CMPE 549: Bioinformatics, Fall 2019
Assignment 1 - Phylogenetic Tree Construction using Sequence-Based Distances
Due: 19/11/2018, 23:55

A. Implement the Needleman-Wunsch dynamic programming *global sequence alignment* algorithm as a function. It should take two protein sequences as input, and should output the maximum alignment score of these sequences, as well as the alignment achieving this maximum score. If there are multiple optimal alignments, you may print one of them. Use the BLOSUM62 scoring matrix (available at `BLOSUM62`).

B. Implement the Smith-Waterman dynamic programming *local sequence alignment* algorithm as a function. It should take two protein sequences as input, and should output the maximum alignment score of these sequences, as well as the alignment achieving this maximum score. If there are multiple optimal alignments, you may print one of them. Use the BLOSUM62 scoring matrix (available at `BLOSUM62`).

C. Construct two phylogenetic trees for the organisms given in the *organisms.txt* file. The first tree should be constructed using the scores generated by the Needleman-Wunsch algorithm and the second one should be constructed using the scores generated by the Smith-Waterman algorithm. The *organisms.txt* file contains the names, GenBank IDs, and the protein sequences corresponding to the *COX3* gene of 15 different organisms in a Python dictionary format. In order to create a dendrogram you can use the following Python code:

```
from scipy.cluster.hierarchy import linkage, dendrogram

def generate_phylogenetic_tree(organisms, distance_matrix, algorithm):
    average = linkage(distance_matrix, ”average”)
    dendrogram(average,
               labels=list(organisms.keys()),
               orientation=”left”,
               leaf_font_size=10)
    pylab.subplots_adjust(bottom=0.1, left=0.2,
                          right=1.0, top=1.0)
    # Save figure as pylab.savefig(”YourNameSurname”+algorithm+”.jpg”)
    # Show figure
```

To perform hierarchical agglomerative clustering the SciPy's `linkage` function and to plot the clustering solution as a dendrogram SciPy's `dendrogram` function can be used. The *distance_matrix* mentioned in the code contains the distances between each pair of organisms. To compute the distances;

1. You should compute the pairwise similarities by using the Needleman-Wunsch and Smith-Waterman algorithms from parts A and B.

2. Then, you should subtract the pairwise similarity scores from the maximum pairwise similarity score to obtain the distance values in the *distance_matrix*.
3. If everything goes well, you should expect to see an output figure like Figure 1.

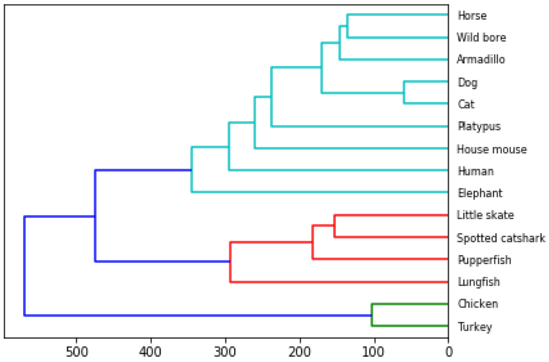

Figure 1: Organisms Dendrogram

You're encouraged to use Jupyter Notebook either with Python or R (A notebook is provided to you, please check *Bio-Assignment1.ipynb* file where all the code details and steps are listed). **If you use Jupyter Notebook**, you can prepare your code and report, which explains your code and evaluations of your outputs, at the same place and only submit your Jupyter Notebook. **If you do not use Jupyter Notebook**, please submit a fully commented code as well as a report that includes;

- detailed explanation of the functions and parameters,
- *screenshots* from running your code,
- the outputs of your Needleman-Wunsch and Smith-Waterman algorithms for the *chicken* and *turkey* organisms (in the *organisms.txt* file) for Parts A and B,
- the resulting dendrograms for Part C.

Submission: Please submit your assignment using Moodle. Upload a single zip file named as YourNameSurname.zip. Your zip file should include your report, your source code, requirements.txt that has the requirements to run, and the corresponding READ.ME file that explains how to run your code. You can use any programming language of your choice. But, your READ.ME file should clearly explain how to run your program. Feel free to reach me from my e-mail address, selen.parlar@boun.edu.tr.

Late Submission: You are allowed a total of 3 late days on homeworks with no late penalties applied. You can use these 3 days as you wish. For example, you can submit the first homework 2 days late, then the second homework 1 day late. After using these 3 extra days, 10 points will be deducted for each late day.