ITX Integrative Transcriptomics
Prof. K. Nieselt,
Institute for Bioinformatics and Medical Informatics Tübingen
Prof. S. Nahnsen,
Institute for Bioinformatics and Medical Informatics Tübingen

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Lecture: Grundlagen der Bioinformatik — SoSe 2022

Assignment 1 (20 points)

Hand out: Thursday, April 28
Hand in due: Thursday, May 5, 18:00
Direct inquiries via the ILIAS forum or to your respective tutor at:
Mathias Witte Paz: iizwi01@uni-tuebingen.de
theresa-anisja.harbig@uni-tuebingen.de
meret.haeusler@student.uni-tuebingen.de
jules.kreuer@student.uni-tuebingen.de
simon.heumos@qbic.uni-tuebingen.de

Theoretical Assignments

1. **Scoring matrices for nucleotide sequence alignments** (4P)

   (a) Find out which scoring matrices for the nucleotide alphabet $\Sigma = \{A, G, C, T\}$ are used by the well-established tool `BLASTn` and which do not use a match score of 1. What is the absolute ratio between the match/mismatch scores?

   (b) Scoring matrices $S(a, b)$ with $a, b \in \Sigma$ have to fulfil the following condition, with p_a and p_b being the probability of appearing for nucleotides a, b respectively:

   $$\sum_{a,b \in \Sigma} p_a p_b S(a, b) < 0$$

   How does a unitary scoring matrix $S(a, b)$ (i.e., a common mismatch and a common match score) for nucleotide sequences have to look in order to assure the above inequality? Assume an equal probability p_a and p_b for all $a, b \in \{A, G, C, T\}$.
   Further, let $k = S(a, b)$ for $a \neq b$ and $m = S(a, a)$. Derive an equation that relates k with m. Give a concrete example for a feasible nucleotide scoring matrix.

Practical Assignments

For the practical assignments you should keep a good structure in your code, e.g. implement functions that solve the sub-tasks presented. All functions should be called within the function `main()` for the program to run without the need of any further modification. For this task, you may use the provided template that includes the *Argument Parser* of `Python`. This facilitates the parsing of the files, since their paths will not be hard-coded.

You can hand-in one single file `Python`-file that solves tasks 2 and 3.

2. Reading and Writing Sequences in FASTA Format (8P)

Implement a Python program that:

- reads sequences from a single or multi FASTA file,
- outputs the length of the read sequences to the console after reading,
- and also allows the user to write the sequences (with the original headers) to a new FASTA file, e.g. after certain modifications.

Your program should handle the sequence data in a way that allows for subsequent processing and saves also other information, e.g. the header of the sequence. For this task, you are not allowed to use the FASTA-reader from any established library, e.g. `BioPython`.

Use the data provided in the `material-A1.zip` file. Furthermore, your program should include a function that:

(a) Writes the **reverse complement** of the sequences in the file 'MultipleSeqs.fasta' in reverse order (i.e., last sequence becomes first sequence, and so on) in FASTA format.

3. Substitution matrix computation (8P)

Write a program that computes the substitution matrix from a multiple sequence alignment (MSA) given as a FASTA file. In more detail, your program should expect a FASTA file containing multiple sequences of the same length (i.e. they have been previously aligned) and then proceed as explained in the lecture. If you had difficulties to solve task 2, you may use the FASTA reader provided by the library `BioPython`.

Apply your program to the provided input file called `msa-scoring-matrix.fasta` from the file `material-A1.zip`. The resulting matrix should be printed in the console with an appropriate style and exported to a txt-file.

Please read the questions carefully. If there are any questions, you may ask them during the tutorial session or in the forum of ILIAS. You will usually get an answer in time, but late e-mails (e.g. the evening of the hand-in) might not be answered in time. Please upload all your solutions to ILIAS. Don't forget to put your names on every sheet **and** in your source code files. Please pack both your source code as well as the theoretical part into one single archive file and give it a name using this scheme: `<name1>_<name2>_<Assignment>_<#>.zip`. The program should run without any modification needed.