<u>ifold - Interactive protein folding</u>

<u>*Abbreviations*</u>
PDB – Protein Data Bank
DMD – Discrete Molecular Dynamics

These are the simulations tasks supported by ifold –

1. **Protein unfolding / Thermal denaturation –** In unfolding simulations, we start with the native structure of the protein, simulated at low temperature and slowly raise the temperature of the system until the protein starts to unfold.
   Input:
   a. Native structure of the protein: PDB format.
   b. List of desired outputs –
      i. Range of trajectory snapshots desired,
      ii. *Graphs and/or raw (ascii) data for*
         1. radius of gyration vs simulation time,
         2. energy vs. simulation time,
         3. energy vs. temperature.
      iii. Mpeg movie of simulation

   Potential Outputs:
   a. Simulation trajectory (zipped PDB),
   b. Graphs and raw data for
      a. Radius of: gyration vs simulation time,
      b. Energy vs. simulation time,
      c. Energy vs. temperature.
      d. Mpeg movie of simulation,

The DMD simulation parameters include initial temperature Ti, final temperature Tf, heat exchange coefficient C, and total simulation time Tmax. By default Ti=0.4; Tf=1.2, C=0.0001, and Tmax=100000. The advanced user can assign arbitrary values for these parameters.

2. **Thermodynamic scan –** In this mode, we perform constant temperature DMD simulations of the given protein structure over a range of temperatures to get thermodynamic properties of the simulated protein.
   Input:
   a. Native structure of the protein: PDB format.
   b. *(Available only for advanced users)* Structural model of the protein used in simulation
      i. *One bead model* – Each backbone Cα atom of the protein is used to model the coarse-grained protein structure.
      ii. *Two bead model* –Cα and Cβ atoms of each residue are used to model the protein beads for DMD simulation. For glycine residues, the beta carbon bead is absent.

iii. *Four bead model* – In this model, we use coordinates of Cα, C', N (backbone atoms) and Cβ (sidechain atoms) of each residue to model a the protein structure. This increases the detail of simulation at the expense of computation time.

iv. *Quasi-all atom representation* – This representation includes all heavy atoms (upto Cγ) present in the given protein structure to make a higher resolution model of protein structure.

v. *All atom model* – This is the highest resolution model used in DMD – all heavy atoms and some hydrogen atoms forming hydrogen bonds are incorporated to model the protein structure.

c. List of desired outputs –
   i. *Graphs and/or raw (ascii) data for*
      1. Heat capacity (C_v) vs temperature
      2. Mean radius of gyration (R_g),
      3. Energy of the protein (E) vs. temperature.
      4. Frequencies of inter-residue contacts made over the given range of temperature

d. Range of temperature scan (*Optional*, available to advanced users)

Potential Outputs:

a. Graphs and raw data for
   a. Heat capacity (C_v) vs. temperature.
   b. Mean radius of gyration (R_g) vs. temperature.
   c. Energy of the protein (E) vs. temperature.

Since the transition temperature is often in the range of 0.7-0.9. We will set the temperatures: 0.6, 0.7, 0.75, 0.80, 0.85, 0.90, 0.95, 1.0, 1.1, 1.2. For each DMD simulation, we set the temperature as constant Ti=Tf , C=0. 1, and Tmax=100000. The advanced user can assign arbitrary values for these parameters.

3. **Folding using simulated annealing** – In simulated annealing mode, we start from a high temperature conformation (e.g. T = 2.0) where the protein is unfolded (i.e. high energy state of protein) and slowly reduce the system temperature to a very low value (i.e. low energy state of protein), whereby the proein folds to a stable conformation. This process is repeated a certain "nRuns" number of times – suddenly raising the temperature from the last stable conformation and allowing the protein to cool at a very slow rate. Multiple runs ensure that the protein reaches the most stable conformation, which is hypothesized to be the native state.

   Input:

   a. Native structure of the protein: PDB format.

   b. *(Available only for advanced users)* Structural model of the protein used in simulation
      i. *One bead model* – Each backbone Cα atom of the protein is used to model the coarse-grained protein structure.
      ii. *Two bead model* –Cα and Cβ atoms of each residue are used to model the protein beads for DMD simulation. For glycine residues, the beta carbon bead is absent.

iii. *Four bead model* – In this model, we use coordinates of Cα, C', N (backbone atoms) and Cβ (sidechain atoms) of each residue to model a the protein structure. This increases the detail of simulation at the expense of computation time.
iv. *Quasi-all atom representation* – This representation includes all heavy atoms (upto Cγ) present in the given protein structure to make a higher resolution model of protein structure.
v. *All atom model* – This is the highest resolution model used in DMD – all heavy atoms and some hydrogen atoms forming hydrogen bonds are incorporated to model the protein structure.

c. Number of runs for simulated annealing. By default 5, each annealing step t =10000, Th = 2.0, Tl=0.4. and C=0.0005
d. List of desired outputs –
   i. Final folded structure of the protein,
   ii. Trajectory of all simulation runs *(available for advanced users)*
   iii. Graph and raw data of
      1. Energy of protein vs temperature.
      2. Energy of protein vs. number of runs.

Potential Outputs:

a. Final folded structure of the protein
b. Graph and raw data of –
   1. Energy of protein vs temperature.
   2. Energy of protein vs. number of runs.

4. *(Available only for advanced users)* **Custom temperature folding simulation** – In this mode, the user enters the desired initial T_{init} and final temperature T_{final} at which he wants to simulate the folding of his/her protein and starting from a linear conformation of the protein at temperature T_{init}, the temperature of the system is reduced at a user-specified rate α, until the system reaches the final temperature T_{final}.

   Input:

   a. Native structure of the protein: PDB format.
   b. Structural model of the protein used in simulation
      i. *One bead model* – Each backbone Cα atom of the protein is used to model the coarse-grained protein structure.
      ii. *Two bead model* –Cα and Cβ atoms of each residue are used to model the protein beads for DMD simulation. For glycine residues, the beta carbon bead is absent.
      iii. *Four bead model* – In this model, we use coordinates of Cα, C', N (backbone atoms) and Cβ (sidechain atoms) of each residue to model a the protein structure. This increases the detail of simulation at the expense of computation time.

        iv. *Quasi-all atom representation* – This representation includes all heavy atoms (upto Cγ) present in the given protein structure to make a higher resolution model of protein structure.
        v. *All atom model* – This is the highest resolution model used in DMD – all heavy atoms and some hydrogen atoms forming hydrogen bonds are incorporated to model the protein structure.
    c. Initial temperature T_{init}
    d. Final temperature T_{final}
    e. Rate of temperature decrease, α
    f. List of desired outputs
        i. Desired range of trajectory snapshots
        ii. Graph and raw data of –
            1. Energy of protein vs temperature.
            2. Energy of protein vs. simulation time.
            3. Heat capacity (C_v) vs. temperature.
            4. Mean radius of gyration (R_g) vs. temperature.
        iii. RMSD of the final conformation with the native structure.
        iv. Mpeg movie of simulation

Potential Outputs:
    a. Trajectory snapshots
    b. Mpeg movie of simulation
    c. RMSD of the final conformation with the native structure.
    d. Graph and raw data of –
        1. Energy of protein vs temperature.
        2. Energy of protein vs. simulation time.
        3. Heat capacity (C_v) vs. temperature.
        4. Mean radius of gyration (R_g) vs. temperature.

5. *(Available only for advanced users)* **Custom temperature unfolding of protein**
In this mode, the user enters the desired initial T_{init} and final temperature T_{final} at which he wants to simulate unfolding of the given protein. Starting from the given native conformation of the protein at temperature T_{init}, the temperature of the system is increased at a user-specified rate α, until the system reaches the final temperature T_{final}.
Input:
    a. Native structure of the protein: PDB format.
    b. Structural model of the protein used in simulation
        i. *One bead model* – Each backbone Cα atom of the protein is used to model the coarse-grained protein structure.
        ii. *Two bead model* –Cα and Cβ atoms of each residue are used to model the protein beads for DMD simulation. For glycine residues, the beta carbon bead is absent.
        iii. *Four bead model* – In this model, we use coordinates of Cα, C', N (backbone atoms) and Cβ (sidechain atoms) of each residue to

model a the protein structure. This increases the detail of simulation at the expense of computation time.

iv. *Quasi-all atom representation* – This representation includes all heavy atoms (upto Cγ) present in the given protein structure to make a higher resolution model of protein structure.

v. *All atom model* – This is the highest resolution model used in DMD – all heavy atoms and some hydrogen atoms forming hydrogen bonds are incorporated to model the protein structure.

c. Initial temperature T_{init}

d. Final temperature T_{final}

e. Rate of temperature increase, α

f. List of desired outputs

   i. Desired range of trajectory snapshots

   ii. Graph and raw data of –

      1. Energy of protein vs temperature.
      2. Energy of protein vs. simulation time.
      3. Heat capacity (C_v) vs. temperature.
      4. Mean radius of gyration (R_g) vs. temperature.

   iii. Mpeg movie of simulation

Potential Outputs:

e. Trajectory snapshots

f. Mpeg movie of simulation

g. Graph and raw data of –

   5. Energy of protein vs temperature.
   6. Energy of protein vs. simulation time.
   7. Heat capacity (C_v) vs. temperature.
   8. Mean radius of gyration (R_g) vs. temperature.

6. **p_{fold} scan** – p_{fold} refers to the probability that a given *conformation* of the protein will fold before unfolding. It is a quantitative measure of progress in protein folding of the given conformation.

   Input:

   a. The native structure of the protein in PDB format.

   b. Structural model of the protein used in simulation

      i. *One bead model* – Each backbone Cα atom of the protein is used to model the coarse-grained protein structure.

      ii. *Two bead model* –Cα and Cβ atoms of each residue are used to model the protein beads for DMD simulation. For glycine residues, the beta carbon bead is absent.

      iii. *Four bead model* – In this model, we use coordinates of Cα, C’, N (backbone atoms) and Cβ (sidechain atoms) of each residue to model a the protein structure. This increases the detail of simulation at the expense of computation time.

   iv. *Quasi-all atom representation* – This representation includes all heavy atoms (upto Cγ) present in the given protein structure to make a higher resolution model of protein structure.
   v. *All atom model* – This is the highest resolution model used in DMD – all heavy atoms and some hydrogen atoms forming hydrogen bonds are incorporated to model the protein structure.

c. The structure of the decoy.

Potential Outputs: the folding probability value p_{fold} for the given conformation.

The software development process for the ifold server may be divided into three stages:

Stage I

- Designing client-side and server-side java programs which communicate with each other (using java.net api) and with the MySQL database.

- The server side application formats input parameters to the prescribed DMD readable input file and stores the simulation type/options in the database. It then schedules the jobs onto one of the available set of compute nodes and sends the input data (protein databank file) and instructions (a txt file readable by DMD) to the compute node. A unique key is assigned to each job in the database and the processing instructions are all associated with this unique key.

- The client side application runs the daemon listening for jobs sent by the ifold master server. The daemon fetches the input files and PDB structure and runs the DMD simulation. Upon completion of the simulation job, it notifies the ifold (master) server. The ifold server then queries MySQL database for analyses to be performed on this set of simulations and schedules the analysis back on the compute node. Once the analysis is complete, the output is transferred from compute node to the ifold server and the user is notified about the results via email. The user receives a text summary of results (listing operations successful /failed) and having a URL for the completed job. A total time of ***N*** hours is allotted for each user to download the simulation results before they are deleted from the disk (by using a cron job deleting all files older than 48 hours in ifold results directory).

Stage II

Improved user interface for output formats –

- Graphs for the output of simulations.
   a. Average energy vs. temperature plot (using gnuplot)
   b. Energy variation with temperature
- Simulation trajectories available for download in compressed (gzip/bzip2) protein databank format.
- Visualization – documentation of how simulation can be visualized using VMD.

Stage III

- Developing easily readable statistics of ifold usage
    a. Statistics for administrator.
        i. How many regular users a
    b. Statistics for ifold users.
- Support for custom "start scripts" for performing commonly used tasks.

These are the user types for ifold –

1. ifold web administrator – The administrator controls privileges for other users (both *regular* and *advanced*) from the web and is able to see the statistics of simulations which are currently running or have previously run on the server.

2. Regular users – this class of users are allowed to submit a maximum of "*n*" jobs (*n* configurable by the ifold web administrator). The privileges of this class of user are restricted, i.e. they have access to only a limited set of features from the ifold web-server (see below), and are allowed only a fixed quota of CPU-time (*t* hours per week) for ifold simulations.

3. Advanced users – this class of users are allocated a maximum of "*m*" jobs (*m*>*n*, but again, configurable by the ifold web administrator). Advanced users are allowed to access all the features available on the ifold web-server. The quota of cpu-time for each simulation submitted by advanced users is unrestricted.

In addition to the login/job compleition notification information, detailed properties of each job submitted by each user are also logged in the MySQL database. This is intended to be used for generating statistics of usage and for vigilance by ifold web administrator.

iFold input parameters

Recall that for regular users many input parameters are pre-specified, while advanced users have the option for giving custom input parameters for their simulations. In what follows, we describe the default set of parameters (for regular users) for each simulation type and also the validations to be made before a certain simulation type is scheduled.

Notice that the runDMD.pl script is essentially a wrapper script for calling the DMD simulation engine. Therefore, when initiated with appropriate parameters, this script is used for folding as well as unfolding simulations.

The idea for organizing simulations is the following - for advanced users, the Ajax engine running at the client parses the input parameters right when they are submitted and validates if the parameters are appropriate for the chosen simulation. If there is a problem, the Ajax script rejects the job, notifying the user about all the associated problems, otherwise if all the required input parameters are obtained, these parameters are passed to the java engine, the simulation data is stored in the MySQL database and the request for simulation is sent to ifold scheduler.

For extensibility and easy manageability of ifold's code, we strongly recommend that the default input parameters be stored as representative constants, whose numerical values are stored in a table in the database - e.g. $UNFOLD_SIMTIME for default unfolding simulation time for regular users. And the administrator can tweak these values by changing $UNFOLD_SIMTIME variable in the "Administrator's Control Panel" page. I have suggested the names for input parameters in the following discussion.

Important Note: For "Native Structure of the Protein", please give two options for specifying the structure file:

(1) An upload menu from where the user can upload a PDB file present on his/her disk.

(2) By specifying the 4 letter Protein DataBank (PDB) code (e.g. 1kx5) assigned to the protein. Use the script fetchpdb.pl to fetch the gzipped structure file from the PDB ftp server and start the simulation based on this structure.

1. Protein unfolding / Thermal denaturation simulation

Regular Users

Input parameters:

Native Structure of the Protein (PDB Format):

(A) File uploaded by user, or

(B) Four letter code of the PDB file.

Initial Temperature	= $UNFOLD_TEMPI = 0.4
Final Temperature	= $UNFOLD_TEMPF = 1.2
Heat Exchange Coefficient	= $UNFOLD_HEATX = 0.0001
Total simulation time	= $UNFOLD_SIMTIME = 100000

Advanced Users

Input parameters:

Native Structure of the Protein (PDB Format):
(A) File uploaded by user, or
(B) Four letter code of the PDB file.

Initial Temperature	= $UNFOLD_TEMPI
Final Temperature	= $UNFOLD_TEMPF
Heat Exchange Coefficient	= $UNFOLD_HEATX
Total simulation time	= $UNFOLD_SIMTIME

Validations to be made before calling runDMD.pl :

1. Both initial temperature and final temperature must be floating point numerals. Note that these are not temperatures values in celsius/fahrenhiet/kelvin scale. Rather, this is a relative scale with values close to T=~0.0 (T>0) representing very low temperatures, T=0.8 being close to the unfolding transition temperature for protein and higher temperature representing unfold.

2. Initial temperature < Final temperature. Because its an unfolding simulation. You may hint the user for "folding simulation" if he/she wishes to go for Initial temperature > Final temperature.

3. Heat exchange coefficient must be less than $UNFOLD_HEATXMAX=0.005 and greater than $UNFOLD_HEATXMIN=0.0. Higher heat exchange coefficient represent unphysical levels of heat exchange undergoing in the protein.

4. Simulation time - The simulation time is specified in number of DMD time steps for which the simulation is carried out. If we have small simulation times, the system is not sufficiently equilibrated, thus the simulations carry no scientific meaning. However large time units for simulations are a big resource hog! So, the simulation time should lie between $UNFOLD_TMIN=1000 and $UNFOLD_TMAX=100000.

5. "Range of Trajectory Snapshots:" should be an integer ranging between $UNFOLD_TRAJMIN=0 to $UNFOLD_TRAJMAX=$UNFOLD_SIMTIME.

Note - Please replace option for "Mpeg movie of simulation?" by "Entire trajectory of simulation (zipped PDB)"

6. Validation for structure -
If PDB code is chosen, the code must start with a numeral. The file must be downloadable from the PDB ftp server. If the file uploaded by user, it may be either stored in the database or kept in a repository of all pdb files. In all cases, use validatePDB.pl to validate the protein structure, i.e. if a DMD model can be generated from the pdb file.

2. Thermodynamic scan

Constant temperature DMD simulations of the given protein structure over a range of temperatures to get thermodynamic properties of the simulated protein.

Regular Users

Native structure of the protein (PDB Format):
(A) File uploaded by user or
(B) Four letter code of the PDB file.

DMD model of the protein used in thermodynamic scan: $TSCAN_DMDMODEL = 2

Default temperatures at which simulation is done:
$TSCAN_TEMP1 = 0.6
$TSCAN_TEMP2 = 0.7
$TSCAN_TEMP3 = 0.8
$TSCAN_TEMP4 = 0.9
$TSCAN_TEMP5 = 1.0
$TSCAN_TEMP6 = 1.1

Heat Exchange Coefficient = $TSCAN_HEATX = 0.0001
Total simulation time = $TSCAN_SIMTIME = 50000

Note that the initial and final temperatures are the same for *Thermodynamic Scan.* So, we perform six simulations at the above constant temperatures.

Advanced Users

DMD model of the protein - $TSCAN_DMDMODEL = 1/2/4/Q/A. This represents the number of beads used in DMD structural model of the protein. 1,2,4,Q,A representing one-bead, two-bead, four-bead, quasi-all atom and all atom models respectively.

Six temperature points for simulation - $TSCAN_TEMP1, $TSCAN_TEMP2, $TSCAN_TEMP3, $TSCAN_TEMP4, $TSCAN_TEMP5, $TSCAN_TEMP6 are specified by the user. Each of these must be a floating-point numeral with values between 0.4 and 1.4.

3. Folding simulation using simulated annealing

Regular Users

DMD model of the protein simulated, $SIMA_DMDMODEL=2

Native structure of the protein (PDB Format):
(A) File uploaded by user or
(B) Four letter code of the PDB file.

Initial Temperature	= $SIMA_TEMPI = 1.0
Final Temperature	= $SIMA_TEMPF = 0.3
Heat Exchange Coefficient	= $SIMA_HEATX = 0.0001
Total simulation time	= $SIMA_SIMTIME = 10000

Number of simulated annealing runs, $SIMA_NUMRUNS = 5

Advanced Users

Validations to be made before calling runDMD.pl :

1. Both initial temperature and final temperature must be floating point numerals.

2. Initial temperature (SIMA_TEMPI) must be greater than final temperature (SIMA_TEMPF), since this is a folding simulation. You may hint the user for "unfolding simulation" if he/she wishes to go for Initial temperature < Final temperature.

3. Heat exchange coefficient must be less than $SIMA_HEATXMAX=0.005 and greater than $SIMA_HEATXMIN=0.0. Higher heat exchange coefficients represent unphysical levels of heat exchange undergoing in the protein.

4. Simulation time - The simulation time is specified in number of DMD time steps for which the simulation is carried out. So, the simulation time should lie between $SIMA_TMIN=1000 and $SIMA_TMAX=100000.

5. Simulation time spent at each annealing step $SIMA_SIMTIME must lie between $SIMA_MINSIMTIME = 2000 and $SIMA_MAXSIMTIME = 20000.

6. Number of simulated annealing runs, $SIMA_NUMRUNS must lie between $SIMA_MINNUMRUNS=1 and $SIMA_MAXNUMRUNS=50

7. Validation for structure -
If PDB code is chosen, the code must start with a numeral. The file must be downloadable from the PDB ftp server. If the file uploaded by user, it may be either stored in the database or kept in a repository of all pdb files. In all cases, use validatePDB.pl to validate the protein structure, i.e. if a DMD model can be generated from the pdb file.

8. DMD model of the protein ($SIMA_DMDMODEL) must be one of these: 1 (1-bead), 2 (2-bead), 4 (4-bead), (Q) quazi all-atom, or A, (all-atom).

4. Custom temperature folding simulation

Advanced Users Only

Input parameters:

Native structure of the protein (PDB Format):
(A) File uploaded by user or
(B) Four letter code of the PDB file.

Initial Temperature	= $CUSTFOLD_TEMPI
Final Temperature	= $CUSTFOLD_TEMPF
Heat Exchange Coefficient	= $CUSTFOLD_HEATX
Total simulation time	= $CUSTFOLD_SIMTIME

Validations to be made before calling runDMD.pl:

1. Both initial temperature and final temperature must be floating point numerals.

2. Initial temperature > Final temperature, since this is a folding simulation. You may hint the user for "unfolding simulation" if he/she wishes to go for Initial temperature < Final temperature.

3. Heat exchange coefficient must be less than $CUSTFOLD_HEATXMAX=0.005 and greater than $CUSTFOLD_HEATXMIN=0.0. Higher heat exchange coefficient represent unphysical levels of heat exchange undergoing in the protein.

4. Simulation time - The simulation time is specified in number of DMD time steps for which the simulation is carried out. The simulation time should lie between $CUSTFOLD_TMIN=1000 and $CUSTFOLD_TMAX=100000.

5. "Range of Trajectory Snapshots:" should be an integer ranging between $CUSTFOLD_TRAJMIN=0 to $CUSTFOLD_TRAJMAX=$CUSTFOLD_SIMTIME.

6. Validation for structure -
If PDB code is chosen, the code must start with a numeral. The file must be downloadable from the PDB ftp server. If the file uploaded by user, it may be either stored in the database or kept in a repository of all pdb files. In all cases, use validatePDB.pl to validate the protein structure, i.e. if a DMD model can be generated from the pdb file.

7. DMD model of the protein ($CUSTFOLD_DMDMODEL) must be one of these: 1 (1-bead), 2 (2-bead), 4 (4-bead), (Q) quazi all-atom, or A, (all-atom).

5. Custom temperature unfolding simulation

Advanced Users Only

Input parameters:

Native structure of the protein (PDB Format):
(A) File uploaded by user or
(B) Four letter code of the PDB file.

Initial Temperature	= $CUSTUNFOLD_TEMPI = 0.4	
Final Temperature	= $CUSTUNFOLD_TEMPF = 1.0	
Heat Exchange Coefficient	= $CUSTUNFOLD_HEATX = 0.0001	
Total simulation time	= $CUSTUNFOLD_SIMTIME = 100000	

Validations to be made before calling runDMD.pl :

1. Both initial temperature and final temperature must be floating point numerals.

2. Initial temperature > Final temperature, since this is a folding simulation. You may hint the user for "unfolding simulation" if he/she wishes to go for Initial temperature < Final temperature.

3. Heat exchange coefficient must be less than $CUSTUNFOLD_HEATXMAX=0.005 and greater than $CUSTUNFOLD_HEATXMIN=0.0. Higher heat exchange coefficient represent unphysical levels of heat exchange undergoing in the protein.

4. Simulation time - The simulation time is specified in number of DMD time steps for which the simulation is carried out. IF we have small simulation times, the system is not sufficiently equilibrated, thus the simulations carry no scientific meaning. However large

time units for simulations are a big resource hog! So, the simulation time should lie between $CUSTUNFOLD_TMIN=1000 and $CUSTUNFOLD_TMAX=100000.

5. "Range of Trajectory Snapshots:" should be an integer ranging between $CUSTUNFOLD_TRAJMIN=0 to $CUSTUNFOLD_TRAJMAX=$CUSTUNFOLD_SIMTIME.

6. Validation for structure -
If PDB code is chosen, the code must start with a numeral. The file must be downloadable from the PDB ftp server. If the file uploaded by user, it may be either stored in the database or kept in a repository of all pdb files. In all cases, use validatePDB.pl to validate the protein structure, i.e. if a DMD model can be generated from the pdb file.

7. DMD model of the protein ($CUSTUNFOLD_DMDMODEL) must be one of these: 1 (1-bead), 2 (2-bead), 4 (4-bead), (Q) quazi all-atom, or A, (all-atom).

5. p_{fold} scan

<u>*Advanced Users Only*</u>

Input parameters:

Structure of the decoy (PDB Format): File uploaded by user

Native structure of the protein (PDB Format):
(A) File uploaded by user or
(B) Four letter code of the PDB file.

Structural model of the protein: $PFOLD_DMDMODEL.

Validations to be made before calling runDMD.pl :

1. Validation for structure -
If PDB code is chosen, the code must start with a numeral. The file must be downloadable from the PDB ftp server. If the file uploaded by user, it may be either stored in the database or kept in a repository of all pdb files. In all cases, use validatePDB.pl to validate the protein structure, i.e. if a DMD model can be generated from the pdb file.

2. DMD model of the protein ($PFOLD_DMDMODEL) must be one of these: 1 (1-bead), 2 (2-bead), 4 (4-bead), (Q) quazi all-atom, or A, (all-atom).