CSE 5370: Bioinformatics Homework-2

Tulasi Sridevi Navuluru

Fall 2022

Collaboration Statement

I have collaborated with Vijitha Kotapati (UTAID: 1001860730) and Divya Boggavarapu (UTAID: 1002086719) to work on the problems 2, 3 and 4 by understanding the concepts of Substitution Matrix, Global Alignment and Local Alignment.

1 Substitution Matrices

Suppose that transition mutations (A $\longleftrightarrow$ G and T $\longleftrightarrow$ C) are less common than tranversions (A $\longleftrightarrow$ T, A $\longleftrightarrow$ C, G $\longleftrightarrow$ T, and G $\longleftrightarrow$ C). The following is the substitution matrix that reflects this:

	A	G	T	C
A	1	-5	-1	-1
G	-5	1	-1	-1
T	-1	-1	1	-5
C	-1	-1	-5	1

2 Global Alignment

We have conducted global alignment with the Needleman-Wunsch algorithm and implemented it in a single python file "**1002010740_NW.py**"

The global alignment function in the file will take in sequence A and sequence B as strings to be aligned (assume that these strings only contain the chars "acgt" for problem 2), a gap penalty, and a substitution matrix and returns a list of tuples representing possible alignments.

2.1 Example

In **1002010740_NW.py**, we executed the global alignment function with input strings "GATA" and "CTAC", substitution matrix d and gap penalty as parameters which returned the tuples ("GATA", "-CTAC"),("GATA-","C-TAC").

3 Local Alignment

We conducted local alignment with the Smith-Waterman algorithm and implemented it in a single python file called **"1002010740_SW.py"**.

The local_alignment function in the file will take in sequence A and sequence B as strings to be aligned (assume that these strings only contain the chars "acgt" for problem 2), a gap penalty, and a substitution matrix and returns a list of tuples representing possible alignments.

4 A Custom Alignment

- I have taken my first name **tulasi**, last name **navuluru** in lowercase and concatenated them to be **'tulasinavuluru'**.
- I have written code to create a custom substitution dict to substitute all the 26 English alphabets as characters and included code for this problem in the file **"1002010740_CUSTOM.py"**

  The file **"1002010740_S.txt"** will provide the output of my pretty matrix.
- I ran "local_alignment" function from Problem 3 with the custom S defined by my name, a gap penalty of -2, my concatenated name (i.e. "tulasinavuluru") as the first string, and the pangram "thequickbrown- foxjumpsoverthelazydog" as the second string. Following are the output tuples:
  [('tulasinavuluru', 'thequickbrownf'), ('__avuluru', 'thequicfoxjump'), ('_lasinavuluru', 'thmpsoverthela')]
- The file **"1002010740_D.txt"** will provide the output of my pretty matrix.

5 Difficulty Adjustment

Please find my answers to the questions asked as below:

- How long did this assignment take you to complete?
  - **I completed my assignment in 15 hours.**
- If the assignment took you longer than the 10 hours, which parts were overly difficult?
  - **The assignment was challenging. I learnt new concepts like Needleman-Wunsch algorithm, Smith-Waterman algorithm in depth while working on this assignment.**