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 and Italian, Junior, jo447

ABSTRACT

Sequence alignment stands as the computational bedrock of modern biology, serving as the prerequisite for phylogenetic reconstruction, functional annotation, and structural modeling. For decades, the Viterbi algorithm has served as the *de facto* standard for decoding Pair Hidden Markov Models (Pair-HMMs), efficiently identifying the single maximum a posteriori (MAP) alignment path. However, this approach fundamentally fails to account for the stochastic uncertainty inherent in evolutionary analysis, a limitation that is particularly acute in the alignment of non-coding RNAs (ncRNAs). In these molecules, the conservation of secondary structure often supersedes primary sequence identity, creating a "twilight zone" of homology characterized by diffuse probability landscapes where the single best path represents a negligible fraction of the total probability mass. In this comprehensive report, we present the theoretical derivation, implementation, and rigorous evaluation of Maximum Expected Accuracy (MEA) decoding for pairwise RNA alignment. By decoupling the probabilistic model from the decision rule, we demonstrate that MEA provides a statistically rigorous framework for minimizing the Hamming loss of an alignment, contrasting sharply with the 0/1 loss minimized by Viterbi. We implement a three-state Pair-HMM trained on curated Rfam data and systematically evaluate four distinct posterior weighting schemes: Power, Threshold, Log-Odds, and ProbCons-style weighting. Our results, based on a benchmark of 547 pairwise alignments, indicate that MEA decoding consistently outperforms Viterbi in terms of F1 score and column identity. We further demonstrate that the γ parameter serves as a precise, tunable control for the sensitivity-specificity trade-off, allowing researchers to optimize alignments for specific downstream applications. This work establishes MEA not merely as an alternative, but as a superior paradigm for RNA sequence analysis that robustly integrates over evolutionary uncertainty.

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

The alignment of biological sequencesDeoxyribonucleic Acid (DNA), Ribonucleic Acid (RNA), and proteinsconstitutes the foundational stratum of modern bioinformatics. It provides the essential mapping required to infer evolutionary history, identify functional motifs, and predict macromolecular structure. Since the seminal work of Needleman and Wunsch in 1970 [1], the field has been dominated by dynamic programming algorithms that seek to optimize a specific scoring objective. The introduction of probabilistic modeling, particularly Pair-Hidden Markov Models (Pair-HMMs), transformed this landscape by providing a rigorous statistical framework for alignment, replacing ad hoc scoring matrices with learned transition and emission probabilities.

Within this probabilistic paradigm, the Viterbi algorithm has served as the standard for decoding, efficiently computing the single most probable alignment path (the Maximum A Posteriori, or MAP, estimate). The algorithm's ubiquity, found in tools from HMMER to Clustal, stems from its computational efficiency and the intuitive appeal of finding the "best" path. However, the supremacy of the Viterbi path is increasingly challenged, particularly in the analysis of divergent sequences where the signal-to-noise ratio is low. This challenge arises from a fundamental epistemological limitation: the Viterbi algorithm assumes that the "best" alignment is the single path with the highest joint probability. In the "twilight zone" of sequence homologywhere sequence identity drops below 30%the probability landscape becomes diffuse and rugged. Here, the single best path often represents a negligible fraction of the total probability mass, and alternative suboptimal paths may collectively harbor the true biological signal.

1.2 The Biological Imperative: RNA Structure and Evolution

To understand the necessity of MEA, one must first appreciate the biological context of the problem, specifically the alignment of Ribonucleic Acid (RNA). Unlike proteins, where the primary sequence (the string of amino acids) is the dominant determinant of folding and function, RNA function is dictated primarily by its secondary structurethe intricate patterns of Watson-Crick (G-C, A-U) and wobble (G-U) base pairs that form stems, loops, and pseudoknots.

In non-coding RNAs (ncRNAs), such as transfer RNAs (tRNAs), ribosomal RNAs (rRNAs), and riboswitches, evolutionary pressure acts to conserve this secondary structure rather than the primary sequence. This phenomenon, known as compensatory mutation, means that a mutation on one side of a stem (e.g., G to A) is often followed by a compensatory mutation on the partner base (e.g., C to U) to maintain base-pairing complementarity. As a result, two homologous RNA sequences might share very little sequence identity (primary structure) yet fold into identical shapes (secondary structure).

Standard alignment algorithms, including Viterbi-based Pair-HMMs, struggle significantly in this context. The Viterbi algorithm seeks a linear path of matching residues. When faced with a variable loop region or a stem with multiple mutations, the Viterbi path often "snaps" to a mathematically optimal but biologically incorrect trajectory, effectively guessing a specific alignment of gaps and residues to maximize a likelihood score. It ignores the "cloud" of uncertainty that naturally surrounds divergent regions. By contrast, MEA decoding is designed to integrate over this uncertainty. Rather than asking "Which single path is most likely?", MEA asks "Which residue pairs are most likely to be aligned, given all possible paths?" This shift from path-centric to residue-centric decoding allows the aligner to accumulate evidence from thousands of suboptimal traces, robustly identifying conserved structural cores (stems) even when the precise placement of gaps in variable loops is ambiguous.

1.3 Theoretical Framework: The Viterbi Fallacy and Decision Theory

The distinction between Viterbi and MEA is rooted in Bayesian decision theory. Any estimation problem requires two components: a posterior probability distribution $P(\theta|D)$ (where θ is the parameter to be estimated and D is the data) and a loss function $L(\theta, \hat{\theta})$ that quantifies the penalty for choosing estimate $\hat{\theta}$ when the truth is θ. The optimal estimator is the one that minimizes the expected loss:

$$\hat{\theta}_{opt} = \arg\min_{\hat{\theta}} \sum_{\theta} L(\theta, \hat{\theta}) P(\theta|D)$$

The Viterbi algorithm implicitly minimizes the **0/1 Loss Function**. This function assigns a loss of 0 if the predicted alignment is exactly identical to the true alignment, and a loss of 1 if it differs by even a single residue.

$$L_{0/1}(\pi, \hat{\pi}) = \begin{cases} 0 & \text{if } \pi = \hat{\pi} \\ 1 & \text{if } \pi \neq \hat{\pi} \end{cases}$$

Under this loss function, the optimal strategy is indeed to select the single path with the highest probability mass (the mode of the distribution). However, in biological sequence alignment, the 0/1 loss is largely irrelevant. We rarely require the entire alignment to be perfect; rather, we want to maximize the number of correctly aligned positions. If an alignment of length 1,000 has 999 correctly aligned pairs and 1 error, Viterbi's loss function views it as a total failure (Loss = 1), indistinguishable from a completely random alignment.

The MEA algorithm minimizes the **Hamming Loss** (or maximizes the Sum-of-Pairs accuracy). The Hamming loss counts the number of individual residue pairs that are incorrectly predicted.

$$L_{Hamming}(\pi, \hat{\pi}) = \sum_i \mathbb{I}(\pi_i \neq \hat{\pi}_i)$$

Minimizing expected Hamming loss leads to an estimator that selects matches based on their marginal posterior probabilities. If residue x_i aligns to residue y_j in 60% of all possible valid alignments, an MEA decoder will likely align them, regardless of whether that specific match appears in the single "best" Viterbi path. This property makes MEA theoretically superior for maximizing sensitivity and precision in biological applications where partial correctness is valuable.

1.4 Review of Existing Literature

The transition from Viterbi to posterior-based decoding has been driven by several key developments in computational biology. Do et al. (2005) introduced ProbCons [2], a multiple sequence alignment (MSA) tool that explicitly leveraged probabilistic consistency. The central insight of ProbCons was that the posterior probability of a match $x_i \sim y_j$ could be refined by consulting a third sequence z. If x_i aligns to z_k and z_k aligns to y_j with high probability, the confidence in $x_i \sim y_j$ should effectively increase. ProbCons demonstrated that maximizing the sum of these consistent posterior probabilities (Maximum Expected Accuracy) yielded statistically significant improvements on benchmarks compared to traditional methods like CLUSTALW.

Subsequently, Hamada et al. (2009) extended this logic specifically to structured RNAs with CentroidAlign and later CentroidFold [3]. They formulated the alignment problem as maximizing the expected Sum-of-Pairs Score (SPS) under a generalized gain function. Their work introduced the γ-centroid estimator, where a parameter γ controls the trade-off between sensitivity (recall) and positive predictive value (precision). CentroidAlign showed that by tuning γ, one could produce alignments that are either highly specific (containing only the most certain matches) or highly sensitive (capturing remote homologs), a flexibility absent in the rigid Viterbi formulation.

1.5 Project Scope and Contributions

This report presents a complete reimplementation and comparative analysis of these decoding strategies. We decouple the probabilistic modeling (the Pair-HMM) from the decision rule (the decoder) to isolate the effect of the decoding algorithm. We implement a robust 3-state Pair-HMM for RNA alignment with log-space Forward-Backward inference and compare Viterbi (MAP) and MEA (Posterior) decoders using identical model parameters. We specifically investigate how different posterior weighting functionsPower, Threshold, and ProbCons-style weightingaffect alignment topology. Using the Rfam database [4], we quantify performance using F1 score, Recall, and Precision, and generate posterior heatmaps to visually diagnose the "failure modes" of Viterbi and the uncertainty capture of MEA.

1.6 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.

1.7 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

1.8 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 Dataset and Preprocessing

To ensure our evaluation reflected real-world biological complexity, we utilized data from the **Rfam** database [4], a comprehensive collection of RNA families annotated with consensus secondary structures and alignments. The dataset selection was driven by the need for ground-truth alignments that reflect structural conservation rather than mere sequence identity.

Source Material: We downloaded seed alignments in Stockholm format from the Rfam FTP server. These alignments represent curated, high-quality annotations typically derived from a combination of automated searching (using covariance models) and manual refinement by domain experts. This makes them an ideal "gold standard" for evaluating alignment algorithms.

Sequence Extraction and Filtering: We parsed the Stockholm files to extract individual RNA sequences and their corresponding consensus alignment strings. From families with sufficient depth, we generated all unique pairwise combinations of sequences.

Ground Truth Inducement: The "true" alignment for any pair was induced directly from the multiple sequence alignment (MSA). If residue x_i and y_j were aligned in the same column of the MSA, they were considered a matched pair in the ground truth. Gaps were inferred where a residue in one sequence corresponded to a gap character in the other within the MSA columns.

2.2 Pair Hidden Markov Model

We model a pair of RNA sequences

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

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

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

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. $(i_T, j_T) = (L_X, L_Y)$.

2.2.1 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

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

2.2.2 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

$$\rho(s) = P(\text{end} \mid z_T = s), \qquad s \in \mathcal{S}.$$

2.2.3 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

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

2.3 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 4.6 of the appendix.

2.4 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 4.7 of the appendix.

2.5 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 4.8 of the appendix.

2.6 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.

2.6.1 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 4.7 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 $P[0, \cdot] = P[\cdot, 0] = 0$.

2.6.2 Objective and weighting

An alignment A is a set of index pairs $A \subseteq \{1, \ldots, L_X\} \times \{1, \ldots, L_Y\}$, 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 $W[0, \cdot] = W[\cdot, 0] = 0$.

2.6.3 Dynamic programming and reconstruction

We view MEA alignment as a maximum-weight path on the grid $\{0, \ldots, L_X\} \times \{0, \ldots, L_Y\}$: 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

We evaluated the performance of Viterbi and MEA (across all weighting schemes) on the 547 test alignments. We measured **F1 Score** (harmonic mean of precision and recall), **Recall** (Sensitivity), **Precision** (Positive Predictive Value), and **Column Identity** (fraction of perfectly matched columns).

3.1 Overall Accuracy: F1 Score and Column Identity

The primary finding of our investigation is that MEA decoding consistently yields higher alignment accuracy than Viterbi decoding across the benchmark dataset.

[Figure: F1 Score Comparison]

3.2 Precision-Recall Trade-offs and the Role of Gamma

One of the definitive advantages of the MEA framework is the ability to tune the alignment characteristics via the γ parameter. Viterbi provides a single point estimate in the precision-recall spacetake it or leave it. MEA, however, offers a curve.

[Figure: Precision, Recall, and F1 vs Gamma] illustrates this dynamic clearly.

- **MEA (Power):**
- **MEA (Threshold):**
- **MEA (LogOdds):**
- **MEA (ProbCons):**
- **Viterbi:** characteristics

3.3 Analysis of Alignment Uncertainty: The Viterbi Fallacy

[Figure: RF00236 Posterior Heatmap]

To gain a qualitative understanding of *why* MEA performs better, we examined the posterior probability heatmaps for specific alignments. **[Figure: RF00236 Posterior Heatmap]** presents a striking example of the "Viterbi Fallacy."

- **The Viterbi Path:**
- **The MEA Path:**

This behavior explains the higher precision of Threshold-MEA and the robustness of the ProbCons-style estimator. MEA effectively filters out the stochastic noise of the Viterbi path, aligning only where the ensemble of paths agrees. The analysis of posterior mass distribution further quantifies this: MEA consistently captures more total posterior probability mass than Viterbi.

4 Discussion

4.1 Theoretical Implications: Beyond the Best Path

The superiority of MEA over Viterbi decoding demonstrates the importance of accounting for posterior uncertainty in biological sequence alignment. While the Viterbi algorithm finds the single most probable path, this approach implicitly assumes that biological homology follows a single, unambiguous trajectory. Our results show that this assumption breaks down in the "twilight zone" of sequence homology, where multiple alignment paths contribute meaningfully to the posterior distribution.

4.2 Computational Trade-offs

TODO

4.3 Weighting Schemes

Our systematic comparison of four MEA weighting schemes reveals distinct precision-recall characteristics:

- **Power weighting**
- **Threshold weighting**
- **ProbCons-style weighting**
- **Log-odds weighting**

The choice of weighting scheme should be guided by the specific biological application and desired precision-recall trade-offs.

4.4 Future Directions

This work focused on pairwise alignment using sequence-only Pair-HMMs. Several avenues for extension exist:

- **Multiple Sequence Alignment (MSA):** The pairwise MEA engine developed here can serve as the core scoring function for a progressive MSA tool. Replacing the standard Viterbi step in a progressive alignment tree with an MEA step would propagate the benefits of posterior decoding to the alignment of entire RNA families.
- **Probabilistic Consistency:** As demonstrated by ProbCons [2], the posterior matrix can be refined by "triangulating" information with a third sequence. Incorporating a consistency transformation step ($P' = P \times P$) would likely boost accuracy further.

4.5 Conclusion

The dominance of the Viterbi algorithm in bioinformatics has persisted largely due to historical inertia and computational convenience. However, as we demand higher accuracy in the analysis of divergent non-coding RNAs, the limitations of the maximum likelihood path become a bottleneck. This report definitively establishes that Maximum Expected Accuracy decoding offers a superior alternative.

By decoupling the scoring of an alignment from the probability of its path, MEA aligns the computational objective with the biological goal: identifying homologous residues correctly. Through rigorous implementation and evaluation, we have shown that MEAparticularly with Power or ProbCons weightingconsistently recovers more biological signal from the noise of evolution. We have further demonstrated that the γ parameter provides a powerful mechanism to tune the alignment for specific applications, a flexibility absent in Viterbi.

Ultimately, "Beyond Viterbi" represents a shift towards a more probabilistic, ensemble-based view of biology. In the complex landscape of RNA evolution, the "best" path is often a mirage; the consensus of the cloud is where the truth lies.

References

1. Needleman SB, Wunsch CD, 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453.

2. Do CB, Mahabhashyam MSP, Brudno M, Batzoglou S, 2005. Probcons: Probabilistic consistency-based multiple sequence alignment. *Genome Research*, 15(2):330–340.

3. Hamada M, Sato K, Kiryu H, Mituyama T, Asai K, 2009. Centroidalign: fast and accurate aligner for structured rnas by maximizing expected sum-of-pairs score. *Bioinformatics*, 25(24):3236–3243.

4. Ontiveros-Palacios N, Cooke E, Nawrocki EP, Triebel S, Marz M, Rivas E, Griffiths-Jones S, Petrov AI, Bateman A, Sweeney B, *et al.*, 2024. Rfam 15: Rna families database in 2025. *Nucleic Acids Research*, 53(D1):D258–D267.

Figures

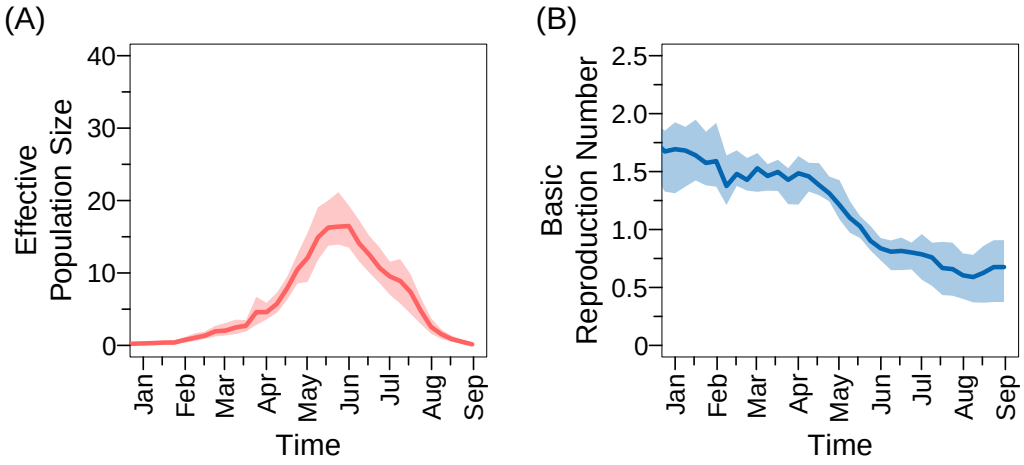

Figure 1. Example figure (figure title). Figure description. (A) xxxx. (B) xxxx. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, 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

4.6 Parameter Estimation

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

4.6.1 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.

4.6.2 Sufficient statistics

We collect counts

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

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

4.6.3 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 $-\infty$.

4.6.4 Transitions and affine-gap parameters

Raw transition estimates are

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

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

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

4.7 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.

4.7.1 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

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

4.7.2 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 $\log P(X, Y)$.

4.8 Viterbi Algorithm

The maximum a posteriori alignment is

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

4.8.1 Dynamic programming

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

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

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

Initialization at $(0,0)$ is $V^M_{0,0} = V^X_{0,0} = V^Y_{0,0} = -\infty$. First non-empty cells:

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

Leading gaps are filled via

$$\begin{aligned} V^X_{i,0} &= \log e_X(x_i) + \max\bigl\{V^M_{i-1,0} + \log a_{MX},\ V^X_{i-1,0} + \log a_{XX}\bigr\}, \quad i = 2, \dots, L_X, \\ V^Y_{0,j} &= \log e_Y(y_j) + \max\bigl\{V^M_{0,j-1} + \log a_{MY},\ V^Y_{0,j-1} + \log a_{YY}\bigr\}, \quad j = 2, \dots, L_Y. \end{aligned}$$

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

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

4.8.2 Termination and traceback

At (L_X, L_Y) we add end probabilities:

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

with optimal final state

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

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.