

# Practice with basic bioinformatics

## Introduction

In order to get some hands-on experience with common practices in bioinformatics, we will work through a guided example in class today. We have three goals for today:

1. Cover the basics of how to use the **Muscle** aligner and **hmmer3**.
2. Discuss some simple principles of how the quality of sequence matches are quantified.
3. Work through as many challenges as we have time.

## Basics of Muscle

Recall that **Muscle** is designed to generate multiple sequence alignments. These alignments attempt to identify and “line up” conserved residues in sequences. Once an alignment is generated it can be used to infer evolutionary relationships as the basis of a molecular-based phylogenetic tree or create profile hidden markov models (HMMs).

In **bash** call the **Muscle** function with the **-h** flag. Based on this help page, what is the basic syntax for generating a sequence alignment? What is the default format for an input sequence file?

## Basics of hmmer

Remember that **hmmer3** is used to find sequences that match a particular pattern of interest. We use **hmmer3** to describe a pattern of interest with a profile HMM. A common application of **hmmer3** is to generate a profile HMM for a particular protein family, domain, or active site sequence. We'll practice that very skill today using bacterial proteomes and reference sequences from a database called *pfam* (<http://pfam.xfam.org/>). The two key **hmmer3** tools we'll need are **hmmbuild** (to build a profile HMM from a sequence alignment) and **hmmsearch** (to search a sequence database with a profile HMM).

In **bash** call **hmmbuild** and **hmmsearch** with the **-h** flag. Based on those help pages, what is the basic syntax for generating a profile HMM and searching a database with a profile HMM?

## Interpreting results

Commonly, any sequence comparison or sequence database searching tool will return values that describe the quality of the match between two sequences or between a profile HMM and a sequence. Two very common values are an **e-value** and a **bit score**. Both of these are based on an **alignment score**.

1. **alignment score** is a value that describes the quality of a pairwise or multi-sequence alignment. An exact match at a position in the alignment adds to the score and mismatches deduct from the score. Often some substitutions incur larger deductions than others based on biological principles. In addition, deductions or penalties are often included for insertion of gaps. The score for the whole alignment is the sum of the additions or deductions at each position.

2. **E-value** is a parameter that describes the number of hits one can expect to see by chance when searching a database of a particular size. It decreases exponentially as the *alignment score* of the match increases. This means that the *E-value* is sensitive to the size of the database, the score of the alignment, and the length of the alignment. The closer the E-value is to zero, the more “significant” the match is. Keep in mind that even identical short alignments have relatively high E-values because shorter sequences have a higher probability of occurring in the database purely by chance.
3. **bit score** is the required size of a database in which the current match could be found just by chance. It is presented on a  $\log_2$  scale. This means each increase by one doubles the required database size. This has become a more popular metric as it does not depend on database size, like *E-value*.

*What is the E-value or bit score that indicates a “true” match?*

## Processing tabular information

**bash:** Two bash tools are useful for editing tabular files.

1. **grep** is useful for getting rid of header lines. Nice bioinformatics tools will indicate header lines with some special character. For example, **hmmsearch** header lines have ‘#’ at the beginning. Remember the **-v** argument will return all lines that DO NOT match your search string.
2. **cut** can be used to capture specific columns from tabular data. Remember it takes a **-d** argument to set the delimiter between columns and a **-f** to specify which columns to return.

*How could we reduce tabular output to the 1st, 4th, and 8th column of a tabular text file returned by **hmmsearch**?*

*How could we find information contained in one table that is associated with sequence identifiers that were identified as sequences of interest by **hmmsearch**?*

## Challenges

On Stuart’s github, there is a repository containing a fasta file with reference sequences for a domain of a gene encoding a gene expression regulator called sigma70 (sigma70.ref) and a file containing sequences for all of the proteins encoded in a bacteria’s genome (Roseobacter.fasta). Fork and clone this repository to obtain these files and work through the challenges below.

- 1) Align the sequences in sigma70.ref using **muscle**. How does the alignment file differ from the original file?
- 2) Build a profile HMM using the sigma70 sequence alignment and **hmmbuild**.
- 3) Search the Roseobacter proteome (Roseobacter.fasta) for sigma factors using **hmmsearch**. How many matches do you get?

*I suggest using the argument for tabular output, as this will be useful in other steps.*