Beyond Viterbi: A Maximum Expected Accuracy Approach to Pairwise Alignment

Lawrence Granda[1], **Joyce Shen**[2], **Soham Goswami**[3], **and Joshua Ochalek**[4]

[1]Computer Science, Junior, lg626
[2]Computer Science, Junior, js3696
[3]Computer Science, Sophomore, sbg226
[4]Computer Science, Junior, jo447

ABSTRACT

Abstract (200 words minimum)

Keywords (minumum 5): Pairwise sequence alignment, Hidden Markov Model (HMM), Maximum Expected Accuracy (MEA), Viterbi algorithm, Forward-backward algorithm, RNA sequence analysis, Posterior probability

Project type: Reimplementation

Project repository: `https://github.com/joucebox/cs4775-final-project`

AI Use Attribution Statements: Required

Note:

- This is the title page (required). It does NOT count toward the minimum page requirement.
- In the main text, do NOT adjust margins, font size, character spacing, line spacing, or add blank lines to artificially increase the number of pages to meet the minimum page limit. Doing so will result in a significant point deduction.
- Please read the "Final Project Report Guide" on Canvas carefully.
- Each group member must submit the "Author Contribution" form individually. The template is available on Canvas. Please note that, while the final report is a group submission, the author contribution form is an individual submission. Failure to submit the author contribution form by the due date will result in a 1-point deduction from the final report grade.

1 Introduction

1.1 Background and Motivation

Pairwise sequence alignment is a fundamental task in computational biology, essential for understanding evolutionary relationships, identifying functional domains, and annotating genomic sequences. For RNA molecules, accurate alignment is particularly important as it enables the identification of conserved structural elements and functional motifs that may not be apparent from primary sequence alone. Traditional alignment methods rely on scoring matrices and gap penalties, but these approaches often fail to capture the complex dependencies and uncertainties inherent in biological sequence comparison.

1.2 Problem Statement

The Viterbi algorithm, widely used in Hidden Markov Model (HMM) based alignment methods, finds the single most probable alignment path — the maximum a posteriori (MAP) estimate. While computationally efficient, this approach has a fundamental limitation: it completely ignores posterior uncertainty in the alignment. In regions where multiple alignments are nearly equally probable, the Viterbi path may not reflect the true underlying biological relationship, potentially leading to incorrect functional annotations.

Maximum Expected Accuracy (MEA) alignment addresses this limitation by maximizing the expected number of correctly aligned positions under the posterior distribution over all possible alignments. Rather than committing to a single path, MEA considers the probability that each position pair should be aligned, providing a more robust approach that accounts for alignment uncertainty.

1.3 Objectives

This work implements and evaluates MEA alignment for RNA sequence pairs using a three-state pair hidden Markov model. Our primary objectives are:

1. Implement a complete HMM framework with parameter estimation from curated pairwise alignments
2. Compare Viterbi and MEA alignment quality across different posterior weighting schemes
3. Analyze the precision-recall tradeoffs enabled by the MEA approach
4. Evaluate performance on a comprehensive dataset of 547 pairwise RNA alignments from Rfam families

1.4 Contributions

This work provides a complete Python implementation of HMM algorithms for RNA alignment, including maximum likelihood parameter estimation, forward-backward inference, Viterbi decoding, and MEA alignment with multiple weighting schemes. We present a thorough empirical evaluation comparing alignment quality against golden standard annotations, demonstrating the benefits of accounting for posterior uncertainty in sequence alignment. Our analysis reveals how different MEA formulations provide tunable precision-recall tradeoffs.

2 Methods

2.1 Pair Hidden Markov Model

We model a pair of RNA sequences

$$X = x_1, \dots, x_{L_X}, \qquad Y = y_1, \dots, y_{L_Y},$$

over alphabet $\mathcal{A} = \{A, C, G, U\}$. Their alignment is represented by a hidden state sequence

$$Z = (z_1, \dots, z_T), \qquad z_t \in \mathcal{S} = \{M, X, Y\},$$

where M denotes a match (or substitution), X an insertion in X (gap in Y), and Y an insertion in Y (gap in X).

We associate to each step t a pair of prefix indices (i_t, j_t) specifying how many characters of X and Y have been consumed. The state z_t determines which symbols are emitted and how indices advance:

$$(i_{t+1}, j_{t+1}) = \begin{cases} (i_t + 1, j_t + 1), & z_t = M, \\ (i_t + 1, j_t), & z_t = X, \\ (i_t, j_t + 1), & z_t = Y. \end{cases}$$

A path Z is valid if all characters are consumed, i.e. $(i_T, j_T) = (L_X, L_Y)$.

2.1.1 Emission model

For $z_t \in \{M, X, Y\}$ we define emission distributions

$$e_M(x,y) = P(\text{emit } (x,y) \mid z_t = M), \qquad e_X(x) = P(\text{emit } x \mid z_t = X), \qquad e_Y(y) = P(\text{emit } y \mid z_t = Y),$$

with normalizations $\sum_{x,y\in\mathcal{A}} e_M(x,y) = 1$, $\sum_{x\in\mathcal{A}} e_X(x) = 1$, $\sum_{y\in\mathcal{A}} e_Y(y) = 1$. At step t the emission probability is

$$e_{z_t} = \begin{cases} e_M(x_{i_t+1}, y_{j_t+1}), & z_t = M, \\ e_X(x_{i_t+1}), & z_t = X, \\ e_Y(y_{j_t+1}), & z_t = Y. \end{cases}$$

2.1.2 Start, transition, and end distributions

The HMM parameters are:

- Start distribution
$$\pi(s) = P(z_1 = s), \qquad s \in \mathcal{S},$$
taken as uniform ($\pi(M) = \pi(X) = \pi(Y) = \frac{1}{3}$) unless noted.
- Transition matrix
$$a_{uv} = P(z_{t+1} = v \mid z_t = u), \qquad u, v \in \mathcal{S},$$
with $\sum_v a_{uv} = 1$ and constraints $a_{XY} = a_{YX} = 0$ (no direct transitions between insert states), allowing an affine gap penalty via distinct gap-open ($M \to X/Y$) and gap-extend ($X \to X$, $Y \to Y$) probabilities.
- End distribution
$$\rho(s) = P(\text{end} \mid z_T = s), \qquad s \in \mathcal{S}.$$

2.1.3 Joint probability and log-space parameterization

For any valid path $Z = (z_1, \ldots, z_T)$,

$$P(X,Y,Z) = \pi(z_1)\Big[\prod_{t=1}^{T-1} a_{z_t z_{t+1}}\Big]\Big[\prod_{t=1}^{T} e_{z_t}\Big]\rho(z_T).$$

The marginal likelihood sums over all valid alignments,

$$P(X,Y) = \sum_{Z\in\mathcal{Z}(X,Y)} P(X,Y,Z),$$

where $\mathcal{Z}(X,Y)$ is the set of paths that consume both sequences.

All probabilities are represented in log-space: $\log\pi(s)$, $\log a_{uv}$, $\log\rho(s)$, $\log e_M(x,y)$, $\log e_X(x)$, $\log e_Y(y)$. Then

$$\log P(X,Y,Z) = \log\pi(z_1) + \sum_{t=1}^{T-1}\log a_{z_t z_{t+1}} + \sum_{t=1}^{T}\log e_{z_t} + \log\rho(z_T).$$

2.2 Parameter estimation from aligned sequences

We estimate pair-HMM parameters from gold-standard alignments using maximum likelihood estimation with pseudocount regularization. Each reference alignment provides labeled training data where columns are classified as match, insertion in X, or insertion in Y states. We collect sufficient statistics across all alignments and normalize to obtain emission and transition probability estimates. The full mathematical formulation, including column-wise state labeling, sufficient statistics computation, and parameter estimation with affine gap penalties, is provided in Algorithm 4.1 of the appendix.

2.3 Forward–backward inference

We implement an equivalent formulation to the forward-backward algorithm covered in class, adapted for the pair-HMM topology. The algorithm computes posterior probabilities over alignment paths using log-space dynamic programming on a 2D grid representing consumed prefixes of both sequences. Forward probabilities accumulate the likelihood of reaching each grid position, while backward probabilities compute the likelihood of completing the alignment from each position. The full algorithm, including initialization, recurrence relations, and log-sum-exp operations for numerical stability, is detailed in Algorithm 4.2 of the appendix.

2.4 Viterbi decoding (MAP alignment)

We implement an equivalent formulation to the Viterbi algorithm covered in class, adapted for the pair-HMM with three states (match, insert-X, insert-Y). The algorithm finds the single most probable alignment path by maximizing the joint probability of sequences and hidden states using dynamic programming. Starting from the origin, it computes the maximum probability of reaching each grid position in each state, then traces back from the optimal terminal state to recover the alignment. The full algorithm, including initialization for leading gaps, recurrence relations, and traceback procedure, is provided in Algorithm 4.3 of the appendix.

2.5 Maximum expected accuracy (MEA) alignment

We also compute an alignment that maximizes the expected number of correctly aligned nucleotide pairs under the posterior over paths.

2.5.1 Posterior match probabilities

For $1 \le i \le L_X$, $1 \le j \le L_Y$, let $M_{ij} = 1$ if x_i is aligned to y_j in state M along some path, and define

$$P_{ij} = P(M_{ij} = 1 \mid X, Y).$$

Using the forward-backward algorithm (detailed in Algorithm 4.2 of the appendix), we compute $\alpha_M(i,j)$, the forward probability of reaching position (i,j) in match state M while aligning (x_i, y_j), and $\beta_M(i,j)$, the backward probability of completing the alignment from position (i,j) in match state M. The posterior match probability is then

$$P_{ij} = \frac{\alpha_M(i,j)\,\beta_M(i,j)}{P(X,Y)},$$

where $P(X,Y)$ is the marginal likelihood of the sequences. We arrange P_{ij} in a matrix $P \in [0,1]^{(L_X+1)\times(L_Y+1)}$ with $P[0,\cdot] = P[\cdot,0] = 0$.

2.5.2 Objective and weighting

An alignment A is a set of index pairs $A \subseteq \{1,\dots,L_X\} \times \{1,\dots,L_Y\}$, where $(i,j) \in A$ iff x_i is aligned to y_j. Its expected accuracy is

$$\mathbb{E}[\mathrm{acc}(A) \mid X, Y] = \sum_{(i,j)\in A} P_{ij}.$$

To provide a tunable precision-recall tradeoff, we introduce a parameter γ and use weighted posterior probabilities $w_{ij} = f(P_{ij};\gamma)$ in the objective function. We then maximize

$$A^\star = \arg\max_A \sum_{(i,j)\in A} w_{ij},$$

where different choices of the weighting function f and parameter γ allow us to prioritize precision or recall. Examples of f include:

- Power ($\gamma > 0$): $w_{ij} = P_{ij}^{\gamma}$, where larger γ emphasizes high-confidence matches
- Threshold ($\gamma \in (0,1]$): $w_{ij} = P_{ij} - (1-\gamma)$, which sets matches below threshold γ to negative weights
- ProbCons-style ($\gamma > 0.5$): $w_{ij} = 2\gamma P_{ij} - 1$, providing linear weighting
- Log-odds ($\gamma \in (0,1)$):
$$w_{ij} = \log\frac{P_{ij}}{1-P_{ij}} + \log\frac{\gamma}{1-\gamma},$$
  which transforms probabilities to log-odds space

Let W be the matrix with entries $W[i,j] = w_{ij}$ and $W[0,\cdot] = W[\cdot,0] = 0$.

2.5.3 Dynamic programming and reconstruction

We view MEA alignment as a maximum-weight path on the grid $\{0,\dots,L_X\} \times \{0,\dots,L_Y\}$: diagonal steps align (x_i, y_j) with weight $W[i,j]$, and horizontal/vertical steps are gaps with weight 0.

Let $D(i,j)$ be the maximum total weight for prefixes $x_{1:i}, y_{1:j}$, with

$$D(0,0) = 0, \qquad D(i,0) = 0, \qquad D(0,j) = 0.$$

For $1 \leq i \leq L_X, 1 \leq j \leq L_Y$,

$$D(i,j) = \max\big\{D(i-1,j-1) + W[i,j],\ D(i-1,j),\ D(i,j-1)\big\}.$$

The MEA score is $D(L_X, L_Y)$. Storing the maximizing move (diagonal/up/left) at each cell and tracing back from (L_X, L_Y) to $(0,0)$ yields the MEA alignment.

3 Results

3.1 Synthetic Data

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

3.2 Real Data

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

4 Discussion

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

References

Figures

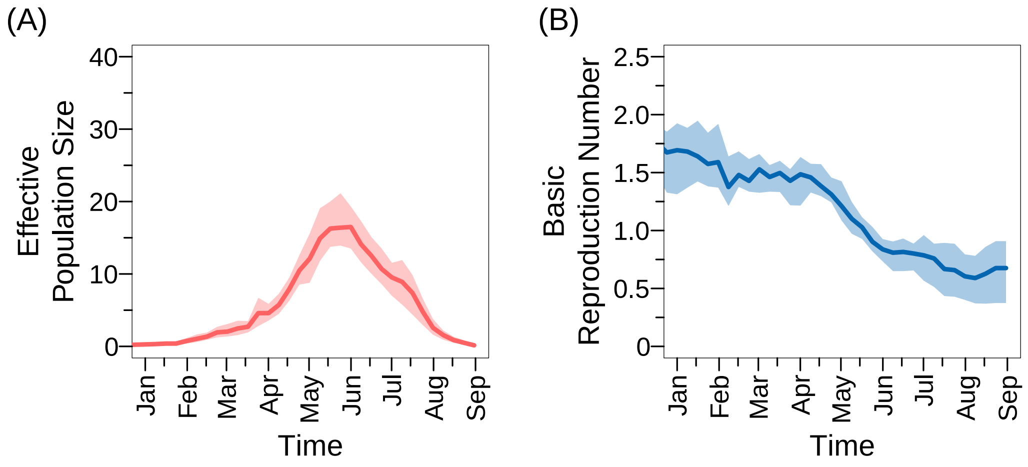

Figure 1. **Example figure (figure title).** Figure description. (A) xxxx. (B) xxxx. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Tables

Table 1. Example table (table title). Table description. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

		A			B		C	
n_1	n_2	Analytic	Method 1	Method 2	Analytic	Method 3	Analytic	Method 4
x	x	x	x	x	x	x	x	x
x	x	x	x	x	x	x	x	x
x	x	x	x	x	x	x	x	x
x	x	x	x	x	x	x	x	x
x	x	x	x	x	x	x	x	x
x	x	x	x	x	x	x	x	x

Algorithms

4.1 Parameter Estimation

We estimate emission and transition parameters from reference alignments. Each alignment yields strings $\tilde{X}, \tilde{Y} \in (\mathcal{A} \cup \{-\})^L$.

4.1.1 Column-wise state labeling

For each column ℓ we assign a state

$$s_\ell = \begin{cases} M & \tilde{x}_\ell \in \mathcal{A},\ \tilde{y}_\ell \in \mathcal{A}, \\ X & \tilde{x}_\ell \in \mathcal{A},\ \tilde{y}_\ell = -, \\ Y & \tilde{x}_\ell = -,\ \tilde{y}_\ell \in \mathcal{A}, \\ \varnothing & \text{otherwise (e.g. both gaps).} \end{cases}$$

Columns with $s_\ell = \varnothing$ are ignored; transitions are not propagated across them.

4.1.2 Sufficient statistics

We collect counts

$$\begin{aligned} c_M(a,b) &= \#\{\ell : s_\ell = M,\ \tilde{x}_\ell = a,\ \tilde{y}_\ell = b\}, \\ c_X(a) &= \#\{\ell : s_\ell = X,\ \tilde{x}_\ell = a\}, \\ c_Y(b) &= \#\{\ell : s_\ell = Y,\ \tilde{y}_\ell = b\}, \\ c_{\text{tr}}(u,v) &= \#\{\ell : s_\ell = u,\ s_{\ell+1} = v,\ u,v \in \mathcal{S}\}, \end{aligned}$$

skipping ℓ with $s_\ell = \varnothing$.

4.1.3 Emissions with pseudocounts

With pseudocount $\eta \geq 0$ and $\mathcal{A} = \{A, C, G, U\}$, define

$$T_M(a) = \sum_{b \in \mathcal{A}} (c_M(a,b) + \eta).$$

Then

$$e_M(a,b) = \begin{cases} \dfrac{c_M(a,b) + \eta}{T_M(a)}, & T_M(a) > 0, \\ \dfrac{1}{|\mathcal{A}|}, & T_M(a) = 0. \end{cases}$$

For insertions, with $T_X = \sum_a (c_X(a) + \eta)$ and $T_Y = \sum_b (c_Y(b) + \eta)$,

$$e_X(a) = \begin{cases} \dfrac{c_X(a) + \eta}{T_X}, & T_X > 0, \\ \dfrac{1}{|\mathcal{A}|}, & T_X = 0, \end{cases} \qquad e_Y(b) = \begin{cases} \dfrac{c_Y(b) + \eta}{T_Y}, & T_Y > 0, \\ \dfrac{1}{|\mathcal{A}|}, & T_Y = 0. \end{cases}$$

In implementation we store $\log e_M, \log e_X, \log e_Y$, with zero probabilities encoded as $-\infty$.

4.1.4 Transitions and affine-gap parameters

Raw transition estimates are

$$\tilde{a}_{uv} = \begin{cases} 0, & (u,v) \in \{(X,Y),(Y,X)\}, \\ \dfrac{c_{\text{tr}}(u,v) + \eta}{\sum_w (c_{\text{tr}}(u,w) + \eta)}, & \text{if denominator} > 0, \\ 0, & \text{otherwise,} \end{cases}$$

followed by row-wise renormalization over allowed v (excluding $X \to Y$ and $Y \to X$) to obtain a_{uv}. Again, $\log a_{uv}$ is stored, with $-\infty$ for zero entries.

We summarize affine-gap parameters as

$$\delta = a_{MX} + a_{MY}, \qquad \epsilon_X = a_{XX}, \qquad \epsilon_Y = a_{YY}, \qquad \epsilon = \tfrac{1}{2}(\epsilon_X + \epsilon_Y).$$

4.2 Forward-Backward Algorithm

We perform log-space dynamic programming on the grid $0 \le i \le L_X$, $0 \le j \le L_Y$ using tables $F_s(i,j)$ and $B_s(i,j)$ for $s \in \{M, X, Y\}$, where (i,j) encodes consumed prefixes. We use the log-sum-exp operator

$$\mathrm{LSE}(v_1, \dots, v_k) = \log\Big(\sum_{m=1}^{k} e^{v_m}\Big)$$

for stable accumulation.

4.2.1 Forward recursion

Initialize all $F_s(i,j) = -\infty$. The first emissions are

$$F_X(1,0) = \log \pi(X) + \log e_X(x_1), \quad F_Y(0,1) = \log \pi(Y) + \log e_Y(y_1), \quad F_M(1,1) = \log \pi(M) + \log e_M(x_1, y_1),$$

if the corresponding indices are in range.

For $1 \le i \le L_X$, $1 \le j \le L_Y$, the recurrences (omitting impossible moves) are

$$F_M(i,j) = \mathrm{LSE}\Big(F_M(i-1,j-1) + \log a_{MM},\ F_X(i-1,j-1) + \log a_{XM},\ F_Y(i-1,j-1) + \log a_{YM}\Big) + \log e_M(x_i, y_j),$$
$$F_X(i,j) = \mathrm{LSE}\Big(F_M(i-1,j) + \log a_{MX},\ F_X(i-1,j) + \log a_{XX}\Big) + \log e_X(x_i),$$
$$F_Y(i,j) = \mathrm{LSE}\Big(F_M(i,j-1) + \log a_{MY},\ F_Y(i,j-1) + \log a_{YY}\Big) + \log e_Y(y_j).$$

The forward log-partition is

$$\log Z_f = \mathrm{LSE}\big(F_M(L_X, L_Y) + \log \rho(M),\ F_X(L_X, L_Y) + \log \rho(X),\ F_Y(L_X, L_Y) + \log \rho(Y)\big).$$

4.2.2 Backward recursion

Initialize all $B_s(i,j) = -\infty$ and at (L_X, L_Y) set

$$B_M(L_X, L_Y) = \log \rho(M), \quad B_X(L_X, L_Y) = \log \rho(X), \quad B_Y(L_X, L_Y) = \log \rho(Y).$$

For $0 \le i \le L_X$, $0 \le j \le L_Y$, let $\mathbf{1}[\cdot]$ denote an indicator and include only in-bounds moves. Then

$$\begin{aligned} B_M(i,j) = \mathrm{LSE}\Big(&\mathbf{1}[i < L_X, j < L_Y]\big(\log a_{MM} + \log e_M(x_{i+1}, y_{j+1}) + B_M(i+1, j+1)\big),\\ &\mathbf{1}[i < L_X]\big(\log a_{MX} + \log e_X(x_{i+1}) + B_X(i+1, j)\big),\\ &\mathbf{1}[j < L_Y]\big(\log a_{MY} + \log e_Y(y_{j+1}) + B_Y(i, j+1)\big)\Big), \end{aligned}$$

$$\begin{aligned} B_X(i,j) = \mathrm{LSE}\Big(&\mathbf{1}[i < L_X, j < L_Y]\big(\log a_{XM} + \log e_M(x_{i+1}, y_{j+1}) + B_M(i+1, j+1)\big),\\ &\mathbf{1}[i < L_X]\big(\log a_{XX} + \log e_X(x_{i+1}) + B_X(i+1, j)\big)\Big), \end{aligned}$$

$$\begin{aligned} B_Y(i,j) = \mathrm{LSE}\Big(&\mathbf{1}[i < L_X, j < L_Y]\big(\log a_{YM} + \log e_M(x_{i+1}, y_{j+1}) + B_M(i+1, j+1)\big),\\ &\mathbf{1}[j < L_Y]\big(\log a_{YY} + \log e_Y(y_{j+1}) + B_Y(i, j+1)\big)\Big), \end{aligned}$$

using $a_{XY} = a_{YX} = 0$. The backward log-partition is

$$\log Z_b = \mathrm{LSE}\big(\log \pi(M) + \log e_M(x_1, y_1) + B_M(1,1),\ \log \pi(X) + \log e_X(x_1) + B_X(1,0),\ \log \pi(Y) + \log e_Y(y_1) + B_Y(0,1)\big).$$

We use $\log Z = \frac{1}{2}(\log Z_f + \log Z_b)$ as a numerically stable estimate of $\log P(X,Y)$.

4.3 Viterbi Algorithm

The maximum a posteriori alignment is

$$Z^\star = \arg \max_{Z \in \mathcal{Z}(X,Y)} P(X,Y,Z) = \arg \max_{Z \in \mathcal{Z}(X,Y)} \log P(X,Y,Z).$$

4.3.1 Dynamic programming

For $0 \le i \le L_X$, $0 \le j \le L_Y$ and $s \in \{M, X, Y\}$ define

$$V_{i,j}^{s} = \max_{Z:\,(i_T, j_T)=(i,j),\, z_T = s} \log P(X_{1:i}, Y_{1:j}, Z),$$

with $V_{i,j}^{s} = -\infty$ if no such path exists.

Initialization at $(0,0)$ is $V_{0,0}^{M} = V_{0,0}^{X} = V_{0,0}^{Y} = -\infty$. First non-empty cells:

$$V_{1,0}^{X} = \log \pi(X) + \log e_X(x_1), \quad V_{0,1}^{Y} = \log \pi(Y) + \log e_Y(y_1), \quad V_{1,1}^{M} = \log \pi(M) + \log e_M(x_1, y_1).$$

Leading gaps are filled via

$$\begin{aligned} V_{i,0}^{X} &= \log e_X(x_i) + \max\bigl\{V_{i-1,0}^{M} + \log a_{MX},\ V_{i-1,0}^{X} + \log a_{XX}\bigr\}, \quad i = 2, \ldots, L_X, \\ V_{0,j}^{Y} &= \log e_Y(y_j) + \max\bigl\{V_{0,j-1}^{M} + \log a_{MY},\ V_{0,j-1}^{Y} + \log a_{YY}\bigr\}, \quad j = 2, \ldots, L_Y. \end{aligned}$$

For $1 \le i \le L_X$, $1 \le j \le L_Y$:

$$\begin{aligned} V_{i,j}^{M} &= \log e_M(x_i, y_j) + \max\bigl\{V_{i-1,j-1}^{M} + \log a_{MM},\ V_{i-1,j-1}^{X} + \log a_{XM},\ V_{i-1,j-1}^{Y} + \log a_{YM}\bigr\}, \\ V_{i,j}^{X} &= \log e_X(x_i) + \max\bigl\{V_{i-1,j}^{M} + \log a_{MX},\ V_{i-1,j}^{X} + \log a_{XX}\bigr\}, \\ V_{i,j}^{Y} &= \log e_Y(y_j) + \max\bigl\{V_{i,j-1}^{M} + \log a_{MY},\ V_{i,j-1}^{Y} + \log a_{YY}\bigr\}. \end{aligned}$$

4.3.2 Termination and traceback

At (L_X, L_Y) we add end probabilities:

$$\ell^{\star} = \max_{s \in \{M,X,Y\}} \bigl(V_{L_X, L_Y}^{s} + \log \rho(s)\bigr),$$

with optimal final state

$$s^{\star} = \arg\max_{s \in \{M,X,Y\}} \bigl(V_{L_X, L_Y}^{s} + \log \rho(s)\bigr).$$

We store backpointers during the DP and trace back from $(L_X, L_Y, s^{\star})$ to $(0,0)$, then reverse the sequence of moves to recover the optimal alignment.