Recall the Needleman-Wunsch algorithm for affine gap penalty:

\[
V^M(i, j) = s(x_i, y_j) + \max \left\{ \begin{array}{c}
V^M(i-1, j-1) \\
V^X(i-1, j-1) \\
V^Y(i-1, j-1)
\end{array} \right. 
\]

\[
V^X(i, j) = \max \left\{ \begin{array}{c}
V^M(i-1, j) - d \\
V^X(i-1, j) - e
\end{array} \right. 
\]

\[
V^Y(i, j) = \max \left\{ \begin{array}{c}
V^M(i, j-1) - d \\
V^Y(i, j-1) - e
\end{array} \right. 
\]

\[
V(m, n) = \max \{V^M(m, n), V^X(m, n), V^Y(m, n)\}
\]

We can now give a probabilistic interpretation of this algorithm using a slightly generalized notion of HMM.
A pair HMM generates an alignment by simultaneously producing two sequences of symbols.

The $M$ (match) state emits a pair of symbols, one for each sequence:

$$(x_i, y_j) \sim p(x_i, y_j)$$

The $X$ ($x$-insertion) state emits only an "$X$ symbol":

$$x_i \sim q(x_i)$$

The $Y$ ($y$-insertion) state emits only a "$Y$ symbol":

$$y_j \sim q(y_j)$$
Pair HMM - cont.

- The model above does not generate a probability distribution over all possible sequences
  - for that we need to add Begin and End states:

The expected length of the generated alignment is $\frac{1}{\tau}$
- The transitions of the Markov chain are given by $p_{MM} = p_{BM} = 1 - 2\delta - \tau$, $p_{MX} = p_{MY} = \delta$, $p_{XX} = \varepsilon$, $p_{XM} = 1 - \varepsilon - \delta$, etc.
**Most probable alignment**

- We can only observe \( x \) and \( y \): unlike in HMMs we cannot observe the joint emission from the \( M \) state.

- Let \( S_{ij} \) be the set of paths \( s \) compatible with an alignment of \( x_{1:i} \) and \( y_{1:j} \):
  - i.e. the path visits states \( \{M, X\} \) exactly \( i \) times and states \( \{M, Y\} \) exactly \( j \) times.

- Given the observed sequences \( x \) and \( y \), \( S_{mn} \) is in 1:1 correspondence with the set of alignments of \( x \) and \( y \).

- The advantage of the pair HMM framework is now we can ask for the most probable alignment given the data:
  - same as maximizing \( p(x, y, s) \) over the path \( s \).
Most probable alignment - cont.

- For $\alpha \in \{M, X, Y\}$ let

$$v^\alpha(i, j) = \max_{s \in S_{ij}: |s| = \alpha} p(x_{1:i}, y_{1:j}, s_{1:|s|})$$

where $|s|$ is the length of the alignment of $s$.

- Clearly,

$$\max_s p(x, y, s) = \max\{v^M(m, n), v^X(m, n), v^Y(m, n)\} \cdot \tau$$

- note that the rhs is in fact $v^E(m, n)$

- The following claim shows how to recursively compute $v^\alpha(i, j)$
Viterbi for pair HMM

• Claim. For $m \geq i \geq 0$, $n \geq j \geq 0$ with $i + j > 0$:

$$v^M(i, j) = p(x_i, y_j) \cdot \max \begin{cases} p_{MM} \cdot v^M(i - 1, j - 1) \\ p_{XM} \cdot v^X(i - 1, j - 1) \\ p_{YM} \cdot v^Y(i - 1, j - 1) \end{cases}$$

$$v^X(i, j) = q(x_i) \cdot \max \begin{cases} p_{MX} \cdot v^M(i - 1, j) \\ p_{XX} \cdot v^X(i - 1, j) \end{cases}$$

$$v^Y(i, j) = q(y_j) \cdot \max \begin{cases} p_{MY} \cdot v^M(i, j - 1) \\ p_{YY} \cdot v^Y(i, j - 1) \end{cases}$$

where $v^\bullet(i, -1) = v^\bullet(-1, j) = v^{[XY]}(0, 0) = 0$, and $v^M(0, 0) := 1$

• $v^M(0, 0)$ is in fact a surrogate for $v^B(0, 0)$
Viterbi for pair HMM - cont.

- This algorithm is similar but still differs from Needleman-Wunsch
  - logarithms should be used
  - log-odds ratio rather than log-odds are computed (BLOSUM/PAM)

- The following random model simply dumps the symbols of \( x \) and the \( y \) without any correlation (no match states)

\[
p_R(x, y) = \eta(1 - \eta)^m \prod_{i=1}^{m} q(x_i) \eta(1 - \eta)^n \prod_{j=1}^{n} q(y_j)
\]
Viterbi for maximal log-odds ratio

- Look for the path $s$ that maximizes the log-odds ratio $\log \frac{p_M(s, x, y)}{p_R(s, x, y)}$

- Let $V^\alpha(i, j) = \max_{s \in S_{ij} : s(|s|) = \alpha} \log \frac{p_M(x_{1:i}, y_{1:j}, s_{1:|s|})}{p_R(x_{1:i}, y_{1:j}, s_{1:|s|})}$

- Analogously to the log-odds case we have

\[
V^M(i, j) = \log \frac{p(x_i, y_j)}{q(x_i)q(y_j)} + \max \left\{ \log \frac{p_{MM}}{(1-\eta)^2} + V^M(i-1, j-1), \log \frac{p_{XM}}{(1-\eta)^2} + V^X(i-1, j-1), \log \frac{p_{YM}}{(1-\eta)^2} + V^Y(i-1, j-1) \right\}
\]

\[
V^X(i, j) = \log \frac{q(x_i)}{q(x_i)} + \max \left\{ \log \frac{p_{MX}}{1-\eta} + V^M(i-1, j), \log \frac{p_{XX}}{1-\eta} + V^X(i-1, j) \right\}
\]

\[
V^Y(i, j) = \log \frac{q(y_j)}{q(y_j)} + \max \left\{ \log \frac{p_{MY}}{1-\eta} + V^M(i, j-1), \log \frac{p_{YY}}{1-\eta} + V^Y(i, j-1) \right\}
\]
Viterbi as Needleman-Wunsch

• To see the equivalence more clearly it is convenient to introduce

\[ s(a, b) = \log \frac{p(a, b)}{q(a)q(b)} + \log \frac{p_{MM}}{(1 - \eta)^2} \]

\[ -d = \log \frac{p_{MX/Y}}{(1 - \eta)} + \log \frac{p_{X/YM}}{p_{MM}} \]

\[ -e = \log \frac{p_{XX/YY}}{1 - \eta} \]

• \( s(a, b) \) “assumes” we come from \( M \)

• \( d \) “pre-corrects” that by adding \( c := \log \frac{p_{X/YM}}{p_{MM}} \)

• Only \( \eta^2 \) and the transitions from \( X/Y \) to \( E \) are left unbalanced:

\[ V^M(0, 0) := -2 \log \eta \]

\[ V^E(m, n) := \max\{V^M(m, n), V^X(m, n) - c, V^Y(m, n) - c\} \]
pair HMM for local alignment

- As before we can look for optimal log-odds or log-odds ratio paths (the latter case will yield Smith-Waterman)
The likelihood that $x$ and $y$ are aligned

- While it is interesting to note that the Needleman-Wunsch algorithm can be cast in the language of HMM

- The real power of the HMM framework is that it allows us to answer questions such as
  - what is the likelihood that $x$ and $y$ are aligned, i.e., that they were generated by the model?

- The answer is the probability that $x, y$ will be generated by the model
  $$p(x, y) = \sum_s p(x, y, s)$$

- An analogue of the forward algorithm computes that: let
  $$f^\alpha(i, j) := P(X_{1:i} = x_{1:i}, Y_{1:j} = y_{1:j}, S(\tau_{ij}) = \alpha),$$

  where
  $$\tau_{ij} := \min\left\{ k : \sum_{l=1}^{k} 1_{S(l) \in \{M,X\}} = i \text{ and } \sum_{l=1}^{k} 1_{S(l) \in \{M,Y\}} = j \right\}$$
The likelihood that $x$ and $y$ are aligned - cont.

- Claim. With the initial conditions

$$f^M(0, 0) = 1 \quad f^{[XY]}(0, 0) = 0 \quad f^\bullet(i, -1) = f^\bullet(-1, j) = 0,$$

for $i \geq 0, j \geq 0$ with $i + j > 0$:

$$f^M(i, j) = p(x_i, y_j)[p_{MM} \cdot f^M(i - 1, j - 1) + p_{XM} \cdot f^X(i - 1, j - 1)$$
$$+ p_{YM} \cdot f^Y(i - 1, j - 1)]$$

$$f^X(i, j) = q(x_i)[p_{MX} \cdot f^M(i - 1, j) + p_{XX} \cdot f^X(i - 1, j)]$$

$$f^Y(i, j) = q(y_j)[p_{MY} \cdot f^M(i, j - 1) + p_{YY} \cdot f^Y(i, j - 1)],$$

and

$$p(x, y) = f^E(m, n) = \tau[f^M(m, n) + f^X(m, n) + f^Y(m, n)]$$
Posterior distribution of an alignment

- With $p(x, y)$ we can find the posterior distribution of any particular alignment $s$: $p(s|x, y) = \frac{p(x, y, s)}{p(x, y)}$
  - In particular we can apply it for $s^*$, the Viterbi solution
  - The answer is typically depressingly small
    ▶ For example in the alpha globing vs. leghemoglobin case:

```
(b) Lupin leghemoglobin
HBA_HUMAN  GSAQVKGHGKKVADALTNAVAHV---D---DMPNALSAALSDLHAKKL
          ++ ++++H+ KV + +A ++ +L+ L+++H+ K
LGB2_LUPLU NNPELQAHAGKVFKLVYEAATQLQVTGVVVTDAKLKNLGSVHVSKG
```

▶ $p(s^*|x, y) = 4.6 \times 10^{-6}$
Sampling from the posterior distribution

- Given the poor posterior probability of the Viterbi alignment
  - are there parts of the alignment which we are more confident of?
  - can we estimate posterior expectation of functionals of the alignment as in posterior decoding?

- We can do that through MC sampling from the posterior distribution
  - backward sampling (using forward algorithm)
  - forward sampling (using backward algorithm)
The backward algorithm

- Analogously to the backward function for HMMs we define

$$b^\alpha(i, j) := P(X_{i+1:m} = x_{i+1:m}, Y_{j+1:n} = y_{j+1:n}, S(\tau_{ij}) = \alpha),$$

where

$$\tau_{ij} := \min\left\{ k : \sum_{l=1}^{k} 1_{S(l) \in \{M,X\}} = i \text{ and } \sum_{l=1}^{k} 1_{S(l) \in \{M,Y\}} = j \right\}$$

Algorithm: Backward calculation for pair HMMs

Initialisation:
- $b^M(n,m) = b^X(n,m) = b^Y(n,m) = \tau$.
- All $b^*(i,m+1), b^*(n+1,j)$ are set to 0.

Recursion: $i = n, \ldots, 1, j = m, \ldots, 1$ except $(n,m)$;

- $b^M(i,j) = (1-2\delta-\tau)p_{x_{i+1}y_{j+1}}b^M(i+1,j+1) + \delta[q_{x_{i+1}}b^X(i+1,j) + q_{y_{j+1}}b^Y(i,j+1)];$
- $b^X(i,j) = (1-\epsilon-\tau)p_{x_{i+1}y_{j+1}}b^M(i+1,j+1) + \epsilon q_{x_{i+1}}b^X(i+1,j);$
- $b^Y(i,j) = (1-\epsilon-\tau)p_{x_{i+1}y_{j+1}}b^M(i+1,j+1) + \epsilon q_{y_{j+1}}b^Y(i,j+1).$

- Durbin et al.:

  - as before we can add $b^M(0,0)$ as a surrogate for $b^B(0,0)$
Forward posterior sampling (backward algorithm)

- Inductively draw from the posterior distribution as follows:
  - start at state $B$ with $(i, j) := (0, 0)$
  - while $(i, j) \neq (m, n)$:
    - given our hitherto path $s \in S(i, j)$ randomly choose our next state $\alpha$ according to $P[S(|s| + 1) = \alpha | x, y, s]$
    - update: $s = s \land \alpha$, and $(i, j) := (i^+(\alpha), j^+(\alpha)) := (i + 1_{\alpha \in \{M,X\}}, j + 1_{\alpha \in \{M,Y\}})$
  - output the resulting $s \in S(m, n)$ (why is $s \in S(m, n)$?)

- Claim. The probability that we draw a path $s \in S(m, n)$ is $p(s | x, y)$

- Proof. To simplify notations assume $s(0) = B$ does not count toward $|s|$. Then

$$p(s | x, y) = \prod_{i=1}^{|s|} p(s(i) | x, y, s_{0:i-1})$$
Forward posterior sampling - cont.

- The algorithm hinges on finding

\[ P[S(|s| + 1) = \alpha | x, y, s] = \frac{p(s \wedge \alpha, x, y)}{p(s, x, y)} \]

- Using the properties of the HMM we have:

\[
p(s \wedge \alpha, x, y) = p(x_{1:i}, y_{1:j}, s) \times P[S(|s| + 1) = \alpha, x(i^+(\alpha)), y(j^+(\alpha)) | x_i, y_j, s] \times p(x_{i+1:m}, y_{j+1:n}) S(|s| + 1) = \alpha] \]

\[ P[S(|s| + 1) = \alpha, x(i^+(\alpha)), y(j^+(\alpha)) | x_i, y_j, s] = \begin{cases} p_s(|s|), M \cdot p(x(i + 1), y(j + 1)) & \alpha = M \\ p_s(|s|), X \cdot q(x(i + 1)) & \alpha = X \\ p_s(|s|), Y \cdot q(y(j + 1)) & \alpha = Y \end{cases} \]
Note that
\[ p[x_i^{+(\alpha)+1:m}, y_j^{+(\alpha)+1:n}|S(|s| + 1) = \alpha] = b^\alpha(i^{+(\alpha)}, j^{+(\alpha)}) \]

Finally,
\[
\begin{align*}
p(s, x, y) &= p(x_{1:i}, y_{1:j}, s)p(x_{i+1:m}, y_{j+1:n}|s) \\
&= p(x_{1:i}, y_{1:j}, s)b^s(|s|)(i, j)
\end{align*}
\]

Thus,
\[
P[S(|s| + 1) = \alpha|x, y, s] = \frac{b^\alpha(i^{+(\alpha)}, j^{+(\alpha)})}{b^s(|s|)(i, j)}
\]

\[
\times \begin{cases} 
p_s(|s|), M \cdot p(x(i + 1), y(j + 1)) & \alpha = M \\
p_s(|s|), X \cdot q(x(i + 1)) & \alpha = X \\
p_s(|s|), Y \cdot q(y(j + 1)) & \alpha = Y
\end{cases}
\]
Posterior probability that $x_i$ is aligned to $y_j$

- We can estimate the posterior probability that $x_i$ is aligned to $y_j$ by posterior sampling of alignments
  - but we can also compute it directly
    - analogous to computing $P(S(i) = k|x)$ for HMMs
- Let $X_i \diamond Y_j$ denote the event $X_i$ is aligned to $Y_j$, then

\[
P(X = x, Y = y, X_i \diamond Y_j) = P(X_{1:i} = x_{1:i}, Y_{1:j} = y_{1:j}, S(\tau_{ij}) = M) \\
\times P[X_{i+1:m} = x_{i+1:m}, Y_{j+1:n} = y_{j+1:n} | S(\tau_{ij}) = M] \\
= f^M(i, j)b^M(i, j)
\]

therefore

\[
P(X_i \diamond Y_j | X = x, Y = y) = \frac{f^M(i, j)b^M(i, j)}{p(x, y)} = \frac{f^M(i, j)b^M(i, j)}{f^E(m, n)}
\]
Optimizing for the “expected accuracy”

- An intuitively appealing measure of the accuracy of $s$ is

$$A(s) = \sum_{M_{ij} \in s} p(x_i \diamond y_j | \mathbf{x}, \mathbf{y})$$

- $A(s)$ is the expected overlap in $M$ states between $s$ and a random alignment drawn according to the posterior distribution.

- Finding a path $s$ which maximizes $A(s)$ is easy: let $A(i, j)$ be the optimal accuracy we can gain using only $\mathbf{x}_{1:i}$ and $\mathbf{y}_{1:j}$

$$A(i, j) = \max \begin{cases} A(i - 1, j - 1) + p(x_i \diamond y_j | \mathbf{x}, \mathbf{y}) \\ A(i - 1, j) \\ A(i, j - 1) \end{cases}$$
Viterbi failure

• Can the Viterbi algorithm discriminate between data generated by the following model vs. the random one?

• Maximizing the log-likelihood or llr is equivalent here

• If $\alpha^4 q(a)q(b)q(a)q(c) > 1 - \alpha$ then $p_S(abac) > p_B(abac)$ and the Viterbi path will never visit state $B$

• Since state $S$ is random, the Viterbi path cannot discriminate between the model and random data
Viterbi failure - cont.

• However, if the data is long enough clearly the model is distinguishable from random:
  • $f_o(abac) \rightarrow p_M(abac) > p_S(abac)$
  • so simply observing the frequency of $abac$ should work for sufficiently long sequences

• Maximizing the likelihood is not always the appropriate approach

• However, comparing $p_M(x)$ and $p_R(x)$ should discriminate the two models
  • as this is the optimal test and we just saw we can discriminate

• Bonus points: figure out Figure 4.8