DNA Motif Alignment and Recognition Pipeline Project Report

At its core, this pipeline is capable of aligning numerous DNA sequences at conserved nucleotide regions shared between all these sequences. It is most applicable towards recognition and confirmation of DNA motifs, which are effectively conserved nucleotide sequences shared in different genes or even species. The pipeline has five general steps and in order of use, they are encoding each of the four nucleotides of the *.fasta* sequences into a mathematical *wyk* format; determining the maximum alignment scores between two sequences and shifting one sequence to be aligned with the other, obtaining an ultimate *wyk* file that represents the summation of all the aligned files, converting this file into a nucleotide probability, or position weight, matrix, and visualizing the matrix via a DNA logo generator. Although the matrix could be considered an output, the best analysis results, at least due to its visual nature, will stem from the DNA logo. Nucleotide regions that are conserved will have more information content and clarity in the logo, but regions that are not conserved will ideally have little to no information content and presence in the logo. It is up to the user to determine what information is significant but typically, conserved regions that are significant biologically can then be considered the motif, and regions that are repeats or other miscellaneous conservations will be disregarded. When this pipeline was applied to a data set of *E.coli* genes that have a similar, but slightly variable motif, the motif was visible if the alignment of all the sequences to the motif region was correct, and selection of that alignment is something this pipeline does not completely address yet.

This pipeline was curated and created entirely by myself computationally, but ideas behind the biological and mathematical aspect of this pipeline were in collaboration with my research lab, specifically Dr. Gary Stormo. The biggest challenges stemmed from the python code, as it involves application of numerous packages such as argparse, BioPython, itertools, matplotlib, numpy, pandas, scipy, and even sys. Understanding the various packages, modules, and functions to see what applies towards my goals was time-intensive and required many hours of troubleshooting. In addition, although steps such as encoding nucleotide sequences into *wyk* format or summation of two files were achieved via list comprehension, gauging whether the products of each step were beneficial towards the greater pipeline required immediate assessment that also proved to be an arduous and challenging task. This phenomenon was emphasized during calculating the alignment score, shift values, or converting a *wyk* matrix into a nucleotide probability format as all three of these steps required mathematical assessment and graphical analysis to confirm if what was being done was correct. Ultimately, however, the challenges faced initially were fruitful and once Unix code was used to harmonize and quicken the pace of usage for this pipeline, results were almost instantaneous. Every step of this pipeline, except inputting the data into a logo generator was achieved computationally, which even increased the benefit for the workflow of my research lab.

To discuss a little more about the computational features of this pipeline, I will begin with the python code. Encoding *.fasta* files into *wyk* format was achieved by replacing each A, C, G, and T with a corresponding value. For example, in *wyk* format, A can be represented by [1, -1, -1]. This translation is fundamental to all other steps of this pipeline. The alignment score generation was achieved by using the signal module of the scipy package. This module's

correlate function allows us to assess, numerically, at what alignment is the highest value obtained between these two sequences. The shift value from the latter step is then used to shift one *wyk* sequence to the reference *wyk* sequence (both of which are the same sequences used in generating the shift value and alignment score). Afterwards, all the shifted files in a data set and the reference are summed together, index position by index position, via list comprehension. The sum of all files is then converted into a nucleotide probability matrix through numpy's matrix algebra functionality. The python files, at most, only assess two file inputs and continuously write output. Incorporating Unix allowed for these python files to be executed across entire directories and the output paths to be well-organized; one can observe that there is significant similarity between the code of all the various Unix scripts in this pipeline for that reason.

Data analysis of this pipeline was obtained via running the pipeline on three test data sets, a dummy data set that mimics an experimental *E.coli* data set, and the aforementioned experimental *E.coli* data set. The analysis output was assessed only through the various data sets; there were no parameters changed in the computation itself for the published output but I did spend time exploring possible parameter changes. The first test showed that the pipeline is capable of aligning similar regions such as motifs, the second test confirmed that this pipeline will show a lack of information content in the logo if there are no similarities between sequences, and the third test represented the opposite of the second, leading to an output that showed high confirmation content if the two sequences were identical. All three tests confirmed that the pipeline's usage is viable, at least in extreme situations. A dummy data set where four sequences had the same motif, identical to the consensus motif explored in the *E.coli* genes, and nothing else in common also resulted in expected results. The motif region was noticeable and had high information content while all other regions that were different were not aligned. When applying this pipeline to the *E.coli* data set, initial results were not ideal. The motif region did not have high information content and it was not discernible what was the motif. To troubleshoot this, it was determined, through hand, what the shift of each of the sequences in the data set were to a reference sequence I determined. It was then observed that the pipeline had an issue when it came to the sliding alignment score algorithm, not the pipeline's computation and execution itself. The motif alignment is not always the highest alignment score. Other regions of similarity can result in a higher value, which is biologically true. There are many regions of repeats or similar nucleotides between genes and species but they may not be biologically significant to be a motif; computers however, cannot differentiate between what is biologically significant or not, especially in a pipeline like this that does not incorporate artificial intelligence and machine learning. Using the same genes, I then created three subsets for genes where the highest peak is the correct shifts, where both values are close, and where both are very different, and ran the pipeline on those three subsets to ultimately receive expected results. If the shift was correct, the motif was visible but the more distance between the peak and the shift values was there, the less information content was present in the logo. One last data "setting" was through determining a logo for all ideal shifts, and it led to the motif being present in the logo. In short, this pipeline is viable as it has reflected the practical, biological, and algorithmic merits and faults as expected.