Tik-61.181 Bioinformatics

Exercises
Ville Viitaniemi 44374L

1 Exercises set 1

1.1 a)

For an alignment with a pair HMM we need to have a hidden Markov model to begin with. Such a model structure with begin, end, match and two symmetric mismatch states is shown in DEKM figure 4.2.

Before an alignment can be made, the model parameters (transition and emission probabilities) have to be specified. This can be done either heuristically or by estimating them from a large database of existing alignments. In this example case I would just pick some reasonable parameter values by hand.

When the model has been specified, it assigns probabilities for every possible alignment of the strings, so in principle we could just compute them all and pick the alignment with maximum probability. In practice, we avoid considering all the alignments by using the Viterbi algorithm to find the most probable path through the HMM.

For a non-probabilistic alignment any non-probabilistic method could be used. For simplicity's sake I would probably use a model which considers each base pair in the alignment independently and assigns a score of $+1$ for a matching base pair, -1 for mismatch and -2 for a gap in alignment. Then I would find out the maximum score by filling a dynamic programming matrix as described in DEKM section 3.2. After finding the maximum score, I would find out the corresponding alignment by backtracking in the DP matrix. (In this simple example, I might also just enumerate all the alignments, calculate their scores, and pick the maximum).

This alignment scheme could be improved for example by using some PAM matrix and some affine gap penalty function for scoring.

1.2 b)

The actual terms probabilistic and non-probabilistic sequence alignment do not seem to be explicitly defined in the book chapters mentioned. My interpretation is that a probabilistic alignment method is one that defines a probability for generating any alignment of the sequences (these probabilities are, of course, non-negative, sum to one etc.) and then somehow utilizes the probability distribution. Important class of probabilistic methods is methods defined around HMM's.

Any other other alignment method is classified as "non-probabilistic". (e.g. silly alignment method, which just randomly chooses one alignment among the possible ones). These methods themselves could include probabilistic elements (e.g. the generative model behind the estimation of PAM matrices for alignment scoring).

The alignment methods - probababilistic or not - typically can find one, optimal alignment (in case of probababilistic models . The "added" value of probabilistic models is, that one readily has a justifiable (mathematical) framework for looking at suboptimal alignments and their relation to the optimal one. One can sensibly ask questions like "how probable it is that the real alignment is among the 10 most likely alignments", that is, e.g. assess the quality of alignment. Also one can ask questions about the probability that the alignment has certain trait, e.g. that a certain section in one of the sequences is aligned with a gap.

An advantage of the non-probabilistic methods is, that some schemes might be very efficient what comes to computing time or memory requirements. It might also sometimes be difficult to devise a plausible probabilistic model.

An important class of non-probabilistic models are methods which assign (deterministically) scores to different alignments. By rescaling the scores one would get a probabilistic model (although possibly very inplausible or complicated). The same works also the other way around.

1.3 c)

(It was deduced, after some moments of confusion, that the abbreviation MC in the problem statement stands for Markov Chain instead of Monte Carlo)

Possible uses for Markov chains and HMM's in bioinformatics include locating CpG islands in DNA, grouping sequences into families or aligning sequences (pairwise or multiple).

For example if we are given a sequence of DNA and asked if it is from a CpG island, we can give answer, if we have a MC-model for both CpG islands and “usual” DNA. These models could easily be estimated from readily annotated DNA data when we use first order MC models with four states, A, T, G and C. Using the two models we could compute the likelihood of the observed sequence with each of the models. Then we could pick one of them using e.g. maximum likelihood or MAP criterion.

MC-models and HMM’s can both be considered as probabilistic state machines emitting symbols on states. In basic MC’s the state wholly determines the emitted symbol, in a HMM an emission probability density (over output symbols) is associated with every state. (MC is thus a HMM with very special, delta function like probability densities). In practice this means that a HMM has much more parameters than a MC with the same number of states. With MC’s usually many more state sequences are completely ruled out (have zero probability) considering the observed symbol sequence than with a HMM.

1.4 f)

In searching one tries to find a sequence from a database which best matches a query sequence.

When we do sequence alignment with FSA’s, we typically get a single best alignment for the sequences and a score for that. This score can then be used as a score for the database sequence.

With HMM’s we get a probability for every possible alignment. We thus can take contribution of each alignment into account when determining the similarity of the query and database sequences (by weighting according to probabilities). The sequence similarity is assessed by comparing the probaility for the sequences being generated by the pair HMM with the probability of the same sequences being generated by a random model. The comparison can be done in a full Bayesian manner.

In short, by using HMM’s one can take all the alignments into account in a principled manner. FSA procedure only considers the single alignment with highest score, ignoring all the others.

1.5 g)

HMMs (Hidden Markov Models) are a class of probabilistic models.

Profile HMMs are a special application (and perhaps the most important in the field of bioinformatics) of the general model. Profile HMMs summarise the statistical properties of biological sequence families.

PSSMs (Position Specific Score Matrices) assign each location in a biological sequence an independent probability density over the sequence alphabet. The score matrix entries are log-odds ratios of these probabalities against probabilities in a random model.

One trivially gets a HMM out of a PSSM by considering the locations as a linear chain of states, with transition probability one from a state to next one.

One gets useful profile HMMs from the above HMMs by adding silent and insert states. The transitions along the linear chain of states can no longer occur with probability one, the transitions to silent and insert states from the linear chain must also get some part of the probability.

1.6 g)

The idea of progressive alignment methods is to heuristically construct a succession of pairwise alignments.

One approach is to construct a distance tree of the sequences to be aligned. In this tree all ancestors jof a root of a subtree are joined together before the whole subtree is joined with some other subtree. This procedure clearly requires methods to do pairwaise alignment, either half can be either a single sequence or an alignment (of some subtree). One algorithm implementing this strategy is the Feng-Doolittle algorithm. There the subtrees are processed starting from ones representing smallest distances. One disadvantage of the approach is that gap locations in the alignment have to be decided in a very early phase during the algorithm (to keep alignments consistent with earlier ones, which are done for other, possibly very distant subtrees).

One way to improve the basic approach above is to use position specific information when performing an alignment between two alignments (either of which could be a single sequence). By this information positien specific gap penalties, knowledge about which sites are highly conserved and which are not, are meant. This profile information can be derived from the to-be-aligned alignments. A widely used algorithm for profile-based alignment is CLUSTALW.

Still, in these methods subalignments are fixed once they have made. To fight this problem one can iteratively take some parts away from the final alignment and try to realign them with the remaining parts. Obviously one can get better quality alignments this way but with the cost of extra computing time.

Instead of progressive aligning also HMMs are often used for multiple alignment. The main advantage of HMMs is that instead of more ore less heuristic sum-of-pairs scoring one can use more principled model for sequence generation from one 'master' template of the family.

HMMs can be used for multiple alignment principally in two ways.

If one already has a profile HMM for a sequence family, it is pretty straightforward to align sequences to this model (e.g. with the Viterbi algorithm). Very apparent problem is to somehow obtain the model. One might have this readily available, for instance, if set of sequences of the are picked and a profile HMM is made from them.

Another alternative is to estimate both the profile HMM and to alignment to it from the data. This can be accomplished e.g. using Baum-Welch type algorithm or Viterbi training. If this method works, it can produce high quality alignments. The obvious problem is that computational cost is high. Also the optimisation easily end in a bad local optimum. This can be helped with different randomisation methods (e.g. noise injection into Baum-Welch estimation) but this makes the computational cost still higher.

1.7 j)

The fragment assembly methods are needed when one wants find out the base sequence of fairly long stretch of DNA. Current technology is namely able to directly sequence only stretches of a few hundred bp's. The solution is to duplicate the DNA stretch under inspection, randomly cut the copies to short pieces, segment each of them individually, and then deduce the sequence for the DNA stretch from these subsequences. It is this last step where fragment assembly methods are needed.

Physical mapping concerns even longer pieces of genome. These pieces are so long that their complete sequence cannot be found out even with fragment assembly methods. To inspect genome on such a larger scale, one uses physical maps, which indicate the location of certain marker sequences in the piece of genome. With help of a physical map one can e.g. locate a given (long enough) piece of sequence in DNA. Once again, special methods are needed since one needs to splice the to-be-mapped DNA in small fragments in order to be able to process them with current technology.

1.8 l)

(Exercise of 5.5 in S&M)

If the restriction sites for two different restriction enzymes would coincide, the enzymes would be sensitive to exactly the same sequence fragments. This would mean that the complementary sequences (which appear in the enzymes) also were the same and the enzymes would not be different.

Actually, I don't have enough knowledge to rule out the possibility of two restriction enzymes having different structures even if their complementary sequence is the same. Neither can I exclude the possibility that some enzymes would have complementary sequences such that one is the substring (or even prefix/suffix) of some other.

Anyway, when doing a hybridisation experiment, one can be pretty sure that these unlikely cases would occur, as we are allowed to pick the set of enzymes ourselves.

2 Exercises set 2

2.1 e)

The information that is inserted in the WAC matrix and to PAM matrices come from different sources.

Essential part of information input into the PAM matrices comes from the fact that it takes as input two aligned sequences. Then it is is pretty straightforward to estimate the matrix entries just by counting frequencies of different mutations. The mutation process itself is modelled as very simple, one just considers which single amino acid is replaced and by which one.

WAC matrix estimation takes different viewpoint to amino acid mutation. One just assumes that amino acids that easily mutate to each other often have similar chemical environments. Therefore, it is necessary to consider just single amino acids, no alignment of sequences is needed. In contrast to total ignorance of PAM method the chemical environments of different amino acids are compared in detail. The WAC matrix is based on similarity between the typical chemical environments.

Because sequence alignment is needed to estimate PAM matrices, only some special amino acid sequences (homologous) and strictly speaking PAM matrices would not necessarily be valid for other types of sequences.

WAC method does not need aligned input data, so it is probably easier to have representative input data for estimation. But one can ask if the chemical environment statistics are optimal or even good enough (could they still be markedly improved ?).

Is it a good idea to reduce the data to matrix form? Of course one throws away a vast amount of data when one summarizes the complete chemical structure, but simplifications are needed to make fast database search possible. One can also say that this is demonstrate the methodology by comparing the results with well known PAM. There are many algorithms which one can very easily change to use WAC instead of PAM matrices.

Experiments show that generally the PAM250 matrix and the WAC matrix are about as sensitive (in protein family search). In some cases the one is more sensitive, in some cases the other. I personally would not know which one to use, probably I'd try both (or just choose the more traditional PAM for convenience).

2.2 e)

The equation to be explained is

$$f(\mathbf{t}) = \sum_i g_1(i,t_i) + \sum_i sum_{j>i} g_1(i,j,t_i,t_j) + \cdots + \sum_i sum_{j>i} \cdots \sum_{l>k} g_m(i,j,\cdots,k,l,t_i,t_j,\cdots,t_k,t_l)$$

There's not that much explaining in the equation. The equation states that f, the score function for a protein threading in general form, is function of the normalized set $\mathbf{t} = (t_1, \cdots, t_m)$ (a set of m integers) when the protein sequence and the core template with m core segments are given.

The equation also states that the score function might contain terms that involve one core segment alone, two of them together, three of them together and so on up to all m together. The summations with conditions on summation indices take care that each set of segment numbers appears exactly once in the summation (excluding any self interaction of a segment with itself).

We make note that the i-segment term is not guaranteed not to include a interaction among less than n segments, so we could as well retain only last term while still keeping the function general, be choosing a suitable g_m. Actually even the summation is not needed since there's only one way of picking a ordered sequence from the set of integers $(1, \cdots, m)$. Therefore, we could keep the same generality and write

$$f(\mathbf{t}) = g(t_1, \cdots t_m)$$

The reason for using the longer form is to make it more explicit what type of terms the score function might contain.

2.3 e)

Nowadays it is straightforward to obtain vast amounts of raw gene regulation data using DNA microchips. In principle it should be possibly infer the regulatory structure of the genome from this data. However, currently problem is still unsolved. It would mean a great advance for mankinds knowledge of biology if a practical solution to this problem is found.

The computatinal problem "Maximum gene regulation" very directly relates to this problem of actual physical world. The possible labeling of edges is given by input data, that is, it is deduced from the measurement data which genes may possibly regulate other ones.

It's then the computational procedure's task to rule out all the labelings of genes, which are inconsistent with the measurement data (the edge labels). This method ruling out impossible alternatives and hoping that a feasible one (or couple of them) remains is often quite efficient method of doing inference. (Of course not the only one). If too many possible alternatives remain after the procedure, we might try to devise more experiments to rule out some of them.

To justify a problem, it is enough to show that its solutions could be useful and on the other hand the problem really is a problem, that is, difficult to solve. As Chen shows, the problem indeed is NP hard, even with some simplifications. So, it is justified to look for methods of solving this problem.

2.4 f)

A small Tamayo-traditional SOM dictionary is in the table: (the "traditional" notation taken from Kohonen's book, 2nd ed.)

Furthermore it can be noted that the Tamayo paper fixes a specific form of decreasing learning rate $\alpha(t)$ $1/(t+const.)$ and uses a bubble neighbourhood of decreasing radius as neighborhood function.

	Tamayo notation	traditional SOM notation
position of a node, reference vector	$f_i(N)$	$m_N(i)$
input data	P	$x(t)$
winning unit	N_P	$c = argmax_i \lvert\lvert x - m_i \rvert\rvert$
combined learning rate and neighbourhood function	$\tau(d(N, N_p), i)$	$h_{ci}(t)$ often $h_{ci} = \alpha(t) \times g(\lvert\lvert m_c - m_i \rvert\rvert)$