Exercise 6: Docking.

We propose the following problem: we have the structures of two proteins. We know that these two proteins interact, but we don't know how does this interaction look like nor the regions of the proteins that are interacting. To find which is the conformation of this interaction we can use docking programs. Besides, we can rebuild the interaction between two proteins using protein superimposition and the structures of homologous interacting proteins, as we learned in practical 3.

Theoretical concepts:

Docking: Docking programs explore the conformational space of the interaction between two proteins. Then, they obtain a set of possible conformations for that interaction that are ranked according to scores. These different conformations are also called docking poses. There are different algorithms to obtain the docking poses. These may use a systematic search, molecular simulations or local shape features. Besides, there are different scoring functions to rank the docking poses. These can be divided in two main groups: those that are based on chemical and physical properties, and those that are knowledge based. If you remember from other practices, statistical potentials were also knowledge based scores. Actually, some of the scoring functions used to rank docking poses are statistical potentials that have been designed to evaluate protein-protein interactions.

There are mainly two types of docking algorithms:

- Rigid body docking: it obtains docking poses by translating and rotating molecular structures as rigid bodies. This means that no structural changes are introduced in the molecules.

- Flexible docking: it works like the rigid body docking, but it allows a certain degree of flexibility in the molecules. Therefore, structural changes are introduced in the molecules.

Keep in mind that docking can be applied to other scenarios than protein-protein interactions. It can be used to study the interactions between other types of molecules. For example, docking approaches are also used to study protein-drug interactions or protein-DNA interactions.

When working with docking programs we have to define the receptor and the ligand. The receptor is the biggest molecule, while the ligand is the smallest. In protein-protein docking the receptor will be the biggest of the two proteins. In protein-drug docking the protein will be the receptor and the drug will be the ligand.

During the process of docking, the receptor remains immobile while the ligand moves around it, testing all their possible interacting conformations. This exploration of the conformational space can be more or less exhaustive, depending on how many conformations you test. The more conformations you test, the higher the computational cost will be. Each one of these conformations is called a docking pose, and can be represented by a pdb file.

During this tutorial we will learn how to work with the Pydock web server.

Tutorial:

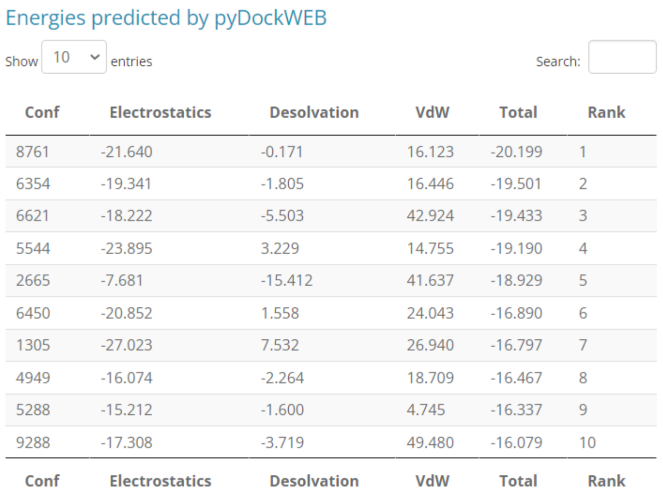
Step 1: Ordinary docking

In this tutorial we will work with the interaction between two proteins: the RhoGAP GTPase and the RAS related protein RAB1A. The interaction between these two proteins is fundamental for proper signal transduction within cells. Here, RhoGAP is the biggest protein, so it will be our receptor, while RAB1A will be our ligand. The PDB entries for these structures are 1F7C_A for RhoGAP and 4FMD_F for RAB1A.

Start by submitting your pdb structures into the pydock web server. Go to https://life.bsc.es/pid/pydockweb and upload the pdb structures for RhoGAP and RAB1A.

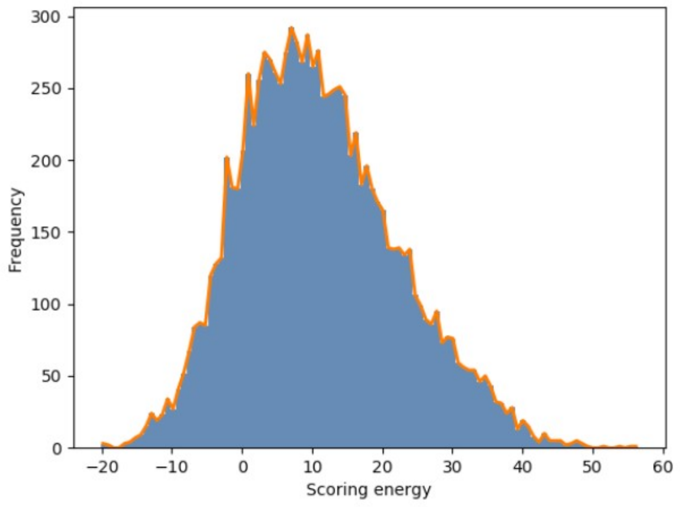
Once you submit your job, this process can take several minutes to finish. So, go to this page where you can see the output pydock page for the same proteins you submitted: https://life.bsc.es/pid/pydockweb/jobs/get_info/2022-02-16_12:34:32_DFLGYBRUCV2I56NM.

Pydock generates 10000 different docking poses. All these poses are scored using a scoring function that assesses electrostatic interactions, interaction with water molecules (desolvatation) and Van der Waals interactions. From the 10000 poses, pydock shows you the best 500, ranked according with their scores. Also, it allows you to download the 500 top scoring poses as pdb files. You can also see the distribution of scores for all the generated poses.


Energies predicted by pyDockWEB

Show 10 entries Search:

Conf	Electrostatics	Desolvation	VdW	Total	Rank
8761	-21.640	-0.171	16.123	-20.199	1
6354	-19.341	-1.805	16.446	-19.501	2
6621	-18.222	-5.503	42.924	-19.433	3
5544	-23.895	3.229	14.755	-19.190	4
2665	-7.681	-15.412	41.637	-18.929	5
6450	-20.852	1.558	24.043	-16.890	6
1305	-27.023	7.532	26.940	-16.797	7
4949	-16.074	-2.264	18.709	-16.467	8
5288	-15.212	-1.600	4.745	-16.337	9
9288	-17.308	-3.719	49.480	-16.079	10
Conf	Electrostatics	Desolvation	VdW	Total	Rank



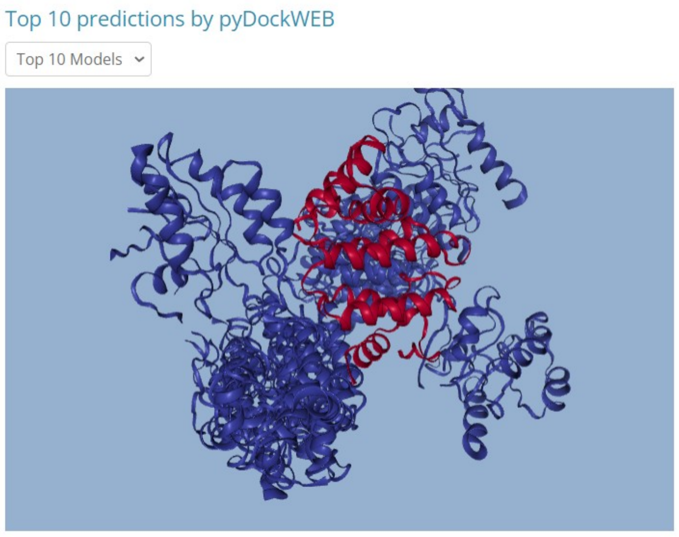
As you can see, the top scoring poses are very different among themselves. However, only one of these poses corresponds with the native state conformation for the interaction of the two proteins. Also, it is not sure that the best scoring pose is the native state one, neither that the native state pose is in the top 10, 50 or 100 best scoring poses. Here you can understand the main limitation of docking: finding the correct docking pose is like finding a needle in a haystack. Luckily, we have other resources besides docking scoring functions to identify docking poses.

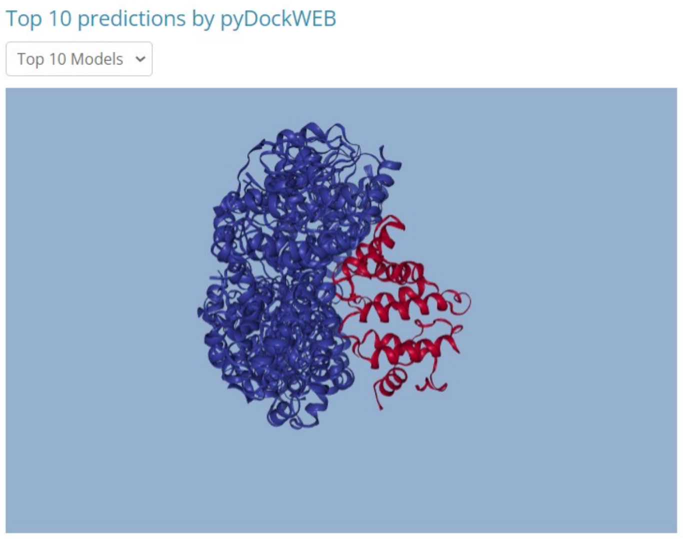
Step 2: Docking with spatial restrictions

If we know some of the amino acids involved in the interaction we can guide docking and produce poses that include these amino acids in the interaction interface. This way, our results will be way more accurate and we will be able to rule out many incorrect docking poses.

The first way to identify amino acids that interact is by conducting wet lab experiments. For example, you could mutate amino acids for the two interacting proteins and see what mutations are disrupting the interaction. There are many experimental methods that can be used to assess protein-protein interactions like yeast two-hybrid or plasmon surface resonance.

For this execution I will give you a tip: Arginine 279 from RhoGAP is interacting with threonine 129 from RAB1A. Now go back to the pydock webpage and submit a new job forcing that these two amino acids are involved in the interface of the interaction. Since this execution can take a long time to run, here you can check the output of this execution: https://life.bsc.es/pid/pydockweb/jobs/get_info/2022-02-16_16:28:12_NWPIM76AMT1QM2DE.





In the previous images you can see the top 10 docking poses without restrictions (left) and with the two residues restriction we just introduced (right). Can you observe some changes in this new pydock execution? Are the scores better? Do you still have very different docking poses?

Step 3: Finding hot-spots to guide the docking

In order to achieve a more precise docking, we can use bioinformatical tools to find what residues are more likely to be interacting. These regions that are more likely to be involved in a protein-protein interaction are called hot-spots. There are several tools we can use to predict, in this practical we will learn how to use iFrag, one program developed by the group of Baldo. To access this method go to the following web page: https://sbi.upf.edu/web/index.php/research/servers/iFrag. Once you are inside the iFrag page, input the fasta sequences of the two proteins we are working with. You can get the fasta sequences from the P6_directory and click on submit.

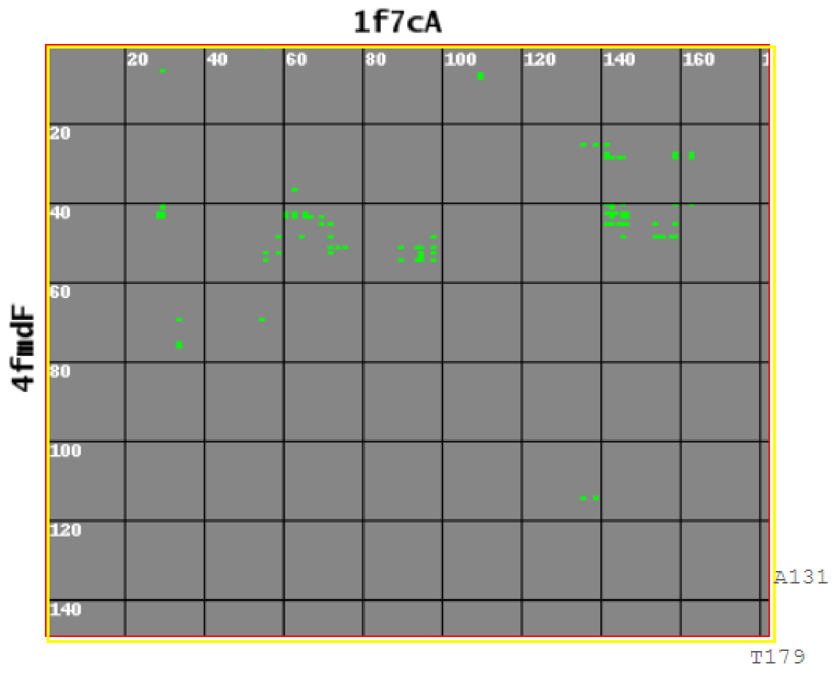
The iFrag server does several things:

- Checks in a database of experimentally determined protein-protein interactions for homologous sequences for the two input sequences. The objective is to find a pair of interacting proteins where each of them is homologous to one of the input proteins.

- Checks all the protein-protein interactions of the PDB and looks for homologous structures for the two input sequences. Again, the objective is to find interacting proteins that are homologous to the two input proteins. In this case, the results will be more precise, because iFrag will only show the regions that are interacting in the PDB structures.

- Checks the PFAM HMMs of the input proteins and looks for information of these two HMMs interacting. This option is not very precise because HMMs cover lots of amino acids, and here we are interested in the few residues that are involved in the interaction.

The results of iFrag are represented in hetmaps where the two sequences are shown in different axis. Residues that are predicted to interact are indicated with color. From all this information we will focus on the information provided by the PDB this corresponds with the next heatmap:



You can check the results for this iFrag execution at the next link: https://sbi.upf.edu/web/index.php/research/servers/iFrag?jobID=01121ba02bded48d1a8ca51ef3f604de

By checking this heatmap we can identify the following pairs of interacting amino acids:

Amino acids in RhoGAP	Amino acids in RAB1A
60-76: TITSALKTYLRMLPGP	26-31: TISTI
140-149: GVVFGPT	41-55: AGQERFRTITSST

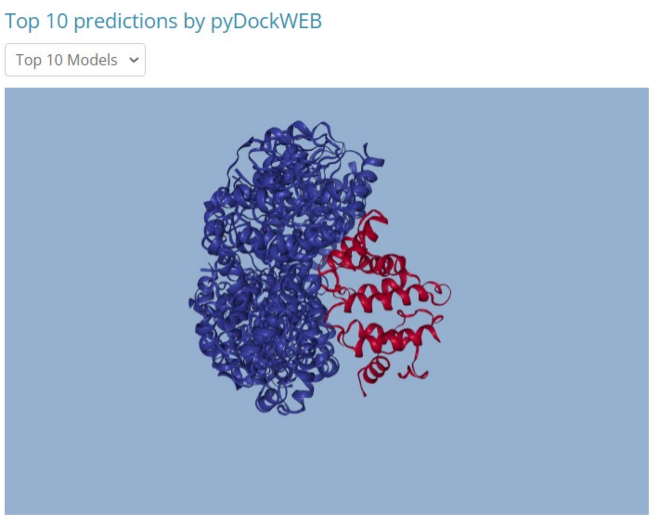
However, when you are going to put these amino acid coordinates in the pydock submission form you can see that the amino acids do not correspond with the ones of iFrag. This is because iFrag is assuming that the structure starts at position 1, and instead of that our structures start at position 191 and 13, respectively. Besides, there are some regions of the protein that are not included in the structure because they couldt be crystalized. The result is a sequence that is missing amino acids at some points:

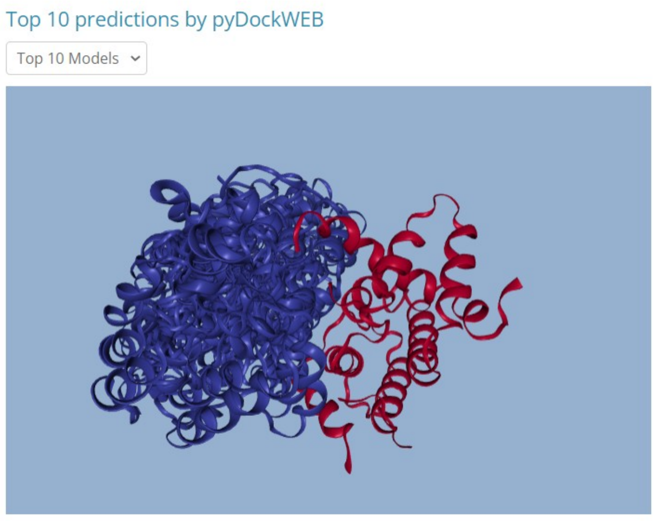
- RhoGAP jumps from position 238 to 245 and from position 347 to 351.
- RAB1A jumps from position 37 to 41 and and from position 51 to 64.

This is very annoying, but it is extremely common when working with structures from the PDB. The solution for this is to check where are the jumps and count to identify where are the regions that you are interested in. To make things faster, I have made this conversion for you, here you have the amino acids that you have to select to execute pydock:

- Amino acids in RhoGAP: THR.256, ILE.257, THR.258, SER.259, ALA.260, LEU.261, LYS.262, THR.263, TYR.264, LEU.265, ARG.266, MET.267, LEU.268, PRO.269, GLY.270, PRO.271, GLY.337, VAL.338, VAL.339, PHE.340, GLY.341, PRO.342, THR.343.
- Amino acids in RAB1A: THR.37, ILE.41, SER.42, THR.43, ILE.44, ALA.68, GLY.69, GLN.70, GLU.71, ARG.72, PHE.73, ARG.74, THR.75, ILE.76, THR.77, SER.78, SER.79, THR.129

Once you have selected these amino acids, submit your pydock execution. Since it will take some time to run, you can access the pydock results for this same execution at this link:
https://life.bsc.es/pid/pydockweb/jobs/get_info/2022-02-17_10:34:52_GB4NCTYY2Z7L1KHJ





In the previous images you can see the top 10 docking poses with two amino acid restrictions (left) and the top 10 docking poses with whole interface restrictions we just made (right). Can you observe some changes in this new pydock execution? Are the scores better? Do you still have very different docking poses?

Step 4: Using 3did to find structures that contain interactions between domains

So far we have been doing *ab initio* docking. As you have seen, it is hard to identify correct docking poses from all the results that docking programs produce. That is why the use of *ab initio* docking has some strong limitations.

Template guided docking on the other hand is way more reliable. This type of docking is what we did on practical 3, when we were using superimposition to reconstruct interactions between proteins. The main limitation of this approach is that there are few structures for protein-protein interactions. Therefore, if you want to model an interaction it is very likely that you cannot find templates for it, and then you have to use *ab initio* docking.

Another limitation of template based docking is that you have to find one PDB entry that is homologous to the two proteins that you are interested in. Then, you have to generate two sequence searches and look close to both files. This is very unproductive. Instead of that you can use 3did: https://3did.irbbarcelona.org/

3did is a database of interacting protein domains. You can search there for domains that you are interested in, and it will tell you what other domains interact with your query. Also, for each of these domain-domain interactions it will tell you if there is any PDB structure representing this interaction. This can be an easy way to find templates for protein-protein interactions.

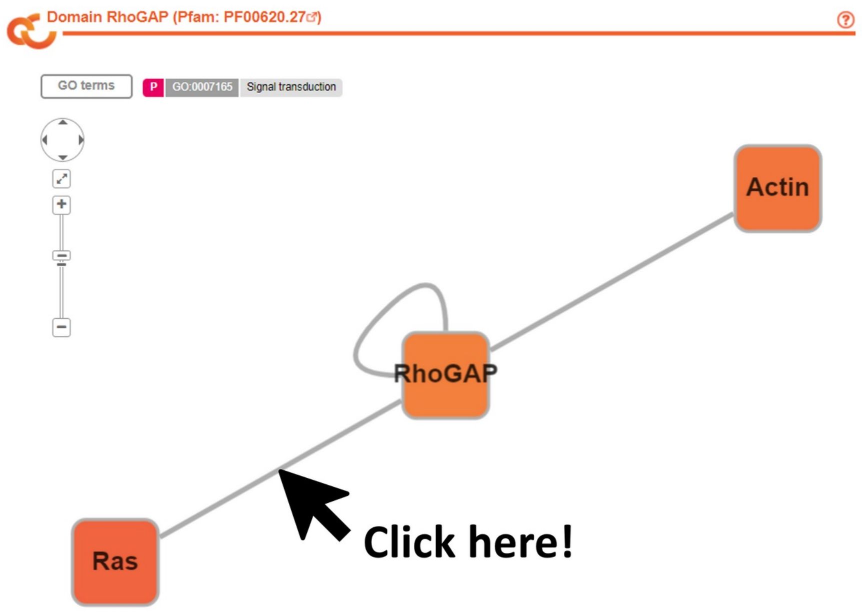
We are going to use 3did to find templates for the interaction we are studying. The first thing we need to know is what are the domains of our protein structures. To do this we will upload them to the cluster and execute hmmscan to see if they match with some HMM from PFAM. Remember to load the modules to execute the hmmer package.

```
hmmscan  /shared/databases/pfam-3/Pfam-A.hmm  1f7cA.fa  >  1f7cA.out
```

```
hmmscan  /shared/databases/pfam-3/Pfam-A.hmm  4fmdF.fa  >  4fmdF.out
```

After this execution you know that the domains of your proteins are RhoGAP and RAS. Now you can go to the 3did database and search for one of these two domains, then check if it says that it interacts with the other.

Start by searching RhoGAP, you will see that this takes you to a page where you can visualize how other domains interact with RhoGAP. One of them is RAS. If you click on the node between RhoGAP and RAS you will arrive to a page describing this interaction.



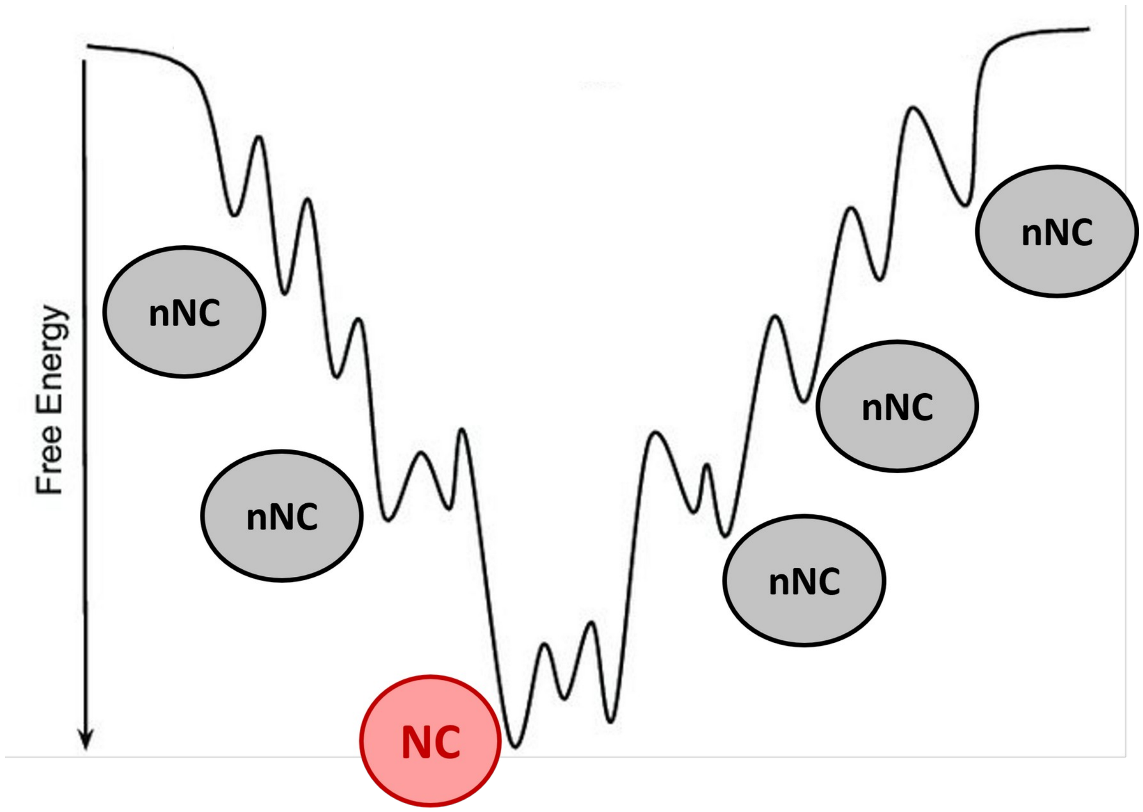
Once in the page describing the interaction between RhoGAP and RAS, you will find many PDB templates for the interaction between these two domains. Take one of these templates and use it to reconstruct the interaction between 1f7cA and 4fmdF. Remember that you have to do just as we did on practical 3 when we were reconstructing protein-protein interactions using superimposition.

Step 5: Using docking to estimate the binding affinity between two proteins

In the last years, several groups working on structural bioinformatics have found that the whole ensemble of docking poses can be used to predict binding affinity between proteins. The reasoning for this consists on the fact that when proteins find each other in the cytoplasm they must be able to recognize who they should interact with. Keep in mind that the cytoplasm is a super crowded environment and yet proteins are able to distinguish their interacting partners in a fast and precise way.

To explain this, there is the theory that when two interacting proteins find each other they interact not through their native state conformation, but through other random conformations. Then, from one of these random conformations, the two proteins recognize each other and change their conformation until reaching the native state conformation. This native state conformation is the one with lowest free energy (ΔG). However, the conformations that take place before should also have low energy, in order to keep the interacting proteins together until they reach the native state conformation.

A way of representing this idea is with an energy funnel, just as we did to describe the folding of proteins:



Here the idea is that if two proteins interact, their non-native conformations will have low energies to allow the finding of the native state conformatio. If two proteins do not interact, the non-native conformations will have high energies, avoiding these two proteins to interact. Therefore, if you take all posible conformations between two proteins, since all conformations contain some information regarding the interaction between them two, you can use them to predict if the proteins will interact.

This is what BADock does, generates an ensemble of docking conformations. Then, it uses them to make a prediction of the change in Gibbs free energy (ΔG) that takes place when the two proteins interact.

You can use BADock by connecting to the following link: http://aleph.upf.edu/BADock/. Here you have the results for a couple of executions in the BADock server, one for the two interacting proteins we have been working with today, and another for two proteins that don’t interact.

Results between the two interacting proteins:
http://aleph.upf.edu/BADock/index.php/results/job/dg_620e683099a8a.html

Results between two non interacting proteins:
http://aleph.upf.edu/BADock/index.php/results/job/dg_620e69919ddae.html

Questions from the tutorial:

1) Use superimposition to compare the interaction you made using template guided docking and some of the docking poses you obtained using *ab initio* docking.

2) Try to execute pydock with a protein that you are working with in your projects. Then comment the results. Are all the top docking poses similar or different? What is the overall score for all the generated poses? Did you use some amino acid restrains to improve your results?

3) Try to find the protein you are working with in the 3DID database. What domains can your protein interact with? Do you have available structures for these interactions? Can you relate some of these interactions with the function of your protein?