Protein Structure and Bioinformatics Databases

Milot Mirdita, Clovis Galiez, Johannes Söding

November 7th, 2017

In this tutorial, we will predict the 3D structure of ACY2_HUMAN[1], the human aspartoacylase enzyme. Its PDB structure **2o4h**[2] is already known but we will pretend it does not exist for the purpose of this exercise (though it will be helpful for evaluating your predicted structure). After this simple target, we will give you another more challenging protein on which you can show what you have learnt!

Note: You can find an online version of this worksheet with clickable links, intermediate results you can use if you get stuck, plus data and a script you will need later on by going to `http://wwwuser.gwdg.de/~compbiol/molbio_course/2017`

1 Sequence-based Analyses on Proteins

In this first section, we will run a few simple sequence-based analyses to get a general overview over what is already known about the protein and its relatives and to get an overview over secondary structure elements, transmembrane regions, etc. For this, we will be using the UniProt[3], PDB[4] and PFAM[5] databases combined with the Quick2D[6] tools for doing sequence-based predictions.

Note: As Bioinformatics methods are continuously being improved on, it is a good idea to learn about the currently best-performing tools by having a look at the results of the biannually occurring CASP competition[7]! For example, to find the results for disorder prediction, you could search PubMed for 'CASP disorder prediction'.

[1] http://www.uniprot.org/uniprot/P45381

[2] http://www.rcsb.org/pdb/explore.do?structureId=2o4h

[3] http://www.uniprot.org

[4] http://rcsb.org

[5] http://pfam.xfam.org

[6] https://toolkit.tuebingen.mpg.de/#/tools/quick2d

[7] http://predictioncenter.org

1.1 Finding Annotation for Unknown Sequences

From the link at the top, download the sequence `unknown.fasta`. Use a BLAST search on the UniProt[8] and the PDB to find the sequence or a related sequence in the databases:

```
>a_mystery_protein
MTSCHIAEEHIQKVAIFGGTHGNELTGVFLVKHWLENGAEIQRTGLEVKPFITNPRAVKK
CTRYIDCDLNRIFDLENLGKKMSEDLPYEVRRAQEINHLFGPKDSEDSYDIIFDLHNTTS
NMGCTLILEDSRNNFLIQMFHYIKTSLAPLPCYVYLIEHPSLKYATTRSIAKYPVGIEVG
PQPQGVLRADILDQMRKMIKHALDFIHHFNEGKEFPPCAIEVYKIIEKVDYPRDENGEIA
AIIHPNLQDQDWKPLHPGDPMFLTLDGKTIPLGGDCTVYPVFVNEAAYYEKKEAFAKTTK
LTLNAKSIRCCLH
```

1.2 Looking Up Existing Annotation

Use the UniProt and PDB databases to learn about what is already known and annotated for our target ACY2_HUMAN. Here are some questions you should be able to answer using the two websites:

- What publications are associated with this protein? What is its clinical relevance?
- What is the molecular function? What are known ligands? What are known cofactors?
- What are the known functionally important residues? Which of them are in the active site? Do we know what happens to them upon mutation?
- How was the 3D structure solved? Are there any modifications from the native sequence? What is the experimental resolution? What other molecules and heteroatoms have been solved with the structure? Are they biological ligands? If not, what are they?

Now is also a good time to download the protein's sequence in FASTA format since you will be needing it for future analyses. Save it as `ACY2_HUMAN.fasta` in a work directory on your computer. Also download the 3D structures of **2o4h** and **3nh4** in the mmCIF format. Examine the downloaded mmCIF files. What do they contain[9]?

1.3 Predicting Secondary Structure

Since sequence-based features of a protein are being predicted by a multitude of different software packages, it can become tedious to run predictions by submitting the same sequence to several webservers. We therefore recommend using the Quick2D tool found in the Bioinformatics Toolkit[10] to run a battery of predictors.

Since the Quick2D tool takes a while to come back with results, we've pre-computed the results for you to find at `https://toolkit.tuebingen.mpg.de/#/jobs/7331514`. What secondary structure elements can you identify? Are there transmembrane regions? Are there disordered regions? Do the methods agree with each other? Do they agree with the annotation from UniProt?

[8] `http://www.uniprot.org/blast`

[9] You can find a documentation on the file formats on `http://www.wwpdb.org/documentation/file-format`

[10] `http://toolkit.tuebingen.mpg.de`

1.4 Detecting Annotated Domains

Working on unknown proteins, it can also be much easier to separately predict individual domains since they form evolutionary units that might appear in another protein where more is known. You can use the Pfam database to detect previously annotated domains in your protein sequence.

How many domains does the protein sequence have? Do these domains appear in other proteins? Is there additional annotation on the domain level not present in the UniProt entry?

2 Template-based Modelling

From the FASTA sequence **ACY2_HUMAN.fasta** you've previously downloaded, we will first enrich our target sequence with evolutionary information by building a multiple sequence alignment using HHblits against the UniProt. With the enhanced evolutionary information, we can then sensitively look for structural homologs in the PDB. We pick a template for modelling and then use the target-template alignment to model the structure with MODELLER.

Note: When working with the Linux command line, be sure to double-check what you write! Commands and file names are case-sensitive and the positions of spaces matter!

2.1 Building a Query Alignment

We build a query alignment on **ACY2_HUMAN** by running HHblits on the Uniclust30[11] database for two iterations[12].

You can find the required databases in `/home/molbio/databases/hhblits`:

```
hhblits -d ~/databases/hhblits/uniclust30_2017_07  # use Uniclust
        -i ACY2_HUMAN.fasta                        # input sequence
        -o ACY2_HUMAN.query.hhr                    # output results in HHR format
        -oa3m ACY2_HUMAN.query.a3m                 # output alignment in A3M format
        -n 2                                       # run 2 iterations (more sensitive), 3 on better computers
        -cpu 1                                     # use just 1 CPU due to the virtual machine
```

After HHblits has finished, you should have two new files in your working directory: The detailed result file **ACY2_HUMAN.query.hhr** and the alignment in A3M format **ACY2_HUMAN.query.a3m**.

2.2 Manually Examining Alignments with JalView

It's always a good idea to look at the generated alignments. To reduce the number of sequences to an amount that can be viewed on a single screen, we first filter the alignment

[11] UniProt clustered down to 30% sequence identity – this way, we don't give a bias to many copies of nearly identical entries in the database resulting in more even predictions and save space!

[12] For very easy targets, you might get away with skipping this step and searching for the target sequence directly in the PDB. We'll practice the full procedure here so that you'll know what to do in more difficult cases.

down to the most diverse set of 30 or so sequences. This is a very useful trick to check if your alignment contains any problematic sequences, for example sequences containing non-homologous stretches, since they will be among the most diverse set of sequences. We filter using the hhfilter binary from the HH-suite software package:

```
hhfilter -i ACY2_HUMAN.query.a3m          # input alignment (must be in A3M format
         -o ACY2_HUMAN.query.filt.a3m     # output alignment (in A3M format)
         -diff 30                         # get most diverse set of ∼ 30 sequences
```

Now we need to convert the A3M-formatted alignment into a more common format such as FASTA that an alignment viewer like **JalView** can use. We employ the reformat.pl script in the HH-suite. To make the alignment as compact as possible, we remove all inserts with respect to the query sequence using the option −r (The result is sometimes called a master-slave alignment):

```
reformat.pl -r                            # remove all gaps w.r.t. the query sequence
            a3m                           # input is in A3M format
            fas                           # output should be in FASTA format
            ACY2_HUMAN.query.a3m          # input alignment
            ACY2_HUMAN.query.fasta        # output alignment
```

Take some time to look at both the HHblits output files using your favorite text editor and the FASTA-formatted alignment using JalView. You can start JalView by going to `http://www.jalview.org` and clicking the purple ‘Launch Jalview Desktop’ button in the top right corner of the page.

- Try out the coloring options. Use e.g. ‘Taylor’, ‘Percentage Identity’ and ‘Hydrophobicity’ schemes to identify conserved sites.
- Compare your results to the ‘Conservation’, ‘Quality’ and ‘Consensus’ annotations at the bottom
- What is the consensus sequence?
- Can you make guesses about the domain structure / evolutionary units of the protein from the alignment?

2.3 Searching for Structural Homologs

Now that we know all about the evolutionary history of ACY2_HUMAN (from a UniProt perspective), we can use this profile data to sensitively search for homologs in the PDB70 using HHblits[13,14]:

```
hhblits -d ~/databases/hhblits/pdb70      # use PDB
        -i ACY2_HUMAN.query.a3m           # use query profile
        -o ACY2_HUMAN.templates.hhr       # output results in HHR format
        -n 1                              # run 1 iteration (we already know the query profile)
        -cpu 1                            # use 1 CPU due to the virtual machine
```

[13] Again, find the database files in /home/molbio/databases/hhblits

[14] On better computers we would use HHsearch, for even more sensitive searches

As you will notice from the output, the hits that were found this time are all PDB entries since we searched in a database containing only PDB entries. You might wonder why our true PDB structure 2o4h is not part of the top-scoring hits, since 2o4h_A was clustered together with the third hit 2q4z_B. The clustering can be inspected in `~/databases/hhblits/pdb70_clu.tsv`.

2.4 Picking a Template for Modelling

Examining the results in ACY2_HUMAN.templates.hhr, you should see the following top-scoring hits:

No.	ID	Description	Seq. Id	Aligned cols	E-value
1	3NH4_A	Aspartoacylase-2	43%	303	6.8E-51
2	3NH8_A	Aspartoacylase-2	40%	300	1.5E-49
3	2GU2_B	Aspa protein	77%	302	2.2E-45
4	2Q4Z_A	Aspartoacylase	77%	302	2.2E-45
5	3CDX_B	Succinylgluta...	16%	263	1.2E-32

Hits 1, 2, 3 and 4 are in the *safe homology modeling zone* while hit 5 is in the *twilight zone*[15]. Since we know that hit 4 contains the true structure and we don't want to make it too easy, we'll pick hit 1 as a template for structural modelling and use the script hhmakemodel.pl together with the template PDB structure 3nh4 to write out a PIR-formatted alignment that can be used in MODELLER:

```
hhmakemodel.py
  ACY2_HUMAN.templates.hhr
  ./                                # directory with downloaded structures in the cif format
  ACY2_HUMAN-3nh4_A.pir             # output PIR-formatted alignment
  ./formatted_cif/                  # output directory for fixed structures in the cif format
  -m 1                              # pick hit number 1 (i.e. 3nh4_A)
```

The script will look through our alignment ACY2_HUMAN.templates.hhr, pick out the first hit '3nh4_A' and look for a file named 3nh4.cif in the directory we specified. Mapping the residues in the PDB file onto the residues in the alignment, it then generates a query-template alignment that can be used for generating restraints in MODELLER.

2.5 Modelling with MODELLER

Now we are almost ready to run the actual modeling! Since MODELLER is quite picky when it comes to file paths, we need to double-check that some identifiers and filenames match up. The filenames MODELLER will search for are determined by the PIR alignment. If you look at ACY2_HUMAN−3nh4.pir, you notice both a FASTA-style identifier line starting with '>' and a second identifier line giving some more details:

```
>P1;UNKP
sequence:UKNP:   1: : 309: :Aspartoacylase OS=Homo sapiens GN=ASPA PE=1
     SV=1: : 0.00: 0.00
[...]
```

Since the name of our target is what the files for our model will be called, we'll rename it from UNKP to simply ACY2_HUMAN (be sure to edit both occurrences!):

[15] See e.g. `http://www.cmbi.ru.nl/~hvensela/EGFR-verslag/homology.html`

```
>P1;ACY2_HUMAN
sequence:ACY2_HUMAN:   1: : 309: :Aspartoacylase OS=Homo sapiens GN=
    ASPA PE=1 SV=1: : 0.00: 0.00
[...]
```

MODELLER does not have a graphical user interface but is controlled through a Python script file. You can download a simple Python script from the address mentioned at the top of this document. As always, it's a good idea to examine the files that you're using! We will now use this script to run the modelling:

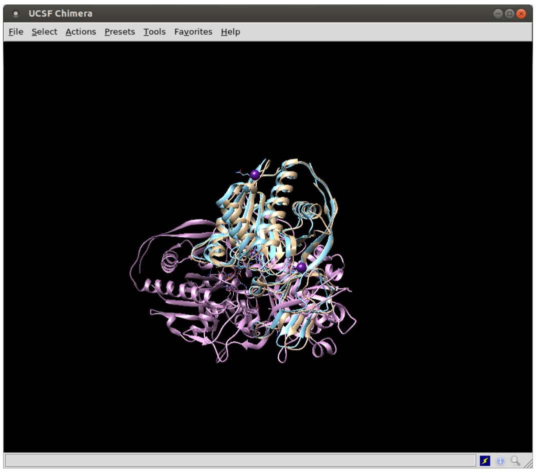
```
model.py
        ACY2_HUMAN-3nh4_A.pir          # our modified input alignment
        ./formatted_cif/               # directory with fixed structures in the cif format
        ACY2_HUMAN                     # the name of our target in the alignment
        3NH4                           # the name of our template in the alignment
```

After a little bit of computation, MODELLER will spit out various quality measures[16] and a model of ACY2_HUMAN, to be found at the file ACY2_HUMAN.B99990001.pdb.

2.6 Visually Examining Generated Models with UCSF Chimera

Time to check out our model!
Start **UCSF Chimera** and load our model (ACY2_HUMAN.B99990001.pdb), the template that was used (**3nh4**) and the true structure (**2o4h**).

/File/Open - select ACY2_HUMAN.B99990001.pdb file
/File/Fetch by ID - in ID field type **3nh4**
/File/Fetch by ID - in ID field type **2o4h**



Note: By default, you can rotate the camera with the left mouse button, zoom with the right mouse button or scroll.

[16] For details on the DOPE and GA341 scores, see https://salilab.org/modeller/9.19/manual/node258.html and https://salilab.org/modeller/9.19/manual/node202.html

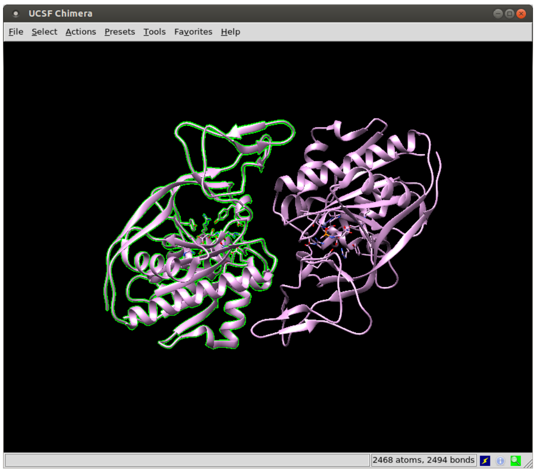
2.6.1 Cleaning up

First, we will clean up the structures to only leave what we're interested in for the purposes of this tutorial.

/Favorites/Model Panel - hide the model and 3nh4 by unmarking **"S"** next to them.

One can easily notice that our true structure is built from two identical chains. Thus, we want to get rid the chain B from **2o4h**.

/Select/Chain/B



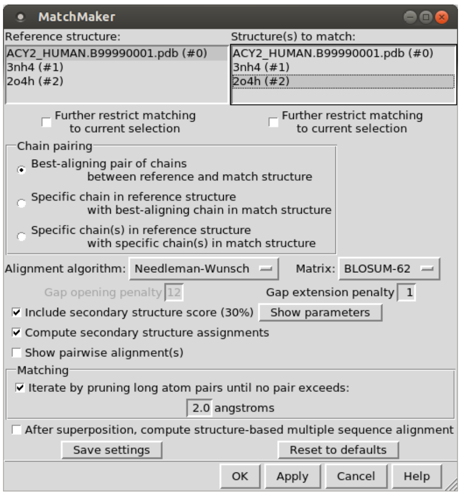
/Actions/Atoms/Bonds/Delete - remove chain B, as we will use only chain A

2.6.2 Structural Alignment

Go back to **Model Panel** and switch on the model and 3nh4 again.

In the main workspace, you should have three molecules, but they are not aligned in 3D. The procedure to do so is called "superposition" and can be easily done by:

/Tools/Structure Comparison/MatchMaker - in the pop-up select the model and the true structure and hit "Apply" or "OK" button. While doing this notice the text in lower left part of the main panel. Note the **RMSD**.



Reference structure:
ACY2_HUMAN.B99990001.pdb (#0)
3nh4 (#1)
2o4h (#2)

Structure(s) to match:
ACY2_HUMAN.B99990001.pdb (#0)
3nh4 (#1)
2o4h (#2)

□ Further restrict matching to current selection
□ Further restrict matching to current selection

Chain pairing
● Best-aligning pair of chains between reference and match structure
○ Specific chain in reference structure with best-aligning chain in match structure
○ Specific chain(s) in reference structure with specific chain(s) in match structure

Alignment algorithm: Needleman-Wunsch Matrix: BLOSUM-62
Gap opening penalty 12 Gap extension penalty 1
☑ Include secondary structure score (30%) Show parameters
☑ Compute secondary structure assignments
□ Show pairwise alignment(s)

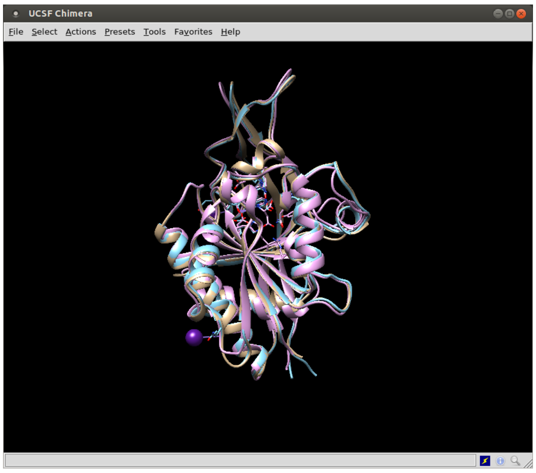
Matching
☑ Iterate by pruning long atom pairs until no pair exceeds:
2.0 angstroms

□ After superposition, compute structure-based multiple sequence alignment

Repeat this to the model and the template. Note **RMSD** again.

How would you rate the overall quality of the model? Is the model closer to the true structure or the template?

Finally, superpose all three together:



You can also superimpose your structures with the more sophisticated TM-align structural aligner by using dedicated software e.g. the TM-align webserver[17].

2.6.3 Examining the Structure

From looking at the small molecules in the template / native structure, can you identify the binding pocket? Is it still conserved in your model?

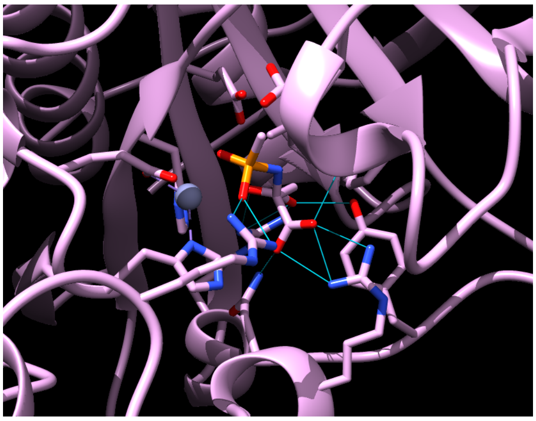
/Select/Structure/Ligand

/Tools/Structure Analysis/FindHBond Mark “Only find H-bonds” - with at least one end selected.

This will give you nice view of catalytic center with ligand inside. You should be able to see H-bonds stabilizing the interaction and Zn ions nearby.

Create some nice figures.

/File/Save Image and change Rendering to “POV-ray”. This option will add shadows, lights etc. Chose file name and hit “Save“ button. Close the ”Reply log“ pop-up when finished.



[17] http://zhanglab.ccmb.med.umich.edu/TM-align

Additionally, some external tools can be used to more advanced analysis. The ConSurf webserver[18] can be used to visualize conserved residues on the protein's surface to identify potential binding pockets or interaction surfaces. Use the webserver interface to identify conserved residues and compare them to the annotations in UniProt.

2.7 Model Quality Ratings

Aside of visualization, the quality of a homology model can also be assessed by computing several quantitative measures, the easiest being overall structural similarity scores on the main chain. You can use the TM-score webserver to compute TM-score, GDT-TS and GDT-HA scores. Check the literature to find out what these individual scores measure.

There are also methods that predict a per-residue quality score for homology models without knowledge of the true structure such as ProQ2[19]. Run ProQ2 on your homology model to determine which residues were modeled more confidently.

2.8 Optional: Homology Model on Point-Mutated Variant

To see an important limitation of homology modeling, repeat the template-based modelling workflow with a protein sequence where you have introduced a deleterious single-residue mutation (e.g. substituting a salt bridge with two same-charge residues, breaking a helix with tryptophan, etc.). Visually examine this new homology model with **UCSF Chimera**. Does it differ significantly from the template that was used?

Hint: To do single-residue mutation:

Close session and start again from **2o4h**. Then select ligand and redo H bonds to it. We will focus on Arg 71. Select it by:

/Favorites/Command Line and type: **sel :71**

Now mutate it:

/Tools/Structure Editing/Rotamers and change Arg into whatever you wish. Redo H-bonds and clash finding. Does changing ARG to GLY affect interaction with the ligand?

Most things shown in this part of tutorial can be automatized and run as commands or simple python script. You can read more about this in the UCSF Chimera user guide[20] Other good alternatives to UCSF Chimera are: PyMOL or VMD.

[18] `http://consurf.tau.ac.il`

[19] Again, use the CASP experiment to find out what works best currently. You could try searching for 'CASP Model Quality Assessment'

[20] `https://www.cgl.ucsf.edu/chimera/current/docs/UsersGuide`

3 Now it's your turn!

Time to show what you've learned! We have prepared a list of UniProt identifiers of lesser-studied proteins on which you can repeat the pipeline. Pick one and see what you can find out:

- G3RZT0
- A0A0F6EHI4
- I7GJP1

Try to paint the most complete picture of the protein's structure and function that you can! Some questions that you can try to answer:

- How many domains does the protein have? Where do they begin and end? What is known about the individual domains?
- Does the protein bind ligands, and if so, which ones and where? Is the binding site conserved?
- What is a likely molecular function for this protein? How would you confirm?

Hint: In the case of multi-domain proteins, you can try to cut the full protein sequence into individual domain sequences to run the HHblits step separately for all domains (look at the alignment with JalView and/or the Pfam results to decide on where to cut). This way, more distantly related protein domains might be found.

Hint: The model.py script used in the MODELLER step supports modelling from multiple templates by adding additional arguments.