

### Homework #2 - Sequence Analysis

1. In this problem we will use dynamic programming to find two different alignments between the sequences  $X = \text{TAGACCCGT}$  and  $Y = \text{TAGGCAGCCAGT}$ .

1.(a) Using the scoring rules:

match            +1  
 mismatch        -1  
 gap creation    -1.5  
 gap extension   0

find the optimal global alignment between  $X$  and  $Y$ . Use the matrix below to maintain the optimal score of the prefixes as you compute the overall optimal score. Recall that the  $i^{\text{th}}$  row and  $j^{\text{th}}$  column of the table contains the score of the optimal global alignment between the prefixes  $X[1..i]$  and  $Y[1..j]$ .

	--	T	A	G	A	C	C	C	G	T
--										
T										
A										
G										
G										
C										
A										
G										
C										
C										
A										
G										
T										

On the matrix above, draw arrows that indicate the traceback of an optimal *global* alignment solution to this problem. Write the alignment below:

List the number of mismatches, matches, gap creations, and gap extensions for the global alignment:

1.(b) Using the scoring rules:

match +1  
 mismatch -1 for transversion(A,G <-> C,T)  
 mismatch 0 for transition (A<->G ; C<->T)  
 gap creation -.5  
 gap extension -1

find the optimal *local* alignment between *X* and *Y*. Again, use the matrix below to keep score as you perform the dynamic programming algorithm.

	--	T	A	G	A	C	C	C	G	T
--										
T										
A										
G										
G										
C										
A										
G										
C										
C										
A										
G										
T										

This time, draw the arrows that traceback through an optimal *local* alignment of the two sequences. Write the optimal alignment below:

List the number of transversions, transitions, matches, gap creations, and gap extensions for the local alignment:

2. Consider a gap-less alignment between two nucleotide sequences of length  $n$ :

$$\begin{array}{cccccc} X_1 & X_2 & X_3 & X_4 & X_5 & \dots & X_n \\ Y_1 & Y_2 & Y_3 & Y_4 & Y_5 & \dots & Y_n \end{array}$$

Where each  $X_i$  and  $Y_i$  is the  $i^{\text{th}}$  amino acid residue in the sequence  $X$  and  $Y$  respectively. Let  $S$  be a measure of similarity between the two sequences. As usual, we define  $S$  to be the sum over the pair-wise similarities:

$$S = \sum_{i=1}^n \sigma(X_i, Y_i)$$

where  $\sigma$  is a pair-wise similarity function between two nucleotides. We can think of  $S$  as a random variable drawn from some distribution,  $F$  (imagine drawing numbers out of a hat -- the hat in this case is  $F$ ). In order to tell how significant an alignment between two sequences is, we need to know  $F$  first. Since we don't know  $F$ , we choose to approximate it with a normal distribution. A normal distribution requires two parameters to be estimated -- an expected value (mean) and a variance. Recall that the expected value of a function,  $g$ , of a discrete random variable,  $T$ , is:

$$E(g(T)) = \sum_{t \in T} P(T = t) \cdot g(t)$$

where the summation sums over all possible values of the random variable and  $P(T=t)$  is the probability that  $T$  equals a particular value  $t$ . The variance of  $g(T)$  is:

$$\text{Var}(g(T)) = E(g(T)^2) - E(g(T))^2$$

2.(a) Let the pair-wise scoring scheme equal the same scoring scheme from part 1(b) where we penalized transitions and transversions unequally:

	A	G	C	T
A	1	0	-1	-1
G	0	1	-1	-1
C	-1	-1	1	0
T	-1	-1	0	1

In the above matrix, each cell represents a particular value of  $\sigma$ . For example,  $\sigma(A,G) = \sigma(G,A) = 0$  since the A->G and the G->A mutation are transitions. Now suppose the two sequences,  $X$  and  $Y$ , are drawn *randomly* and independently from the yeast genome where the percentage of adenine is 31%, thymidine is 31%, cytosine is 19%, and guanine is 19%. Assuming each position in the alignment is independent, compute an estimate for  $E(S)$ , the expected value of  $S$ . [Hint: You will need to use the identity  $E(A+B) = E(A) + E(B)$  for two random variables  $A$  and  $B$ .]

2.(b) Using the same setup as in part (a) compute an estimate of  $\text{var}(S)$ .

2.(c) Suppose we run a dynamic programming algorithm to align two nucleotide sequences of length  $m$  and  $l$  and obtain an alignment of length  $n$ . Can we use a normal distribution described by the above parameters to calculate an accurate p-value that establishes how deviant our alignment score is from a randomly generated alignment? Why or why not?

3. In this problem, we will explore the Basic Local Alignment Search Tool (BLAST) algorithm. You used this algorithm to search GENBANK in Homework 1. Due to issues of efficiency it is used instead of the dynamic programming algorithms described in class to conduct large database searches.

Imagine you have the following nucleotide *query* sequence:

AATCGAATCGAA

And also you have a *database* of the following 6 sequences:

1. TTAATCAATCCCC
2. TATATATAAAAA
3. AAAAACCTCGA
4. GAAGAAAATAGAA
5. ATATCGGGGAA
6. ATTCGATTCGAA

3.(a) The first step in BLAST works by scanning the query sequence to generate all sub-strings (words) of consecutive letters of a particular length  $w$ . From the query sequence generate all such words of length 5.

3. (b) The second step generates a list of high scoring words. This is done by taking the list of words from part (a) and finding all words similar to them. Two words are similar if their gap-less alignment score is greater than  $T$ . Assume we score a match as +1, and a mismatch as -1. For example, if  $w=6$  and  $T=3.5$  two high scoring words generated by AAAAAA is itself (score = 6) and AAATAA (score =4). However, the word AATAAT (score =2) does not meet our criteria.

Choose the word that begins at position 1 and the word that begins at position 7 in the query sequence. From these two words, find all of the possible high scoring words with scores greater than  $T=2.5$ , scoring matches as +1, mismatches as -1.

NOTE: feel free to use \* as a wildcard indicating any amino acid – this may save you some writing.

3.(c) The next step is to scan the database for *hits*. A *hit* is a word in the database that exactly matches one of the words in the list of high scoring words generated from the query.

1. When we find a *hit* we align it to the original query sequence against the database sequence so that the high scoring word that was found in the database sequence is aligned with the word that generated the high scoring word from the query sequence. The gap-less alignment score of length  $w$  is identical to the score of the high scoring word.
2. We attempt to extend the alignment on the right. We add the nucleotides one position at a time, keeping track of the best score of the alignment as we go. When the current score falls below the best score by a threshold  $d$ , we stop extending. We then return the extension that achieved the maximal score.
3. We repeat on the left
4. Finally, both extensions are combined and the alignment is scored in its entirety. The combined score is reported for the hit.

In this problem use only the high scoring words that you found in (b) to find hits in the database. Extend each of the hits you find, calculate the score of each hit, and report the resulting alignment. Use  $d=1.5$ .