Introduction to Bioinformatics – AS 250.265
Laboratory Assignment 5

Mulple sequence alignment is very useful for many applications. We have already seen its application in several settings: It can be used to create position specific scoring matrices in order to increase BLAST sensitivity. It is also useful for creating genetic scoring matrices, as we have seen in BLAST. In addition, multiple sequence alignment can be applied in other contexts as well.

Until now, we have not examined in detail how multiple sequence alignment can be done in detail: in essence, we simply swept the algorithm under the rug and assumed that it could be done somehow. The goal of this lab is to resolve that problem. Here, we will manually perform a multiple sequence alignment between five variants of the protein myoglobin. The approach we will use will follow the progressive alignment technique of Feng and Doolittle, with a few modifications. At the end of the lab, we will compare our alignment with the alignment generated by the popular multiple sequence alignment program ClustalW. Then, for something completely different, we will examine how another technique, profile hidden Markov models (pHMM's) can be used to describe protein domains in the web database Pfam.

This lab will constitute the first part of your assignment for the coming week. Your answers from this lab will count as 30 points toward assignment number eight. You may submit your answers in class on April 21st, or you may upload them as part of your solutions for assignment eight on the course webpage.

Part 1: Manual Multiple Alignment of Myoglobins

Myolobin is an average-sized α-helical protein expressed in skeletal and cardiac muscle cells. It binds a heme group and is involved in storage of oxygen in these cells. As you might expect, it is a protein common to many vertebrate animals, but even some plants like soybeans express globin-like proteins. Here, we will examine an alignment of the following four myoglobins:

Common Name	Species	NCBI Accession #
Human	*Homo sapiens*	NP_976312
Mouse	*Mus musculus*	NP_038621
Chicken	*Gallus gallus*	P02197
Zebrafish	*Danio rerio*	NP_956880

As we will be using these sequences throughout the lab, your first task is to download and store them on your local computer in FASTA format. You can get all of these sequences from the NCBI website under Entrez Protein, and the FASTA format is available as a display option. By default, the display is set to GenPept.

You will have to save the sequences by copying them from your web browser and saving them to an `xemacs` or `vi` file. My files, for example, are named "`cow.fasta`," etc. When you paste your sequence, make sure the result is correct—all of your FASTA files should have one line with a comment (>) and several lines of sequence.

Stage 1: Initial Alignment

We will do much of our manual work using the biology workbench site. Recall that that BioWorkbench web page is `http://workbench.sdsc.edu/`. On this page, upload each of your FASTA file sequences so that you can access them. As we will be comparing sequences to themselves, copy the protein sequences once you have uploaded them—that way, you can click the identical pair when you need to align an sequence to itself.

The first step in our manual sequence alignment is to compare each protein to every other protein. Using the ALIGN program, determine the scores for every possible alignment. You may use the default ALIGN parameters—recall the observation that matrix selection only marginally affects the resulting alignment. The matrix below is given to help you—remember that it should be symmetric.

	Human	**Mouse**	**Chicken**	**Zebrafish**
Human				
Mouse				
Chicken				
Zebrafish				

Now that you have filled in the table above, we must calculate the distance between each pairwise alignment. Recall that the distance is given by the following formula.

$$D(i,j) = -\ln S_{eff}(i,j)$$

The tricky part here is to calculate S_{eff}. S_{eff} involves several parameters: The real alignment score, given in the table above, the maximum alignment score, defined as the average alignment between both sequences aligned with themselves, and a random component, representing the average score of the shuffled sequences.

The definition of S_{eff} is

$$S_{eff}(i,j) = \frac{S_{real}(i,j) - S_{rand}(i,j)}{S_{best}(i,j) - S_{rand}(i,j)}$$

In this equation, S_{best} and S_{rand} are given by the following equations:

$$S_{best}(i,j) = \frac{S(i,i) + S(j,j)}{2}$$

$$S_{rand}(i,j) = \frac{1}{L}\sum_{a=Ala}^{Val}\sum_{b=Ala}^{Val} M(a,b)N_i(a)N_j(b) - G(i,j)$$

While you will be required to calculate S_{best} yourself, a program is provided for you on the lab 5 website that will calculate S_{rand} using the BLOSUM50 scoring matrix. To use this program, simply specify the two FASTA sequence files you wish to use as command line arguments. For example:

```
python random_score.py human.fasta cow.fasta
```

The printed result is the random score between two proteins assuming no gaps—a fairly good assumption for this case.

a. Using the equations above, calculate the distance table for your four alignments (the *D(i, j)* values). You may submit your table electronically, or you may use the form found at the end of the packet. (5 points)

b. Which two non-identical proteins are most similar based on your distance matrix? Which two are the most dissimilar? (3 points)

Stage 2: Construction of a Guide Tree

The next phase of progressive multiple sequence alignment is to hierarchically cluster the proteins using the distance matrix result. The algorithm for clustering are similar (if not identical) to those we used in the analysis of microarray data. The only difference is the initial calculation: when clustering the microarray data, the distance matrix must first be determined from the expression intensities. Here, we have already calculated the distance matrix, so we can skip the initial step.

Unfortunately, the Cluster program on your machines does not support clustering given a precalculated distance matrix, even though the library of functions it uses supports this. On the lab website, however, you will find a program called `matrix_cluster.py` that does will perform this next step of the operation for you. You will, however, have to construct a text file that contains the matrix you calculated in part (a). The format of your text file is given on the next page.

Format of Matrix Input File

Line 1: A list of row and column names, separated by spaces. In our case, this will be something like "`human mouse chicken zebrafish`". These headings will be used not only as labels on your clustering file, but they also specify how many rows and columns the matrix will have.

Line 2: A list of numbers, separated by spaces, corresponding to the first row of the matrix. If there are four columns declared in line 1, there should be four columns in this row.

Line 3: As with line two, corresponding to the second row of the matrix…

Line N: The final row that contains information. This row should correspond to the final heading given in line one.

You will have to convert your matrix to a text file in the above format. When this is complete, you can use the `matrix_cluster.py` program to construct a clustering tree that is viewable with JavaTreeView on your Macintosh machine. To run the clustering program, use the following syntax (the `/usr/local/bin/python` is required to use this program, as this uses a special clustering module not available in the default `python`):

```
/usr/local/bin/python matrix_cluster.py <matrix file> <job name>
```

The matrix file is simply the name of the file to which you saved your matrix, and the job name is the name that will be used in storing your output files. When the program has completed, you will be informed of the names of the files where your tree data is stored.

c. Use the Java TreeView program on your Macintosh to visualize the hierarchic tree produced by `matrix_cluster.py`. Copy the tree to the worksheet at the end of this packet and submit it. (5 points)

Recall that in progressive alignment, the guide tree is used to determine how to build up the multiple sequence alignment. Your guide tree will be almost trivial, but it is not generally the case that this is so. In your case, you will build up the alignment one sequence at a time, but occasionally groups of alignments must be merged. We will discuss specifically how this is done below, but for now, you should be able to start at the two most similar sequences in your tree and identify the order in which you should build up your alignment.

d. Looking at the guide tree, what pairwise alignment should you start with? In what order should you add the other two alignments to your multiple sequence alignment? (3 points)

Stage 3: Building Up the Alignment

Now there is only one final thing to do: we must construct the final multiple sequence alignment by following the tree. As we do this, there are three possibilities that can arise, and we will follow the ideas of Feng and Doolittle:

- We may have to perform another pairwise alignment for two proteins that are very dissimilar to the original pair that we started with. When this happens, we simply align the two sequences as we would normally.

- We may have to add a single sequence into a multiple sequence alignment. To do this, we examine the original set of alignments, and we look for the sequence already present in the multiple sequence alignment that was closest to the new sequence we are trying to insert. We use that best alignment as a starting point for augmenting the multiple sequence alignment, pairing residues and inserting gaps as necessary.

- Finally, we may have to merge two multiple sequence alignments. In this case, we find the best pair of sequences from each MSA, and, following the same idea as above, we would augment the new alignment using the best pairing alignment as a template.

Using these ideas, we also assert that, "once a gap, always a gap." That is, a gap that already exists in the multiple sequence alignment persists even as new sequences are added to the alignment.

Pause here in the lab for a brief example for how this works.

Here, we will use Xemacs to construct a multiple sequence alignment manually. To do this, we will copy and paste our best alignment into a text file being edited by Xemacs in several steps so that the alignment occupies only two lines. Then, we will add additional lines based on the rules given above. You will find it easiest to set the Xemacs variable "truncate-lines" to be true. Using this mode will allow you to scroll over the entire length of one long line rather than having the lines wrap around when the window is not big enough. This can be done with the following commands:

Engaging Xemacs in Truncate Lines Mode

Press:	`<esc>, x`	(Press escape, then type x, you'll see M-X appear at the bottom of your screen)
Type:	`set-variable <enter>`	
Type:	`truncate-lines <enter>`	
Type:	`t <enter>`	

As a first step, open up Xemacs and make the window as wide as you can. Using the instructions above, tell Xemacs to truncate lines of text. Then, open up your initial pairwise alignment on the Biology Workbench site, and copy the entire thing into Xemacs. Again, use all of the default option of the ALIGN program on Biology Workbench.

Once the alignment is in Xemacs, you can delete additional text and copy and paste so that this alignment only occupies two lines. Then, using your tree is a guide, you can past additional alignments into Xemacs and construct the finished multiple sequence alignment.

e. Once you have completed the alignment in this way, clean it up by breaking it into screen-sized chunks (~75 columns) and labeling each protein (human, mouse, etc.) at the left. Submit your alignment as a text file as a part of homework eight. (5 points)

Now that we have calculated our own multiple sequence alignment, let's compare it to a real program. Visit the ClustalW website, available at `http://www.ebi.ac.uk/clustalw/`. Using all the default options, paste the complete contents of your FASTA files into the text box for input. It's important that you include the comment line (which begins with a greater than sign), as this is how the program knows when one sequence ends and another begins. When you submit your result, you should see a multiple sequence alignment for your four sequences (completed in far less time than it took you to do all this manually!)

f. Compare your sequence alignment to the ClustalW result. Does the result differ? Knowing what you know about ClustalW, how do you think your results would compare for more sophisiticated protein alignments? (3 points)

Part 2: Using the Pfam Database

Pfam is a database of multiple sequence alignments created by profile hidden Markov models (HMM), which we discussed in class. Pfam A us a curated collection of several hundred protein domains as represented by the pHMM constructed by their multiple sequence alignments. Pfam B stores the domains as well, although it is generated automatically and not curated. For our purposes, we will focus on Pfam A.

Visit the US Pfam mirror at `http://pfam.wustl.edu/`. At this site, you can search Pfam using several methods: First, you can enter a protein sequence, and Pfam will identify the domains that correspond to that protein sequence, similar to the CDD search functionality in BLAST. You can also search DNA sequences in this manner. Pfam also provides a simple text search, where you can search for a known domain type and browse the protein members of that type. Finally, you can simply browse the domains by name or by taxonomy.

In this part of the lab, we will continue our investigation of the globin structural family. First, do a keyword search for "globin," and locate the main Pfam page. You will see a well-written summary about the globin fold, followed by some options. Here, we can view the alignment, a graphical picture of domains, or we can view the taxonometric distribution of the protein across species. The "seed" proteins are those used to form the initial HMM, whereas the "full" list of proteins are all those proteins identified by the HMM and confirmed to be a genuine member of

the family. Recall that this is Pfam A, so user input is used to curate the database. You will also find links to PDB files and other databases toward the bottom of the page.

g. Examine the domain structure of the globin seed proteins. Is the globin domain most often found by itself or in a multidomain protein? (3 points)

One useful tool that Pfam provides is the Jalview Java viewer. This Java web applet allows you to view and manipulate the multiple sequence alignment. From the main Globin page, select "Jalview java viewer" under the alignment format and click on "Retrieve alignment." You will be directed to a page where you will be allowed to start Jalview. When the applet loads, try viewing the multiple sequence alignment. If you notice a particular aspect of the alignment that seems off, you can click and drag the sequence to change where the gaps are.

There are several features of the Jalview viewer that may be useful to you as you work on your project. First, notice that (at least on my Mac) the menu options appear to the right of the browser menu at the top of the screen. Under the "Colour" menu, you are presented with various ways to color your alignment. For example, coloring by hydrophobicity produces a plot where red residues are hydrophobic and blue residues are hydrophilic, allowing you to see how where patterns might exist in the alignment.

The "Calculate" menu allows you to calculate a phylogenetic tree from the alignment, as well as residue conservation—when conservation is calculated and when the alignment is colored by hydrophobicity, the most highly conserved are colored with the highest intensity. Similarly, you can filter other coloring schemes by hydrophobicity.

h. What is the most highly conserved hydrophilic residue in the globin fold? What about the most hydrophobic residue? (3 points)

Finally, let's examine whether there are any subclasses in the set of alignments. Under "Calculate," select "Principal component analysis." This calculation will attempt to determine the fundamental features of each sequence and plot those features on a graph in three dimensions. Each point on the graph represents a different sequence in the alignment. You will notice that there are approximately four clusters: three small clusters and one large cluster of sequences. Clicking on an individual point will highlight its identifier on the sequence alignment window.

i. Identify the cluster containing the sequence `GLB10_CHITH`. Looking at the other proteins in this cluster, why do you think they cluster differently? (Hint: open up another Pfam browser window, and view the seed alignment in hypertext form, which has clickable links to the SWISS-PROT website. Then, view the SWISS-PROT pages for some of the members of that cluster.) (5 points)

Name: _____

Introduction to Bioinformatics – AS 250.265
Laboratory Assignment 5 Worksheet

You may use this worksheet when submitting your answers to several questions in assignment five. For the other questions, please submit your written responses on a separate page or by uploading them to the assignment eight web server.

Question (a): Distance Matrix

Fill in the matrix below with the distances you calculate from part (a).

	Human	**Mouse**	**Chicken**	**Zebrafish**
Human				
Mouse				
Chicken				
Zebrafish				

Question (c): Guide Tree Diagram

Draw the hierarchical tree you obtain using the `matrix_cluster.py` program and Java TreeView below. Be sure to label the ends of the tree!