mutation [30% in the following]), tournament_num (the number of tournaments defined by tournament selection), Inputpdb (the input pdb),

#Input_Chain (the name of the chain that introduced the mutation).

#The hisol flag is on (The flag would be on when the program was executed by adding -HISOL)

#rosetta score values of average, elite, and original scores. (The important parameters were second and third column in each lines. For examples, at 2nd generation, the rosetta score and HiSol score of the elite were -635.740 and 421.004, respectively)

```
#parameters:sample_num=100, mutation_rate=30 percent, next_genet_num=30
#Inputpdb: GA_NitA_cut.pdb-clean-EM.pdb-clean-EM.pdb-clean-EM.pdb
#Input Chain: A
#The hisol flag is on
#rosetta score values of average, elite, and original scores (for the case of -HISOL and -
EVOLV) of the next_generation pdb
0: 26981.021, -637.114, 421.004
1: 30298.818, -637.078, 421.004
2: 30295.965, -635.740, 418.430
3: 7059.850, -644.908, 415.487
4: 3729.652, -635.404, 403.795
```

NOTE: Make sure that the Rosetta scores are converged and that the final pdb sequences obtained from the three independent trials are consistent with each other.