In practice, as for pairwise alignment, one of the local alignment methods may be more sensitive for finding distant matches. We discuss these in the next section.

We have a choice of ways to score a match to a hidden Markov model. We can either use the Viterbi equations to give the most probable alignment π^* of a sequence x together with its probability $P(x,\pi^*|M)$, or the forward equations to calculate the full probability of x summed over all possible paths $P(x|M)$.

In either case, for practical purposes the result we want to consider when evaluating potential matches is the log-odds ratio of the resulting probability to the probability of x given our standard random model

$$P(x|R) = \prod_i q_{x_i}.$$

We therefore show here versions of the Viterbi and forward algorithms that are designed specifically for profile HMMs, and which result directly in the desired log-odds values. Note that changing to log-odds does not change the result; we could have subtracted the random model log score afterwards. However, it is cleaner and more efficient. Another practical reason for working in log-odds units is to avoid problems of underflow when working with raw probabilities, as we discussed in Section 3.6.

Viterbi equations

Let $V_j^{\mathrm{M}}(i)$ be the log-odds score of the best path matching subsequence $x_{1\ldots i}$ to the submodel up to state j, ending with x_i being emitted by state M_j. Similarly $V_j^{\mathrm{I}}(i)$ is the score of the best path ending in x_i being emitted by I_j, and $V_j^{\mathrm{D}}(i)$ for the best path ending in state D_j. Then we can write

$$\begin{aligned}
V_j^{\mathrm{M}}(i) &= \log\frac{e_{\mathrm{M}_j}(x_i)}{q_{x_i}} + \max\begin{cases} V_{j-1}^{\mathrm{M}}(i-1) + \log a_{\mathrm{M}_{j-1}\mathrm{M}_j}, \\ V_{j-1}^{\mathrm{I}}(i-1) + \log a_{\mathrm{I}_{j-1}\mathrm{M}_j}, \\ V_{j-1}^{\mathrm{D}}(i-1) + \log a_{\mathrm{D}_{j-1}\mathrm{M}_j}; \end{cases} \\
V_j^{\mathrm{I}}(i) &= \log\frac{e_{\mathrm{I}_j}(x_i)}{q_{x_i}} + \max\begin{cases} V_j^{\mathrm{M}}(i-1) + \log a_{\mathrm{M}_j\mathrm{I}_j}, \\ V_j^{\mathrm{I}}(i-1) + \log a_{\mathrm{I}_j\mathrm{I}_j}, \\ V_j^{\mathrm{D}}(i-1) + \log a_{\mathrm{D}_j\mathrm{I}_j}; \end{cases} \\
V_j^{\mathrm{D}}(i) &= \max\begin{cases} V_{j-1}^{\mathrm{M}}(i) + \log a_{\mathrm{M}_{j-1}\mathrm{D}_j}, \\ V_{j-1}^{\mathrm{I}}(i) + \log a_{\mathrm{I}_{j-1}\mathrm{D}_j}, \\ V_{j-1}^{\mathrm{D}}(i) + \log a_{\mathrm{D}_{j-1}\mathrm{D}_j}. \end{cases}
\end{aligned} \tag{5.1}$$

These are the general equations. In a typical case, there is no emission score for $V_j^{\mathrm{I}}(i)$ because we assume that the emission distribution

We need to take a little care over initialisation and termination of the dynamic programming. We want to allow the alignment to start and end in a delete or insert state, in case the beginning or end of the sequence does not match the first or the last match state of the model. The simplest way to ensure this mechanistically is to rename the Begin state as M_0 and set $V_0^{\mathrm{M}}(0) = 0$ (as we did in Chapter 3). We then allow transitions to I_0 and D_1. Similarly, at the end we can collect together possible paths ending in insert and delete states by renaming the End state to M_{L+1} and using the top relation without the emission term to calculate $V_{L+1}^{\mathrm{M}}(n)$ as the final score.

If these recurrence equations are compared with those for standard gapped dynamic programming in (2.16), it can be seen that apart from renaming of variables this is the same algorithm, but with the substitution, gap-open and gap-extend scores all depending on position in the model, j.

Forward algorithm

The recurrence equations for the forward algorithm are similar to the Viterbi equations, but with the max() operation replaced by addition. We define variables $F_j^{\mathrm{M}}(i), F_j^{\mathrm{I}}(i)$ and $F_j^{\mathrm{D}}(i)$ for the partial full log-odds ratios, corresponding to $V_j^{\mathrm{M}}(i), V_j^{\mathrm{I}}(i)$ and $V_j^{\mathrm{D}}(i)$. The recurrence equations are then:

$$\begin{aligned}
F_j^{\mathrm{M}}(i) &= \log\frac{e_{\mathrm{M}_j}(x_i)}{q_{x_i}} + \log\big[a_{\mathrm{M}_{j-1}\mathrm{M}_j}\exp\big(F_{j-1}^{\mathrm{M}}(i-1)\big) \\
&\qquad + a_{\mathrm{I}_{j-1}\mathrm{M}_j}\exp\big(F_{j-1}^{\mathrm{I}}(i-1)\big) + a_{\mathrm{D}_{j-1}\mathrm{M}_j}\exp\big(F_{j-1}^{\mathrm{D}}(i-1)\big)\big]; \\
F_j^{\mathrm{I}}(i) &= \log\frac{e_{\mathrm{I}_j}(x_i)}{q_{x_i}} + \log\big[a_{\mathrm{M}_j\mathrm{I}_j}\exp\big(F_j^{\mathrm{M}}(i-1)\big) \\
&\qquad + a_{\mathrm{I}_j\mathrm{I}_j}\exp\big(F_j^{\mathrm{I}}(i-1)\big) + a_{\mathrm{D}_j\mathrm{I}_j}\exp\big(F_j^{\mathrm{D}}(i-1)\big)\big]; \\
F_j^{\mathrm{D}}(i) &= \log\big[a_{\mathrm{M}_{j-1}\mathrm{D}_j}\exp\big(F_{j-1}^{\mathrm{M}}(i)\big) + a_{\mathrm{I}_{j-1}\mathrm{D}_j}\exp\big(F_{j-1}^{\mathrm{I}}(i)\big) \\
&\qquad + a_{\mathrm{D}_{j-1}\mathrm{D}_j}\exp\big(F_{j-1}^{\mathrm{D}}(i)\big)\big].
\end{aligned}$$

Initialisation and termination conditions are handled as for the Viterbi case, with $F_0^{\mathrm{M}}(0)$ being initialised to 0.

Although these appear a little complicated, in a practical implementation the operation $\log(e^x + e^y)$ can be performed efficiently to adequate accuracy by function lookup and interpolation; see Section 3.6.

Alternatives to log-odds scoring

Some of the earlier papers on HMMs, rather than calculating the log-odds score relative to a random model, the logarithm of the probability of the sequence