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 and via github.

- download the files "`unaligned1.fa`" and "`unaligned2.fa`" from OLAT and store them to your local disk (alternatively, you may also retrieve them from the github repository).
- 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/jdispatcher/msa/clustalo
- then on the Clustal Omega web page, click 'choose file' and choose the file '`unaligned1.fa`' from your computer.
- then click "submit" button at the end of the page
- after switching to the result page, click the "Download" button to download the aligned file and rename it 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 "Sent to MView" at the bottom. On the MView input form, under "Parameters", choose an "Alignment Width" of 120, then click "Submit" at the bottom. Finally click "Download" and rename the downloaded pure html 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.

- Let's install 'muscle' on your local computer:

  on Windows:

  ```
  curl -L -o muscle https://github.com/rcedgar/muscle/releases/download/v5.3/muscle-win64.v5.3.exe
  ```

  On Mac:

  download one of the two binaries below, depending on your CPU:

  ```
  curl -L -o muscle https://github.com/rcedgar/muscle/releases/download/v5.3/ muscle-osx-arm64.v5.3
  curl -L -o muscle https://github.com/rcedgar/muscle/releases/download/v5.3/ muscle-osx-x86.v5.3
  ```

  then, run this command to make the file executable:

  ```
  chmod +x 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/jdispatcher/msa/mview; then click "choose file" button to select the "`aligned.muscle.easy.fa`" file we just made; set the alignment width to 120, then click "Submit" at the bottom. Finally, "Download" the generated html file and

rename it 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.

- Let's install HMM-align on your local computer:

  on Windows:

  download the two files precompiled executable to a directory of your choice:

  ```
  curl -L -o hmmalign https://string-db.org/bio334/hmmalign_windows
  curl -L -o cygwin1.dll https://string-db.org/bio334/cygwin1.dll
  ```

  On Mac:

  choose one of the two binaries below, depending on your CPU:

  ```
  curl -L -o hmmalign https://string-db.org/bio334/hmmalign_x86_mac
  curl -L -o hmmalign https://string-db.org/bio334/hmmalign_arm64_mac
  ```

  then, run this command to make the file executable:

  ```
  chmod +x hmmalign
  ```

- **Optional** on Mac/Linux (but not on Windows) you can also choose to compile the source code by following commands in exact order (this will take some time):

  ```
  curl -OL  http://eddylab.org/software/hmmer/hmmer.tar.gz
  tar zxf hmmer.tar.gz
  cd hmmer-3.4
  ./configure -prefix=[your_directory_here]
  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 "Profile HMM", then towards the top of that section find the link to download the raw HMM file, then story it onto your laptop. Uncompress it with command "`gunzip PF07690.hmm.gz`" and give it the filename "`MFS_1.hmm`"
- now, we can use the hmmalign commend to produce the alignments (make sure you are in a directory where both the executable and the input files are stored:

  ```
  ./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. However, most software tools will

only accept input with upper'case letter and "-" character as gaps. We thus use python to further modify the output to transform all '.' to '-' and all amino acid letters to upper'case.

- Download the python script "`stoTransform.py`" either from OLAT or from github, and store it locally.
- the script requires the "biopython" module, so let's install that. Make sure your python environment is active, then use this command:

```
pip install biopython
```

- after that, we can run the stoTransform.py script:

```
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?