19.1.1 Probability that two sequences are related

For example, given a pair sequences, x and y, we can ask what probability that two sequences are related without relying on any particular alignment. This is given by the quantity

$$P(x, y) = \sum_{\pi} P(x, y, \pi)$$

where the sum is over all possible alignments, π. As in the standard HMM case, we calculate this quantity via the forward algorithm or the backward algorithm.

19.1.2 Sampling alignments

Further, instead of relying on a single alignment, we could sample possible alignments in proportion to their probability using the techniques described in Section 18.8.

That is, we could calculate the forward $f_M(i, j)$, $f_X(i, j)$ and $f_Y(i, j)$ from the forward algorithm and then traceback to find a state path which is an alignment. The traceback from the current state chooses the next state (M, X, or Y) at each step according to it's contribution to the probability of the current state. This is exactly the technique described in the Section 18.8.

19.1.3 Probability that x_i and y_j are aligned

Consider two residues, x_i and y_j, in our given sequences x and y. What is the probability that they are actually aligned to each other? Write $\langle x_i, y_j \rangle$ to mean x_i and y_j are aligned, so the probability of interest is $\Pr(\langle x_i, y_j \rangle | x, y)$. We could estimate this probability by sampling many alignments using the technique described above and counting the proportion of times that $\langle x_i, y_j \rangle$.

But it turns out that we can calculate this number exactly using the general technique of finding the posterior probability of being in a state at some time described in Section 18.4. x_i is aligned to y_j exactly when the HMM is in state M at (i, j). Thus

$$\Pr(\langle x_i, y_j \rangle | x, y) = P(\pi(i, j) = M | x, y) = \frac{P(\pi(i, j) = M, x, y)}{P(x, y)} = \frac{f_M(i, j) b_M(i, j)}{P(x, y)}.$$

The last equality is exactly the same as the one derived in Section 18.4 and, as usual, the quantity $P(x, y)$ can be calculated using either the forward or backward algorithm.

19.2 Profile HMMs

The canonical problem in genetics is to find a group of sequences that are homologous. In particular, we are interested in finding homologous genes that share a similar function. We say such sequences or genes belong to the same family in the sense that they share a common ancestor and have maintained the same (or similar) functionality. They may be the same sequence in different species or in the same species but in different parts of the

genome (having arrived there through duplication). Sequences in the same family will often have features in common, particularly where they share the same function and, therefore, the same basic secondary structure.

If we can characterise these families accurately, by finding features that almost certainly share in common and identifying regions where more variation is seen, we will be able to better align sequences known to belong to the family and more easily identify other members of the family. We achieve this characterisation by modelling the family using an HMM, known as a *profile HMM*.

We'll start by assuming that we are given a family of homologous sequences that are already aligned into a multiple sequence alignment (MSA). See a very small example in Figure 9.

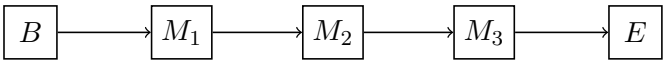
```
VGA--HAGEY
V----NVDEV
VEA--DVAGH
VKG------D
VYS--TYETS
FNA--NIPKH
IAGADNGAGV
```

Figure 9: Ten columns from a given multiple alignment of seven globin sequences.

From a given alignment, we wish to characterise a typical sequence in the alignment at each position. Once we have made this characterisation, we could use it to search for other sequences (or parts of sequences) that fit the profile and so are candidates to be members of this family.

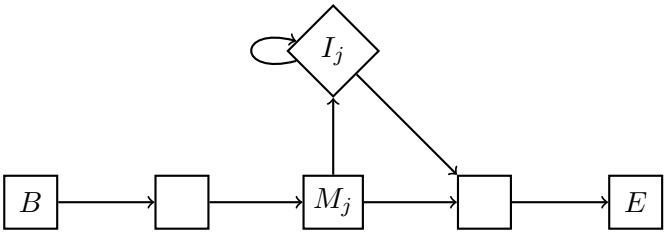
We model the alignment as an HMM, where each position in the alignment is a state with a distinct probability of emitting the various residues.

We'll start by supposing the alignment that is largely free of gaps (by ignoring, say, columns in the alignment that are more than 50% gaps). With each position in the alignment, associate a state in the HMM. Call this state a match state and label the ith match state M_i. The (ungapped) alignment is then modelled as a HMM with only the trivial transitions, M_i to M_{i+1} (with additional begin and end states). Let emission probabilities from the i match state be e_{M_i}. A model for a MSA of length 3 looks like:



Now let's allow gaps in the alignment. How do we handle them? To handle an insertions (with respect to the alignment — parts of a sequence x that are not matched by anything in the model) we introduce a new set of states I_i which matches the residues in x after i to a gap. These states have emission probabilities $e_{I_i}(a)$ which we set to the background rate: $e_{I_i}(a) = q_a$.

There is a transition from M_i to I_i, a loop at I_i, and a transition from I_i to M_{i+1}.

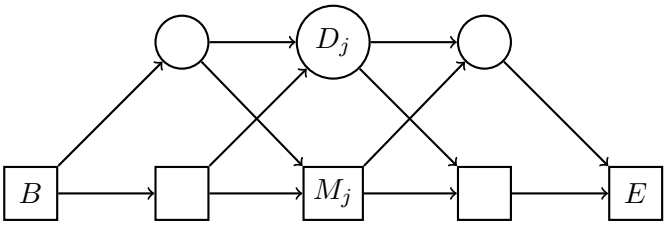


A gap of length k therefore has log-odds score

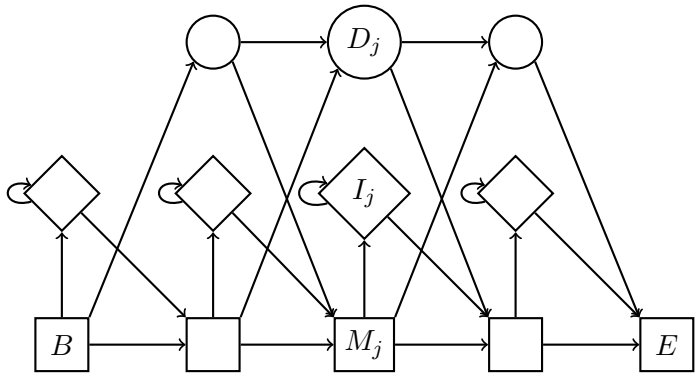
$$\log a_{M_i I_i} + \log a_{I_i M_{i+1}} + (k-1) \log a_{I_i I_i},$$

the same as an affine gap penalty.

A deletion relative to the model corresponds to skipping ahead in the model. To allow transitions from every match state to every other state ahead of it in the model would introduce too many transitions (how many would we need?) so we introduce instead a silent delete states D_i next to every match state. Silent states emit no residues. We allow transitions $a_{M_{i-1}D_i}, a_{D_i M_{i+1}}$ and $a_{D_i D_{i+1}}$.



A full model incorporates both insert and delete states and is drawn below. Notice that we have not drawn transitions between I and D states though it simplifies computation to allow them (we'd allow $D_j \to I_j$ and $I_j \to D_{j+1}$).



This profile HMM model is equivalent to a pair HMM model for pairwise alignment where the given MSA is summarised into a single sequence y . So when we seek to fit a candidate sequence x to our profile HMM, it is as though we are aligning x and y with a pair HMM (with appropriately chosen emission probabilities).

19.2.1 Estimating the parameters of a profile HMM

We can choose the parameters of the profile HMM using the empirical counts (A_{kl} and $E_k(b)$) with pseudo-counts added. Remember that the number of possible transitions is limited: in our full drawn model we only allow non-zero transitions for $a_{M_iM_{i+1}}$, $a_{M_iD_{i+1}}$, $a_{M_iI_i}$, $a_{I_iI_i}$, $a_{I_iM_{i+1}}$, $a_{D_iD_{i+1}}$ and $a_{D_iM_{i+1}}$. We have emissions for all possible residues at each site so we add pseudo-counts to ensure $e_{M_i}(a) > 0$ for all a. Then we use the rule we saw earlier to estimate transition and emission probabilities:

$$a_{kl} = \frac{A_{kl}}{\sum_j A_{kj}} \text{ and } e_k(b) = \frac{E_k(b)}{\sum_j E_k(j)}.$$

A simple way of assigning pseudo-counts is to add 1 to all scores (including zero scores). There is a lengthy discussion of which pseudo-count values to choose in the Durbin et al book and many more sensible schemes are proposed – none are particularly complicated but we don't have time to cover them all here.

We assume emissions from insert states are at the background rate (that is, that rate the residue occurs at at any position).

In the example in Figure 9, the first column has all seven sequences in the match state with 5 Vs, and 1 each of F and I. The 17 other possible residues are not observed, so have frequency 0. Adding a pseudo-count of 1 to each observed frequency gives us emission probabilities $e_{M_1}(V) = 6/27$, $e_{M_1}(F) = e_{M_1}(I) = 2/27$ and the other 17 residues have $e_{M_1}(b) = 1/27$. 6 of the 7 transitions to the next column are to the match state again while one is to a delete state. Again, adding pseudo-countsof one gives $a_{M_1M_2} = 7/10$, $a_{M_1D_2} = 2/10$ and $a_{M_1I_1} = 1/10$. Transitions from the insert and delete states are just based on the pseudo-counts here.

19.2.2 Finding matches

Once the model has been set and the parameters estimated, we can set about seeing whether other sequences match the family. Call M the model and x a sequence we wish to test against the model. To do so, we align proposed sequences to the family using the now familiar tools of the Viterbi algorithm, giving the Viterbi path π^* (and the associated score, $P(x|\pi^*, M)$) or, better, the full probability of a sequence summed over all alignments, $P(x|M)$.

These scores can be used to compare the hypotheses that x belongs to family M or x is no more like M than we would expect at random. Call R the random model and let $P(x|R) = \prod_{i=1}^{L} q_{x_i}$ be the likelihood of x under the random model where q_a is simply the background rate of occurrence of residue a. If the log ratio $\log(P(x|M)/P(x|R)) > 0$, x is more likely associated with the modelled family than not. To find more certain matches, we may chose a threshold that this ratio must exceed before we call it a member of the family.

Note that if we wish to fit new found sequence x to the family, we can simply use the Viterbi alignment to align it to the family and update parameters of the model.

19.2.3 Alignment with a known profile HMM

The simplest case is when we have a known aligned family to which we have already fitted a profile HMM and we wish to add a number of sequences to the profile. In this case, we use the Viterbi algorithm to find the most probable alignment for the new sequences.

The Viterbi path will consist of matches, insert and delete states. At the delete states, we add a gap character, –, to the sequence we are aligning. At an insert state, the unaligned residue of the sequence we are aligning is emitted, forcing a gap like character in the already aligned profile. In the profile, we placeholder character, . , at these positions. Note that if there are multiple insertions in different sequences at a position, there are many ways to align them against each other. Since we believe insertions are not shared between all members of the family, we can set an arbitrary rule for aligning them, such as using simple left-justification.

19.2.4 Alignment from unaligned sequences with HMMs

If we do not have a pre-existing aligned family and/or profile HMM, we need to first specify an HMM and then use the Baum-Welch algorithm to estimate the parameters. After we have done that, we are in the same position as above and can construct the MSA from the fitted profile HMM.

A rule of thumb for specifying the HMM in the absence of prior knowledge is to allow M match states where M is the average length of the training sequences. In general, it is difficult to fit an HMM of this size. A number of heuristics have been developed to avoid local maxima.

ClustalΩ implements this method and is quick enough to align thousands of sequences in reasonable time. Currently, ClustalΩ can only be used for protein sequences. See Fabian Sievers et al, 2011, Molecular Systems Biology 7, `http://www.nature.com/msb/journal/v7/n1/full/msb201175.html` for a full description.

19.3 Gene finding

A DNA sequence can roughly be divided into two types of region: genes and non-genes or inter-genic regions. Genes are regions that code for proteins which actually perform biological functions in an organism so these regions are of primary interest. The role played by intergenic regions is not yet clear and it is often referred to as 'junk DNA'.

We are interested, then, to find a method of finding regions of a given sequence that are genes. To do so, we need to look at how a gene is structured along a sequence. Recalling that a sequence has a direction with one end being 5' end, the other being the 3' end. Moving forward in the sequence is going from the 5' end towards the 3' end.

Our model of a gene has the following elements occurring in the order listed here: an inter-genic region, a promoter region, a 5' un-transcribed region, a series of *exons* and

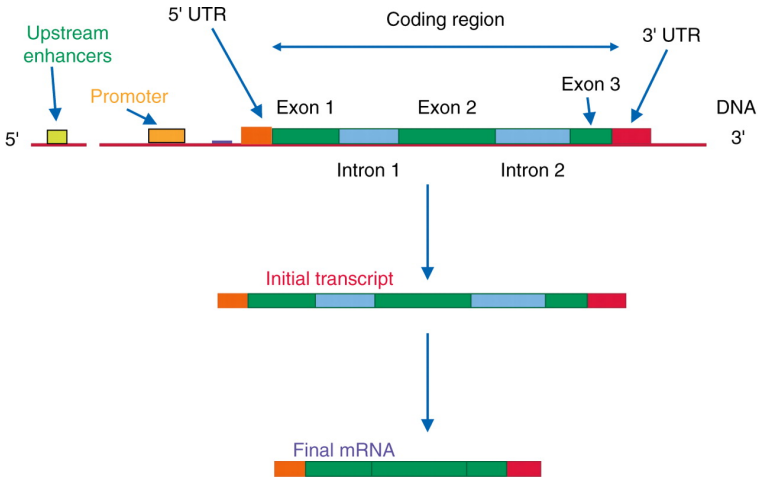


Figure 10: Gene structure and transcription. The DNA of the coding region is composed of exons (coding DNA) interspersed with introns (non-coding DNA) and is flanked by untranslated regions (UTRs). Upstream of the coding regions within the gene are DNA sequences that control (promoter) and regulate (enhancers) gene expression. During transcription, the initial nuclear transcript includes RNA sequence complementary to the entire coding region and the UTRs. In a subsequent step, the introns are spliced out to form mRNA which translocates to the cytoplasm where it is translated into protein. Source: `http://dx.doi.org/10.1093/bja/aep130` by UoA subscription

introns, a 3' un-transcribed region, a poly-A region and then back to another inter-genic region.

These regions are described in more detail below.

Inter-genic region: A non-coding region between genes.

Promoter: a region of DNA that facilitates the transcription of a particular gene. Promoters are located near the genes they regulate, on the same strand and typically upstream (towards the 5' region of the sense strand). For the transcription to take place, the enzyme that synthesizes RNA, known as RNA polymerase, must attach to the DNA near a gene. Promoters contain specific DNA sequences and response elements that provide a secure initial binding site for RNA polymerase and for proteins called transcription factors that recruit RNA polymerase. These transcription factors have specific activator or repressor sequences of corresponding nucleotides that attach to specific promoters and regulate gene expressions.

As promoters are typically immediately adjacent to the gene in question, positions in the promoter are designated relative to the transcriptional start site, where transcription of RNA begins for a particular gene (i.e., positions upstream are negative numbers counting back from -1, for example -100 is a position 100 base pairs upstream).

In eukaryotes, the process is complex and promoters may occur hundreds of base-pairs upstream.

In prokaryotes, the promoter consists of two short sequences at -10 (that is 10 bases upstream from the UTR and is called the Pribnow box which typically looks like TATAAT) and at -35 (the -35 element, usually TTGACAT). The promoter regions are not transcribed to RNA.

Untranslated regions (UTR 5' or 3') : These are regions immediately flanking the translated region. They are transcribed into RNA but not translated into proteins.

Exons and introns: An exon is transcribed into RNA and is further translated into a protein. An intron is transcribed into a form of RNA and then spliced out of the RNA sequence that finally gets translated in a protein. In any gene, there could be one or many exons and zero or many introns. Exons and introns alternate along the sequence.

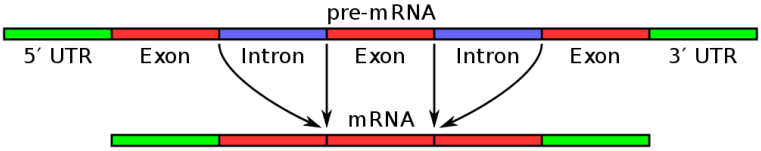


Figure 11: A figure showing how the transcribed precursor to messenger RNA includes the UTRs, exons and introns. The introns are spliced out to form the messenger RNA. The Exons in the mRNA are translated into proteins. Souce: `http://en.wikipedia.org/wiki/File:Pre-mRNA_to_mRNA.svg`

Poly(A) signal: After the 3' UTR on the RNA, a number of adenines (As) is added — this is called the poly(A) tail. The poly(A) signal, or polyadenylation signal, is thus a stretch of DNA that signals to the RNA transcription mechanism to begin the addition of the poly(A) tail.

A method first described in Burge and Karlin 1997 (see `http://www.ncbi.nlm.nih.gov/pubmed/9149143`) describes a generalized HMM incorporating all these regions. See Figure 12 for a sketch of the structure of the HMM. The states of the HMM are N (corresponding to an inter-genic region), P (promotor), F (5' UTR), E (exons), I (introns), T (3' UTR) and A (poly(A) signal). The multiple states for introns, I_0, I_1 and I_2 and exons E_0, E_1 and E_2 indicate the relation of the reading frame of the exon to the reading frame of the initial exon (E_{init}). Recall that three bases of DNA code for a single amino acid, with each group of 3 bases call a codon. If an exon or intron does not have length that is a multiple of 3, then the start of the next exon or intron may be out of phase with it. A subscript of 0, 1 or 2 represents in phase or lagging 1 or 2 steps out of phase, respectively.

The GHMM produces a set of states $q = q_1 \dots q_n$ with an associated set of lengths (durations) $d = d_1 \dots d_n$ and for each state q_i, it produces a sequence of length d_i according to a probability model associated with the state q_i. Algorithms to analyse sequences according to this model are implemented in Genscan and GlimmerHMM.

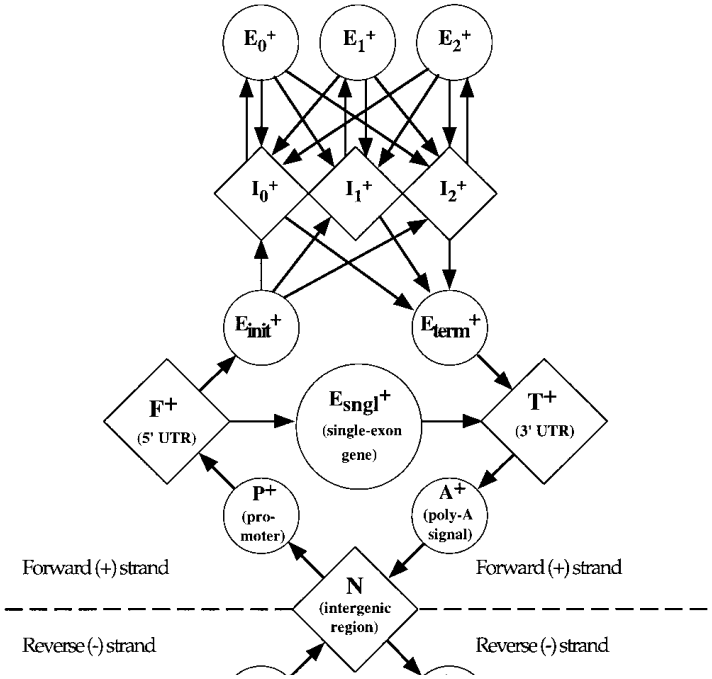


Figure 12: The structure of the (generalised) HMM used for gene finding from Burge et al 1997. See text for details. The full HMM includes a mirror image corresponding to the reverse strand which has been deleted here.

20 Reconstructing trees

The fastest ways of constructing trees rely on defining a distance between sequences. We have already one method that does this: UPGMA in Section 17.3. We looked at UPGMA in the context of multiple sequence alignment where a sensible choice of distance between sequeces to use was $D(x, y) = -\log S_{eff}(x, y)$. We'll briefly look at other, more widely used distance measures.

20.1 Defining distances between sequences

There are numerous ways of defining distances between sequences. The simplest, for an aligned pair of sequences x and y of length L is to count the number of positions where they differ, D_{xy} say, and define the distance to be

$$f_{xy} = D_{xy}/L,$$

which is simply the fraction of sites at which they differ. This method works well for related sequences where f is expected to be small, but doesn't grow as much as we would like as sequences become less and less related since even unrelated sequences share many bases in common due to chance.

The Jukes-Cantor distance is based on the simplest model of sequence evolution where mutations between all four bases are equally likely. The distance includes a correction for the fact that unrelated sequences will agree simply due to chance. The distance is defined by

$$d_{xy} = -\frac{3}{4}\log\left(1 - \frac{4f_{xy}}{3}\right).$$

Since the background level of dissimilarity (given by f_{xy}) for unrelated sequences is $\frac{3}{4}$, $(1 - \frac{4f_{xy}}{3})$ tends to zero as sequences become more unrelated so d_{xy} tends to infinity for unrelated sequences.