**Exercise 3 – Multiple Alignment on your Computer**

**Objectives:**

- to perform a multiple alignment of proteins using three different programs.

- to prepare the input files, bringing them into the right format.

- to check whether the results are identical.

**1) Prepare the input files.**

In the last two exercises, we’ve made two files with protein sequences already. One of them came from BLAST, it will form the basis for an ‘easy’ alignment because the proteins are very similar. The other is from a domain database, this will be the ‘hard’ alignment because the proteins span a very large area in sequence space. We’ll use those input files to create six different multiple alignments; but first we’ll have to properly prepare the input. Both input files should be un-aligned, and for one of them we also still need to add our test protein.

Prior to the course, the files “input\_proteins\_1.fa” and “input\_proteins\_2.fa” were unaligned using regular expressions. The character “-“ was replaced with empty character “” and empty lines were removed. This removed all the gaps, and hence destroyed the alignment. This was saved into “unaligned1.fa” and “unaligned2.fa” respectively; the files are available on OLAT.

- download the files “unaligned1.fa” and “unaligned2.fa” from OLAT and store them to your “Documents/Bio334\_Data” folder (create this folder in case it does not exist yet).

- now, to deal with our test protein: in one of the files, it simply needs to be renamed … and in the other file it is missing, so we need to add it.

- open the File ‘unaligned1.fa’ in an editor of your choice, and change the name of the very first protein, to name ‘query\_protein’. Then save the file. The first lines of the file should now look something like this:

>query\_protein

GFFGDRVGRKFIIWFSILGTAPFALWLPYADADTTAILVILIGFIIS

SAFASILVYSQELLPKKIGMISGVFYGFAFGMGGLASALLGKLIDLTDITFVYKVCSFLP

LMGLIAYFLPNLRKVKMKE

>ref|WP\_002666381.1| MFS transporter [Capnocytophaga gingivalis]

METKQRTQYLIIILISL

SHCLNDLLQGVLPSIYPALQSKFALSMAQIGLITFCYQIAASILQPIVGAYTDKHPKPYA

QVVGMAFSALGIGLLSWVDSYTLVLCSVVFVGIGSSIFHPEASRISFLASGGKRSFAQAV

[…]

- open the second File ‘unaligned2.fa’ in an editor of your choice, and add the query protein sequence to the beginning of the file (simply copy-and-paste from the example output above).

*- optional extra task for the geeks among you: can you do the above file manipulations using the unix-editor “****vi****“ as your editor? If you can’t, you’re not a geek … ☺*

**2) Multiple alignment using Clustal Omega.**

- Open your browser (Chrome/Firefox), and take it to the EBI Clustal Omega website: <https://www.ebi.ac.uk/Tools/msa/clustalo/>

- then on Clustal Omega web page, click ‘choose file’ and choose the file ‘unaligned1.fa’ at your computer.

- then click “submit” button at the end of the page

- after switching to result page, click “Download Alignment File” button to a new page. Right click at anywhere in the new page and choose “Save as” to save to your laptop and name the file as ‘aligned.clustalo.easy.aln’

- now let’s make a nice graphical overview (for example to print it out later). Go back to result page. Choose “Results Viewers” at the top, then click “View in MView” at the bottom. On the MView input form, under “Set Output Parameters”, choose an “Alignment Width” of 120, then click “Submit” at the bottom. Finally click “Download Alignment File”, this will lead us to a pure html page. Right-click on some empty part of that page, choose “Save Page As”, set the format to “Page Source”, and name the file as ‘aligned.clustalo.easy.html’

- repeat the exact same procedure for the second input file. This time, name the output file as ‘aligned.clustalo.hard.aln’. The alignment is more difficult; it should take around five to ten minutes. Again, create a nice graphical overview (‘aligned.clustalo.hard.html’).

**3) Multiple Alignment with Muscle on the Command Line**

the program ‘muscle’ is somewhat faster and often produces better alignments than ClustalX in benchmarks, so it is often preferred for larger and/or more difficult alignments.

- We can install ‘muscle’ easily via Anaconda by the following commands:

. /opt/miniconda3/etc/profile.d/conda.sh

conda create --prefix ~/Documents/py38\_envs python=3.8

conda activate ~/Documents/py38\_envs

conda install -c bioconda muscle

- now, we can run muscle on both of our input files (make sure you are in directory directory “~/Documents/Bio334\_Data/” before running following command):

muscle -align unaligned1.fa -output aligned.muscle.easy.fa

muscle -align unaligned2.fa -output aligned.muscle.hard.fa

- ok, let’s use Mview to format this new alignment in a pretty way again: first go to the URL: <https://www.ebi.ac.uk/Tools/msa/mview/>; then click “choose file” button to select “aligned.muscle.easy.fa” file we just made; set the alignment width to 120, then click “Submit” at the bottom. Finally click “Download Alignment File”, this will lead us to a html page. Right click and choose “Save Page As”, set output format to “Page Source” and name file as ‘aligned.muscle.easy.html’. Do the same for the other file (‘aligned.muscle.hard.fa’) and save file as ‘aligned.muscle.hard.html’

**4) Multiple Alignments using HMMAlign**

the program HMM-align produces very good alignments, but it can only be used if the proteins to be aligned have a previously known domain.

- We can install HMMAlign easily via conda by the commands:

. /opt/miniconda3/etc/profile.d/conda.sh

conda activate ~/Documents/py38\_envs

conda install -c biocore hmmer

- **Optional** you can also choose to compile the source code by following commands in exact order (this will take some time):

cd ~/Documents/Bio334\_Data/

curl -OL <http://eddylab.org/software/hmmer/hmmer.tar.gz>

tar zxf hmmer.tar.gz

cd hmmer-3.3.2

./configure --prefix ~/Documents/Bio334\_Data/HMMER3

make

make check

make install

* next, we need a so-called HMM-file, which describes all the knowledge that has been assembled for a given domain. In our case, follow the steps below:
* Go to this link: <https://www.ebi.ac.uk/interpro/entry/pfam/PF07690/>

- Next, on the left, find Curation, then go towards the bottom of that section and download the raw HMM file and store it into your home directory. Give it the filename “MFS\_1.hmm” and move it to folder “Documents/Bio334\_Data/”

- now, we can use the hmmalign commend to produce the alignments (make sure you are in directory “Documents/Bio334\_Data/” before running following command):

cd ~/Documents/Bio334\_Data/

hmmalign --outformat Stockholm MFS\_1.hmm unaligned1.fa > aligned.hmmalign.easy.sto

hmmalign --outformat Stockholm MFS\_1.hmm unaligned2.fa > aligned.hmmalign.hard.sto

- One thing to notice here is that that hmmalign uses both lower case and upper case residues, and it uses two different characters for gaps. In a match column, residues are upper case, and a ’-’ character means a deletion relative to the consensus. In an insert column, residues are lower case, and a ’.’ is padding. While most software only accept input with upper case letter and “-” character as gaps. We thus use python to further modify output to transform all ‘.’ to ‘-’ and all amino acid letter to upper case.

- Upload python script “stoTransform.py” from OLAT to “Documents/Bio334\_Data/”

folder then run:

cd ~/Documents/Bio334\_Data/

conda activate ~/Documents/py38\_envs

conda install biopython

python stoTransform.py -i aligned.hmmalign.easy.sto -o aligned.hmmalign.easy.fa

python stoTransform.py -i aligned.hmmalign.hard.sto -o aligned.hmmalign.hard.fa

- as before, upload the alignments (“aligned.hmmalign.easy.fa” and “aligned.hmmalign.hard.fa”) that you just created to the Mview website, and create visual representations for them (save them accordingly).

**5) compare the alignments**

You should now have six different alignments: two each from the three different algorithms (one easy, one hard). Compare them side-by-side … are there differences? If so, which of these alignments looks ‘better’? What criteria might be useful for deciding that?

As expected, the ‘easy’ alignments overall appear to be more similar to each other. With one exception: the hmmalign may put our query protein in the first half of the alignment, whereas the others usually put it in the second half (!). Any idea what this might mean?