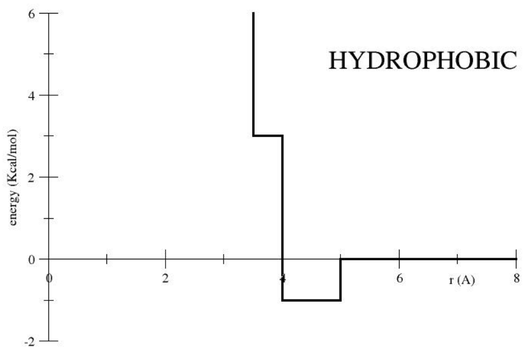
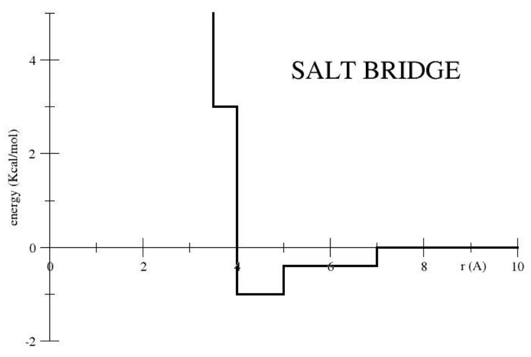
FLEXIBLE DOCKING WITH DISCRETE

DISCRETE and the DMD method

DISCRETE is a package devised to simulate the dynamics of proteins. The simulation code generates trajectories for the atoms of the protein, writing the structure in pdb format with a frequency specified by the user. The method used to perform the simulations is Discrete Molecular Dynamics (DMD). The difference respect to a standard molecular dynamics simulation is that the particles are considered to move with constant velocity until a collision occurs. Upon each collision there is a transfer of linear momentum between the colliding particles, where total momentum and energy are conserved. As the particles move in ballistic regime, the equations of motion do not have to be integrated. This leads to a drastic saving in computing time. In DMD the interaction potentials are stepwise potentials, and the events (collisions) occur when the distance between two particles corresponds to a step in its interaction potential. DMD is an event-driven method, so the timestep is not predefined like in a standard molecular dynamics simulation, but it is the time between consecutive collisions. The frequency of the collisions in the system increases linearly with the number of particles, and with the number of steps in the interaction potential. For more details on DMD potentials see the Appendix.

DISCRETE uses a united-atom representation of the protein, so it does not consider the hydrogens. Solvation effects are considered in an implicit way using the Lazaridis-Karplus (LK) solvation potential [1], devised for a united-atom representation of the proteins. The potential associated to the hydrophobicity of the atom is represented with a square well potential whose depth/height is the minimum/maximum value of the LK potential. The Van der Waals interaction is included by adding the value of the VdW potential at the distance corresponding to the minimum. Since the solvation potential is approximated with a simple square well, these values are modified with a multiplying factor, whose value has been adjusted to fit the flexibility pattern obtained with explicit-solvent all atom molecular dynamics for an ensemble of representative proteins [2]. This simplicity in the representation of the potentials leads to a decrease in accuracy, but an increase in speed, since the method gets slower as the resolution (the number of steps) is increased. Coulomb interactions are considered between some atoms of acceptor/donor residues that are assigned a charge in the model. To take into account the long range nature of the electrostatic interaction, a second step in the potential is defined at a longer distance (see figure below). In our model, with the aim of simplicity, the two types of interaction are considered to exclude each other: if two particles are charged, they are considered to interact via the Coulomb potential (forming a salt bridge or repeling each other), otherwise via the LK potential. Both types of potential have an infinite barrier at very short distances to prevent steric clashes.





Due to the simplicity in the DMD potentials, the secondary structure should be constrained, otherwise it would be lost. The hydrogen bonds of the starting structure are conserved during the simulation. The program identifies the hydrogen bonds based on geometric criteria, considering the distances between the backbone atoms involved in the hydrogen bond (see the Appendix)

With the secondary structure conservation and using the values for the potential interaction parameters that are defined by default in the program, the RMSD respect to the experimental structure is, on average, 50% higher that RMSD obtained with an all-atom explicit solvent MD method [2]. This deviation respect to the real minimum in the free energy landscape is due to the simplification in the potentials, but this is still a very small deviation in exchange for a remarkable increase in speed. The program can be also used to sample the conformational transitions of a partially unfolded protein, as long as the starting structure conserves the secondary structure of the native conformation.

Multiscale simulations

The new release of DISCRETE is able to simulate the dynamics of a structure (constituted by several proteins) with different levels of resolution in different regions of the structure. This allows to remove particles from the regions of the structure that do not provide useful information and therefore to speed up the simulations.

The setup program is able to recognize the interface between two proteins and remap the structure into a structure with atomistic resolution at the interface and residue-level resolution elsewhere, writing only the C_α . This treatment of the problem is useful for protein-protein docking predictions, where the relevant magnitude is the interaction energy between the two proteins (receptor and ligand). The setup program allows both receptor and ligand to be constituted by more than one protein.

We have provided in this package 100 different ligand positions, that together with the receptor constitute 100 docking conformations such that the complete space of relative positions is covered.

Installation

Two programs have to be compiled, DMDSetup and discrete

To compile DMDSetup, go to the src/setup directory

To compile with gfortran:

```
make -f Makefile.gfortran
```

To compile with ifort

```
make -f Makefile.ifort
```

To compile discrete, go to src/discrete and proceed equally.

Input and output files

To open an interactive session, type

```
mnsh
```

discrete, the program that performs the simulation, needs as inputs a coordinate and a topology file. The coordinate file provides the position of each particle to start the simulation, and the topology file provides all the other data relevant for the simulation, like position restraints due to covalent bonding, bond angle restraints and dihedral restraints, interaction potential parameters etc. These files are generated with the program **DMDSetup**

As an example simulation we will show here how to generate the DMD trajectory for the first docking pose.

```
exe/DMDSetup -i test/parmsetup.dat -rlib dat/dmd_reslibrary.dat -pot dat/dmd_potentials.dat
-pdbin test/receptor.pdb -ligpdbin test/ligand/ligand1.pdb
```

Input files:

-i: *parmsetup.dat* is a parameter file for **DMDSetup**

```
 &SETUP
  TIPCALC=1
  IRIG=0
  DINT0=8
  DINT1=12
 &END
```

TIPCALC=1 tells DMDSetup to prepare the topology and coordinates for a docking simulation (therefore make multiscale representation of the system). If TIPCALC=0, the normal simulation with atomistic resolution in the whole structure is made.

If IRIG=1, the C_α positions are restrained in all the structure, therefore only the sidechains are allowed to move.

DINT0 is the width of the interface. Around the interface we have defined another layer of atoms whose positions are restrained, whose thickness is DINT1. The role of this outer atomistic resolution region is to produce a physical environment around the interface equivalent to what would exist if a full atomistic representation over all the structure was used. The positions of the particles far from the interface (only C_α) are fixed with spacial restraints.

-pdbin: the PDB file of the receptor.

-ligpdbin: the PDB file of the ligand.

-rlib: is the residue library where each atom is assigned an atom type that will be considered to define the interaction potentials, and where the bonded interactions between atoms within one residue are defined.

-pot: interaction potential parameters for each atom type

Output files (input files for **discrete**):

-top: topology file (default *topology*)

-r: coordinate file (default *coordinates*)

Once generated the coordinate and topology files, you can run the simulation:

```
exe/discrete -i test/dmdparm.dat
```

dmdparm.dat is the parameter file for the simulation:

```
 &INPUT
 ,TSNAP=10000
 ,NBLOC=200
 ,TEMP=300
 ,SEED=2381
 &END
```

TSNAP is the period between each time the program writes the structure, i.e. the length of each block in fs.

NBLOC is the number blocks, i.e. the number of times it writes the structure. The parameters of this file correspond therefore to a 1 ns long simulation.

TEMP is the temperature of the system, in K.

SEED is the random seed for the simulation.

The output files of **discrete** are:

-ener: energy (default *energy.dat*)

-traj: trajectory file in pdb format (snapshots), that is continously updated during the simulation (default *trajectory.pdb*)

-rst: coordinates at the end of the simulation, in the same format as the coordinate file generated by DMDSetup, so it can be used as starting point for another simulation. (default *restart.crd*)

Analyzing the DMD trajectories

You can visualize the trajectory with VMD, PYMOL...

The analysis tools are in the *analysis* directory

You should use the program **interface**, that you have to compile this with gfortran. Use the script *compile.bat*

```
cd analysis
./compile.bat
```

The input file for the program interface is *read.dat*:

```
 &INPUT
 ,FILE20='../trajectory.pdb'
 ,FILE16='atomtypes.dat'
 ,FILE17='potentials.dat'
 ,TSNAP=10
 ,NBLOC=200
 ,NRES1=98
 ,NATOM=1261
 &END
```

NATOM is the number of atoms of the structure in the trajectory, and NRES1 the number of residues of the receptor. You can see it in the *trajectory.pdb* file where the molecules are labeled (A is the receptor, B is the ligand). In this way the program knows where is the receptor and the ligand, and recognizes the interface (its placement is different for each docking conformation). NBLOC is the number of snapshots that you want to analyze (starting from the beginning of the trajectory). The time TSNAP is now in ps

Run the program from the directory where you have the trajectory:

```
interface < read.dat > energies.dat
```

interface computes the interaction energy terms with DMD stepwise potentials, like DISCRETE does. Therefore all the interactions have a cutoff distance, leading to a sparse energy matrix (see below).

The standard output that we have redirected to *energies.dat* is

```
 #VdW,SOLV,COUL  -35.905174714500909        62.402842952512849       0.10323998902142112
   10.000000000000000       -31.952872650975504        44.231662163552194    -0.590160010001
   20.000000000000000       -31.355674769701476        42.314264528214046    -0.590760007014
...
```

The program has calculated three terms of the interaction energy: Van der Waals, desolvation and Coulomb. The first row are the values at t=0 (rigid docking pose).

The total energy is the sum ot the three terms. You can extract it doing

```
cat energies.dat | awk '{print $1,$2+$3+$4}' > energy.dat
```

You can visualize the evolution of the interaction energy between receptor and ligand during the simulation doing

```
xmgrace energy.dat
```

interface also produces the file *matrix.dat*, that shows the interaction energy between all the residues of the receptor and all the residues of the ligand. The total energy is the sum of all the matrix elements. *matrix.dat* looks like this:

```
           1           1   0.0000000000000000        0.0000000000000000
           1           2   0.0000000000000000        0.0000000000000000
           1           3   0.0000000000000000        0.0000000000000000
           1           4   0.0000000000000000        0.0000000000000000
           1           5   0.0000000000000000        0.0000000000000000
           1           6   0.0000000000000000        0.0000000000000000
           1           7   0.0000000000000000        0.0000000000000000
           1           8   0.0000000000000000        0.0000000000000000
...
```

The majority of the matrix elements are 0, since only a few pairs of residues interact. The first two columns are the sequence number of the residues in receptor and ligand, the third column is the interaction energy before the simulation (rigid docking pose) and the fourth after the simulation (structure relaxed with DMD)

To visualize matrix.dat as a 2D map, use **gnuplot**, with the commands:

```
gnuplot> set pm3d map
gnuplot> splot 'matrix.dat' u 1:2:3
```

With this you will see the interactions before in the rigid docking poses. If you want to see the interacions after the DMD relaxation, type:

```
gnuplot> splot 'matrix.dat' u 1:2:4
```

And now you can see how it changed.

To leave gnuplot:

```
gnuplot> quit
```

Examples of the interaction energy 2D maps can be found in

`http://mmb.pcb.ub.es/~agusti/scalalife/CECAM/`

APPENDIX

In discrete molecular dynamics (DMD) the particles are considered as hard spheres interacting through discontinous potentials, therefore in ballistic regime (moving at constant velocity) until an event (collision) occurs. No forces have to be calculated, and it is not necessary to integrate the equations of motion, speeding up the calculation compared with conventional molecular dynamics. Hardcore potentials preventing steric clashes are defined between unbound particles and infinite square wells are defined between bound particles to keep bond distances, bond angles and torsional angles (bonds and pseudobonds).

The system evolves from event to event, and the velocity of the particles not involved in the collision is unaltered. Thus in DMD the timestep is the time between consecutive events, and most of the calculation time is spent in finding out which will be the next pair of particles to collide.

$$\vec{r}_i(t+t_c) = \vec{r}_i(t) + \vec{v}_i(t)t_c$$

t_c=min(t_{ij}) $\qquad t_{ij} = \frac{-\vec{r}_{ij}\cdot\vec{v}_{ij}\pm\sqrt{(\vec{r}_{ij}\cdot\vec{v}_{ij})^2-v_{ij}^2(r_{ij}^2-d^2)}}{v_{ij}^2}$

being d the sum of the radii of particles i and j.

Conservation of momentum and energy is imposed at each event

$$m_i\vec{v}_i + m_j\vec{v}_j = m_i\vec{v}_i' + m_j\vec{v}_j'$$

$$\frac{1}{2}m_iv_i^2 + \frac{1}{2}m_jv_j^2 = \frac{1}{2}m_iv_i'^2 + \frac{1}{2}m_jv_j'^2 + \Delta V$$

The transferred momentum upon the collision

$$m_i\vec{v}_i = m_i\vec{v}_i' + \Delta\vec{p}$$

$$m_j\vec{v}_j + \Delta\vec{p} = m_j\vec{v}_j'$$

depends on the change in the potential energy. The particles can overcome the potential step as long as

$$\Delta V < \frac{m_im_j}{2(m_i+m_j)}(\vec{v}_i - \vec{v}_j)^2$$

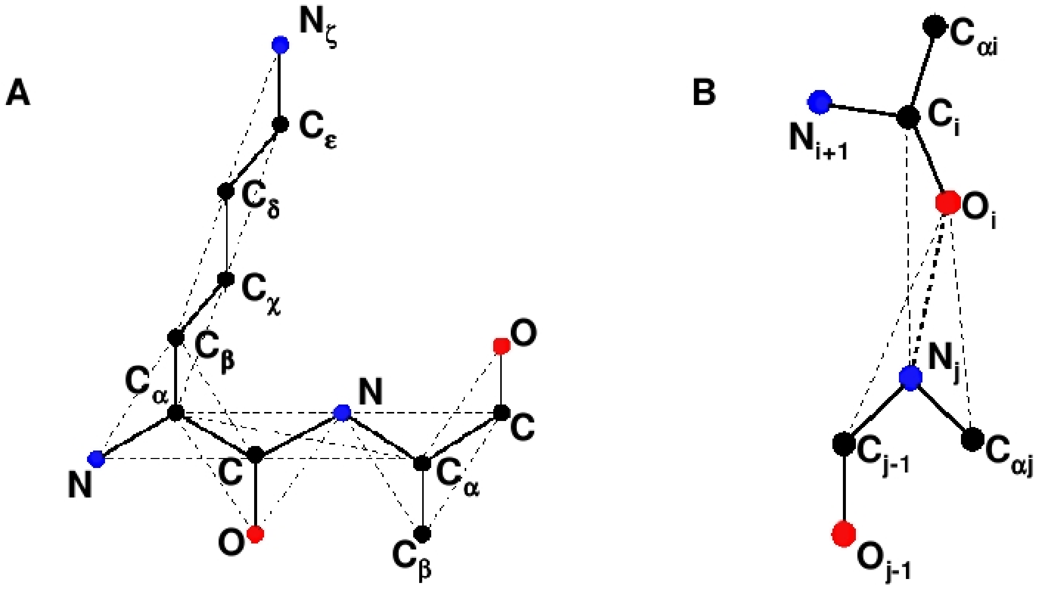
then

$$\Delta p = \frac{m_im_j}{m_i+m_j}\left[\sqrt{(\vec{v}_i - \vec{v}_j)^2 - 2\frac{m_i+m_j}{m_im_j}\Delta V} - v_j + v_i\right]$$

otherwise the particles keep bound and the transferred momentum is

$$\Delta\vec{p} = \frac{2m_im_j}{m_i+m_j}(\vec{v}_i - \vec{v}_j)$$

We defined a simple united atom force-field consisting of bonded and non-bonded terms. As usual bonded terms account for stretchings (a-b), bendings (a-b-c) and torsions (a-b-c-d). Stretching, bendings and torsions involving double or conjugated bonds were represented by means of infinite square wells with a width corresponding to 1% the length of the bond/pseudobond distance (a-b bond for stretching, a-c pseudobond for bending and a-d pseudobond for torsions). See figure below (pannel A):



Backbone hydrogen bonds (pannel B in the figure above) where defined between amide nitrogens and carbonyl oxygens such that their distance was $2.5 < d < 4.1$ Å, provided the dipole-dipole hydrogen bond geometry is the proper one. In order to keep this geometry, pseudobonds were defined between N_i and C_j, C_{i-1} and O_j, and $C_{\alpha i}$ and O_j.

REFERENCES

[1] Lazaridis T., Karplus M. *Proteins Struct. Funct. Genet.***35**, 133-152 (1999)

[2] Emperador A., Meyer T., Orozco M. *Proteins Struct. Funct. Genet.***78** 83 (2010)

[3] Humphrey, W., Dalke, A. and Schulten, K., 'VMD - Visual Molecular Dynamics', J. Molec. Graphics 1996, 14.1, 33-38.