Practice 2: Search for homologous proteins using hidden markov models and the HMMer package

Theoretical concepts:

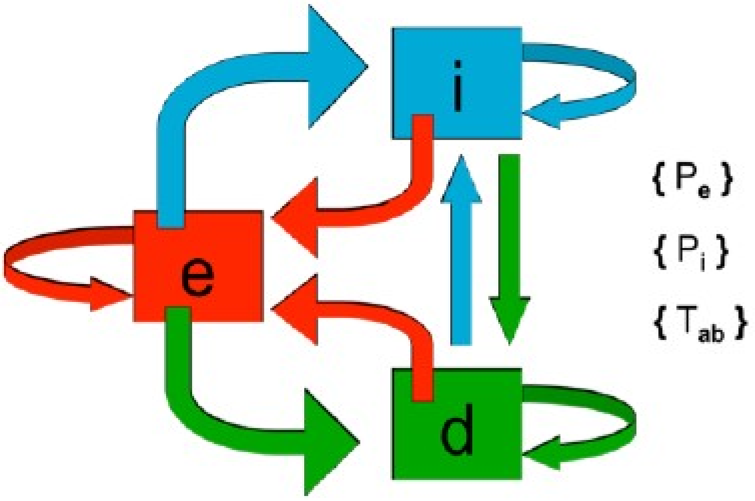
Hidden Markov Models (HMMs): In the previous practice we found similar sequences to our target protein using BLAST and PSI-BLAST. These programs perform and score alignments using substitution matrices as statistical models. In this practice we will identify similar sequences to our target protein using the programs from the HMMer package. The statistical models that these programs use are Hidden Markov Models (HMM).

Consider a HMM as a model that produces, one by one, amino acids. These amino acids are produced according to the probabilities contained in the HMM. At the end, the probability of a HMM producing a protein sequence is the product of all the individual probabilities of producing its amino acids in the correct order.

Besides, HMMs have different states. Each state produces amino acids with different frequencies. Then, HMMs change from one state to another while they are producing the protein sequence. In particular, we will work with HMM which contain the following states:

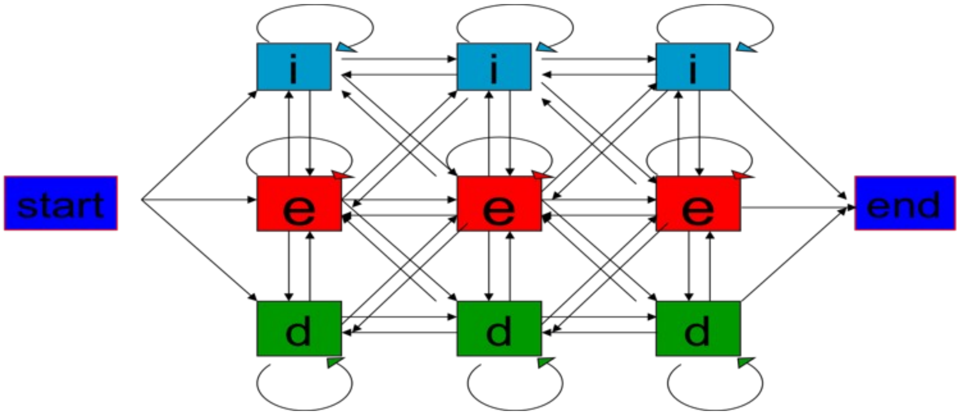
> Emission: conserved position of the sequence in reference with the ancestral sequences contained in the HMM.
>
> Insertion: insertion of an amino acid in the sequence in reference with the ancestral sequences contained in the HMM.
>
> Deletion: elimination of one amino acid in the sequence in reference with the ancestral sequences contained in the HMM.

The change from state to state is determined also by the probabilities contained in the HMM.



As you can see, only two of these states produce amino acids (emission and insertion) while the deletion state doesn't. The deletion state considers the probability of having a gap, in a position of the protein, in comparison with the ancestral proteins contained in the HMM.

Probabilities of amino acid emission, insertion and deletion; as well as the probabilities of changing from one state to another are specific for each position considered. So at the end, the production of a protein sequence can be simplified as a path across different states:



One single HMM can produce the same sequence following different paths. Each one of these paths has an associated probability. This probability is the product of all the probabilities of individual events that take place during the path (inclusion of amino acids and gaps in the sequence and changing from one state to another). Finally, the probability of a HMM producing one sequence is the addition of the probabilities of the different paths that take to that common output.

All the frequencies contained in a HMM are calculated from a Multiple Sequence Alignment (MSA). Therefore, all these frequencies are the representation of the evolutionary relations between the sequences contained in that MSA. We will start this tutorial by building a HMM.

Tutorial:

Step 1: Creating a HMM using hmmbuild

To generate a HMM of a particular family of sequences we need a previous alignment of these sequences. This MSA, named seed, will be turn into a HMM by using the program hmmbuild. Here is an example of HMM usage:

- **hmmbuild [model_HMM] [alignment]**

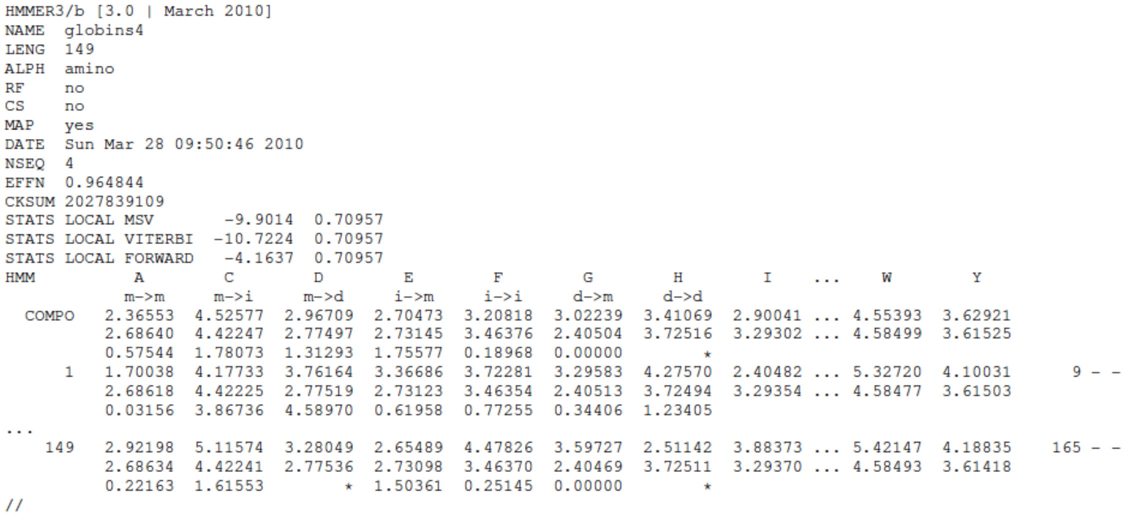
The alignment has to be in STOCKHOLM format, like **globins4.sto**. You will find the required files in the folder HMMER within the directory of exercise_2. We run this as an example:

```
hmmbuild  globins4.hmm  globins4.sto
```

The file globins4.sto contains an alignment with the STOCKHOLM format, from the PFAM database. This alignment contains many sequences of globin domains aligned. Therefore, the resulting HMM will be informative for the evolutionary relationships of the globin domain. HMMs, like the previous one, that are informative for specific protein domains can also be called **profiles**.

Now you can open the file globins.hmm to check each column ad row. Each position has the logarithm of the probability of emission of a residue. The Aa order is defined in two specific rows of the header (HMM). For each position we have probabilities on two different states (insertion and main), then we have a third row per position with the probabilities of transitions

MODEL PARAMETERS



```
HMMER3/b [3.0 | March 2010]
NAME  globins4
LENG  149
ALPH  amino
RF    no
CS    no
MAP   yes
DATE  Sun Mar 28 09:50:46 2010
NSEQ  4
EFFN  0.964844
CKSUM 2027839109
STATS LOCAL MSV       -9.9014  0.70957
STATS LOCAL VITERBI  -10.7224  0.70957
STATS LOCAL FORWARD   -4.1637  0.70957
HMM          A        C        D        E        F        G        H        I    ...    W        Y
            m->m     m->i     m->d     i->m     i->i     d->m     d->d
  COMPO   2.36553  4.52577  2.96709  2.70473  3.20818  3.02239  3.41069  2.90041 ... 4.55393  3.62921
          2.68640  4.42247  2.77497  2.73145  3.46376  2.40504  3.72516  3.29302 ... 4.58499  3.61525
          0.57544  1.78073  1.31293  1.75577  0.18968  0.00000        *
      1   1.70038  4.17733  3.76164  3.36686  3.72281  3.29583  4.27570  2.40482 ... 5.32720  4.10031      9 - -
          2.68618  4.42225  2.77519  2.73123  3.46354  2.40513  3.72494  3.29354 ... 4.58477  3.61503
          0.03156  3.86736  4.58970  0.61958  0.77255  0.34406  1.23405
...
    149   2.92198  5.11574  3.28049  2.65489  4.47826  3.59727  2.51142  3.88373 ... 5.42147  4.18835    165 - -
          2.68634  4.42241  2.77536  2.73098  3.46370  2.40469  3.72511  3.29370 ... 4.58493  3.61418
          0.22163  1.61553        *  1.50361  0.25145  0.00000        *
//
```

The columns for A C D etc. are the values to score the probabilities of the residues Ala, Cys, Asp etc. in the main state (first row) or insertion state (second row). The third row is for state transition scores (see section 5 of the UserGuide manual HMMER3.0). The COMPO row refers to average scores for all residues. Next two rows refer to the BEGIN state (emissions and transitions, respectively). The END state is defined by LENG, that indicates the last state of the model.

As we see, HMMs are composed by collections of position specific probabilities. Then, another way to understand HMM is to consider them as connected Position Specific Substitution Matrices (PSSMs), each one belonging to a different state, plus the probabilities of having deletions and the probabilities of changing from one state to another.

Step 2: Searching for similar sequences using hmmsearch

Sequence search with HMM is based on finding sequences that fit with that model. In other words, finding those sequences that would be generated by the HMM with high probability.

This can be done using **hmmsearch**. Here is a usage example:

- **hmmsearch (options) [model_HMM] [database] > [output]**

In our example, we can search for sequences with known structure fitting with the family of globins:

```
hmmsearch  globins4.hmm  ~/Documents/databases/pdb_seq
> globins_pdb.out
```

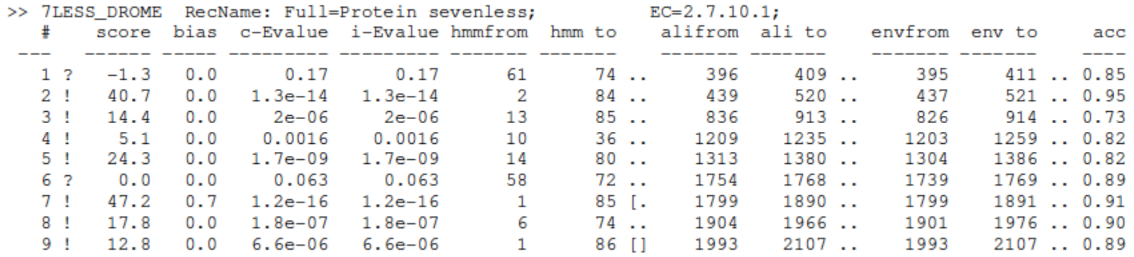
The results are sequences sorted by E-value. As in BLAST, this statistic informs of the probability of obtaining a match with that score just by chance.

Besides searching similar sequences in a database, we can also search a domain **within a single sequence** with hmmsearch. By doing this we will identify the regions within the sequence that fit in our HMM. Run the following commands:

```
hmmbuild  fn3.hmm  fn3.sto
```

```
hmmsearch  fn3.hmm  7LESS_DROME.fa  >  fn3.out
```

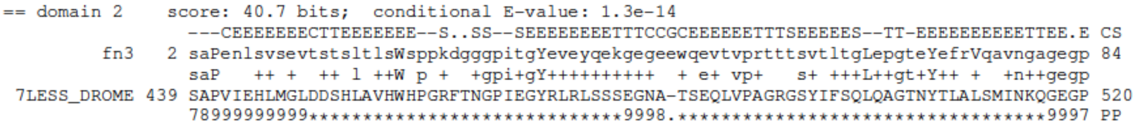
The file 7LES_DROME only contains one sequence; therefore only one E-value is retrieved. Nevertheless, the HMM profile finds several domains within this sequence. The output looks like this:



```
>> 7LESS_DROME  RecName: Full=Protein sevenless;          EC=2.7.10.1;
   #    score  bias  c-Evalue  i-Evalue hmmfrom  hmm to    alifrom  ali to    envfrom  env to     acc
 ---   ------ ----- --------- --------- ------- -------    ------- -------    ------- -------    ----
   1 ?   -1.3   0.0      0.17      0.17      61      74 ..     396     409 ..     395     411 .. 0.85
   2 !   40.7   0.0   1.3e-14   1.3e-14       2      84 ..     439     520 ..     437     521 .. 0.95
   3 !   14.4   0.0     2e-06     2e-06      13      85 ..     836     913 ..     826     914 .. 0.73
   4 !    5.1   0.0    0.0016    0.0016      10      36 ..    1209    1235 ..    1203    1259 .. 0.82
   5 !   24.3   0.0   1.7e-09   1.7e-09      14      80 ..    1313    1380 ..    1304    1386 .. 0.82
   6 ?    0.0   0.0     0.063     0.063      58      72 ..    1754    1768 ..    1739    1769 .. 0.89
   7 !   47.2   0.7   1.2e-16   1.2e-16       1      85 [.    1799    1890 ..    1799    1891 .. 0.91
   8 !   17.8   0.0   1.8e-07   1.8e-07       6      74 ..    1904    1966 ..    1901    1976 .. 0.90
   9 !   12.8   0.0   6.6e-06   6.6e-06       1      86 []    1993    2107 ..    1993    2107 .. 0.89
```

This shows the E-values of the domains: C-Evalue is the significance of matching the domains under the condition that the sequence is a member of the family defined by the HMM profile, I-Evalue is the independent E-value showing he significance of the domain within the whole set of sequences of the searched database (in this case there is only one sequence, so C-Evalue and I-Evalue coincide). The rest of columns indicate the residues for starting and ending the alignment of the profile and the sequence.

Finally, the alignment between sequence and profile is reported:



```
== domain 2    score: 40.7 bits;  conditional E-value: 1.3e-14
                  ---CEEEEEEECTTEEEEEEE--S..SS--SEEEEEEEETTTCCGCEEEEEEETTTSEEEEES--TT-EEEEEEEEEEETTEE.E CS
          fn3   2 saPenlsvsevtstsltlsWsppkdgggpitgYeveyqekgegeewqevtvprtttsvtltgLepgteYefrVqavngagegp 84
                  saP   ++ +  ++ l ++W p +  +gpi+gY++++++++++  + e+ vp+   s+ +++L++gt+Y++ +  +n++gegp
  7LESS_DROME 439 SAPVIEHLMGLDDSHLAVHWHPGRFTNGPIEGYRLRLSSSEGNA-TSEQLVPAGRGSYIFSQLQAGTNYTLALSMINKQGEGP 520
                  7899999999******************************9998.**********************************9997 PP
```

The alignment shows the secondary structure of the profile, the model (fn3) and the sequence aligned (7LESS_DROME) with the matches between both, as in PSI-BLAST. The bottom line shows the % of expected accuracy of the match (7 is 65-75%, 8 is 75-85%, 9 is 85-95%, * is 95-100%)

Step 3: Using HMM databases: hmmpress and hmmscan

Consider that we want to perform a search for similar sequences to a target sequence using hmmsearch. Then we need to get one HMM that fits with this target sequence, and then use it to perform the search. Many HMM are available in several databases in the internet, but which one should we choose? Having a target sequence, can we have a method to assign the best HMM? Yes, this method is **hmmscan**.

To learn how it works, we will start by creating a database of profiles. How can this database be generated? The initial MSAs of the families are required. These are named seed-MSAs. As an example, we use the alignments of fn3.sto (transcription factor FN3) and pkinase.sto (one family of kinases) as seeds. Create the HMMs for these alignments running:

```
hmmbuild  fn3.hmm  fn3.sto
```

```
hmmbuild  Pkinase.hmm  Pkinase.sto
```

Then, concatenate all the generated HMMs in one file:

```
cat  globins4.hmm  fn3.hmm  Pkinase.hmm  >  minifam
```

In order to check sequences and profiles very fast, we compress and index the database file using **hmmpress**. Here is a usage example:

- **hmmpress [database]**

We run then:

```
hmmpress  minifam
```

Now we can search what is the best profile for a given target sequence using the command hmmscan. Here is a usage example:

- **hmmscan (options) [Database_HMM] [sequence] > [output]**

For example we can use the sequence of 7LES_DROME to search for the best profile in the database previously generated. Run:

```
hmmscan  minifam  7LESS_DROME.fa  >  7LESS_DROME_minifam.out
```

When looking at the output we can see that the target sequence may fit with more

than one profile and more than once with the same profile. This is because sequences can have more than one domain, and in fact each profile is considered a domain, formed by similar sequences.

Step 4: Performing multiple sequence alignments using hmmalign

In addition, the alignment of several sequences can also be done using the HMM profile. This represents a relevant improvement with respect to the alignment of sequences, such as clustalw, for two reasons: 1) the control of gaps and substitutions is referred to the initial seed alignment, and this can be guided by the user (i.e. using structural and/or functional knowledge about the family); and 2) it is faster than any agglomerative approach (i.e. clustalw), as it aligns all sequences with the profile at the same time. The program is **hmmalign**, here is a usage example:

- **hmmalign [model_HMM] [file_with_sequences] > [output]**

We can show this with the file globins45.fa. Run the following commands and test the speed of both approaches, hmmalign and clustalw:

```
hmmalign  globins4.hmm  globins45.fa  >  globins45_hmm.sto
```

```
clustalw  globins45.fa
```

Programs of the HMMer package work with alignments in Stockholm format. If you want to transform this format into clustalw format and back, you can use the script **“aconvertMod2.pl”**, it allows us to change several alignment formats. The format of the globins45_hmm.aln file contains additional rows showing the accuracy of the alignment. This can be transformed into clustalw format running the script:

```
perl  ~/Documents/perl-scripts/aconvertMod2.pl  –in  h  –out  c
<globins45_hmm.sto>globins45_hmm.clu
```

Step 5: Looking for homolog sequences using phmmer and jackhmmer

A straightforward way to find homolog sequences in a database is to use **phmmer** and **jackhmmer**. These are the equivalents to BLAST and PSI-BLAST (respectively) in the HMMER3 package. While PSI-BLAST is the iterative version of BLAST, jackhmmer is the **iterative** version of phmmer.

The commands to run these programs are:

- **phmmer [target_sequence] [database_of_sequences] > [output]**
- **jackhmmer [target_sequence] [database_of_sequences] > [output]**

The output has the same format as for hmmsearch, and the input are sequences with FASTA format. The difference between phmmer and jackhmmer is that jackhmmer perform several iterations (up to a maximum of five by default, which can be changed using the –N flag) and it generates internally a HMMER profile at each iteration, while phmmer is like jackhmmer at iteraton 1. Iterations are shown in the output entitled by "Round" (search this word in the file to compare the results at different iterations). The internal HMMER profile is used for the search (like with hmmsearch) and the profile is generated with the first sequences matched with the target and taken from the database. We can show this with the file globins45.fa as database, and a globin sequence (i.e. hbb_human) as target sequence.

```
jackhmmer  hbb_human.fa  globins45.fa  >  globins_jackhmmer.out
```

```
phmmer  hbb_human.fa  globins45.fa  >  globins_phmmer.out
```

The problem of using jackhmmer is that if the database of sequences where the search is performed is small or redundant, the profiles are biased (i.e. searching in "pdb_seq"). We already tackled this problem in "TUTORIAL Step 2" of practice 2.1. Do you remember how do we solve this problem?

Step 6: Using HMMs from the PFAM database

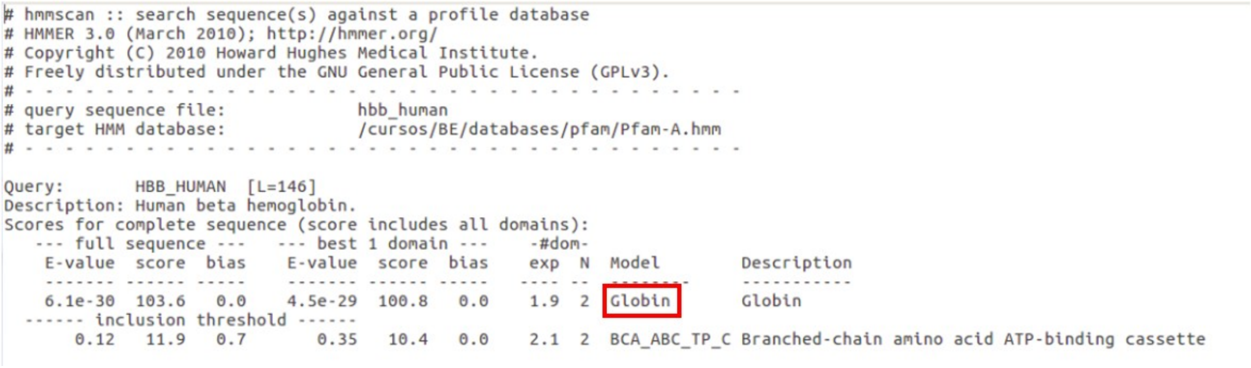
PFAM is a database that contains the profiles of several known families of sequences (see BOX, at the end of the tutorial). A good strategy would be to find the best HMMER profile in PFAM for our target sequence. We will select the best profile of your choice and fetch it from the database. Then, we will run a search of sequences in pdb_seq using this profile. These options are done with the commands **hmmscan** (to scan in PFAM) and **hmmfetch** (to fetch a profile from PFAM).

Step 6.1) Assign the best profile(s) to the target sequence (i.e. hbb_human) using **hmmscan**:

```
hmmscan  ~/Documents/databases/Pfam-A.hmm  hbb_human.fa

>  hb_human_db.out
```

Then, we have to inspect the output of hmmscan to find which HMM from PFAM fits the best with our target sequence. Once we have find it, we must copy the name that this HMM has in PFAM. We will use this name to fetch the HMM from PFAM using the hmmfetch program.



```
# hmmscan :: search sequence(s) against a profile database
# HMMER 3.0 (March 2010); http://hmmer.org/
# Copyright (C) 2010 Howard Hughes Medical Institute.
# Freely distributed under the GNU General Public License (GPLv3).
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# query sequence file:             hbb_human
# target HMM database:             /cursos/BE/databases/pfam/Pfam-A.hmm
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

Query:       HBB_HUMAN  [L=146]
Description: Human beta hemoglobin.
Scores for complete sequence (score includes all domains):
   --- full sequence ---   --- best 1 domain ---    -#dom-
    E-value score   bias    E-value  score  bias    exp  N  Model        Description
    ------- ------  -----   -------  ------ -----  ---- --  --------     -----------
    6.1e-30  103.6   0.0    4.5e-29  100.8   0.0    1.9  2  Globin       Globin
  ------ inclusion threshold ------
       0.12   11.9   0.7       0.35   10.4   0.0    2.1  2  BCA_ABC_TP_C Branched-chain amino acid ATP-binding cassette
```

Next, we need to know the HMM of the profile(s) assigned to our target sequence. They can be extracted from the PFAM database using the program **hmmfetch**, here is a usage example:

- **hmmfetch [database_HMM] [name_HMM] > [file_HMM]**

Therefore, in our example, assuming we have found a domain_target:

Step 6.2) extract the profile(s) from PFAM that correspond to the domains of the target sequence which are found in the column indicated as “model” (see example in step 3 of this tutorial). Let’s assume the name of the model we have found for hbb_human is “domain_hbb”, then we execute the command:

```
hmmfetch  ~/Documents/databases/Pfam-A.hmm  “domain_hbb”
>  domain_hbb.hmm
```

Step 6.3) Search for sequences with known structure that contain the same domain as our target using **hmmsearch**:

```
hmmsearch  domain_hbb.hmm  ~/Documents/databases/pdb_seq
>  hbb_pdb_by_HMM.out
```

Step 7: Some format changes

Additionally, we have seen how to use aconvertMod2.pl to transform a STOCHOLM format in CLUSTALW. We also need other programs and scripts to transform some of these formats:

1) Transforming a CLUSTALW format of a MSA (Multiple Sequence Alignment) in a STOCKHOLM format. To do this transformation first we need to transform into a FASTA format the alignment using aconvertMod2.pl:

```
perl  ~/Documents/perl_scripts/aconvertMod2.pl  –in  c  –out  f
<alignment.aln>alignment.fa
```

Then we transform the FASTA alignment to STOCKHOLM:

```
perl  ~/Documents/perl_scripts/fasta2sto.pl  alignment.fa  >  alignment.sto
```

We can also transform a STOCKHOLM format MSA file in a FASTA MSA file:

```
perl  ~/Documents/perl_scripts/sto2fasta.pl  –g  alignment.sto  >  alignment.fa
```

2) Transform HMMER3.0 format files in HMMER2.0 format files (old) and ASCII to binary and viceversa.

```
hmmconvert [options] [name_HMM] > [name2_HMM]
```

3) Or generate a set of sequences derived from a profile

```
hmmemit [options] –N #number [name_HMM] –o [output]
```

Where the file output contains the generated sequences in FASTA format and #number is the number of sequences to be generated

QUESTIONS FROM THE TUTORIAL

1) Compare the results of phmmer, jackhmmer with the results of hmmsearch using “domain_hbb.hmm” (see hbb_pdb_by_HMM.out) when searching homologs in pdb_seq for hbb_human.

2) If a protein sequence has more than one domain in PFAM, do you think the result of using hmmsearch and jackhmmer will be the same? Why? Test the example with 7LES_DROME in SwissProt.

3) In practice 2.1 we used PSI-BLAST to fish sequences in the database uniprot_sprot.fasta and generate a PSSM profile which was used for searching homologs in PDB. Check the manual of HMMER3.0 and create your own protocol in which you use the program jackhmmer in a similar approach: use SwissProt database to generate the HMM profile and perform the search in pdb_seq.

4) Use hmmscan to search the best model(s) for 7LES_DROME in PFAM and search the homologs in PDB with this/these model(s). Compare the results of this search with the results of your protocol search in question 3. What are the differences? Why?

5) Use your protocol described in question 3 to search homologs of 7LES_DROME in PDB and compare with the results of the protocol described in practice 2.1 when using PSI-BLAST.

6) Use the sequence target.fa from practice 2.1. Apply phmmer, jackhmmer and the protocols of questions 3 and 4 to find homologs in PDB. What’s the fold of this sequence? Compare the result with the homologs found in practice 2.1

7) Use hmmalign and FetchFasta.pl to align the sequence of target.fa and its homologs of PDB

8) If you have to align the sequence 7LES_DROME and its homologs of PDB what’s the best model to use? Produce the alignment with the models from question 4 and your protocol in question 3 to show your answer.

9) What are the folds of the following sequences?

   a. *problem1/serc_myctu.fa*

   b. *problem2/p72_mycmy.fa*

   c. *problem3/lip_staau.fa*

   d. *problem4/orc1_human.fa*