TIK-61.181 BIOINFORMATICS

AUTUMN 2000

EXERCISES

Julián Cobos Aparicio
St. Number: 55810J
julianc@cc.hut.fi

EXERCISES 1

a) Explain in detail how you would align the following sequences with pair HMM:s and with non-probabilistic methods:

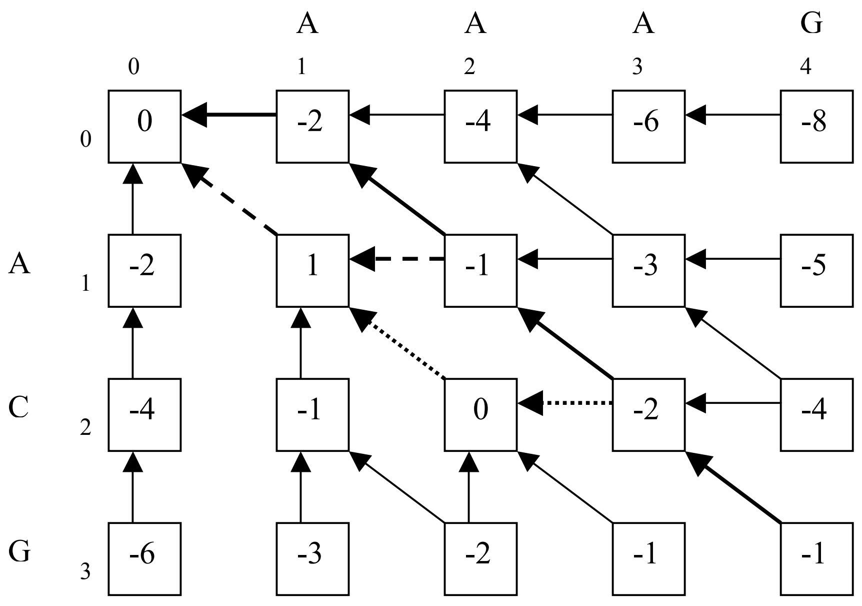
AAAG
ACG

To align the sequences with non-probabilistic methods we can use dynamic programming, with the basic algorithm. This is the formula for getting the score of each box:

$$a[i,j] = \max\begin{cases} a[i,j-1]-2 \\ a[i-1,j-1]+p(i,j) \\ a[i,j-1]-2 \end{cases}$$

$$where\ p(i,j) = \begin{cases} 1 & s[i]=t[j] \\ -1 & s[i]\neq t[j] \end{cases}$$

The resulting matrix from the application of the algorithm is the following:

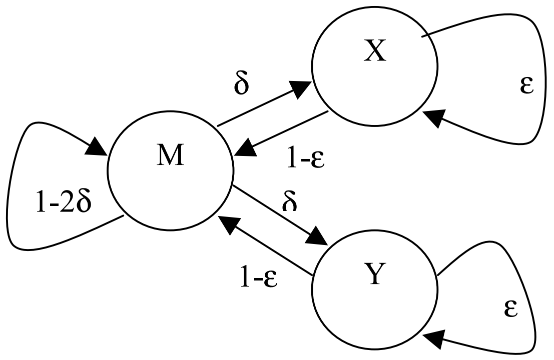


Therefore we obtain three possible alignments:

AAAG	AAAG	AAAG
– ACG	AC – G	A– CG
Optimal	Sub optimal	Sub optimal

To search the best alignment for the given sequences with probabilistic methods we can use pair HMMs with the Viterbi algorithm.

The HMM model would be:



The Viterbi algorithm calculates the alignment finding the most probable path in the model following these steps:

Initialization: $v^M(0,0) = 1$; all other $v^*(i,0)$, $v^*(0,j)$ are set to 0.

Recurrence: i = 1..n, j = 1..m;

$$v^M(i,j) = p_{x_i y_j} \max \begin{cases} (1-2\delta)v^M(i-1,j-1), \\ (1-\varepsilon)v^X(i-1,j-1), \\ (1-\varepsilon)v^Y(i-1,j-1); \end{cases}$$

$$v^X(i,j) = q_{x_i} \max \begin{cases} \delta v^M(i-1,j), \\ \varepsilon v^X(i-1,j); \end{cases}$$

$$v^Y(i,j) = q_{y_j} \max \begin{cases} \delta v^M(i,j-1), \\ \varepsilon v^Y(i,j-1); \end{cases}$$

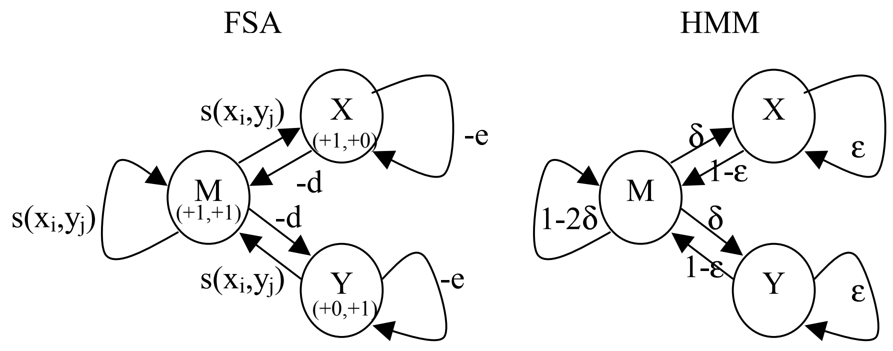
Termination:

$$v^E = \max(v^M(n,m), v^X(n,m), v^Y(n,m))$$

To get the best alignment we only have to keep pointers to trace back, saving at the end the residue emitted at each step of the path in the trace back.
At the end we also would get the probability of obtaining this optimal path.

b) Analyze briefly the relation of the probabilistic approach and the non-probabilistic approach for sequence alignment.

The Finite State Automaton (FSA) is the description of the most complex dynamic programming pair wise alignment algorithms. We also are able to take it as the basis for a more advanced probabilistic model (HMMs).



Using this probabilistic model and applying to it all the HMM theory and algorithms we will obtain some complementary information about the alignments that we cannot get with non-probabilistic methods, for example the reliability of the optimal alignment obtained, or other possible but not optimal alignments (suboptimal). With all this information we will be able to score the similarity of two sequences by using statistical probabilistic methods. It is also possible to modify the model for recognising local alignments.

By measuring the similarity of two sequences (we can do it using the Forward algorithm), with the HMM we can avoid the problem of identifying the correct alignment with dynamic programming when similarity is weak. Other advantage of HMMs is that we are able somehow to train the model with pairs of related sequences to find the optimal parameters of the model matrices to maximize the final probabilities. And then we can use these parameters for searching with new sequences.

c) Give an example of the possible uses of the MC-model and HMMs in Bioinformatics. Outline briefly the main differences between the two models.

Some possible uses of the Markov chains models and Hidden Markov models (HMM) are:

- Pair alignment of two sequences. Finding the best alignment of two sequences and measuring the similarity between them.
- Multiple alignments. Finding the best possible alignment of several sequences and giving a score to this alignment.
- Check if a particular DNA sequence belongs to a certain family based on common properties of the family, and what is the probability for it to belong to the family.
- Pattern searching in DNA sequences. (e.g. CpG islands)
- Identifying α-helix or –sheet regions in a protein sequence.
- Classification of sequences and fragments inside a database.

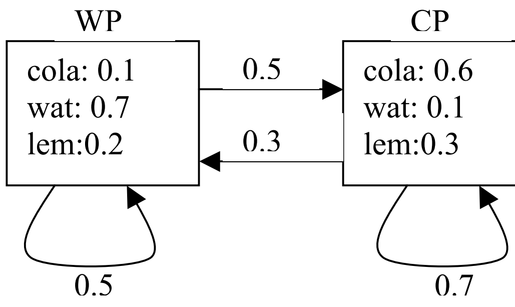
A HMM can be seen like a finite state machine in which the following state only depends on the present state (present but not passed memory), and we have a observations or parameters vector associated to each transition between states. It is possible thus to be said that a model of Markov have two processes associated: one hidden, no directly observable, corresponding to the transitions between states, and another observable one (and directly related to first), whose accomplishments are the vectors of parameters that take place from each state and which they form the pattern to recognize.
In a normal Markov model there is a one-to-one correspondence between the states and the symbols, this means that in a normal Markov model each state corresponds to a particular emission of a symbol, whereas in a HMM usually every state is able to emit all the symbols.

d) Relax. You are buying a drink from the notorious crazy coke machine while absent-mindedly solving some exercises. The machine can be in either of two states: cola preferring state (CP) and mineral water (WP) preferring state. When you put in a coin, the output of the machine can be described with the following probability matrix:

	cola	***Mineral water***	***lemonade***
CP	***0.6***	***0.1***	***0.3***
WP	***0.1***	***0.7***	***0.2***

Draw the state model graph with transition probabilities P(CP→WP) = 0.3 and P(WP→CP) = 0.5 and calculate the probability of seeing the output sequence {cola,lemonade} if the machine always starts off in the WP state. What kind of behaviour would make this HMM to a (visible) Markov model?

The state model graph for the HMM is the following:



We can calculate the probability of seeing the sequence {cola, lemonade} starting in WP, by using the forward algorithm as follows:

$f_{wp}(\text{cola}) = e_{wp}(\text{cola})\, \pi_{wp} = 0.1 * 1 = 0.1$
$f_{cp}(\text{cola}) = e_{cp}(\text{cola})\, \pi_{cp} = 0.6 * 0 = 0$

$f_{wp}(\text{lemonade}) = (f_{wp}(\text{cola})*a_{wp\text{-}wp} + f_{cp}(\text{cola})*a_{cp\text{-}wp})\, e_{wp}(\text{lemonade}) = (0.1*0.5+0*0.3) = 0.01$
$f_{cp}(\text{lemonade}) = (f_{wp}(\text{cola})*a_{wp\text{-}cp} + f_{cp}(\text{cola})*a_{cp\text{-}cp})\, e_{cp}(\text{lemonade}) = (0.1*0.5+0*0.7) = 0.015$

Therefore:

$P(\text{cola, lemonade} / \lambda) = f_{wp}(\text{lemonade}) + f_{cp}(\text{lemonade}) = 0.01 + 0.015 = 0.025$

f) Compare briefly the FSAs and pairwise HMMs in searching.

In probabilistic modelling we are able to sample the model with a very large amount of data, if this amount tends to infinite then the likelihood takes its maximum value for the model, and the parameters of this optimal model (emissions and transitions matrices) could be used to maximize the likelihood of the data.

Due to this, if the parameters of a pair HMM describe the statistics of pairs of related sequences well, then we should use that model with those parameter values for searching. We also could use the Bayesian model comparison with other model that gives a good description of the generation of random sequence.

However most currently used algorithms for these probabilistic models have two weaknesses. First they do not compute the full probability of the pair of sequences, summing over all alignments, but instead find the best match, or Viterbi path, in this case, a model whose parameters match the data is not necessarily the best search model. Second, regarded as FSAs, their parameters may not be readily translated into probabilities.

Therefore we can conclude that probabilistic models may underperform standard alignment methods if Viterbi is used for database searching, but if the forward algorithm is use to provide a complete score independent of specific alignment, then pair HMMs may improve upon the standard methods.

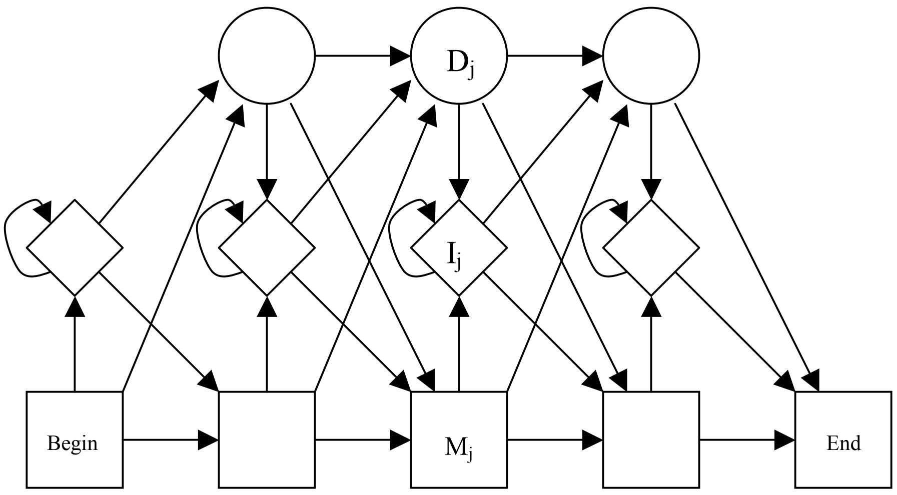
g) Analyze the relation of the profile HMMs, HMMs, and PSSMs.

Profile HMMs are a particular type of HMMs especially good to modelling multiple alignments. A particular feature of protein family multiple alignment is that gaps tend to line up with each other leaving solid groups without any gap.

A first probabilistic model to make comparison with these solid blocks is to specify probabilities to the observation of amino acid a in position i ($e_i(a)$) comparing it to the probability under a random model:

$$S = \sum_{i=1}^{L} \log \frac{e_i(x_i)}{qx_i} \rightarrow \text{Probability of a new sequence x}$$

These values behave like elements in a score matrix; this is why this method is called Position Specific Score Matrix (PSSM). And this method is used for searching for a match of the sequence x in a longer sequence. PSSMs are useful to find conservation information in protein families, but they do not take account of gaps. The best approach to do this is to build a HMM with repetitive structure of states, but different probabilities in each position. A PSSM can be viewed as a trivial HMM, to which we can add features to take into account insertions and deletions.



The profile HMM comes up from conditioning the pair HMM used for pair wise alignment (for example comparing sequences x and y) on emitting the sequence y as one of the sequences in its alignment. Because of this, it is possible to use the same Viterbi equations to find the most probable alignment of x to our profile HMM.

h) Assume we have a number of sequences that are 50 residues long, and that a pairwise comparison of two such sequences takes one second of CPU time on our computer. An alignment of four sequences takes $(2L)^{N-2} = 10^{2N-4} = 10^4$ seconds (a few hours). If we had unlimited memory and we were willing to wait for the answer until just before the sun burns out in five billion years, how many sequences could our computer align?

We have to align N sequences of length 50, and knowing that a single pariwise comparison takes 1 second and the alignment of 4 of those sequences takes 10^4 seconds, and we have five billion ($5*10^{12}$) years:

$5*10^{12}$ years = $15768*10^{16}$ seconds

Therefore:

$$(2L)^{N-2} = 10^{2N-4} = 15768*10^{16}$$

Taking logarithms:

$$(2N-4)*\log(10) = \log(15768*10^{16})$$

And:

$$N = \left\lfloor \frac{\log(15768*10^{16})+4}{2} \right\rfloor = \lfloor 12.09888 \rfloor = 12 \text{ sequences}$$

It means that in five billion years we can align 12 sequences with this computer.

i) Explain briefly the pros, cons, and the most useful applications of the different progressive alignment methods and profile HMMs in multiple alignment.

There are several methods that can be used for multiple alignment:

- Feng-Doolittle progressive alignment
- Profile alignment (CLUSTALW)
- Iterative refinement (Barton-Sternberg)
- Profile HMMs

The Feng-Doolittle method is quite fast due to its clustering algorithm (Fitch & Margoliash) but uses only pair wise alignment to determine the multiple alignments. This means that when a group of sequences has been aligned the method uses position-specific information from it to align the following sequence. Therefore when a subalignment has been built up, it does not change during the rest of the process.

The CLUSTALW method has a higher accuracy and flexibility than the previous one, and it is very simple for linear gap scoring, but it has similar weak points, the subalignments that already have been done remains without changes to the end of the alignment, and it aligns always whole sequences, never taking into account what fragments of the sequence are alignable or not.

The Barton-Sternberg method has the advantage that is more flexible than the progressive alignment methods, but its complexity is also bigger.

The use of profile HMMs for multiple alignment give us a more biologically realistic view of the problem, it is semantically better because it takes into account if the current fragment of the sequence is meaningfully alignable. On the other hand, it is difficult to apply it because the construction of the model is not obvious and the estimation of a multiple alignment from unaligned sequences is a very hard task.

j) Explain the reasons why there are methods for fragment assembly and physical mapping.

The problem of DNA sequencing becomes unapproachable when the continuous stretches are over a few hundred bases. However there are methods to split a long DNA molecule into random pieces and to produce enough copies of the pieces to sequence. Therefore, we can sequence fragments of them, but this method leaves us the problem of assembling the fragments, and that's why these methods exist.

However, even using these methods to sequence DNA, we are restricted to a few tens of thousands of DNA base pairs whereas a chromosome is a DNA molecule with around 10^8 base pairs. This means that these methods only enable us to analyze a very small fraction of the chromosome, not allowing us to analyze bigger structures in the DNA molecule. In order to do this there are methods to create maps of entire chromosomes or of significant parts of them, and they are called techniques for Physical Mapping of DNA.

l) Explain why two restriction sites for two different restriction enzymes cannot coincide.

The restriction site mapping technique uses these kind of "fingerprints" that describes uniquely the information in a given fragment of the DNA molecule (restriction enzyme). These fingerprints are used to compare the fragments mainly by overlapping to accomplish the DNA mapping.

One restriction enzyme could recognize one or more restriction sites, however two different restriction enzymes cannot recognize the same restriction site. A restriction site is obtained from information on the length of the fragments, a fragment length is thus its fingerprint. Therefore the restriction site represents uniquely a particular restriction enzyme.

EXERCISES 2

a) Prove that any unoriented permutation over n elements can be sorted with at most n-1 reversals.

In order to sort an unoriented permutation of n elements we can do reversals over 2 to n elements:

Reversal over 2 elements:	$\underline{62}374 \rightarrow 26374$
Reversal over 5 elements:	$\underline{62374} \rightarrow 47326$

It allows us to make, in the worst case, a reversal for every single element but the last two elements that can be ordered with a single reversal over them. Thus we would make n-1 reversals to sort the whole permutation simply choosing always the following number to order as the end of the reversal.

For example in a 6 elements permutation:

$$\underline{246531} \rightarrow 1\underline{35642} \rightarrow 12\underline{4653} \rightarrow 123\underline{564} \rightarrow 1234\underline{65} \rightarrow 123456$$

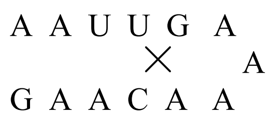
Therefore, we have sorted the permutation in 5 reversals.

b) Give an example of an RNA sequence where a knot may appear.

With a RNA sequence like:

```
1  2  3 4  5  6  7  8  9 10 1112 13
AAUUGAAAACAAG
```

We may find a knot if the base pairs: (r_5,r_{10}) and (r_4,r_9) belong to the set of pairs (S) of a possible RNA secondary structure for the given sequence:



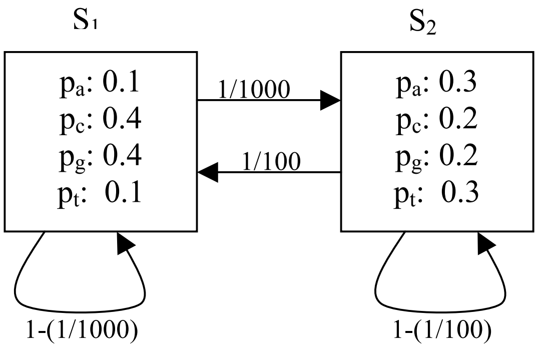
c) Consider a simple HMM that models two kinds of base composition in DNA. The model has two states fully interconnected by four state transitions. State 1 emits CG-rich sequence with probabilities (p_a, p_c, p_g, p_t) = {0.1, 0.4, 0.4, 0.1} and state 2 emits AT-rich sequence with probabilities (p_a, p_c, p_g, p_t) = {0.3, 0.2, 0.2, 0.3}.

a) Draw this HMM.

b) Set the transition probabilities so that the expected length of a run of state 1s is 1000 bases and the expected length of a run of state 2s is 100 bases.

c) Give the same model in stochastic regular grammar form with terminals, non-terminals, and production rules with their associated probabilities.

The HMM could be represented as follows with the transition probabilities set to a_{12}=1/1000 and a_{21}=1/100:



This model can be represented as the following stochastic regular grammar:

Terminals = {A, C, G, T}
Non-terminals = {S_1, S_2}

Production rules:

$S_1 \rightarrow AS_2$ (0.3*1/1000)	$S_1 \rightarrow AS_1$ (0.1*(1-1/1000))
$S_1 \rightarrow CS_2$ (0.2*1/1000)	$S_1 \rightarrow CS_1$ (0.4*(1-1/1000))
$S_1 \rightarrow GS_2$ (0.2*1/1000)	$S_1 \rightarrow GS_1$ (0.4*(1-1/1000))
$S_1 \rightarrow TS_2$ (0.3*1/1000)	$S_1 \rightarrow TS_1$ (0.1*(1-1/1000))
$S_2 \rightarrow AS_1$ (0.1*1/100)	$S_2 \rightarrow AS_2$ (0.3*(1-1/100))
$S_2 \rightarrow CS_1$ (0.4*1/100)	$S_2 \rightarrow CS_2$ (0.2*(1-1/100))
$S_2 \rightarrow GS_1$ (0.4*1/100)	$S_2 \rightarrow GS_2$ (0.2*(1-1/100))
$S_2 \rightarrow TS_1$ (0.1*1/100)	$S_2 \rightarrow TS_2$ (0.3*(1-1/100))

Note that this satisfys the requirement that the probabilities for all the transitions leaving each state sum to one.

d) Show that the Nussinov folding algorithm can be trivially extended to find a maximally scoring structure where a base pair between residues a and b gets a score s(a,b). (For instance, we might set s(G, C) = 3 and s(A, U) = 2 to better reflect the increased thermodynamic stability of GC pairs.)

The basic Nussinov algorithm, given a sequence x of length L with symbols $x_1,..,x_L$ scores as follows:

$$\delta(i,j)=\begin{cases}1 & \textit{if } x_i \textit{ and } x_j \textit{ are base pair}\\ 0 & \textit{if not}\end{cases}$$

So we can modify it in such way that:

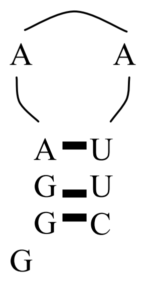
$$\delta(i,j)=\begin{cases}3 & \textit{if } x_i = G \textit{ and } x_j = C \textit{ or viceversa}\\ 2 & \textit{if } x_i = A \textit{ and } x_j = U \textit{ or viceversa}\\ 0 & \textit{else}\end{cases}$$

And apply the algorithm in the same way, and then at the end of the algorithm the score at the upper right corner of the matrix is the value for the maximum scored valid secondary RNA structure for the given sequence.
After that we can make the backtrace in the same way that with the basic Nussinov algorithm and we will obtain the structure with the highest score.

Applying the new algorithm to an example sequence (GGGAAAUCC) we obtain the following matrix:

	G	G	G	A	A	A	U	C	C
G	0	0	0	0	0	0	2	5	8
G	0	0	0	0	0	0	2	5	8
G		0	0	0	0	0	2	5	5
A			0	0	0	0	2	2	2
A				0	0	0	2	2	2
A					0	0	2	2	2
U						0	0	0	0
C							0	0	0
C								0	0

By back tracing we obtain the following structure of score 8:



e) The authors [SSK, Ch. 11] propose a method for computing a substituting matrix for amino acids. Characterize the differences from the PAM matrices. Comment the proposal. Where would you use each?

The method suggested by the author is to build a substitution matrix (WAC matrix), based on statistical descriptions of the amino acid environments, taking a measure of the tolerance of a microenvironment for one new amino acid based on the properties of the environment surrounding the lost amino acid.

Other substitution matrices (score matrices) like PAM matrices or BLOSUM have been obtained by empirical methods from measures of the frequency of substitution of alternative residues in each position in some experiments. These substitution matrices are mostly used for general protein searching and homology measuring by pairwise alignment, but they have demonstrated also a good performance in detecting related protein sequences to a certain family.