**FluProCad**

**Introduction**

FluProCad is a package with short python and bash scripts which can introduce amino-acid mutations in pdb files for fluorescent proteins, generating Gromacs compatible topologies, for molecular dynamics, extracting 1 or more solution structures for the mutant protein (comparable/equivalent to experimental crystal structure) with additional trajectory analysis.

**Software Requirements**

* Python
* Modeller
* Gromacs

We will start by downloading the scripts from github and extracting the files from the **FluProCad** tarball file on a cluster.

* **git clone https://github.com/vavmodi/FluProCad-2.0.git**
* **scp -r FluProCad-2.0.tar.gz username@cluster.domain:/<work-dir>**
* **tar -xzvf FluProCad-2.0.tar.gz**

**Installing Modeller package**

Before we start with the installation, run the following command in your terminal/console to check your python version and system architecture:

* **python –version**
* **uname -a**

This will allow us to install the version of MODELLER supported by your system specific architecture. The currently supported architecture should be one of these options:

1. **Linux x86 PC (e.g. RedHat, SuSe).**
2. **IBM AIX OS.**
3. **x86\_64 (Opteron/EM64T) box (Linux).**
4. **Alternative x86 Linux binary (e.g. for FreeBSD).**

Next, [install](https://salilab.org/modeller/download_installation.html) the molecular modelling package MODELLER. You will also need to [register for a license](https://salilab.org/modeller/registration.html) which is available free of charge for academic non-profit institutions.

The license key is delivered by email which can take ~5-20 minutes.

Download the “*Generic Unix tarball*” version and change to the directory where you downloaded the.tar.gz file. Unpack the file with following command:

(replace “modeller-v.xx” with the MODELLER version you have downloaded. *Ex:” modeller-9.22*”):

* **mv ~/Downloads/modeller-v.xx.tar.gz FluProCad-2.0/**
* **cd FluProCad-2.0**
* **tar -xvzf modeller-v.xx.tar.gz**

Since MODELLER is used here as a python module, check the version of python installed on your system. Currently any version between 2.3 and 3.7 is supported by MODELLER.

We will now run the following command to add the MODELLER library path into the installation file:

(replace “modeller-v.xx” with the MODELLER version you have downloaded. *Ex: “modeller-9.22*”):

* **cd modeller-v.xx**
* **cat Install ../mod\_path.dat > Install\_new**
* **sed -I “s/mkdir -p/mkdir -p -m 755/” Install\_new**
* **chmod u+x Install\_new**

Run the installation script and answer the questions as prompted. If you make any mistakes, you can re-run the script.

**#NOTE:**  The default installation directory for Modeller can be changed during installation

* **./Install\_new**
* **source ~/.bashrc**
* **cd ../**

1. **Preparing Input files and Topologies**

**#NOTE:** For using the pre-existing shared installation of Modeller export the following paths or add them to the bashrc file.

* **export MODINSTALL="/projappl/vamodi/modeller9.22"**
* **export EXECUTABLE\_TYPE="x86\_64-intel8"**
* **export PYTHONPATH=”${PYTHONPATH}:${MODINSTALL}/modlib:${MODINSTALL}/lib/${EXECUTABLE\_TYPE}/${PYTHON\_VERSION}”**
* **export LD\_LIBRARY\_PATH=”${LD\_LIBRARY\_PATH}:${MODINSTALL}/lib/${EXECUTABLE\_TYPE}”**

We are now going to set up our first mutation simulation. With this example we will prepare the starting structure for our mutant protein. Firstly, edit the file mutations.dat to add the residue index of the residue you want to mutate and what you want to mutate it into.

**<residue\_index> <3-letter-Residue\_name> <single-letter-ChainID>**

**Ex: 206 ALA A**

1. **L A**

Load the GROMACS and supporting modules on your local system/cluster if necessary:

* **module load gcc/9.1.0**
* **module load hpcx-mpi/2.4.0**
* **module load gromacs/2020.3**

Next, check the GMX prefix compiled for the loaded/installed Gromacs version on your system (Ex: gmx, gmx\_mpi, or none). Now, run the script md\_setup.sh to prepare the mutant structure and topologies for our simulations.

**#NOTE:** Here we use the GROMACS version 2020.3 so for every time the script prompts asking for the chosen Gromacs version you can select "2020.x" and "gmx\_mpi" for the prefix notation.

**./md\_setup.sh <PDB\_filename> <output\_suffix>**

(Ex: For the input PDB files names protein.pdb we will run the following command:)

**./md\_setup.sh protein test**

The script will call MODELLER package to generate a new mutant structure called *“PDB\_filename-suffix.pdb”.* Further, it will prepare the Gromacs topologies and solvate the protein in a neutral water solvent box with 0.15M concentration.

There should now be a new directory called *<PDB\_filename-suffix>* with the topologies, forcefield and *.mdp* files that we will use to start out MD simulations.

**2. Equilibration and Production MD Simulations**

We are once again going to use grompp to assemble the structure, topology, and simulation parameters into a binary input file (.tpr).

If you have access to a supercomputer/supercluster or local cluster, we can use a bash script submit a parallel job as our system is rather large and may need significant computational time.

* **cd *<PDB\_filename-suffix>***

Open the file *gmx-jobs.sh* with a text file editor of your choice or in the terminal. Here you can edit the number of nodes/cores you prefer to use for the job. Edit the name of the Gromacs module corresponding to the modules available on the cluster in use.

Below are the contents of the bash script file which can be used for submitting a batch job on the ***PUHTI*** cluster at CSC:

(#NOTE: IF YOU PREFER USE YOUR LOCAL SYSTEM TO RUN THE SIMULATIONS, REFER TO THE NEXT SECTION)

* **vi gromacs\_jobs.sh**

#!/bin/bash

#SBATCH --job-name=FPC-test

#SBATCH --account=<*project\_name\_CSC*>

#SBATCH --partition=large

#SBATCH --time=24:00:00

#SBATCH --nodes=4

#SBATCH --tasks-per-node=24

#SBATCH --mem-per-cpu=4000

# this script runs a 192 core (8 full nodes) gromacs job.

export OMP\_NUM\_THREADS=1

module load gcc/9.1.0

module load hpcx-mpi/2.4.0

module load gromacs/2020.3  #change if you want a different version

gmx\_mpi grompp -f em.mdp -p topol.top -c ions.gro -o em

srun gmx\_mpi mdrun -deffnm em

gmx\_mpi grompp -f nvt.mdp -p topol.top -c em.gro -r em.gro -o nvt

srun gmx\_mpi mdrun -deffnm nvt

gmx\_mpi grompp -f npt.mdp -p topol.top -c nvt.gro -t nvt.cpt -o npt

srun gmx\_mpi mdrun -deffnm npt

gmx\_mpi grompp -f md.mdp -p topol.top -c npt.gro -t npt.cpt -o topol

srun gmx\_mpi mdrun -s topol -dlb yes

* **cp gromacs\_jobs.sh <PDB\_filename>-<output\_suffix>/**

Here is an example for submitting a SLURM/PBS batch job on a cluster. (Please check the correct of commands for submitting a batch job if your cluster supports only PBS commands.)

* **sbatch gromacs\_jobs.sh**

To check the state of the submitted jobs use the following slurm command:

* **squeue -u *<CSC-username>***

If you intend to run the simulations on your local system use the following commands. For the following commands, replace *<gmx\_version>* by the prefix compiled for the GROMACS version installed on your system.

We will run the energy minimization first:

* **<gmx\_prefix> grompp -f em.mdp -p topol.top -c ions.gro -o em.tpr**
* **<gmx\_prefix> mdrun -deffnm em**

Then continue with a 2-phase equilibration step:

* **<gmx\_prefix> grompp -f nvt.mdp -c em.gro -r em.gro -p topol.top -o nvt.tpr**
* **<gmx\_prefix> mdrun -deffnm nvt**
* **<gmx\_prefix> grompp -f npt.mdp -p topol.top -c nvt.gro -r nvt.gro -t nvt.cpt -o npt.tpr**
* **<gmx\_prefix> mdrun -deffnm npt**

We will now start the final production run of 100ns:

* **<gmx\_prefix> grompp -f md.mdp -p topol.top -c npt.gro -r npt.gro -t npt.cpt -o topol.tpr**
* **<gmx\_prefix> mdrun -s topol**

**3. Analysis**

Now that we have run the MD simulations, let’s run some analysis on the system in the 100ns trajectory. As we want to extract solution structure(s) from the trajectory that represent the ensemble generated with MD, we will use a set of RMSD and clustering analysis (GROMOS algorithm) tools available in GROMACS.

Check if you are in the directory with all the trajectory files generated from the MD production run. The following command will run the analysis on the trajectories:

* **sbatch analyze-jobs.sh <*output-tag>***

Once the job is completed you can find all the outputs from the above script in a new sub-directory called **MD-analysis**. Have a look at the ***‘.xvg’*** files to check for the structural stability (RMSD) and flexible residue regions (RMSF).

The directory should also include clustering output log and pdb files. The log file summarizes the number of clusters found in the trajectory and extracts a single representative structure for each cluster into the pdb file.

Clusters with reasonable population size (>200 frames) and distribution are considered as the representative solution structures for the selected mutation.

With your choice of visualization program (PyMol, VMD,..), now open the PDB file (*clusters-0.30-<output-suffix>.pdb)* with cluster centres of different populations extracted from the MD-trajectory.