Ab initio protein structure modeling using Rosetta

1. This tutorial is based on:
https://www.rosettacommons.org/demos/latest/tutorials/denovo_structure_prediction/Denovo_structure_prediction

2. Setup

Make a new directory and change into it:

mkdir rosetta

md rosetta

All work for this exercise will be conducted and retained in this directory.

3. Perform *ab initio* folding on the HOP protein

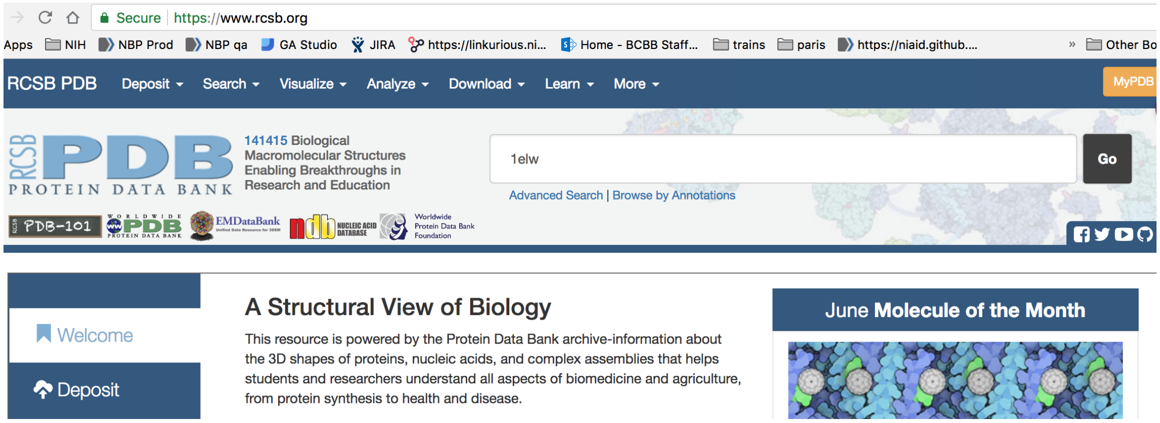
The required files to run an ab initio calculation are:

- A sequence in fasta format
- 9mer fragment file
- 3mer fragment file

3.1. Obtain a sequence to fold by going to:

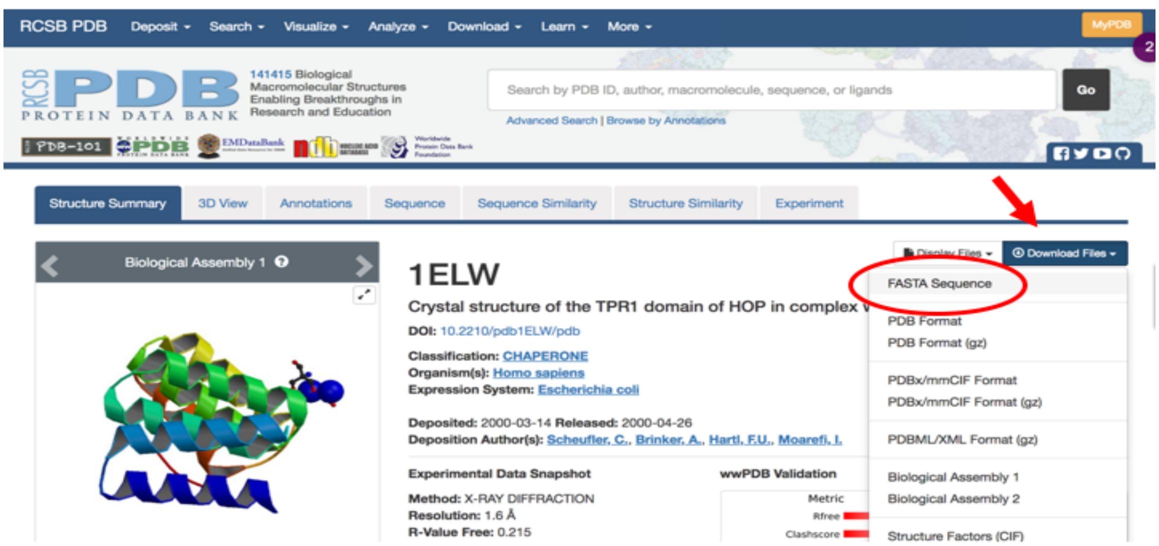
https://www.rcsb.org

3.2. Enter 1elw in the search field and click 'Go'



3.3. An entry for the crystal structure of the TPR1 domain of the Hsp70-Hsp90 Organizing Protein (HOP) appears.

3.3. Click on ‘Download Files’ (red arrow below) and choose ‘FASTA Sequence’



3.4. If possible, save the file as ‘1elw.fasta,’ otherwise use the following command to change the filename:

mv 1elw.fasta.txt 1elwA.fasta

If the file is in a directory other than your working directory, go to that directory and type the following to move it to your ‘rosetta’ directory:

mv 1elwA.fasta ~/Desktop/rosetta/

3.5. The file contains sequences of two HOP proteins as well as peptides, all found in the unit cell. Edit the file using a text editor so that it only contains only 1 HOP sequence:

>1ELW:A|PDBID|CHAIN|SEQUENCE
MEQVNELKEKGNKALSVGNIDDALQCYSEAIKLDPHNHVLYSNRSAAYAKKGDYQKAY
EDGCKTVDLKPDWGKGYSRKAA

3.6. The next step is to generate fragment files based on the sequence. These files will contain short backbone fragments that will be randomly inserted at all positions during the simulation.

There are two ways to do this, one from the command line and one using the Robetta server.

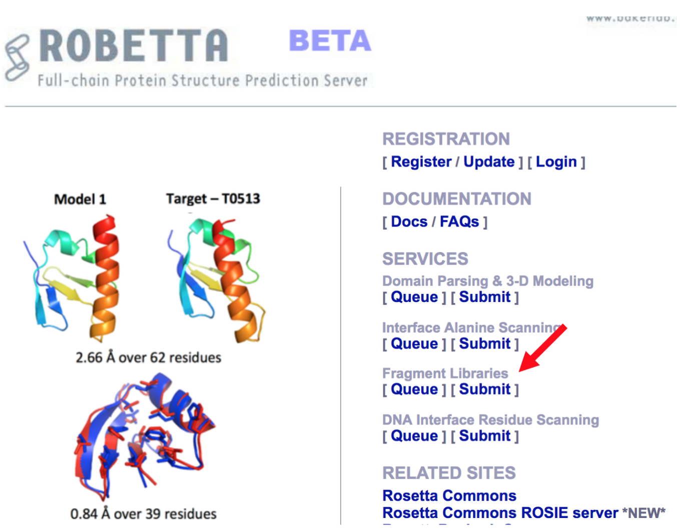
A. To generate fragment files locally, follow these instructions:

https://www.rosettacommons.org/demos/latest/public/fragment_picking/README

or use the much easier way….

B. Go to the following and click ‘Submit’ under the Fragment Libraries service:

http://robetta.bakerlab.org/



3.7. From the fragment generation run (either from the Robetta output or from your local calculation), obtain the following files:

aa1elwA03_05.200_v1_3
aa1elwA09_05.200_v1_3

Note: Pre-calculated files can be copied as follows:

cp ~/Desktop/model/abinitio/input_files/aa* .

3.8. Create/obtain a 'flags' file where all options (flags) can be nicely organized.

```
-in:file:fasta ./input_files/1elwA.fasta
-in:file:frag3 ./input_files/aa1elwA03_05.200_v1_3
-in:file:frag9 ./input_files/aa1elwA09_05.200_v1_3
-abinitio:relax
-nstruct 1
-out:pdb
-use_filters true
-abinitio::increase_cycles 10
-abinitio::rg_reweight 0.5
-abinitio::rsd_wt_helix 0.5
-abinitio::rsd_wt_loop 0.5
-relax::fast
```

Note: A pre-generated flags file can be copied as follows:

cp ~/Desktop/model/abinitio/flags .

3.9 View the contents of the flags file using nano:

nano flags

3.10. To see the available options for the ab initio executable, type the following:

AbinitioRelax.static.linuxgccrelease -help

Note: Unfortunately, the Rosetta developers have created very long executable names. Two ways to reduce typing are to:

a. Create an alias such as:

alias ab='AbinitioRelax.static.linuxgccrelease'

B. Make use of tab completion. Type in 'Ab' and hit the tab key to complete the Command.

3.11. Run the job:

AbinitioRelax.static.linuxgccrelease @flags

Note: 1 decoy is not enough, need 50,000-100,000 decoys.

3.12. Use Chimera or other program to examine the results.

3.13. Examine the scoring file:

nano score.fsc

3.14. Note: In this next section, we'll set up the job again to generate 10 decoys and a 'silent' file.
A silent file is a compact format file which stores information from multiple structures.Silent files can be converted into PDB files using the `extract_pdb` application. In addition, we'll add constraints in the form of secondary structure predictions which are provided by Robetta alongside the fragment file output.

Copy the secondary structure prediction file to your working directory:

cp ~/Desktop/model/abinitio/input_files/1elwA.psipred_ss2 .

3.15. Modify the flags file using nano:

nano flags

a. Add the following line:

-psipred_ss2 ./1elwA.psipred_ss2

b. Change out:pdb to:

-out:file:silent ./silent.out

3.16. Remove all files except the .fasta, the two fragment files, the flags file and the psipred file:

rm scores.fsc
rm default.out

Rename the .pdb for later:
mv S_00000001.pdb one.pdb

3.17. Run the job:

AbinitioRelax.static.linuxgccrelease @flags

3.18. Extract the silent file using:

extract_pdbs.static.linuxgccrelease   -in::file::silent   silent.out

3.19. Examine the scoring file:

nano score.fsc

3.20. Rename the .pdb for later:
mv S_00000001.pdb two.pdb

3.21. For the final run, use a native structure to guide folding. Copy a native structure to your working directory:

cp ~/Desktop/model/abinitio/input_files/1elw.pdb .

3.22. Add the following line to the flags file:

-in:file:native ./1elw.pdb

3.23. Run the job:

AbinitioRelax.static.linuxgccrelease @flags

3.24. Extract the silent file using:

extract_pdbs.static.linuxgccrelease   -in::file::silent   silent.out

4. Analysis

Note: For larger numbers of decoys, an algorithm can be used to cluster decoys and provide representative structures to visualize.

4.1. Read one.pdb, two.pdb and the S_0000001.pdb (from the final run) into Chimera, PyMOL or other visualization program. Also read in 1elw.pdb.

How does the native structure compare?

4.2 Sort and isolate the top x decoys (this will find the top 5 in terms of lowest energy):

```
sort -n -k2 score.sc | head -l 5
```

A great video tutorial is offered here:
https://www.youtube.com/watch?v=y6-1UUEf4Pw