Introduction to Bioinformatics – AS 250.265
Assignment 8

The application of multiple sequence alignments in bioinformatics is pervasive and important to understand. As we will learn shortly, many of the contemporary arguments for molecular evolution are based on the data contained in multiple sequence alignments. This homework is designed so that you will practice interpreting and scoring multiple sequence alignments. When we finish, you should have an increased understanding of how MSAs are scored and used in real-world applications. This assignment will also encourage you to develop your programming knowledge in Python.

This assignment is worth 100 points toward the homework portion of the class. The laboratory questions you do in class constitute 35 of those points, and this handout makes up the remainder. Both parts are due on April 21 at the start of class. Electronic portions of the assignment should be submitted through the course website with appropriate filenames (e.g. `hw08-q4e.py`). The written portions of the assignment may be submitted by hand or uploaded as a Word document or PDF file.

Question 1: Laboratory Number 4 (35 points)

Submit your solutions the laboratory questions on MSA that you completed in class.

Question 2: Sum of Pairs Scoring (10 points)

Using the sum of pairs scoring method, score the following multiple sequence alignments. You should use the BLOSUM62 scoring matrix (in your book on page 61), and use the following additive affine gap penalties: gap opening (-4), gap extension (-2). Gaps aligned with residues or other gaps score zero. Each alignment is worth five points.

a.
```
seq1 - HAKKVL
seq2 - HGRNAI
seq3 - HYKQQL
```

b.
```
seq1 - Y--TSEE
seq2 - FF-TTDE
seq3 - YFVTSEE
```

Question 3: Problems with Sum of Pairs Scoring (15 points)

In class, we discussed that the popular sum of pairs scoring methods used to score multiple sequence alignments overcounts certain combinations and gives non-intuitive scores for multiple sequence alignments. In this problem, we will work out an example.

Consider the following scenario, where we examine two hypothetical "one column" multiple sequence alignments. In the first alignment of N sequences, every residue is a tyrosine. In the second alignment, there are N-1 tyrosines and one proline.

a. Given that the BLOSUM62 Y ↔ Y score is 7, calculate the score of the first alignment as a function of N. This is $S_{Y_N}(N)$. (5 points)

b. Given that the BLOSUM62 Y ↔ P score is -4, calculate the score of the second alignment as a function of N. This is $S_{Y_{N-1}P_1}(N)$. (3 points)

c. Evaluate and simplify the following expression, representing the fractional difference between the two different sequence alignments. (2 points)

$$f(N) = \frac{S_{Y_N}(N) - S_{Y_{N-1}P_1}(N)}{S_{Y_N}(N)}$$

d. Construct a plot of the expression you derived in part (c), and explain why this scoring behavior is incorrect. (5 points)

Question 4: Random Scores and Python (15 points)

In this part of the assignment, you will answer questions about one of the programs you used in lab 5, `random_score.py`. To answer these questions, you will have to use the python documentation available on the web at `http://www.python.org/doc/`. You may find the module index particularly helpful for questions about the `sys` module.

a. Examine the end of the program. You will notice that several references are made to `sys.argv`. What is the type of this object (i.e., number, string, etc.)? What does it store? (3 points)

b. Explain the use of the `try`… `except` block at the end of the program. (3 points)

Now consider the N function, which counts how many times an amino acid appears in a protein. This function uses some Python constructs that you have likely not seen before. First, consider the `lambda` expression. This expression is simply a way of defining a simple function in a single line. The expressions:

```
add = lambda x, y: x + y
```

and

```
def add(x, y):
     return x + y
```

are equivalent. Thus, it is possible to pass a simple `lambda` expression as a function argument, as is done twice in the return statement of the N function.

c. What does the lambda expression `lambda x: int(x == aa)` do? In other words, how many arguments does it take? What will the return values be in the context of the N function? (Hint: it's not cheating to plug it into the Python interpreter.) (2 points)

The other Python statements that you may not be familiar with are the `map` and `reduce` functions (along with their cousin `filter`). These functions are carry-overs from functional programming languages such as lisp and ML. Since they are implemented in as Python primitives, they are often faster than simply using a `for` or `while` loop, which must be interpreted. You will never be responsible for using these functions in this class, but you should understand what they are doing.

Look up these functions in the Python Library Reference (they are under section 2.1, "Built-In Functions").

d. As the program is currently written, explain how the N function works. (3 points)

e. [programming] Rewrite the N function using a `for` or `while` loop, and submit only your re-written function to the course web server. (4 points)

Question 5: Position Specific Substitution Matrix (25 points)

In this final question, you are to implement a function that calculates position specific substitution frequencies for every position in a multiple sequence alignment. While this is not exactly the same as a true PSI-BLAST PSSM, it is close, and the calculations you will perform are similar.

On the course website, you will two files, `lipocalins.mfasta` and `hw08_q5.py`. The first file contains a Pfam multiple sequence alignment for the lipocalin family of proteins. The second file is a template that you will use to create your substitution matrix program. You will also need the `matrix.py` library of matrix functions to construct your matrix and provide a means of printing it to the screen.

Your goal for this assignment is to construct a matrix with 20 columns and N rows, where N corresponds to the length of the sequence alignment. At each position of the sequence alignment, you should compute the frequency of each amino acid using the simple maximum-likelihood estimation:

$$M_{pos,aa} = \frac{N_{pos,aa}}{\sum_{i=Ala}^{Val} N_{pos,i}}$$

In this expression, the matrix value M for a position (row) *pos* in the alignment, for a particular amino acid type (column) *aa* is given by the number of amino acids (N) of type *aa* at that position in the multiple sequence alignment divided by the total number of amino acids as that position in the alignment. Though not ideal, we will not count gaps in computing our matrix.

Your program should calculate this matrix and print it out with a reasonable precision. It is recommended that you use the same column ordering found in previous assignments: A, R, N, D, … If you choose to use a different order, please make this explicitly clear at the top of your program file.

In the template you will already find code to open the MFASTA file and a skeleton function for computing your matrix. When you finish, you should be able to check your work by examining the Pfam alignment for lipocalins: The highly conserved residues in the Pfam alignment should also be identified by your substitution matrix as having frequencies close to or exactly one. (25 points)