McGill

December 2016
Final Examination

COMP 462 - COMPUTATIONAL BIOLOGY METHODS
COMP 561 - COMPUTATIONAL BIOLOGY METHODS AND RESEARCH

December 14th 2016, 18:00-21:00

Examiner: Mathieu Blanchette　　Assoc Examiner: Jérôme Waldispühl

Signature: ______________________　　Signature: ______________________

Student Name:		**McGill ID:**									

INSTRUCTIONS:

- This is an OPEN BOOK examination
- Answer **DIRECTLY ON THE EXAM**
- This examination paper MUST BE RETURNED
- COMP 462 students: This examination is worth 40% of your final grade.
- COMP 561 students: This examination is worth 30% of your final grade.
- The total of all questions is 100 points. The value of each question is found in parentheses next to it.
- Two blank pages are added at the end in case you need extra space.
- Calculators are allowed
- No laptop computer, cell phones, etc. is allowed.
- This exam comprises 6 questions on 11 pages, including this cover page and two blank pages at the end.

Question 1. (28 points)

Answer each question with a <u>short but precise</u> explanation for each. *Credits will be given only if the justification is clear and correct.*

a) (4 points) True or False? *Justify.* If gene *G* has 10 mRNA copies present in a cell at time *t*, and if the expression of *G* does not change afterwards, then the protein encoded by *G* will never be present in the cell in more than 10 copies.

False. The number of copies of the protein encoded by gene G depends on a variety of factors, including the translational efficiency, the stability of the encoded protein, etc.

b) (4 points) Consider two samples S_1 and S_2 of cells where the gene expression profiles (number of mRNA copies of each gene) are absolutely identical, except for a single gene, called G_1, that is expressed 1,000,000 times more in sample 2 than in sample 1. You perform RNA-seq experiments on S_1 and S_2, and obtain expression levels for each gene, measured in FPKM (Fragment per Kilobase per Million reads). How will the FPKM expression of gene G_2 (a gene other than G_1) compare in the two samples, and why?

The FPKM value for G2 will be lower in sample 2 than in sample 1, because a larger fraction of reads will be mapped to G1. The key thing to remember is that RNA-seq measures the relative gene expression, not the absolute expression level.

c) (4 points) What is a transcription factor?

A transcription factor is a protein that regulates the expression of one or more genes by binding the DNA near)or not so near) the start of a gene and helping recruit the RNA polymerase.

d) (4 points) Consider a RNA sequence *S* containing 10 A's, 10 C's, 10 G's, and 10 U's, and sequence *T* also contains the same number of each nucleotide, but in a different order. *True or False? Explain.* The score of the RNA secondary structure produced by the Nussinov algorithm for *S* will always be equal to that for *T*.

False. This depends on the arrangement of the nucleotides. With AAA..CCC…GGG…UUU, we can form 19 pairs (10 A-U pairs, 9 C-G pairs, and 2 unpaired nucleotides for the loop).

e) (4 points) Consider two single nucleotide polymorphisms (SNPs) found the human population:

	Major allele (%)	**Minor allele (%)**
SNP1:	A (98%)	T (2%)
SNP2 :	C (55%)	G (45%)

Give two reasons why the minor alleles of the two SNPs have frequencies that are so different.
1) In SNP1, allele T may be deleterious (i.e. cause a reduction in fitness, e.g. a disease), which would lead to a lower allele frequency. Or allele A may cause an advantage.

2) In SNP1, allele T may be recent and may not have had time to spread to a large fraction of the population.

3) SNP2 may be under balancing selection (i.e. a case where it is beneficial to be heterozygous).

f) (4 points) Suppose you performed a RNA-seq experiment to measure gene expression (for m = 20,000 genes) in 100 diabetic patients and 100 non-diabetic patients. You then used a Student t-test to obtain a p-value for each gene. The result shows that the average expression level of gene G is 200 in diabetic patients and 100 in non-diabetic patients. However, the variances of the expression levels of G in the two groups are large, so the p-value obtained from the t-test is 0.15, i.e. the difference is not statistically significant. The expression level of G is then measured on 1000 patients from each group, and the average expression values and variances remain exactly the same as before. How will the p-value of the t-test for G change, if at all?

The t-value will remain the same, but the p-value will become smaller, because the number of degrees of freedom of the Student distribution increases with sample size.

g) (4 points) As seen in class, the Sankoff algorithm calculates the parsimony score of a given alignment column on a given phylogenetic tree. The algorithm is based on the following recurrence, which calculates a table S_u at node u based on the previously computed tables S_v and S_w, where v and w are the two children of u. For nucleotide a:

$$S_u[a] = \min(\, S_v[a],\ 1+ \min\{S_v[b] : b \neq a\}) + \min(\, S_w[a],\ 1+ \min\{S_w[b] : b \neq a\})$$

The algorithm can be modified to calculate the *weighted* parsimony score of an alignment column, where instead of trying to minimize the total number of substitutions performed along the branches of the tree, we assign each possible substitution from nucleotide x to nucleotide y a weight $w(x,y)$, and seek to minimize the total weight of the substitutions performed along the branches of the three. This is done by redefining $S'_u[a]$ as the *weighted* parsimony score of the subtree rooted at node u if u is labeled with a. Give the recurrence to calculate $S'_u[a]$.

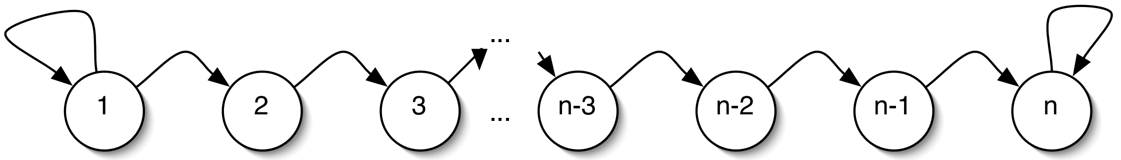
$$S'_u[a] = \min(\, S_v[a],\ \min\{S_v[b]+w(a,b) : b \neq a\}) + \min(\, S_w[a],\ \min\{S_w[b]+w(a,b) : b \neq a\})$$

Question 2. (25 points)

a) (3 points) Consider a hidden Markov model with *n* states. What is the worst-case running time of the Viterbi algorithm on a sequence of length *L* ? Use the Big-O notation. No justification is needed.

$O(L\ n^2)$

b) (10 points) Now, consider the following hidden Markov model with *n* states. Describe how to modify the Viterbi algorithm so that, on this particular type of linear HMM, the running time is asymptotically faster than in (a) ? Describe the modification required, and give the running time of the improved algorithm. Your answer could be in pseudocode, or simply refer to changes to the pseudocode given in class.



We first note that in this HMM, all but one of the states has a single possible previous state. This allows an optimization of the Viterbi algorithm:

*For state s=1: V(i,s) = V(i-1,s)*T(s,s)*E(s,i)*
*For state 1<s<n: V(i,s) = V(i-1,s-1)*T(s-1,s)*E(s,i)*
*For state s=n: V(i,s) = max{ V(i-1,s-1)*T(s-1,s)*E(s,i), V(i-1,s)*T(s,s)*E(s,i).*

*The execution of the dynamic programming algorithm then proceeds as in the normal Viterbi algorithm. Since the computation of each entry of the dynamic programming matrix is done in constant time, we get a running time of O(n*L).*

c) (8 points) Consider a 2-state HMM with states set S = {A, B}, alphabet $\Sigma = \{0, 1\}$, and with transition, emission, and initial state probabilities given below:

Transition probability matrix:

	A	B
A	0.5	0.5
B	0.5	0.5

Emission probability matrix:

	0	1
A	0.7	0.3
B	0.4	0.6

Initial state probability:

A	B
1	0

Consider the sequence of observations:

```
X =   1  0  1  1  1  1  0  1  1  0  0  0

P=    A  A  B  B  B  B  A  B  B  A  A  A
```

What is the path *P* that is mostly likely to have generated *X,* i.e. the path that maximizes Pr[*P* | *X*]? Write the path below the sequence *X*, each state aligned to the symbol it emitted.

Hint: You could obtain the answer by manually executing the Viterbi algorithm, but this would take a lot of time. Instead, take a good look at the probability tables; this should suggest a big shortcut.

d) (4 points) Explain briefly (2-3 lines) how you got the solution you gave in (c).

We note that the transition probabilities are all equal. This means that the only relevant variables are the emission probabilities and initial state probability. Since the initial state probability of A is 1, the path must start in that state. Afterward, the optimal path is the one that use A when emitting 0 and B when emitting 1.

Question 3. (15 points)

Consider the following alignment problem:
Given:

- A short DNA sequence *R*
- A long DNA sequence *G*
- A substitution cost matrix *M*
- A cost *c* for gaps (linear gap penalty)

Find: A substring *G*' of *G* that, when aligned to the entire sequence *R*, achieves the highest possible alignment score. The output should be in the form of the starting and ending positions of *G'* in *G*. The algorithm does not need to report the alignment itself.

a) (10 points) Describe a modification to one of the alignment algorithms seen in class to obtain an algorithm that is guaranteed to produce the optimal solution to the problem. The algorithm should run in time O(|*R*| * |*G*|), where |*R*| is the length *R* and |*G*| is the length of *G*.

We can find the optimal alignment by a modification of the Needleman-Wunsch algorithm where the leading and ending gaps placed in sequence R have score zero.:

Let X(j,j) be the score of the optimal alignment of R(1…i) to G(1…j), where the leading gaps in placed in R have cost zero. X(i,j) is obtained using the standard NW algorithm, with two modifications:

1) *Initialization:*
   X(0,j) = 0 for all j
   *X(i,0) = c * i*

2) *Recurrence : same as in NW*
3) *Traceback: instead of performing traceback from X(|R|,|G|), we start from argmax$_j$ {X(|R|,j)}. The traceback procedure ends when we reach i=0.*

b) (5 points) The Blast algorithm was designed to solve problems similar to the problem presented in part (a). However, this algorithm is not guaranteed to produce an optimal solution. Suppose that the sequence R that is given as input is actually an exact substring of G, i.e. R matches a portion of G exactly. True or False: Once the indexing of G is done, the running time of Blast on input R will be $O(|R|)$, i.e. it will be independent of $|G|$. Explain your answer.

False. The running time will depend on the number of short perfect matches found between R and G. That number is proportional to $|G|$.

Question 4. (10 points)

Consider a set of 10 random DNA sequences, each of length *L*=1000 bp, where each nucleotide is selected randomly and independently with the following probabilities:
A: 10%, C: 30%, G: 40%, T: 20%.

a) (2 points) In the 10 sequences described above, which DNA word of length 6 has the largest expected number of occurrences? How many occurrences would you expect for that word?

GGGGGG is the word with the largest expected number of occurrences. That number is $10*(1000-6+1)*0.4^6$.

b) (3 points) Now, suppose the sequence ACTAGT is inserted once in each of the random sequences described in (a), at a randomly chosen position, by overwriting the 6-nucleotide sequence that was already there. What is the expected number of occurrences of ACTAGT in these 10 new sequences?

10 + expected number of matches occurring by chance at locations other than those where the motif was inserted. Since this motif is not self-overlapping, i.e. it is not possible to have two matches for the motif in overlapping portions of a sequence. We are left with 1000 – 11 – 5 positions where the motif could occur by chance. So the answer is:

$10 + 10*(1000-11-5)*0.1*0.3*0.2*0.1*0.4*0.2 = 10.47$

c) (2 points) In the sequences obtained in (b), what is the expected number of occurrences of the word found in (a)?

Word GGGGGG is also non-overlapping with ACTAGT, so the expected number of occurrences is:

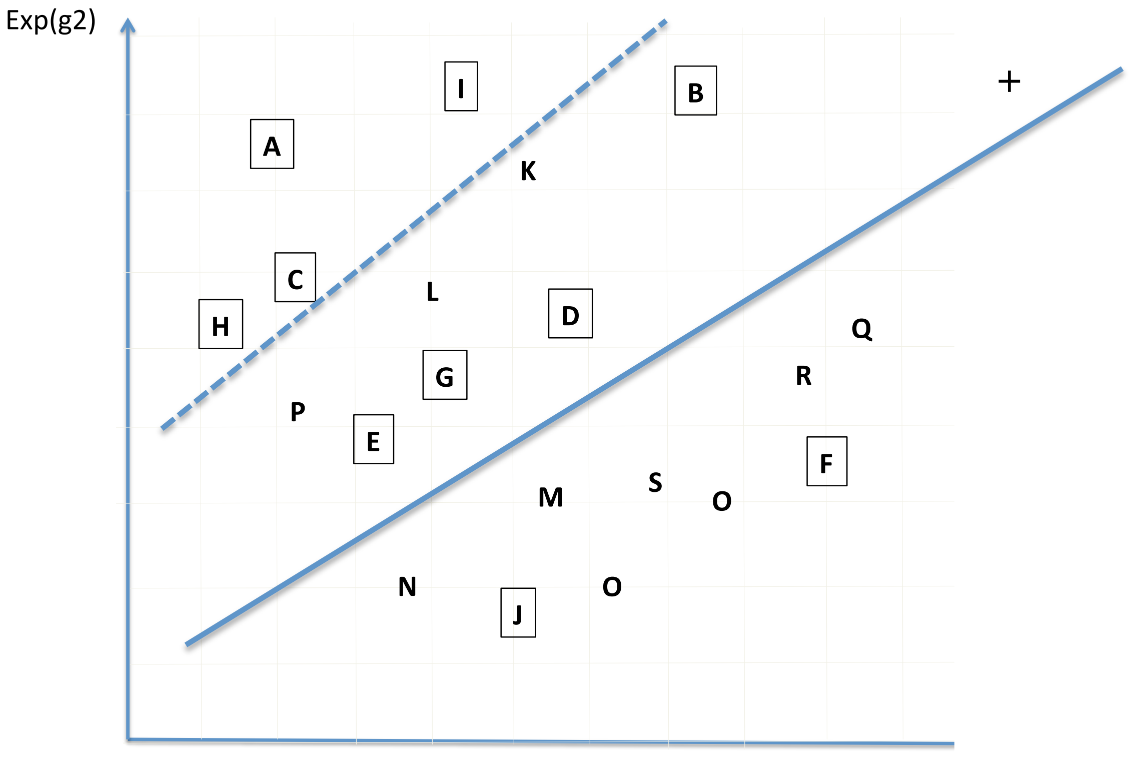
$10*(1000-11-5)*0.4^6 = 40.3$

d) (3 points) Your result in (b) and (c) should show that despite the fact that ACTAGT was inserted in each sequence, this word is not the one that is expected to be most common. However, if the random sequences were not of length 1000 bp but were instead of smaller length, at some point ACTAGT would become the word with the highest expected count. How short would the sequences to be in order for this to happen? More specifically: What is the largest value of *L* for which the expected count of ACTAGT exceeds the expected count of the word found in (a)?

We need to solve for L in the inequality:
$10 + 10*(L-11-5)*0.1*0.3*0.2*0.1*0.4*0.2 > 10*(L-11-5)*0.4^6$
This gives $L< (1 - 16*0.1*0.3*0.2*0.1*0.4*0.\ 2 + 16*0.4^6\ ) / (0.4^6 - 0.1*0.3*0.2*0.1*0.4*0.2)$, i.e. $L<263.2$

Question 5. (12 points)

Consider the following result of an experiment where we measured the expression of two genes, g1 and g2. Assume that samples A, B, …, J (shown with boxes in the figure below) come from patients with a specific disease (positive examples), while samples K, L, …, S (shown without boxes) come from healthy patients (negative examples).



a) (2 points) Draw the linear classifier that obtains the smallest number of classification errors (False-positives + False-negatives) on this data set.

b) (2 points) What is the sensitivity of your classifier on this data?

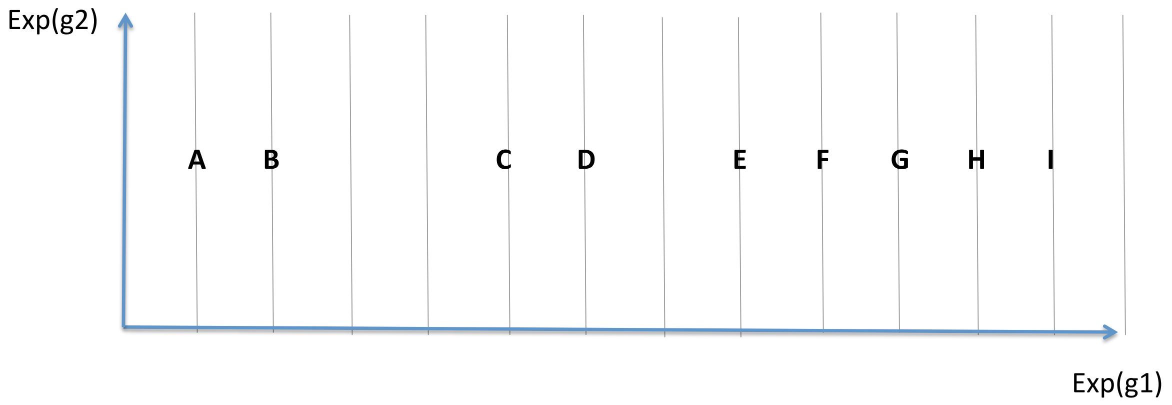
Sensitivity = 8/10

c) (2 points) What is the specificity of your classifier on this data?

Specificity = 7/10

d) (2 points) The positive predictive value of a predictor is defined as PPV = TP / (TP + FP). Draw using a <u>dotted line</u> the linear classifier that achieves the largest PPV. If there are multiple different classifiers that achieve the same optimal PPV, choose the one with the highest sensitivity.

e) (4 points) Now consider a different gene expression data set shown below.



Show what the result of both the single-linkage (left) and the complete-linkage (right) agglomerative clustering algorithm on this data, using the trees (dendrograms) showing the results of algorithm.

Single-linkage clustering dendrogram

(too lazy to draw it… using the standards tree parenthesis notation instead)

((A,B), ((C,D), ((E,F), ((G,H),I))))

Complete-linkage clustering dendogram

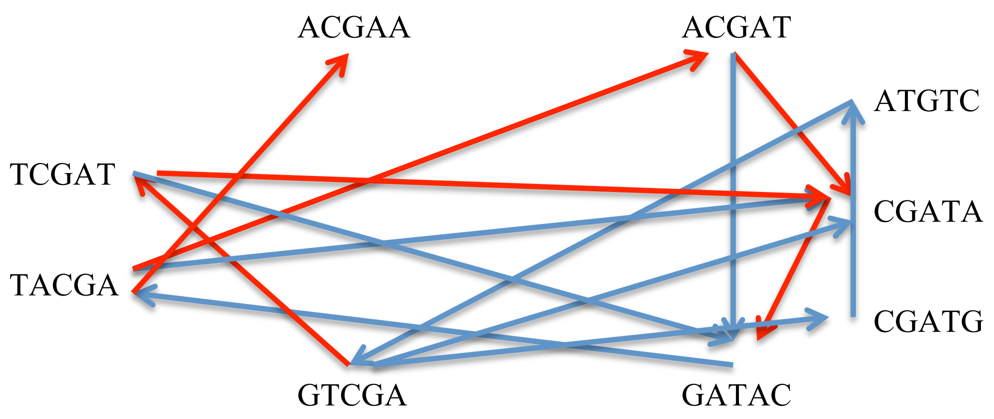
((A,B), ((C,D)), ((E,F), ((G,H),I)))

Question 6. (10 Points)

Consider the following set of 8 short reads obtained from a (toy) genome sequencing experiment.

ACGAA, ACGAT, ATGTC, CGATA, CGATG, GATAC, GTCGA, TACGA, TCGAT

a) (5 points) Suppose we want to assemble the genome from which these reads were obtained. Draw the overlap graph of this set of reads, and specify the weight of the edges. Only consider overlaps of 3 bases or more.



In the graph, red edges have a weight of 1 and blue edges have a weight of 2.

b) (5 points) What is the sequence of the mini-genome that is the most likely to have generated this set of reads?

There was an error in the question, because the graph does not have a Hamiltonian path.

Actually, the more meaningful translation of the genome assembly problem uses not the shortest Hamiltonian path (which may not exist), but the path corresponding to the Chinese Postman Problem, which is the shortest path that visits all the vertices *<u>at least</u>* *once.*

Page left blank intentionally. Use it if you need extra space.

Page left blank intentionally. Use it if you need extra space.