**Practical 3: Phylogenetic reconstruction**

Group no. 11

Fuqi Xu, Shuhan Xu, Milda Valiukonytė

**Exercise 1.**

1.

1.1

–in name of the input file

-dbtype database type (nucl - nucleotide)

1.2

Ran the program makeblastdb for each FASTA file.

2. Gathered all the genomes in one file called genomes\_all and made a database out of it.

3.

3.1 The best hits were found and put in a FASTA file using blastResultParser.py

3.2 ?

3.2.1

–outfmt Output format:

(5 = BLAST XML)

-query Name of the query file

-db Name of the database file

-out Name of the output file

3.2.2 The output is found within the directory the command was run in. In the output you can see hits and for each hit the high scoring pairs (hsps) aligned with the query sequence, as well as various parameters for each of the alignments such as identities, gaps, etc.

**Exercise 2.**

**B.**

a. It takes the first object from the list of high scoring pairs (hsps) of a single alignment.

print alignment.hsps [0].sbjct

b. BLAST record corresponds to results of a BLAST run for a single query sequence. If multiple query sequences are used, the output file will contain multiple records. In this case, only one query sequence was used (16S rRNA), hence the output file will be parsed into a single BLAST record object. A BLAST record object contains all the information of a BLAST run, including information about the program, query sequence, database, as well as the alignments.

c. The script assumes the output file contains multiple BLAST records, hence it uses *parse* function (instead of *read*) and then iterates over the list of records.

d. It outputs the FASTA file with the best hits from the run.

**Exercise 3.**

1.

1.1 Gap penalties are methods to score a multiple sequence alignment. When sequences are aligned, gaps must be introduced to maximize the similarity between them. Each time a gap is introduced, the alignment score is lowered by subtracting a penalty from it, and when it is extended, a smaller penalty is subtracted (gap extension penalty).

1.2 ?

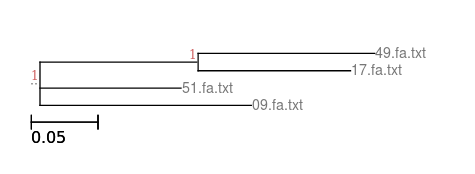
**Exercise 4.**

1.

1.1 Scoredist distance correction, Jukes-Cantor distance correction, Kimura distance correction, Storm & Sonnhammer distance correction.

1.2 Storm & Sonnhammer distance correction method makes the tree more compact whereas Kimura distance correction makes it more spaced out. UPGMA tree has a root whereas neighbor-joining based tree does not have a root.

2. The most likely tree:

****

2.3

-f An option which determines the algorithm used by the program (-f a is the rapid Bootstrap analysis and search for best-scoring ML tree in one program run)

-x An option which specifies the random seed number and turns on rapid bootstrapping

-N

-T Specifies the number of threads to run

-p Specifies a random number seed

-m ?

-s Name of the alignment data file

-n Name of the output file

2.4

Distance based methods derive the distance measure for each pair of aligned sequences and the maximum likelihood methods estimate the likelihood of each tree topology and reports the one with the maximum likelihood.

Distance-based

Advantages: Suitable for large datasets as only a single tree is constructed

Disadvantages:

Maximum likelihood

Advantages: Evaluates all possible tree topologies and gives the one with the highest probability.

Disadvantages: Computationally demanding and cannot analyze large datasets

**Exercise 5.**

1. Bootstrapping is a method to evaluate the uncertainty of the result when one cannot perform a repeated measurement. It uses the original dataset with N data points and picks another N datapoints from it, which is then called a bootstrap replicate (might be the same points – sampling with replacement). The replicate dataset is then analyzed as the original dataset. Multiple replicate datasets can be created this way and analyzed. In the case of sequences, new sequences are generated from the original ones and then branch lengths are calculated.

2.1 It specifies the number of bootstrap samples.

2.2 ?