Molecular Simulations

Universitat Pompeu Fabra

Project 1

Inés Sentís
Mariona Torrens

April 29, 2016

1 Introduction

We have created a python module that, given a transmembrane protein of PDB, adds a membrane bilayer, represented as two leaflets of dummy atoms, and orientates the protein according to it.
The goal of the project has been to get used to HTMD and VMD, as well as NGL, while making an approximation of the results that OPM authors report in: `http://opm.phar.umich.edu/`
The program can be executed typing:
$python3 membraneprot.py -i [PDB ID of the protein]

2 Methods

2.1 Membrane creation and integration of the protein in the membrane

To begin with, we created two planes of dummy atoms, which represent the two layers of the cellular membrane. These two layers can be differentiated thanks to the fact that one is made of atoms of O and the other of atoms of N. To obtain each layer, we built the function *dummy_leaflet()*. First, it creates an empty molecule using the *empty(N*N)* method, where N is the maximum distance of the proteins atoms from the center. We used this N to make sure that the leaflet is be bigger than the protein. Once we have the empty molecule, we set the characteristics of its atoms:

- The name of the atoms as O or N
- The atom type as heteroatom (HETATM)
- the residue IDs
- The coordinates of the atoms, considering that all atoms of a dummy leaflet are at the xy plane, separated by a distance of 3Å.

When the membrane was creates, we centered the protein and the membrane and created a new molecule object with both of them.

2.2 Obtaining the protein vector

As it was suggested in the assignment file, we have taken advantage of the fact that some aromatic rings of the aromatic amino acids play the role of anchors to the membrane, so they are placed parallel to it. Therefore, the orientation of the rings was supposed to help in finding the orientation of the protein. We decided to obtain the general orientation of the protein by calculating a vector for which each of the 3D components (x,y,z) were the mean of the ones of the vectors orthogonal to the plane formed by the atoms of each aromatic ring of the protein. In other words, to obtain a plane it is required to have two vectors; those were calculated as the distance between the CG carbon of the ring and the two consecutive atoms of the polygon. Then, we computed the scalar product of these vector to obtain the orthogonal vector to the plane. Once we had all the orthogonal vectors we computed the mean of each component x,y and z to get the vector of the protein (a general vector which approximated the orientation of the molecule). This is done by the function emphprot_vector.

2.3 Rotating the protein

The previously obtained vector, which represents the average vector perpendicular to the aromatic atoms of the protein, needs to be perpendicular to the membrane, which is placed in the xy plane. This way, the aromatic rings will end up, on average, parallel to the membrane. Therefore, we need to rotate the protein until its vector reaches $[0, 0, z]$. We can do this applying two rotations: one around the y axis and one around the z. With this purpose, we made a projection of the vector in the planes xy and xz and, then, we calculated two angles:

1. The angle α between the projection of the protein vector at the xy plane and the x axis $(1, 0)$.

2. The angle β between the projection of the proten vector at the xz plane and the z axis $(0, 1)$.

The angles were calculated as: $cos\Theta = \frac{u*v}{||u||*||v||}$
With the module *rotationmatrix* we built two matrices, which we later multiplied to rotate the protein with the method °emphrotationBy. These two matrices correspond to a rotation of angle β around the y axis and a rotation of angle α around the z axis. The calculation explaned above are performed by the function *rotate*.

2.4 Position of the membrane

The position of the membrane, computed by the function *hydroph_calculation*, was done based on hydrophobicity. In other words, we "cut" the protein along the z axis as if it were been a "cake", so we computed the hydrophobicity for each "slice". The length of the slice represents the window size of the angstroms in the z axis analyzed at each iteration. We knew which residues belonged to each portion of "cake" because we first got their coordinates, and only those with a z component between the values of the current window were kept to sum their hydrophobicity.

After that, we plotted the hydrophobicity of the protein along the z axis (at the plot, the x axis corresponds to the window of z values and the y axis the hydrophobicity value). Theoretically, the maximum hydrophobicity should be reached at the part of the protein inserted on the membrane, and decrease where the protein is anchored to the membrane.

Since the layers of the cellular membrane are separated by at least 20 Å, we considered that the two layers wouldn't be at least at a distance of 12 Å from the z point of maximum hydrophobic. We took 12 Å instead of 20 in order to leave some margin. We focused only on the extremes left after going from the maximum point to 6 angstroms to the right and to the left. That was set in the function *extreme_list*, where we determined a cutoff (3 steps x 2Å = 6Å) and stored the extremes in two arrays, one for the right side and the other for the left side. We used two types of lists: one that kept the hydrophibicity scores and one that kept the steps or intervals of z when we had "cut" the protein (the intervals of the x axis of the plot). Each interval of z corresponds to an hidrophobicity value.

The function *find_z_position* finds the local/relative minimums of the plot of hydrophobicity, which are candidate points to place the leaflet for each of the side/extreme lists (right/left). We considered the most probable candidate as the local minimum nearest to the cutoff (6Å from the maximum) for each side list.

Finally, we set the new coordinates for each leaflet as the the z coordinate of the most probable points with the *set_new_membrane_coords* function.

3 Results

We tested the algorithm described above with different types of proteins, ranging from β barrels and α helix, and from small proteins to big ones. Here we present some of the examples:

(a) ***(b)***

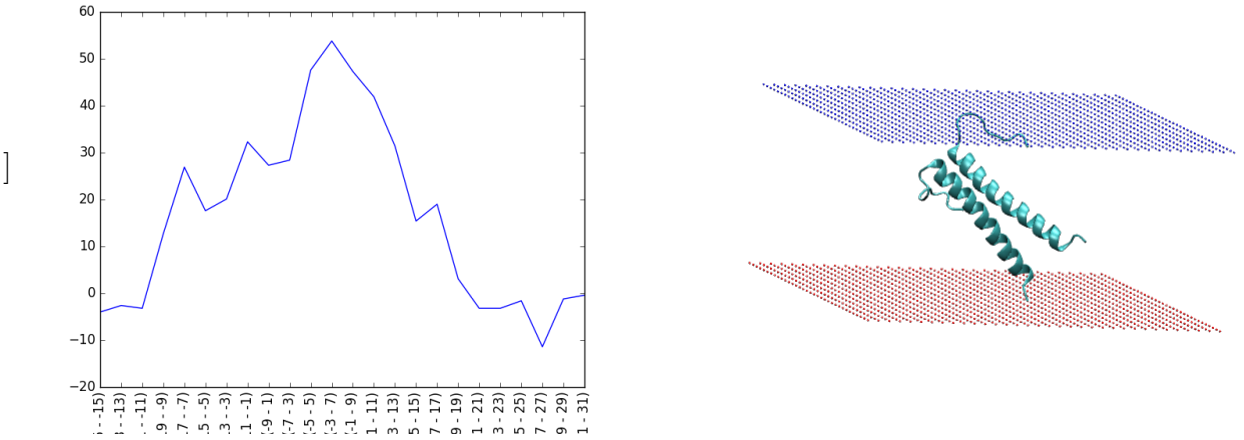


Figure 1: *a)* Hydrophobicity plot for 1afo protein. The central region of high hydrophobicity corresponds to the part that that needs to be inserted in the membrane in order to be stable. *b)* Result for 1afo after rotation and insertion of the membrane. Probably the helix should be more perpendicualar to the membrane. Also, the layers seem too separated from eachother.

(a) ***(b)***

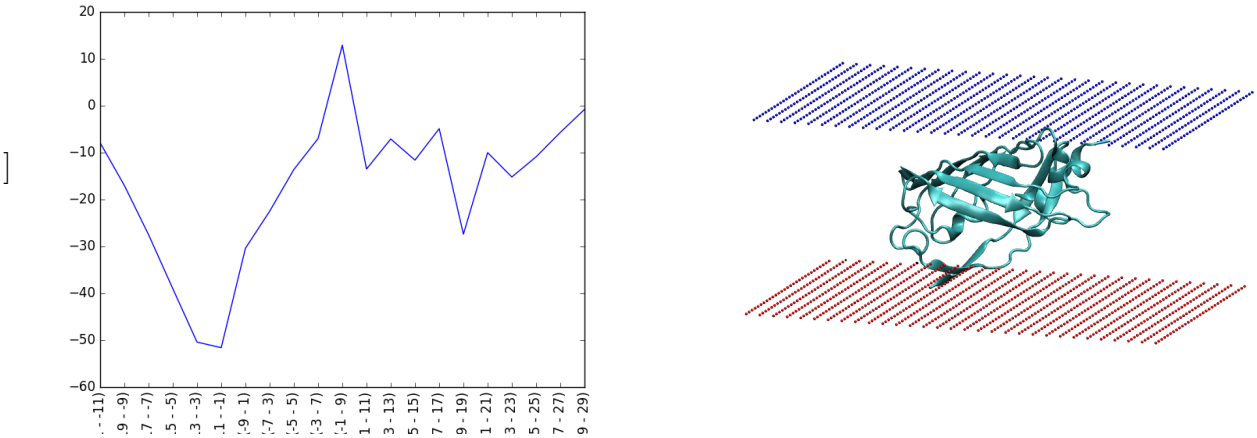
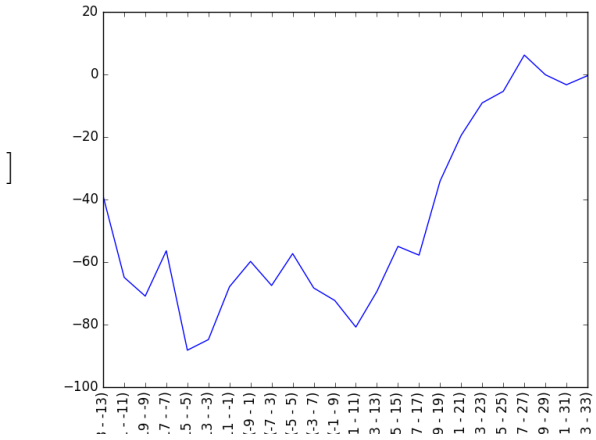


Figure 2: *a)* Hydrophobicity plot for 4zbl protein. *b)* Result for 4zbl after rotation and insertion of the membrane. Again, the beta barrel should be perpendicular to the membrane.

(a)



(b)

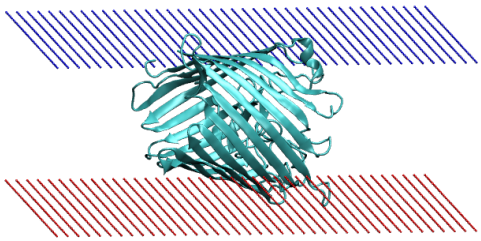


Figure 3: *a)* Hydrophobicity plot for 5dl7 protein. This is an example of the problems of our algorithm with β barrels, which usually don't show a clear hydrophobic central region. This may be because beta barrels have an hydrophilic core. *b)* Result for 5dl7 after rotation and insertion of the membrane. Clearly, the protein is not well rotated..

4 Discussion and Conclusions

The results are not accurate, so there is plenty of room for improvement, of course. For some of the proteins we found that it did not rotate as expected. The problem might be in the approach used for finding the protein vector. When we observed the disposition of the aromatic rings we found that, instead of being parallel to the membrane, they have very different orientations. Therefore, another criteria to determine the protein vector would have been more accurate. Or in case we would have had more time, we could have also tried to filter the aromatic residues and keep only some of them with clear anchor functions and orientations.

On the other hand, we also applied a very rough approach to decide where to put the membrane. Initially, the hidrophobicity seemed to be a good criteria, and in fact it is, but it was difficult to find a way to automatize it for all the proteins, so the result was not precise. We found difficult to code what was the better local minimum in the graph that would leaded us to the right position of the membrane. However, the main problem of the collocation of the membrane was that when the protein is not correctly rotated, the criteria

of hydrophobicity along the z axis loses capacity to give accurate results, because the regions of high hydrophobicity are not aligned to the membrane.

Another issue we had was that for big proteins with different chains it seems that the results are worst than for small proteins of one chain. This makes sense, as the bigger the more complex it becomes (more interactions etc). Also, β barrels give worse hydrophobicity plots than α helixs, without a clear central hydrophobic zone. This may be, a part from the fact that the protein is not well orientated, because β barrels have a central hydrophilic core, so aminoacids whose lateral chain is oriented to the center of the beta tend to be polar.

Finally, context/environment is important. In our project we neglected it and as proteins belong to systems they interact with many things.