Miguel Guardado
BMI 203
Michael Keiser
01/27/2020

Project 1 Write Up

1) For both Smith-Waterman and Needleman-Wunsch algorithms:
   a) What are the parameters and variables required for algorithm initialization, execution, and termination?

      Some of the key distinctions between these two algorithms include the use of negative scores or limiting negative values to just be zero, as well as the location where backtracking begins. Both of these algorithms require a scoring matrix. They require creating and cleaning the sequences of unwanted variables, as well as thinking about storing the raw sequence vs the cleaned sequence. Both of these algorithms are going to require two functions where we fill the traceback as well and running the traceback. You will also need to initialize which scoring matrix you are creating and using for you amino acid alignment. Another parameter required will be the desired gap_open and gap_ext penalty for the sequence you are trying to align.

   b) What quantities are returned?

      Traceback + Score, as well as 6 matrices that dictate the scores and direction of each of the quantities M,Mp,X,Xp,Y,Yp.

   c) What is the runtime complexity?

      The runtime complexity for both of these algorithms is the length of both of the sequences O(nm).

2) What functionalities in initialization, execution and termination are shared between these algorithms? Which are not shared?

Shared:

Seq1- First Sequence
Seq2- Second Sequence
Align-show – Alignment sequence, filled with * and | based on value
Score-matrix - Score Matrix depending on BLOMSUM/PAM
Pointer Matrix – 3 separate matrix of the individual pointer matrix of the three items
M – Matrix of best alignment score
Ix – Matrix of best alignment score for seq1
Iy – Matrix of best alignment score for seq2
Gap_start – gap start penalty
Gap_end – gap extend penalty

Not Shared:

The score that is assigned to the algorithm, since for SW it will be the maximum score of the M,X,Y matrix while for the NW it will be the value in the bottom right corner of the M matrix. Where the traceback starts will also not be the same, since the starting point for both of these algorithms are where the score is found.

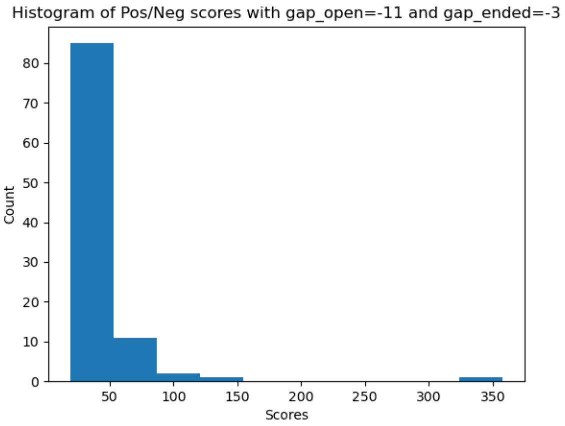
3) How does affine-gap based alignment differ from linear-gap alignment in terms of implementation?

Affine gap alignment will take into account a fixed penalty values for both opening and extending a gap. Gap opening will refer to the cost of opening any new gaps in a sequence while gap extension takes into account the cost to extend the gap. This can allow you to play around with the open/extension penalty parameters depending on the type of sequences you are dealing with. If the interest it to find closely related matched, a higher gap opening penalty should be used in order to reduce gap openings. If the size of the gap is important, then the extension parameter will be used. You can use a linear gap alignment strategy inside an affine gap alignment if you choose to let

Gap_opening = Gap_ext.

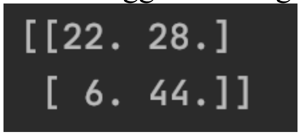
Part 2

1) With the BLOSUM50 matrix and a gap opening cost of 11 and a gap extension cost of 3, locally align these sequences and visualize the distribution of alignment scores. How would you describe this distribution?



This distribution seems very skewed right with most of the distributions falling under 50. With the threshold of this algorithm falling under 46.7, this is showing that most of the scores aligned in this distributions are under the threshold of positive significance, causing most of the scores here to be identified to be a negative alignment.

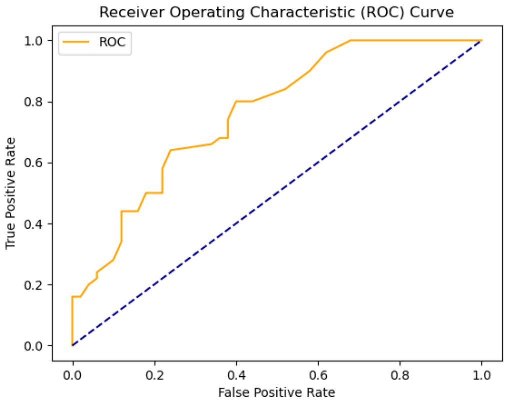
2) Generate a confusion matrix indicating the frequency of false positives, false negatives, true positives, and true negatives when using the average alignment score as a threshold. What is the threshold value, and how does the confusion matrix suggest this algorithm performed.



```
[[22. 28.]
 [ 6. 44.]]
```

The threshold value for this confusion matrix is 46.7, as shown in the distribution is left skewed with most of the scored being under 50. This will lead to most of the sequences to be identified as negative, this idea holds true with most of the positive being identified as false positives, and only 22/50 positive correctly identified. Conversely using this threshold correctly identified most of the negative sequences in the matrix only misclassifying 6/50. This confusion matrix suggest that the algorithm okay, with the potential to identify negative cases much better compared to the true sequence cases.

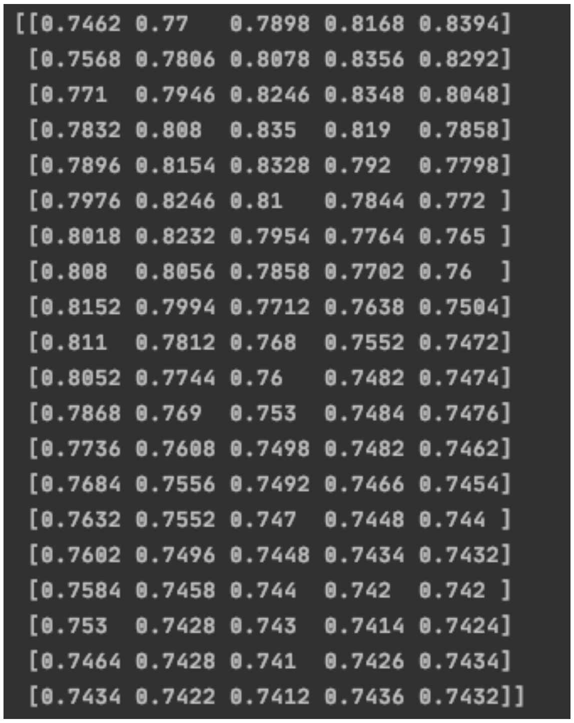
3) Create a ROC plot which shows the fraction of true positives on the Y axis and the fraction of false positives on the X axis. Please take care to make your ROC plots square, with both X and Y axes limited to the range [0:1].



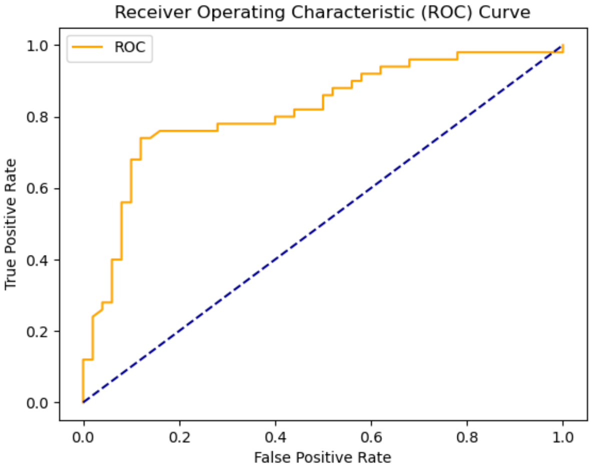
4) Determine the area under the ROC curve (AUROC). What does this value indicate about the performance of this algorithm? Can you confidently assess the performance of this algorithm with this value alone? Why or why not?

The AUROC in this model was .76, which is ideal. This shows that the model, smith waterman, was able to have an ideal measure of separability, with the distribution of FP overlapping more than FN cases. I would say this is not the best measure of the performance of this algorithm, since there are many ways to measure the classification preformane of a model. I think most importantly, the biggest downfall of this model is creating a threshold alignment from our score distribution, having the algorithm determine the best threshold compared to the true threshold score.

5) Once again, using local alignment, try a range of gap opening (1-20) and gap extension (1-5) costs with the BLOSUM62 matrix. Using the AUROC of each approach, determine which gap penalty performs the "best". What does this pair of values suggest about the evolution of these sequences and the likelihood of insertions / deletions?



```
[[0.7462 0.77   0.7898 0.8168 0.8394]
 [0.7568 0.7806 0.8078 0.8356 0.8292]
 [0.771  0.7946 0.8246 0.8348 0.8048]
 [0.7832 0.808  0.835  0.819  0.7858]
 [0.7896 0.8154 0.8328 0.792  0.7798]
 [0.7976 0.8246 0.81   0.7844 0.772 ]
 [0.8018 0.8232 0.7954 0.7764 0.765 ]
 [0.808  0.8056 0.7858 0.7702 0.76  ]
 [0.8152 0.7994 0.7712 0.7638 0.7504]
 [0.811  0.7812 0.768  0.7552 0.7472]
 [0.8052 0.7744 0.76   0.7482 0.7474]
 [0.7868 0.769  0.753  0.7484 0.7476]
 [0.7736 0.7608 0.7498 0.7482 0.7462]
 [0.7684 0.7556 0.7492 0.7466 0.7454]
 [0.7632 0.7552 0.747  0.7448 0.744 ]
 [0.7602 0.7496 0.7448 0.7434 0.7432]
 [0.7584 0.7458 0.744  0.742  0.742 ]
 [0.753  0.7428 0.743  0.7414 0.7424]
 [0.7464 0.7428 0.741  0.7426 0.7434]
 [0.7434 0.7422 0.7412 0.7436 0.7432]]
```

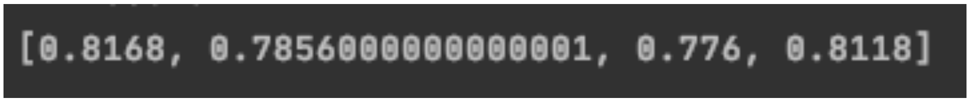


Running over 100 different test, I found I get the best AUROC curve when using a gap opening penalty of 0, and a gap extension cost of 5, with a AUROC value of 0.8394. With the intuition that increasing the gap opening penalty is used for closely related species, and increasing the gap closing penalty will impact the size of indels in my alignments. These sequences are genetically distant from each other due to the low gap opening cost, and that you are more likely to have more indels in your alignment since we have a high gap extension cost. This is all dependent on using a AUROC as your metric of best classification, with this showing the best classification for true and negative cases were found with said gap opening/closing cost.

6) Using the BLOSUM50, BLOSUM62, PAM100 and PAM250 scoring matrices, evaluate the performance of the global alignment algorithm using the selected pair of best performing gap penalties.

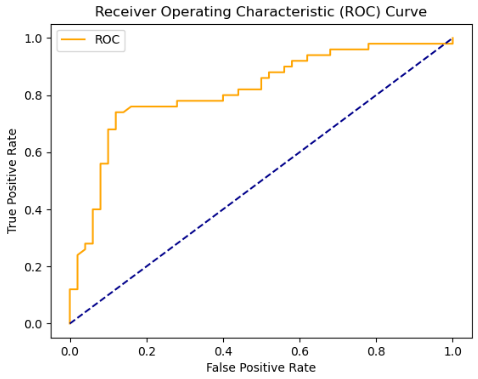
This list captures the alignment score order that corresponds with the auc matrix.

[BLOSUM50,BLOSUM62,PAM100,PAM250]

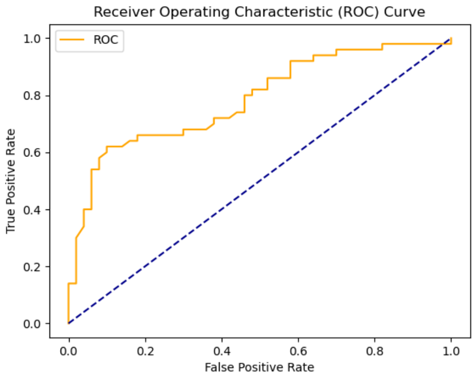


```
[0.8168, 0.7856000000000001, 0.776, 0.8118]
```

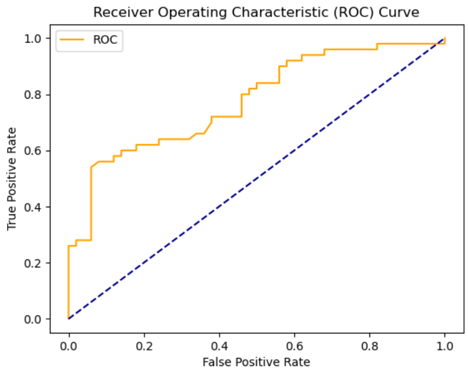
7) For each algorithm, generate a ROC plot demonstrating performance using each of the 4 matrices, given the fixed gap costs. Of all these algorithms, which performs the best as measured by AUROC?



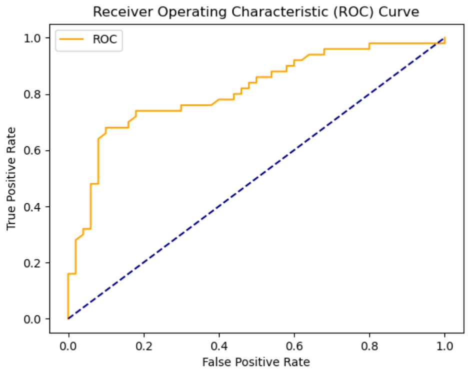
BLOSUM50=0.8168



BLOSUM62=.7856



PAM100=0.776



PAM250=0.8118

From my analysis of using these 4 different scoring matrices with a gap opening penalty of 0 and a gap extension of -4, the scoring matrix that returned the best result was using the BLOSUM62 matrix. This means that using this metric was able to create a threshold small enough that was able to capture the distribution of positive cases without sacrificing creating more FN cases.

8) Comment qualitatively on the best algorithm. What does the best performing algorithms indicate about the origin of these sequences.

We found the best sequencing alignment was found when looking using an affine gap penalty, with gap opening of 0, and gap extension penalty of -4. This indicates that the sequences must be evolutionary distant to each other, due to the low gap opening score, as well as the sequences being more likely to have more placed indels since the optimal extension cost being -4. Showing that the best score was found using a BLOSUM50 matrix, which uses block of conserved sequences that have a pairwise average sequence identity no greater than 50%, which shows that this model will perform best when accounting for larger effects of multiple solutions, which indicated that these sequences are further apart from each other, with the low gap opening cost furthering this claim. This leads me to claim these sequences are evolutionary distant to each other, indicating they come form distantly related species, such as dog/wolfs, humans/gorillas, different SARS sequences, though deeper analysis would be required to pinpoint the exact species these sequences derive from.