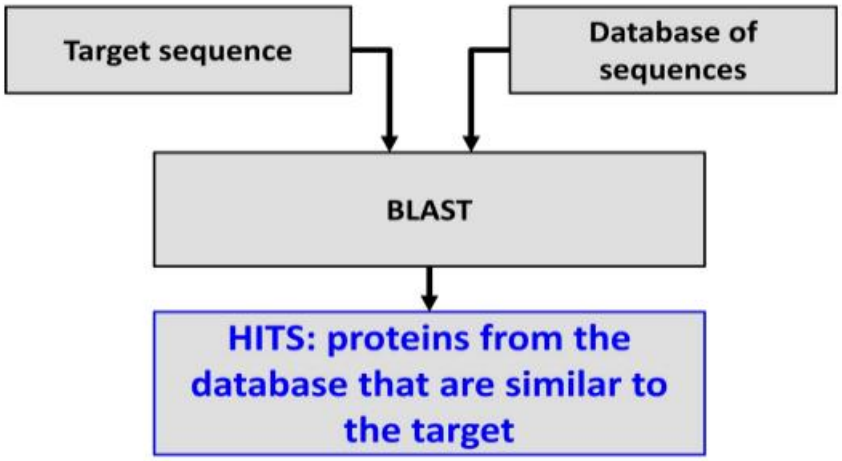
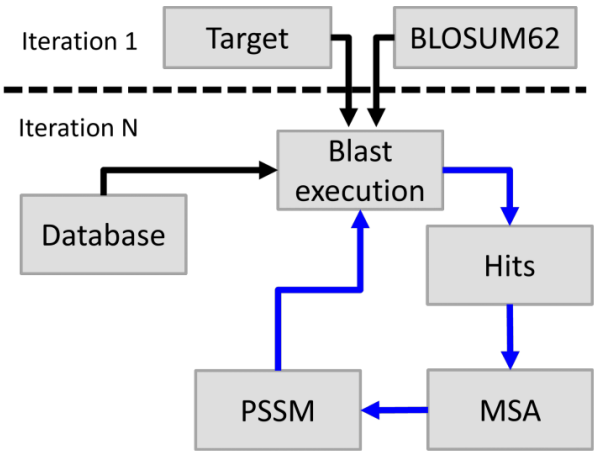
BLAST (basic local alignment search tool)



High score = two proteins have similar sequences

E-value = the smaller the best

Substitution matrix: contain scores associated to the frequency of substitution between two amino acids. They are obtained from multiple sequence alignments (MSA) and contain the information regarding residue substitution and conservation of the MSA.



STRUCTURE OF A BLAST COMMAND

blastp -query [target_fasta_format]-db [database]-out [output]

[target_fasta_format] = File that you have and want to look for proteins structural similar.

[database] = the database you will use (can be a direct path)

[output] = file created with the outputs

To improve our search, we use psi-blast to create a MSA. It looks for sequences similar to the target sequence, it aligns them into a MSA and with this it recalculates a new substitution matrix. The matrix is stored and used to add more similar sequences.

PSIBLAST starts with a target sequence and searches for similar sequences in a database. In each iteration, sequences found in the previous round are aligned, and a new PSSM is calculated based on the updated alignment. PSIBLAST then uses this new PSSM to refine the search in the next iteration.

Iteration 0 → Target + Blosum62 → [similar sequences] 0 → MSA 0 + Target → PSSM 0

Iteration 1 → PSSM 0 → [similar sequences] 1 → MSA 1 + Target → PSSM 1

Iteration 2 → PSSM 1 → [similar sequences] 2 → MSA 2 + Target → PSSM 2

PSIBLAST CODE

Same structure than blast but we just add the number of iterations we want.

```
psiblast -query target.fa -db /mnt/NFS_UPF/soft/databases/blastdat/pdb_seq
-num_iterations 5 -out target_pdb_5.out
```

In order to our PSSM to not be biased because PDB is such a small database we will use Uniprot or SwissProt as they have a wide spectrum of protein sequences for a lot of different species. Also, they are much bigger than the PDB. The PSSMs built using these databases will represent the evolutionary information of the target protein family with increased accuracy, and this will improve the results of our sequence search.

```
psiblast -query target.fa -num_iterations 5
-out_pssm target_sprot5.pssm -out target_sprot_5.out
-db /mnt/NFS_UPF/soft/databases/blastdat/uniprot_sprot.fasta
```

-out_pssm target_sprot5.pssm: Specifies the output file where the resulting PSSM will be stored after the five iterations.

-out target_sprot_5.out: Specifies the output file where the results of the PSIBLAST search will be stored. This file will likely contain information about the sequences found in each iteration.

```
psiblast -db /mnt/NFS_UPF/soft/databases/blastdat/pdb_seq -in_pssm
target_sprot5.pssm -out target_pdb_sprot5.out
```

Now we won't do iterations in the PDB database so it won't be biased.

PERL/CLUSTALW

```
perl /mnt/NFS_UPF/soft/perl-lib/FetchFasta.pl -i file.list
-d /mnt/NFS_UPF/soft/databases/blastdat/uniprot_sprot.fasta -o file.fasta
```

Now we will get the fasta sequences for a set of selected PSI-BLAST outputs. We use the program FetchFasta.pl. It takes as input a list file **(the file needs to have the extension .list)** that you generate by copying and pasting several lines of an output from a PSI-BLAST search on SwissProt (target_sprot_5.out) and fetches FASTA sequences from a database based on a list of identifiers.

-d /mnt/NFS_UPF/soft/databases/blastdat/uniprot_sprot.fasta: Path to the Uniprot SwissProt database in FASTA format. Then it will fetch sequences based on the identifiers from this database.

-o file.fasta: Output file where the fetched sequences will be stored. Each sequence in the output file corresponds to an identifier from the input list.

Now we want the alignment between these sequences and also the target sequence, we will put it all together in a file, and do a clustalw.

```
cat target.fa > pssm.fasta
```

```
cat file.fasta >> pssm.fasta
```

```
clustalw2 pssm.fasta
```

From clustalw to fasta format

```
perl /mnt/NFS_UPF/soft/perl-lib/aconvertMod2.pl -in c -out f
<pssm.aln>pssm.fa
```

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.

hmmbuild [model_HMM] [alignment]

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

For alignments we will always use Stockholm format.

globins4.hmm: This is the output file name where the generated HMM will be saved.

We can open the file globins4.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.

Sequence search with HMM is based on finding sequences that fit with the model (finding those sequences that would be generated by the HMM with high probability). This can be done using hmmsearch

hmmsearch [model_HMM] [database] > [output]

```
hmmsearch  globins4.hmm  /mnt/NFS_UPF/soft/databases/blastdat/pdb_seq
                                  >  globins_pdb.out
```

Our outputs will also be classified with e-values.

CONCATANATE HMM

cat globins4.hmm fn3.hmm Pkinase.hmm > minifam (output file)

To check sequences and profiles very fast, we compress and index the database file using **hmmpress**

hmmpress minifam

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

Performing multiple sequence alignments using hmmalign

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

```
clustalw globins45.fa
```

As hmm works with Stockholm extension and clustalw has it's own we have a code in order to pass from one format to another.

```
perl /mnt/NFS_UPF/soft/perl-lib/aconvertMod2.pl -in h -out c
<globins45_hmm.sto>globins45_hmm.clu
```

HOMOLEG SEQUENCES WITH HMM

phmmer [target_sequence] [database_of_sequences] > [output]

jackhmmer [target_sequence] [database_of_sequences] > [output]

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 iteration 1.

```
hmmscan /mnt/NFS_UPF/soft/databases/pfam-3/Pfam-A.hmm hbb_human.fa
> hb_human_db.out
```

Assign the best profile(s) to the target sequence (hbb_human) using hmmscan.

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.

hmmfetch [database_HMM] [name_HMM] > [file_HMM]

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

```
hmmsearch domain_hbb.hmm /mnt/NFS_UPF/soft/databases/blastdat/pdb_seq
> hbb_pdb_by_HMM.out
```

PYMOL COMMANDS

Fetch : introduce a structure

Remove res hoh: take out water molecules

Super structure1, structure2, object = superimposition object : Superimpose two structures

To save the sequences of the alignment we can use the command**: save filename.aln superimpose object**

To superimpose only a specific part of a sequence

```
align 5t0u and resi 405-460, 5yel and resi 405-460
```

From aln to sto

```
perl /shared/PERL/aconvertMod2.pl -in c -out f
<aln4_corrected.aln>aln4_corrected.fa
```

```
perl /shared/PERL/fasta2sto.pl aln4_corrected.fa > aln4_corrected.sto
```