Fast **S**tatistical **A**lignment

Robert K. Bradley[a,b*], Adam Roberts[c],

Michael Smoot[d], Sudeep Juvekar[c], Jaeyoung Do[e],

Colin Dewey[e,f], Ian Holmes[g], Lior Pachter[a,b]

[a]Department of Mathematics, University of California, Berkeley, CA 94720, USA

[b]Department of Molecular & Cellular Biology, University of California, Berkeley, CA 94720, USA

[c]Department of EECS, University of California, Berkeley, CA 94720, USA

[d]Department of Bioengineering, University of California, San Diego, CA 92093, USA

[e]Department of Computer Sciences, University of Wisconsin, Madison, WI 53706, USA

[f]Department of Biostatistics & Medical Informatics, University of Wisconsin, Madison, WI 53706, USA

[g]Department of Bioengineering, University of California, Berkeley, CA 94720, USA

Abstract

We describe a new program for the alignment of multiple biological sequences that is both statistically motivated and fast enough for problem sizes that arise in practice. Our **F**ast **S**tatistical **A**lignment program is based on pair hidden Markov models which approximate an insertion/deletion process on a tree and uses a sequence annealing algorithm to combine the posterior probabilities estimated from these models into a multiple alignment. `FSA` uses its explicit statistical model to produce multiple alignments which are accompanied by estimates of the alignment accuracy and uncertainty for every column and character of the alignment—previously available only with alignment programs which use computationally-expensive Markov Chain Monte Carlo approaches—yet can align thousands of long sequences.

*To whom correspondence should be addressed: rbradley@math.berkeley.edu.

Moreover, FSA utilizes an unsupervised query-specific learning procedure for parameter estimation which leads to improved accuracy on benchmark reference alignments in comparison to existing programs. The centroid alignment approach taken by FSA, in combination with its learning procedure, drastically reduces the amount of false-positive alignment on biological data in comparison to that given by other methods.

The FSA program and a companion visualization tool for exploring uncertainty in alignments can be used via a web interface at http://orangutan.math.berkeley.edu/fsa/ and the source code is available at http://fsa.sourceforge.net/.

Author Summary

Biological sequence alignment is one of the most fundamental problems in comparative genomics, yet it remains unsolved. Over sixty sequence alignment programs are listed on `wikipedia` and many new programs are published every year. However, many popular programs suffer from pathologies such as aligning unrelated sequences and producing discordant alignments in protein (amino acid) and codon (nucleotide) space, casting doubt on the accuracy of the inferred alignments. Inaccurate alignments can introduce large and unknown systematic biases into downstream analyses such as phylogenetic tree reconstruction and substitution rate estimation. We describe a new program for multiple sequence alignment which can align protein, RNA and DNA sequence and improves on the accuracy of existing approaches on benchmarks of structural alignments. Our approach, which seeks to find the alignment which is closest to the truth under our statistical model, leaves unrelated sequences largely unaligned and produces concordant alignments in protein and codon space. It is fast enough for difficult problems such as aligning orthologous genomic regions or aligning hundreds or thousands of proteins. It furthermore has a companion GUI for visualizing the estimated alignment reliability.

Contents

1 Introduction

The field of biological sequence alignment is very active, with numerous new alignment programs developed every year in response to increasing demand driven by rapidly dropping sequencing costs. The list of approximately 60 sequence alignment programs on the `wikipedia` compilation provides a lower bound for the number of available tools, and illustrates the confusing choice facing biologists who seek to select the "best" program for their studies. Nevertheless, the `ClustalW` program [1, 2], published in 1994, remains the most widely-used multiple sequence alignment program. Indeed, in a recent review of multiple sequence alignment [3], the authors remark that "to the best of our knowledge, no significant improvements have been made to the [`ClustalW`] algorithm since 1994 and several modern methods achieve better performance in accuracy, speed, or both." Therefore, it is natural to ask, "Why do alignment programs continue to be developed, and why are new tools not more widely adopted by biologists?"

A key issue in understanding the popularity of `ClustalW` is to recognize that it is difficult to benchmark alignment programs. Alignments represent homology relationships among the nucleotides, or amino acids, of the genomes of extant species, and it is impossible to infer the evolutionary history of genomes with absolute certainty. Comparisons of alignment programs therefore rely on databases of structural alignments for proteins and RNA or on gene loci or simulated data for DNA. Each type of benchmark is vulnerable to manipulation, and furthermore may not represent the problem setups which are most relevant to biologists. The result is that biologists are confronted with many programs and publications, but it is frequently unclear which approach will give the best results for the everyday problems which biologists seek to address.

Adding to the difficulty of selecting software tools is the blur between programs and ideas. Developers of alignment programs make choices about the objective functions to optimize, the statistical models to use, and the parameters for these models, but the relative impact of individual choices is rarely tested [4]. Discordance among programs is frequently noted, e.g., [5], but the different architectures of individual programs, and in some cases the lack of open software, makes

it difficult to explore novel combinations of existing ideas for improving alignments.

In lieu of these issues, biologists have favored the conservative approach of using the tried and trusted `ClustalW` program, although they frequently use it in conjunction with software which allows for manual editing of alignments [6]. The rationale behind alignment-editing software is that trained experts should be able to correct alignments by visual inspection, and that effort is better expended on manually correcting alignments rather than searching for software that is unlikely to find the "correct" alignment anyway. Although manual editing approaches may be cumbersome, they have been used for large alignments (e.g., [7]).

We therefore approached the alignment problem with the following goals in mind:

1. An approach which seeks to maximize the expected alignment accuracy (in a statistical sense). Our approach seeks to find the alignment with minimal expected distance to the true alignment of the input sequences, where the true alignment is treated as a random variable, with the probability of each true alignment determined under a statistical model. Explicitly incorporating a statistically-motivated objective function, this "expected accuracy" approach to alignment allows us to visualize alignments according to different quality measures, including their expected accuracy, sensitivity, specificity, consistency and certainty. We therefore offer biologists a way to compare alignments that allows them to quantitatively judge differences in alignment quality when they are performing manual edits.

2. A method which is robust to variation in evolutionary parameters. We sought a method which is robust to phenomena such as sequences of differing evolutionary distances and base composition. While "phylogenetic alignment" methods seek to accomplish this by explicitly modeling alignments on trees [8, 9, 10, 11], a computationally-costly procedure, we use machine-learning techniques to estimate model parameters on the fly based on only pairwise comparisons of sequences.

3. Robust results when faced with the wide range of alignment problems encountered today. We

sought to create a single program which is capable of achieving high accuracies on protein, RNA and DNA sequences without additional input from, e.g., database homology searches. We additionally sought to make our approach fast enough for large-scale problems such as aligning many sequences or orthologous regions of genomes. (When aligning genomic-size sequences, we assume that the sequences are colinear; we do not attempt to solve the problem of resolving duplications or inversions.)

4. Creation of a modular code base so that future improvements in one aspect of alignment could easily be incorporated into our approach. In particular, we aimed to create a collaborative infrastructure so that "bioinformaticians with expertise in developing software for comparing genomic DNA sequences [can] pool their ideas and energy to produce a compact tool set that serves a number of needs of biomedical researchers" [12].

The "distance-based" approach to sequence alignment, proposed in [13] and implemented in the alignment program `AMAP` [14], offers a useful framework for these goals. Much as distance-based phylogenetic reconstruction methods like Neighbor-Joining build a phylogeny using only pairwise divergence estimates, a distance-based approach to alignment builds a multiple alignment using only pairwise estimations of homology. This is made possible by the sequence annealing technique [14] for constructing multiple alignments from pairwise comparisons.

We have implemented our approach in `FSA`, a new alignment program described below. We give an overview of the structure of `FSA` and explain the details of its components below. The Supporting Text contains instructions for using the `FSA` program and webserver, as well as an introduction to the mathematics of distance-based alignment.

2 Methods

`FSA` builds on the distance-based approach to alignment implemented in `AMAP` [14], introducing:

1. A redesigned sequence annealing algorithm which is fast enough for large problem sizes.

The approach described in [14] and implemented in `AMAP` has approximate time complexity $O(N^4 \cdot L^3)$ for N sequences of length L, making it unusable for many or long sequences. We have redesigned the algorithm and thereby reduced its complexity to approximately $O(N^2 \cdot L^2 \cdot \log^2(N \cdot L))$ in default mode and to $O\left(N \cdot \log N \cdot L^2 \cdot \log^2(N \cdot L)\right)$ in --fast mode.

2. An iterative refinement procedure for escaping from local minima after sequence annealing terminates. As a greedy algorithm, sequence annealing is only guaranteed to find a local optima of the objective function.

3. A "unsupervised query-specific learning" approach to alignment. While many alignment programs use a model with parameters tuned for best performance on a particular dataset such as BAliBASE 3 [15], `FSA` re-estimates all model parameters, including gap and substitution parameters, on the fly. This "unsupervised query-specific learning" alignment method, inspired by query-specific structure learning for graphical models, makes `FSA` very robust: it can produce superior estimates of homology for sets of sequences which are subject to very different evolutionary constraints.

4. An "anchor annealing" strategy for identifying a consistent set of alignment anchors identifying homologous regions across many long sequences. Assuming that long, (near-) exact matches have a posterior probability ~ 1 of being aligned, `FSA` finds a consistent set of such matches across the multiple alignment and uses these anchors to subdivide each pairwise comparison. This eliminates the need for prohibitively-expensive inference on long sequences and permits `FSA` to align genome-scale data.

5. Software for visualizing alignment reliability statistics. While `FSA` utilizes only computationally-tractable pairwise comparisons and never explicitly models all sequences simultaneously, it can nonetheless produce the estimates of correctness of single characters in the alignment previously available only for computationally-expensive programs such as `BAli-Phy` [9] and `StatAlign` [11]. The included GUI allows the user to visualize the estimated reliabilities

of alignments as well as the sequence annealing process itself.

2.1 Overview

Figure 1 shows an overview of the components of the FSA alignment algorithm, described in detail below.

The **input** to FSA is a set of protein, RNA or DNA sequences. These sequences are assumed to be homologous, although FSA is robust to non-homologous sequence (Section 3). The **output** of FSA is a global alignment of the input sequences which is a (local) optima of the expected accuracy under FSA's statistical model.

FSA first performs pairwise comparisons of the input sequences to determine the posterior probabilities that individual characters are aligned (note, however, that it does not yet actually align any sequences). While the number of possible pairwise comparisons is quadratic in the number of sequences being aligned, giving unfavorable runtimes for datasets of many sequences, FSA overcomes this problem by reducing the number of pairs of sequences that are compared. This is accomplished using a randomized approach inspired by the Erdös-Rényi theory of random graphs, thereby drastically reducing the computational cost of multiple alignment.

After obtaining pairwise estimates of homology at the single-character level, FSA uses the sequence annealing technique [14] to construct a multiple alignment. This approach to alignment seeks to maximize the expected accuracy of the alignment using a steepest-ascent (greedy) algorithm.

The architecture of FSA reflects the inherent modularity of the distance-based approach to alignment. FSA's inference engine uses the flexible HMMoC code-generation tool [16] and has been parallelized with a separate module, alignments of long sequences are anchored with candidate matches found by the MUMmer suffix trie matching tool [17], and FSA's sequence annealing algorithm improves on the original algorithm and implementation in AMAP [14]. The stand-alone visualization tool uses statistical information produced by FSA, but otherwise is completely independent.

Each of these components can be improved independently of the others, allowing for rapid future improvements in distance-based alignment. For example, FSA's entire statistical model could easily be altered to incorporate position-specific features or completely replaced with a discriminative or hybrid generative-discriminative model.

2.2 Core components

The components described here correspond roughly to the simplest mode of operation for FSA, outlined in bold in Figure 1.

Input and Output. FSA accepts FASTA-format input files and produces alignments in multi-FASTA or Stockholm format. The server also outputs Phylip and ClustalW formatted files.

Estimating posterior probabilities of alignment. Distance-based alignment, relying on pairwise estimations of homology, operates on the posterior probability distributions that characters in two sequences are aligned. FSA uses the standard three or five-state pair hidden Markov model (Pair HMM) shown in Figure 2 to infer these posterior probabilities of alignment, as well as posterior probabilities that characters are unaligned or gapped. The structure of the Pair HMM and its parameters can be controlled through the command-line options (see Text S5 for details).

The standard Forward-Backward algorithm on a Pair HMM has time complexity $O(L^2)$ for two sequences of length L.

Merging probabilities. After calculating the posterior probabilities of alignment for characters in all sequence pairs, $\mathbb{P}(x_i \sim y_j|X,Y)$ that individual characters x_i and y_j are aligned and $\mathbb{P}(x_i \sim -|X,Y)$ that a character x_i is gapped to sequence Y, FSA sorts these probabilities according to a weighting function which gives a hill-climbing procedure which is a steepest-ascent algorithm in the weighting function (Section 2.4).

Sequence annealing. After estimating these posterior probabilities and sorting them, FSA creates a multiple alignment with the sequence annealing technique [14]. Sequence annealing attempts to find the alignment with the minimum expected distance to the truth, computed for two sequences X and Y as

$$\mathcal{A}_{optimal} = \text{argmin}_{\mathcal{A}^*} \mathbb{E}\left[d\left(\mathcal{A}^*, \mathcal{A}\right)\right]_{\mathbb{P}(\mathcal{A}|\text{X,Y})} \ .$$

The distance $d\left(\mathcal{A}^*, \mathcal{A}\right)$ between two alignments is defined as the number of characters for which they make different homology statements, taking into account both matches and gaps (Equation 1). This distance is a true metric on alignments [14]. The alignment with the minimum expected distance to the truth is equivalent to the alignment with the maximum expected accuracy,

$$\mathcal{A}_{optimal} = \text{argmax}_{\mathcal{A}^*} \mathbb{E}\left[Acc\left(\mathcal{A}^*, \mathcal{A}\right)\right]_{\mathbb{P}(\mathcal{A}|\text{X,Y})} \ ,$$

where we define the accuracy $Acc\left(\mathcal{A}^*, \mathcal{A}\right)$ of an alignment $\mathcal{A}^*$ with respect to a reference, "true" alignment $\mathcal{A}$ as the fractional number of characters for which they make identical homology statements. This accuracy measure provides a natural way to evaluate alignment quality (Equation 3). (Note that it linearly penalizes incorrectly-placed gaps.)

The posterior probabilities over alignments $\mathbb{P}\left(\mathcal{A}|X, Y\right)$ used in the optimization are given by FSA's statistical model (a Pair HMM). FSA extends this definition of an optimal pairwise alignment to an optimal multiple alignment by taking sum-of-pairs over all sequences.

Using this expected accuracy as an objective function for a greedy maximization, sequence annealing begins with the null alignment (all sequences unaligned) and merges single columns (aligns characters) according to the corresponding expected increase in $\mathbb{E}\left[Acc\left(\mathcal{A}^*, \mathcal{A}\right)\right]_{\mathbb{P}(\mathcal{A}|\text{data})}$, the similarity to the truth under FSA's statistical model. Whereas progressive alignment methods take large steps in alignment space by aligning entire sequences at each step, the distance-based approach takes the smallest-possible steps of aligning single characters.

Section 2.4 gives an in-depth discussion of the objective function and how it arises naturally from FSA's representation of an alignment as a partially ordered set (POSET) or directed acyclic graph (DAG).

Inferring indel events. In FSA's definition of an alignment, an alignment consists solely of a specification of homology. This definition differs from the standard definition of a multiple alignment, which implies an evolutionary history of substitution and indel events. For example, FSA (internally) considers the two alignments shown in Figure 3 to be equivalent.

This is problematic for comparative genomics analyses which use an alignment to infer evolutionary parameters such as indel frequencies across a clade. In order to output a single global alignment from which such evolutionary parameters can be inferred, we choose a topological ordering of the alignment POSET which corresponds to a maximum-parsimony interpretation of indel events. FSA outputs the global alignment with the minimum number of "gap openings" across the individual sequences (the right-hand alignment in Figure 3).

2.3 Optional components

Selection of a subset of pairs for alignment speedup. FSA can efficiently align hundreds or even thousands of sequences. By default it performs exhaustive distance-based alignment, which requires an all-pairs comparison of the N sequences, costing $O(N^2 \cdot L^2)$. However, this prohibitive computational cost can be sharply reduced by only performing pairwise comparisons on a subset of all possible $N \cdot (N-1)/2$ sequence pairs.

In order to ensure a complete alignment, where no sequence is left unaligned, each sequence must be connected to every other sequence by a series of pairwise comparisons. For N input sequences, a minimum of $(N-1)$ pairwise comparisons are necessary to give a complete alignment; this corresponds to building a spanning tree on the sequences. In order to select a randomized subset of pairs for comparison which falls between the extremes of $(N-1)$ and $N \cdot (N-1)/2$ pairs, FSA uses the theory of Erdös-Rényi random graphs. By the Erdös-Rényi theory, a random graph will

almost surely be connected if the edge-creation probability satisfies $p > (1+\epsilon)\log N/N$, which is very low for large N (ϵ is a small positive number). Guided by this result, FSA performs fast alignments by first constructing a spanning tree on the input sequences and then performing each possible pairwise comparison with some probability p proportional to the connectedness threshold. The savings are dramatic—for $N = 1,000$ sequences, the Erdös-Rényi threshold probability is 0.007, corresponding to an algorithm which is over 100 times as fast as an all-pairs comparison—and we have observed little loss of accuracy from considering only a subset of pairs (see, e.g., Table 2).

This speedup reduces the time complexity of both inference and sequence annealing. The cost of inference is reduced to $O\left(N \cdot \log N \cdot L^2\right)$ and the "worst average case" runtime of sequence annealing to $O\left(N \cdot \log N \cdot L^2 \cdot \log^2(N \cdot L)\right)$, where we align N sequences of length L by making $O(N \cdot \log N)$ pairwise comparisons (Section 2.4).

Finding anchors. FSA can align megabase-long sequences using an "anchor annealing" strategy. Analogously to other genome alignment programs such as MAVID [18], MLAGAN [19], CHAOS/DIALIGN [20] and Pecan [21], FSA uses long matches to anchor regions of the alignment and performs inference with dynamic programming in between anchors. FSA uses the fast suffix trie matching program MUMmer [17] to find candidate anchors, and can find anchors in either nucleotide or protein space (by translating the sequence in all frames). FSA requires that anchors be maximal unique matches in both sequences ("MUMs"). The restriction to unique matches helps to prevent false-positive anchors due to, e.g., repetitive sequence: for example, a microsatellite can appear as a candidate anchor only if it appears exactly once, with identical copy number, in each sequence.

FSA utilizes its distance-based approach to find a consistent set of anchors across all sequences simultaneously, thereby making maximal use of additional constraints from other sequences. This "anchor annealing" strategy is conceptually similar to the procedures used in programs for aligning long sequences such as CHAOS/DIALIGN, MAVID, Pecan and TBA, which return partially-ordered sets of anchors, thereby permitting constraints to be projected across multiple sequences.

As with sequence annealing, described in Section 2.2 and Section 2.4, this "anchor annealing" can be accomplished efficiently with a greedy algorithm based on the Pearce-Kelly algorithm. FSA uses the same code for both sequence and anchor annealing, although the objective function is different: Anchor "scores" correspond to p-values under a null model rather than probabilities of homology, and so there are no "gap" probabilities $\mathbb{P}(x_i \sim -|X,Y)$ or $\mathbb{P}(- \sim y_j|X,Y)$ which contribute to the anchor-annealing analog of the expected accuracy $\mathbb{E}[Acc]$.

Rather than aligning entire anchors across the multiple alignment in order to find a consistent set of anchors, FSA finds a consistent set of anchor centroids across all sequences and then prunes the resulting anchors at a pairwise level. The result is a set of anchors between pairs of sequences whose centroids are consistent across all sequences and which are non-overlapping between pairs of sequences. This heuristic approach permits FSA to quickly find consistent anchors across many sequences.

After finding a consistent set of anchors across the multiple alignment, FSA assumes that these anchors are correctly aligned with probability ~ 1 and then aligns the regions between anchors using dynamic programming. When anchored alignment is performed, the posterior probabilities that individual characters are aligned which FSA uses to inform construction of the multiple alignment are conditioned on the set of anchors chosen. Therefore, if all anchors correspond to true homology then these probabilities will be correctly estimated despite the anchoring heuristic; however, if incorrect anchors are chosen, then individual probabilities of alignment can be similarly incorrect.

While FSA's restriction of candidate anchors from MUMmer to MUMs produces a very specific set of anchors, the restriction can be too stringent to obtain sensitive alignments of diverged or very long sequences, for which there are few (unique) exact matches. To address this potential problem, FSA can use the fast homology-search tool exonerate [22] to find inexact matches to use as anchors as well. Furthermore, when performing whole-genome alignment, homologous genomic regions are typically first identified with a program such as Mercator [23] and then each region is aligned with a nucleotide-level alignment program. FSA can use the seed matches, frequently

coding genes, which define the homologous genomic regions to inform its anchor annealing.

Because FSA uses the fast tool MUMmer to find anchors, it can rapidly align closely-related sequences, such as mitochondrial DNA, for which anchors span most of the alignment, making costly dynamic programming largely unnecessary.

The Pair HMM and parameter estimation. The distinct functional constraints acting on biological sequences give rise to very different patterns of molecular evolution, each implying distinct parametrizations of an appropriate model for alignment. For example, if substitutions or indels are more frequent in one lineage than in the others, then using a single model for all sequences (which does not reflect these differing constraints) can give misleading results. Nonetheless, sequence alignment algorithms traditionally use a single model for all sequences.

In order to overcome these difficulties, FSA uses "query-specific learning," wherein a different model is learned for each pairwise comparison (the "query"). This is done in a completely unsupervised framework: FSA uses an unsupervised Expectation Maximization (EM) algorithm to estimate transition (gap) and emission (substitution) probabilities of the Pair HMM on the fly.

Despite its unsupervised nature, FSA's query-specific learning needs remarkably little sequence data to effectively learn parameters. Standard alignment algorithms estimate parameters from thousands or tens of thousands of pairs of *aligned* sequences; in contrast, we empirically observe good results with as little input data as two *unaligned* DNA or RNA sequences of length 60 nucleotides or four *unaligned* protein sequences of length 266 amino acids. These figures correspond to observing each of the independent parameters of a substitution matrix four times.

While FSA learns distinct transition parameters for every pair of query sequences regardless of the sequence composition, it uses different learning strategies for nucleotide and amino acid emission matrices. Because a pair emission matrix over aligned nucleotides has only $(4^2 - 1) = 15$ free parameters, FSA can learn a different emission distribution for every pairwise comparison of all but the shortest RNAs or DNAs. In contrast, emission matrices over aligned amino acids have $(20^2 - 1) = 3,999$ free parameters, thereby preventing FSA from learning independent models for

each pair of proteins. FSA therefore learns a single emission matrix using an all-pairs comparison for protein sequences.

Because FSA uses unsupervised learning on very sparse data, overfitting is a serious concern. FSA attempts to prevent overfitting by (1) using a weak Dirichlet regularizer (prior) when estimating both transition and emission probabilities, and (2) terminating parameter learning before the EM algorithm converges. By default the Dirichlet emission priors are scaled such that total number of pseudocounts for aligned characters is equal to the number of free parameters in a symmetric pair emission matrix. As is the case for other machine-learning algorithms, it can be shown that termination before convergence of query-specific learning is equivalent to a form of regularization (likelihood penalty).

If there is insufficient sequence data for effective learning, then FSA does not estimate parameters and instead uses default parameters to construct an alignment. The default parameters values, as well as seeds used for the learning algorithm, are discussed in Text S5.

Parallelization mode. While FSA can align hundreds or thousands of sequences on a single computer, it can handle truly large-scale problems by running in a parallelized mode on a computer cluster. FSA's distance-based approach to alignment gives the multiple alignment a natural independence structure: each pairwise alignment is independent of all other pairs, allowing dramatic runtime reduction by distributing the individual pairwise computations to different processors.

Two factors were considered for the parallelization of FSA: (1) communication overhead between nodes, and (2) workload distribution over the different processors. For example, distributing jobs in very small batches may reduce processor idle time but lead to high overhead; in contrast, using large batches may increase idle time but minimize overhead. FSA's parallelization mode uses the "Fixed-Size Chunking" strategy described in [24], whereby each of the P processors runs on chunks of $N \cdot (N-1)/(2 \cdot P)$ pairwise comparisons.

While the pairwise comparisons can be naturally parallelized, sequence annealing does not have the same obvious independencies. Therefore, even when running in parallelized mode, FSA

performs sequence annealing on a single node. The parallelization of the annealing step is a future aim for this project. A schematic of the current parallelization strategy is given in Figure 4.

Iterative refinement. As a greedy algorithm, sequence annealing is only guaranteed to find a local optima of the expected accuracy. `FSA` therefore uses an iterative refinement strategy after sequence annealing terminates to locally improve the alignment. For each round of iterative refinement, `FSA` looks at every character's position in the multiple alignment and sees whether the objective function can be improved by moving it to another position (without violating the consistency constraints of the multiple alignment). `FSA` assembles a list of such candidate character shifts, orders the list by the expected change in the objective function, and then performs the shifts. Iterative refinement terminates when there are no candidate shifts which improve the objective function.

Visualization. `FSA`'s included GUI permits the user to visually assess alignment quality under `FSA`'s statistical model according to different measures, including expected accuracy, sensitivity, specificity, consistency and certainty. This permits biologists and bioinformaticians to incorporate reliability measures into downstream analyses, such as evolutionary rate measurements and phylogenetic reconstruction. Incorporating such information can produce distinctly different results. For example, over-aligned non-conserved sequence can cause a systematic bias towards long branch lengths; this can be ameliorated by incorporating the expected accuracy statistics produced by `FSA` into reconstruction algorithms. Figure 5 shows a sample protein alignment colored by the expected alignment accuracy under `FSA`'s statistical model as well as the true accuracy (based on a reference structural alignment).

`FSA`'s GUI can color alignments according to five different measures of alignment quality. Characters x_i in a multiple alignment can be colored according to:

- Accuracy: The expected accuracy with which x_i is aligned to other characters or gaps.
- Sensitivity: The expected sensitivity with which x_i is aligned to other characters.

- Specificity: The expected specificity with which x_i is aligned to other characters.
- Certainty: The certainty with which x_i is aligned correctly (whether there is a good alternate choice).
- Consistency: The consistency of the many pairwise comparisons used to construct the multiple alignment and the implied optimality of the alignment of x_i to other characters or gaps in the multiple alignment.

See Text S3 for detailed descriptions of how these measures are defined and calculated using `FSA`'s statistical model.

The GUI also provides a visual and statistical guide when manually editing alignments.

2.4 The mathematics of distance-based alignment

The POSET view of alignments. `FSA` defines an alignment to be a set of homology statements between characters. This definition of an alignment is expressed mathematically as a partially ordered set (POSET) [25], which is equivalent to viewing an alignment as a Directed Acyclic Graph (DAG) as described in [26] and implemented in the `POA` alignment program.

A biologically-intuitive metric on alignments (introduced in [14]) emerges naturally from this definition. We define the distance $d(\mathcal{A}^*, \mathcal{A})$ between two alignments to be just the number of characters for which they make different homology statements. For two sequences X and Y of lengths L_X and L_Y, the distance between two alignments is just $d(\mathcal{A}^*, \mathcal{A}) = L_X + L_Y - Sim(\mathcal{A}^*, \mathcal{A})$, where the similarity measure $Sim(\mathcal{A}^*, \mathcal{A})$ is the number of characters for which $\mathcal{A}^*$ and $\mathcal{A}$ make identical homology statements. The distance is therefore computed as:

$$d(\mathcal{A}^*, \mathcal{A}) = L_X + L_Y - \left[2 \cdot \sum_{i,j\,:\, x_i \sim y_j \in \mathcal{A}^*} \mathbf{1}\{x_i \sim y_j \in \mathcal{A}\} + \sum_{i\,:\, x_i \sim - \in \mathcal{A}^*} \mathbf{1}\{x_i \sim - \in \mathcal{A}\} + \sum_{j\,:\, - \sim y_j \in \mathcal{A}^*} \mathbf{1}\{- \sim y_j \in \mathcal{A}\} \right]. \quad (1)$$

This distance $d(\mathcal{A}^*, \mathcal{A})$ between alignments also has a natural mathematical expression. An alignment $\mathcal{A}$ induces a hierarchy on (or partition of) the set X of all characters in the aligned sequences. The distance between two alignments is just the size of the symmetric difference of the two corresponding hierarchies induced on the set X, $d(\mathcal{A}^*, \mathcal{A}) = \mu(\mathcal{A}^* \Delta \mathcal{A})$, where the size μ of the symmetric difference is the cardinality of the set Y which the resulting symmetric difference is defined over.

Inferring indel events. As discussed earlier, the POSET definition of alignments does not attach significance to gap orderings which do not affect the homology specifications of the alignment. `FSA` outputs the global alignment with the minimum number of "gap openings" across the individual sequences.

Finding this minimum-indel global alignment corresponds to finding a minimal-chain decomposition of the alignment POSET. Such a minimal-chain decomposition (Dilworth decomposition) of an arbitrary POSET can be found in time cubic in the number of nodes [27]. Thanks to the special structure of sequence graphs, however, by Theorem 2.1 `FSA` can find this minimum-indel global alignment in time linear in the number of vertices and edges in the graph with a depth-first search.

Theorem 2.1 *The minimal-chain decomposition of a POSET which arises from a linear extension of the POSET can be found in time linear in the number of nodes by a depth-first search of the corresponding DAG.*

Proof Let $M(\mathcal{P})$ be the minimal number of chains in a chain decomposition arising from a linear extension of a POSET $\mathcal{P}$ with elements $\{x\}$. Note that $M(\mathcal{P}) \geq I(\mathcal{P})$, where

$$I(\mathcal{P}) = \sum_{x\colon\, indegree\,(x)>1} indegree\,(x)\,.$$

Here we are treating the POSET as a DAG. More precisely, the DAG represents the transitive

reduction of the partial order, wherein nodes of the DAG represent one or more characters of the alignment and edges of the DAG represent the ordering imposed by the sequences.

Now consider imposing a topological ordering on the corresponding DAG according to the reverse finishing times of a postorder traversal of the graph. This topological sort, consisting of a sequence of chains of the POSET, terminates each chain when the depth-first search reaches a vertex x of the graph for which $indegree(x) > 1$. The topological sort obtained by a postorder traversal therefore has exactly $I(\mathcal{P})$ chains. As $M(\mathcal{P}) \geq I(\mathcal{P})$, this is a minimal chain decomposition. ∎

The objective function. `FSA` seeks to find the alignment with the minimum expected distance to the truth, where the distance $d(\mathcal{A}^*, \mathcal{A})$ between two alignments is defined in Equation 1 as the number of characters for which they make different homology statements. The alignment with the minimum expected distance to the truth is therefore

$$\begin{aligned} \mathcal{A}_{optimal} &= \text{argmin}_{\mathcal{A}^*} \mathbb{E}\left[d(\mathcal{A}^*, \mathcal{A})\right]_{\mathbb{P}(\mathcal{A}|X,Y)} \\ &= \text{argmin}_{\mathcal{A}^*} \mathbb{E}\left[L_X + L_Y - Sim(\mathcal{A}^*, \mathcal{A})\right]_{\mathbb{P}(\mathcal{A}|X,Y)} \\ &= \text{argmax}_{\mathcal{A}^*} \mathbb{E}\left[Sim(\mathcal{A}^*, \mathcal{A})\right]_{\mathbb{P}(\mathcal{A}|X,Y)} . \end{aligned} \tag{2}$$

The alignment with the minimum expected distance to the truth is therefore equivalent to the alignment with the maximum expected similarity to the truth. This gives a natural measure of alignment accuracy, where we define alignment accuracy as the similarity to the truth and the expected alignment accuracy as the expected similarity to the truth,

$$\begin{aligned} Acc(\mathcal{A}^*) &= Sim(\mathcal{A}^*, \text{truth}) / (L_X + L_Y) \\ \mathbb{E}\left[Acc(\mathcal{A}^*)\right]_{\mathbb{P}(\mathcal{A}|X,Y)} &= \mathbb{E}\left[Sim(\mathcal{A}^*, \mathcal{A})\right]_{\mathbb{P}(\mathcal{A}|X,Y)} / (L_X + L_Y) . \end{aligned} \tag{3}$$

We divide by $(L_X + L_Y)$ to normalize the accuracy to the interval $[0, 1]$. We will speak equivalently of finding the alignment with the highest expected similarity to the truth and the highest expected

accuracy.

The posterior probabilities over alignments $\mathbb{P}(\mathcal{A}|X,Y)$ used in the optimization are given by FSA's statistical model (a Pair HMM). Recalling that $Sim(\mathcal{A}^*, \mathcal{A})$ is the number of characters for which $\mathcal{A}^*$ and $\mathcal{A}$ make identical homology statements, we can express the optimal alignment by taking the expectation of Equation 1 and using the posterior probabilities that characters are aligned or gapped (these are computed with the Forward-Backward algorithm):

$$\begin{aligned}
\mathcal{A}_{optimal} &= \text{argmax}_{\mathcal{A}^*} \, \mathbb{E}\left[Sim(\mathcal{A}^*, \mathcal{A})\right]_{\mathbb{P}(\mathcal{A}|X,Y)} \\
&= \text{argmax}_{\mathcal{A}^*} \sum_{\mathcal{A}} \left[\mathbb{P}(\mathcal{A}|X,Y) \, Sim(\mathcal{A}^*, \mathcal{A})\right] \\
&= \text{argmax}_{\mathcal{A}^*} \left[2 \cdot \sum_{i,j \colon x_i \sim y_j \in \mathcal{A}^*} \mathbb{P}(x_i \sim y_j|X,Y) \right. \\
&\qquad \left. + \sum_{i \colon x_i \sim - \in \mathcal{A}^*} \mathbb{P}(x_i \sim -|X,Y) + \sum_{j \colon - \sim y_j \in \mathcal{A}^*} \mathbb{P}(- \sim y_j|X,Y) \right],
\end{aligned} \tag{4}$$

FSA extends this definition of an optimal pairwise alignment to an optimal multiple alignment by taking sum-of-pairs over all sequences.

FSA allows the user to control the sensitivity/specificity trade-off of the method by introducing a gap factor gf into the objective function (Equation 4) being optimized,

$$\begin{aligned}
\mathcal{A}_{optimal} = \text{argmax}_{\mathcal{A}^*} \left[2 \cdot \sum_{i,j \colon x_i \sim y_j \in \mathcal{A}^*} \mathbb{P}(x_i \sim y_j|X,Y) \right. \\
\left. + \mathrm{gf} \cdot \left(\sum_{i \colon x_i \sim - \in \mathcal{A}^*} \mathbb{P}(x_i \sim -|X,Y) + \sum_{j \colon - \sim y_j \in \mathcal{A}^*} \mathbb{P}(- \sim y_j|X,Y) \right) \right],
\end{aligned}$$

A gap factor $\mathrm{gf} = 1$ corresponds to optimizing the expected accuracy, in which case we stop aligning characters when the probability that a character is aligned is equal to the probability that it is unaligned (aligned to a gap); a lower gap factor emphasizes sensitivity and a higher gap factor specificity. By default FSA uses $\mathrm{gf} = 1$.

A steepest-ascent algorithm. The sequence annealing algorithm greedily attempts to maximize the objective function given in Equation 4, where the posterior probabilities of alignment are given by FSA's statistical model. Because exact optimization is tractable only for two (short) sequences, sequence annealing depends on effective heuristics for optimizing this function.

Sequence annealing begins with the null alignment (all characters unaligned) and iteratively aligns single columns until a stopping criteria is reached. In FSA's view of an alignment as a POSET (DAG), aligning single columns during sequence annealing is equivalent to adding relations (adding edges), maintaining alignment consistency is equivalent to ensuring that the partial order is valid (ensuring that the graph is acyclic) and producing a "global" alignment is equivalent to choosing a linear extension (choosing a topological order).

This graph-based approach to sequence alignment, which is used by programs including DIALIGN[28], POA [26], AMAP and FSA, was given an early graph-based formalization in [29]. The resulting algorithm in [29] is qualitatively similar to the sequence annealing approach which AMAP and FSA use. DIALIGN's approach [28], wherein homologous segments of sequences, rather than individual characters, are aligned, uses results from this formalization [30].

The order in which columns are aligned is given by a weighting function on the posterior probabilities of alignment. For two columns, each containing a single character (x_i and y_j), the weighting function is just

$$2 \cdot \mathbb{P}(x_i \sim y_j | X, Y) \Big/ (\mathbb{P}(x_i \sim - | X, Y) + \mathbb{P}(- \sim y_j | X, Y))$$

(this is the "tgf" weighting function described in [14]). If the two columns each contain one or more characters, then the numerator ("match probabilities") and denominator ("gap probabilities") become sums over the characters pairs which are newly-aligned after aligning the two columns (see Theorem 2.2).

As a greedy algorithm, sequence annealing first aligns the columns which will give the largest (incremental) increase in the expected accuracy of the alignment (strictly speaking, the first columns

aligned have maximum weight; see Section 2.4 for details). FSA therefore internally creates a series of alignments with increasing expected sensitivity and decreasing expected positive predictive value or precision. The final alignment, which has the highest expected accuracy, is the one most relevant in the majority of applications, but the user can view all intermediate alignments produced by the sequence annealing process with the GUI (see Section 2.3).

In the current implementation in FSA, sequence annealing is a steepest-ascent hill-climbing algorithm for maximizing the expected accuracy criterion (Section 2.4), and as such will find a local (rather than global) optimum of the objective function. Most common variants of the multiple alignment problem have been proved to be NP-hard [31], strongly suggesting that the expected accuracy approach described here is NP-hard as well. However, our experiments with simulated annealing strategies, which have given minimal performance increases, suggest that the expected-accuracy alignment landscape is relatively smooth at the large scale, thereby permitting a greedy approach to be very effective. We have additionally observed that the majority of the steps taken by the greedy algorithm (the first 80-90% on typical datasets) involve aligning characters whose homology is completely certain under FSA's model, further suggesting that a greedy strategy is effective on typical accuracy landscapes. Sequence annealing (the greedy algorithm) followed by iterative refinement appears to be the best strategy to use.

Sequence annealing begins by constructing a heap of all possible alignments of (edges between) single characters using this weighting function. At each step of the algorithm, a single candidate edge is popped from the heap. If the columns joined by the edge have been modified since the weight was last calculated, then the weight is re-calculated as above. If the new weight is no longer greater than the weight of the next edge on the heap (if the heap ordering is incorrect), then the edge is re-inserted into the heap and the next edge is popped. If the edge does have the highest weight (if the heap ordering is correct), then the edge is added to the alignment if it is not inconsistent with edges added previously. Treating a multiple alignment as a DAG, sequence annealing quickly performs this consistency-checking of candidate edges using the Pearce-Kelly algorithm [32] for

online topological ordering of directed graphs. While we have phrased the algorithm in the edge-insertion framework, the implementation uses a node-contraction framework, wherein two nodes are merged when they are connected by an accepted edge.

The sequence annealing algorithm described above and introduced in `AMAP` has unfavorable computational complexity of $O(N^4)$ for aligning N sequences. Consider the simple case of building a complete alignment of N sequences, for which there will be $O(N^2)$ edges if we perform an all-pairs comparison. Because a complete alignment of N sequences requires only $O(N)$ edges to assemble, the majority of these candidate edges must be eventually rejected due to consistency constraints. Most of these will not be examined (popped from the heap) until the alignment is largely complete. This introduces an enormous computational cost: re-weighting a candidate edge between two completely-assembled columns costs $O(N^2)$ due to the sum-of-pairs operation required, and so in the worst case, re-weighting of these candidate edges will therefore cost $O(N^2)$ for each edge. Because the number of inconsistent edges grows as $O(N^2)$, whereas the number of consistent edges grows only as $O(N)$, the sequence annealing algorithm of [14] has a worse-case complexity of $O(N^4)$, making it impractical for many sequences.

The sequence annealing algorithm therefore has two costly parts, consistency-checking with the Pearce-Kelly algorithm and edge re-weighting. While the analysis above may suggest that it is best to always perform consistency-checking before edge re-weighting, this is only clearly true once the alignment is largely assembled. There are two phases in the alignment construction, an initial phase, in which most candidate edges are accepted and the basic structure of the alignment is assembled, and a completion phase, in which most candidate edges are rejected and the structure of the alignment changes little. In the initial phase it is best to perform re-weighting first, since re-weighting is cheap on a sparse graph and most edges will pass the consistency checks; in the completion phase it is best to perform consistency-checking first, since re-weighting is expensive and most edges will fail consistency checks.

Finding an optimal solution to this trade-off is an open problem. `FSA` uses a hybrid approach,

wherein it performs several fast partial consistency checks first, such as checking that candidate edges do not map to a single column of the current alignment, re-weights edges which pass these checks, and only then performs a "full" consistency check with a search of the DAG. `FSA` avoids performing repeated searches between identical columns by maintaining lookup tables of consistent and inconsistent edges which are iteratively constructed as graph searches are performed for consistency-checking of new candidate edges. It additionally amortizes re-weighting calculations to minimize their cost and keeps track of re-weighted candidate edges to avoid the cost of re-weighting "duplicate" candidate edges which join columns of the current alignment for which we have already seen and re-weighted a candidate edge. The resulting new sequence annealing algorithm avoids the bad scaling described above, giving it an approximate complexity of $O(N^2)$ for typical problems, with the true complexity depending on the structure of the graph.

By Theorem 2.2, the sequence annealing algorithm is a true steepest ascent algorithm in the weighting function [14].

Theorem 2.2 *The weight of a candidate edge $w\left(col_1, col_2\right)$ between columns can only decrease as more characters are aligned to the columns. Sequence annealing is therefore a true steepest-ascent algorithm in the weighting function given by*

$$w\left(col_1, col_2\right) = 2 \cdot \sum_{x_i\,:\,X \in col_1} \sum_{y_j\,:\,Y \in col_2} \mathbb{P}(x_i \sim y_j | X, Y) \Bigg/ \left(\sum_{x_i\,:\,X \in col_1} \sum_{y_j\,:\,Y \in col_2} \left[\mathbb{P}(x_i \sim - | X, Y) + \mathbb{P}(- \sim y_j | X, Y)\right] \right)$$

between two columns col_1, col_2. This follows from this fact:

If $l_1, \ldots, l_k$ and $m_1, \ldots, m_k$ are positive numbers, then

$$\max_k \frac{l_k}{m_k} \geq \frac{\sum_k l_k}{\sum_k m_k}.$$

(This theorem and its proof are from [14].)

Proof Let $C = \frac{\sum_k l_k}{\sum_k m_k}$ and assume the contrary. Then we have

$$\max_k \frac{l_k}{m_k} < C\,,$$

from which it follows that $l_k < C \cdot m_k \,\forall\, k$. However, then $\frac{\sum_k l_k}{\sum_k m_k} < C$, which is a contradiction. ∎

We have observed that in practice, FSA's sequence annealing algorithm is approximately as fast as inference. This is supported by recent theoretical analysis: The average running time of the Pearce-Kelly algorithm on a complete random graph with n nodes is $O\left(n^2 \log^2 n\right)$ [33]. If we perform distance-based alignment of N sequences of length L, then this gives a runtime of $O\left(N^2 \cdot L^2 \cdot \log^2(N \cdot L)\right)$, so in the "worst average case" for sequence annealing in which we have a near-complete graph (sequence graphs are generically sparse), sequence annealing costs only logarithmically more than inference. FSA's alignment speedup techniques can further reduce this. While this average-case analysis does not take into account the cost of re-weighting candidate edges, FSA's revised annealing algorithm reduces the cost of re-weighting to approximately $O(N^2)$, allowing the bound to hold.

Theoretical justification of distance-based alignment. Our distance-based approach to the multiple alignment problem can be viewed as an approximation to more complex models of multiple alignments. Analogously to the case of Neighbor-Joining, which uses only pairwise comparisons to approximate full models of likelihoods on trees, distance-based alignment uses only pairwise inference of alignment probabilities to approximate complete models of sequences evolving on trees.

Consider finding the optimal multiple alignment of sequences $X_1, ..., X_N$ related by a phylogenetic tree $\mathcal{T}$, so that our statistical model defines probabilities $\mathbb{P}\left(\mathcal{A}|X_1, ..., X_N, \mathcal{T}\right)$ over multiple alignments $\mathcal{A}$ given a tree $\mathcal{T}$. The generalization of Equation 2 to this case is straightforward:

$$\mathcal{A}_{optimal} = \text{argmin}_{\mathcal{A}^*}\, \mathbb{E}\left[d\left(\mathcal{A}^*, \mathcal{A}|\mathcal{T}\right)\right]_{\mathbb{P}(\mathcal{A}|X_1,...,X_N,\mathcal{T})}\,.$$

We can rewrite the objective function as

$$\begin{aligned}\mathbb{E}\left[d\left(\mathcal{A}^*,\mathcal{A}|\,\mathcal{T}\right)\right]_{\mathbb{P}(\mathcal{A}|X_1,\ldots,X_N,\mathcal{T})} &= \sum_{\mathcal{A}} \mathbb{P}\left(\mathcal{A}|X_1,\ldots X_N,\mathcal{T}\right) d\left(\mathcal{A}^*,\mathcal{A}|\mathcal{T}\right) \\ &= \frac{1}{\binom{N}{2}} \sum_{i,j} \sum_{\mathcal{A}_{ij}} \sum_{\mathcal{A}|\mathcal{A}_{ij}} \mathbb{P}\left(\mathcal{A}|X_1,\ldots,X_N,\mathcal{T}\right) d\left(\mathcal{A}^*,\mathcal{A}|\mathcal{T}\right) ,\end{aligned} \tag{5}$$

where we have introduced a sum over pairwise alignments $\mathcal{A}_{ij}$ of sequences X_i and X_j.

Motivated by Equation 5, we derive a distance-based alignment method from this explicit model of alignments on a tree by making two restrictions. We define the distance $d\left(\mathcal{A}^*,\mathcal{A}|\mathcal{T}\right)$ between two multiple alignments of sequences related by a tree $\mathcal{T}$ as a weighted sum of pairwise distances,

$$d\left(\mathcal{A}^*,\mathcal{A}|\mathcal{T}\right) = \sum_{i,j} w_{ij}(\mathcal{T})\, d\left(\mathcal{A}^*_{ij},\mathcal{A}_{ij}\right) , \tag{6}$$

and use a pairwise approximation to the full probabilistic model,

$$\sum_{\mathcal{A}|\mathcal{A}_{ij}} \mathbb{P}\left(\mathcal{A}|X_1,\ldots,X_N,\mathcal{T}\right) = \mathbb{P}\left(\mathcal{A}_{ij}|X_i,X_j\right) . \tag{7}$$

The first restriction, on distances between multiple alignments, is related to the approximations made by distance-based phylogenetic reconstruction methods such as Neighbor-Joining [34] and BIONJ [35]. In this paper we use weights $w_{ij}(\mathcal{T}) = 1 \,\forall\, i,j,\mathcal{T}$, which corresponds to a phylogeny-free approach to alignment. The second restriction, on computation of the likelihood function, is a well-studied: The Pair HMM used by `FSA` is an approximation (in a precise sense; see, e.g., [36]) of pairwise stochastic models of substitutions and indels such as the Thorne-Kishino-Felsenstein '91 model ("TKF91") [37] and its extensions. Substituting Equations 6 and 7 into Equation 5, we obtain the sum-of-pairs objective function use by `FSA` (Equation 2),

$$\mathbb{E}\left[d\left(\mathcal{A}^*,\mathcal{A}|\,\mathcal{T}\right)\right]_{\mathbb{P}(\mathcal{A}|X_1,\ldots,X_N,\mathcal{T})} = \sum_{i,j} w_{ij}(\mathcal{T}) \sum_{\mathcal{A}_{ij}} d\left(\mathcal{A}^*_{ij},\mathcal{A}_{ij}\right) \mathbb{P}\left(\mathcal{A}_{ij}|X_i,X_j\right) . \tag{8}$$

We do not have theoretical results on the degree of approximation involved in Equations 6 and 7, and so at present these approximations remain well-motivated heuristics. However, much as Neighbor-Joining was long seen as a murky heuristic before being revealed as a greedy optimization of a principled objective function [38], we believe that the approximations involved in our distance-based method stand on firm theoretical ground.

We note that the presentation above can easily be further generalized to include, for example, sequences which have undergone recombination.

3 Results

We benchmarked `FSA` against databases of multiple alignments compiled from reference structural alignments, including protein databases (BAliBASE 3 [15] and SABmark 1.65 [39]), small RNA databases (BRAliBase 2.1 [40]) and large RNA databases (`Consan` mix80 [41]).

Alignment programs are commonly used to detect homology among input sequences. As reported in [42], we conducted a series of false-positive experiments to test whether commonly-used alignment programs can reliably identify homologous and non-homologous sequence. Surprisingly, we found that for most alignment programs, aligned sequences are not necessarily homologous, indicating that biologists should use caution when interpreting the multiple alignments produced by many commonly-used tools.

We additionally performed a simple test of the consistency of common programs when aligning coding sequence: We aligned 1,502 genes orthologous across seven species of yeast in both nucleotide and protein space and compared the resulting alignments. Many programs gave surprisingly discordant results, indicating that at least one of these two alignments produced by commonly-used programs is incorrect.

3.1 Protein sequence

Table 1 shows benchmarks of FSA and other alignment programs, including AMAP [14], ClustalW [1, 2], DIALIGN [28], MAFFT [43], MUMMALS [44], MUSCLE [45], Probalign [46], ProbCons [47], T-Coffee [48], and SeqAn::T-Coffee [49], against the BAliBASE 3 [15] and SABmark 1.65 databases [39]. FSA in maximum-sensitivity mode had accuracy similar to those of the better-performing programs on BAliBASE 3 and had accuracy superior to that of any other program on SABmark 1.65. FSA had higher positive predictive values (in both default and maximum-sensitivity modes) than any other tested program on all datasets. Remarkably, FSA was the only tested program which achieved a mis-alignment rate $< 50\%$ on the standard SABmark 1.65 datasets. All other programs produced more incorrect than correct homology statements.

In order to test the robustness of alignment programs to incomplete homology, we modified the BAliBASE 3 database such that every alignment included a single false-positive, an unrelated (random) sequence. This is a realistic setup for biologists who might want to decide whether a sequence is orthologous to a particular protein family. With the exception of FSA, the tested alignment programs suffered a false-positive rate increased by over 25% on this BAliBASE 3 + fp dataset, indicating that they aligned the random sequence with the homologous set. In contrast, FSA left the random sequence unaligned and had an essentially-unchanged false-positive rate.

3.2 RNA sequence

Table 2 shows benchmarks of FSA and the other tested alignment programs against the BRAliBase 2.1 [40] and Consan mix80 [41] databases. FSA outperformed all other programs on both datasets.

BRAliBase 2.1 was assembled from the RFAM [50] RNA database and consists of small RNAs (average length of ~ 150 nt). FSA gave improved performance even on this high-identity dataset where most programs did relatively well.

The Consan mix80 dataset of alignments of Small and Large Subunit ribosomal RNAs, assembled from the European Ribosomal RNA database [51], was created for training RNA structural

alignment programs and provides a test of alignment programs on difficult, large-scale alignments. The four alignments contain from 107 to 254 sequences, each 1-4 kilobases in length, with average percentage identity less than $< 50\%$. Two tested alignment programs, `ProbConsRNA` [52] and `SeqAn::T-Coffee`, were unable to align these large datasets. This dataset demonstrates that `FSA`'s alignment speedup options (Section 2.3), including performing inference only on a subset of all possible pairs (--fast) and anchoring alignments instead of using the full dynamic programming matrix (--anchored), are effective heuristics for large datasets.

3.3 DNA sequence

3.4 Unrelated sequence

As reported in [42], in order to further test the appropriateness of using popular alignment programs to detect homology between sequences, we conducted a large-scale random-sequence experiment. We generated datasets of random sequences to simulate unrelated protein and DNA sequences. The results, show in Table 3, clearly demonstrate that while for most alignment programs, aligned sequences are not necessarily homologous, `FSA` leaves random sequences largely unaligned.

3.5 Concordance between amino acid and nucleotide alignments

Biologists commonly align coding regions in both amino acid and nucleotide space, but there have been few studies of the effectiveness of alignment programs across the two regimes. We tested the consistency of alignment programs on coding sequence by aligning all 1,502 genes in *Saccharomyces cerevisiae* identified as having orthologs in the six related yeast species *S. paradoxus*, *S. mikatae*, *S. kudriavzevii*, *S. bayanus*, *S. castellii*, and *S. kluyveri* ([53]; this dataset was also analyzed in [5]). As shown in Table 4, alignments produced by `FSA` had higher concordance (0.943) than those produced by any other program. If a program produces alignments with low concordance between amino acid and nucleotide alignments, then at least one of the alignments must be incorrect (and quite possibly both are).

This simple test additionally indicates the effectiveness and robustness of FSA's query-specific learning. While very different learning procedures are used for amino acid and nucleotide sequence (Section 2.3), the concordant alignments inferred by FSA indicate that our results are robust with respect to the details of the learning procedure.

3.6 Ablation analysis of FSA's components

We conducted an ablation analysis of FSA's components to test the effectiveness of each component and ensure that they all contributed to the accuracy of the program. As indicated by the results in Tables 5-7, each optional component of FSA contributes to its accuracy.

The importance of each component depends strongly upon the alignment problem. As indicated by the mix80 benchmark (Table 6), iterative refinement is important for aligning many sequences and less so for small alignment problems. Query-specific learning is important for maintaining FSA's robustness to non-homologous sequence: While FSA aligned only 4% of random protein sequences in default mode, when run without learning it aligned 13% (Table 7), similar to the 14% aligned by AMAP (Table 3). The --fast heuristic for reducing the number of sequence pairs used to construct an alignment results in little loss of accuracy, at least on the benchmarks used on this paper. Similarly, the anchor annealing procedure appears to be an effective heuristic for aligning long sequences. Anchoring causes negligible loss of accuracy on the mix80 dataset.

3.7 Runtimes and parallelization

Biologists commonly perform alignments of hundreds or thousands of 16S ribosomal DNA sequences in order to elucidate evolutionary relationships and build phylogenetic trees. We performed alignments of $> 1,000$ prokaryotic 16S sequences to demonstrate the effectiveness of FSA's parallelization mode. As shown in Table 8 and Table 9, parallelizing the pairwise comparisons dramatically reduces runtime. When running in --fast mode on a small cluster with 10 processors, FSA can align 500 16S sequences in under 45 minutes.

4 Discussion

In the Introduction we highlighted four design criteria which we emphasized in the development of FSA. The first goal was to find alignments with high expected accuracy, where an accurate alignment has minimal distance to the truth. This objective function is markedly different from both the maximum-likelihood approaches used by programs such as ClustalW and MUSCLE and the maximum expected sensitivity approaches used by programs such as ProbCons. Note that while the objective function used in ProbCons is called "maximum expected accuracy" in the paper [47], it is actually a maximum expected sensitivity objective function, where there is no penalty for over-aligning sequence (c.f., the results shown in Table 3). Their objective function can be recovered as a special case of our approach by ignoring the gap probabilities in FSA's objective function; see Section 2.4 for details. FSA's explicit search for the most accurate, rather than most likely or most sensitive, alignment is what allows FSA to so dramatically outperform other programs on tests on unrelated sequence (Table 3).

The second goal was to create alignments which are robust to evolutionary distances and different functional constraints on patterns of molecular evolution. FSA's unsupervised query-specific learning for parameter selection frees the user from unknown systematic biases implicitly introduced by using an alignment program whose parameters were trained on a training set whose statistics may not reflect those of the sequences to be aligned. For example, it is traditionally challenging to align sequences with unusual base composition, but FSA can easily handle such alignments by automatically learning appropriate parameters. As indicated by our ablation analysis, query-specific learning further increases FSA's robustness to non-homologous sequences beyond that offered by the minimum-distance objective function alone.

We believe that FSA's unsupervised query-specific learning is the first time a multiple alignment program has been capable of dynamically learning a complete parametrization, wherein parameters can vary for each pair of sequences to be compared, on the fly. This learning method is related to the "pre-training" option in ProbCons, which permits users to learn different models for families

of homologous sequences, but does not permit parametrizations to vary between sequence pairs. We also note that the MORPH program for pairwise alignment of sequences with *cis*-regulatory modules learns model transitions parameters from data [54]. While supervised training on curated data can give superior performance on test sets which are statistically-similar to the training data, the practical alignment problems encountered everyday by biologists do not fit into this rigid problem setup. Query-specific learning consistently gives reasonable performance.

The third and fourth goals, to develop a single, modular program which can address all typical alignment problems encountered by biologists, are naturally achieved within FSA's architecture. While almost all alignment programs are designed to either align many short sequences or a few long sequences, we have demonstrated that it is feasible to create a single program which can address both situations. This is made practical by FSA's modular nature, where the statistical model for pairwise comparisons, the anchoring scheme for finding homology between long sequences and the sequence annealing procedure are entirely separate and can be individually modified and improved. For example, the parallelization of FSA was designed and developed with minimal changes to the rest of FSA's code base. Similarly, while FSA's basic anchoring relies only on exact matches from MUMmer, the anchoring scheme was easily extended to incorporate different information from exonerate [22] and Mercator [23]. In fact, this flexibility permits FSA to incorporate almost any sources of potential homology information, from predicted transcription factor binding sites to entire gene models. One natural extension of FSA's approach is to models of RNA alignment which take structure into account. The program Stemloc-AMA [55] uses a model of the pairwise evolution of RNA secondary structure in conjunction with the sequence annealing algorithm to create accurate multiple alignments of structured RNAs. By using Stemloc-AMA's probabilistic model rather than a Pair HMM and taking advantage of techniques such as query-specific learning, FSA could sum over possible pairwise structural alignments in order to get better estimates of posterior probabilities of character alignment.

FSA is a statistical alignment program insofar as it uses an explicit statistical model of align-

ments and a probabilistic objective function for optimization, but as discussed in Section 2.4, it also is a distance-based approximation to the "phylogenetic alignment" models of alignments on trees [56, 57, 58, 8, 9, 10, 11]. While traditional phylogenetic alignment algorithms are currently too computationally-expensive to use on datasets of more than a few sequences, FSA's distance-based method allows biologists to use the sophisticated tools of statistical alignment algorithms on practical problems. Furthermore, while we have not addressed the phylogenetic aspects of FSA in detail in this paper, our methods can be adapted to incorporate a complete phylogenetic model. However, we believe that FSA's current approach, which is agnostic to phylogeny, offers many practical advantages for common genomics analyses. For example, because FSA uses a sum-of-pairs objective function, there is no guide tree to potentially bias downstream phylogenetic reconstructions based on the alignment. Similarly, while most whole-genome alignment programs require a species tree to perform the alignment, our phylogeny-free approach will be avoid potential biases introduced by using a single species tree to align regions which may have undergone recombination.

Acknowledgments

FSA borrows heavily from previous work, both in its code base and its intellectual foundations.

Ideas. The distance-based approach to multiple alignment was proposed in [13, 14]. This included the idea of modifying the accuracy criterion suggested in Durbin [59] and Holmes and Durbin [60] to include gaps and the demonstration that the resulting modified expected accuracy could be used to control the expected modeler and developer scores. Furthermore, [13, 14] introduced the sequence annealing approach to building multiple alignments, via the description of alignments using partially ordered sets [28, 25, 26]. The graph-based approach to alignment was formalized by [29] and these results were used in the DIALIGN program [30].

The query-specific learning method for re-estimating alignment parameters on the fly was inspired by [16] and conversations with Joseph Bradley about query-specific structure learning of graphical models.

The iterative refinement technique is based on ideas in [61].

The FSA algorithm was parallelized using a modification of the approach in MW [62].

The coloring in the GUI according to posterior probabilities of alignment is inspired by the AU viewer of BAli-Phy [9].

Software. The sequence annealing implementation in FSA is based on Ariel Schwartz's AMAP program [14], which implements the Pearce-Kelly algorithm [32].

FSA's query-specific learning uses code created with Gerton Lunter's HMMoC compiler for HMMs [16]. The "aligner" example distributed with HMMoC, which implements a learning procedure for gap parameters, was an inspiration for FSA's learning strategies. FSA's banding code is taken directly from the "aligner" example.

FSA includes sequence and alignment representation code from the dart library. Several Perl packages distributed with FSA are derived from packages of the same name in dart.

Author contributions

RKB led the development of `FSA`, wrote most of the code base, and developed the query-specific learning method.

RKB, CD, and LP redesigned the sequence annealing algorithm, constituted the core development team, and together managed the project.

RKB, JD, and CD conducted the benchmarks.

AR developed the GUI using a preliminary version developed by MS.

SJ developed the iterative refinement technique.

JD and CD developed the parellelization and database modes.

IH provided advice on the `dart` library, including its algorithms, programming and software components.

RKB and LP formulated the theoretical justification of distance-based alignment. LP created the `FSA` webserver.

Tables

Program	BAliBASE 3 (Acc / Sn / PPV)	BAliBASE 3 + fp (Acc / Sn / PPV)	SABmark 1.65 (Acc / Sn / PPV)
AMAP	0.70 / 0.62 / 0.83	0.73 / 0.61 / 0.80	0.57 / 0.43 / 0.46
ClustalW	0.66 / 0.63 / 0.62	0.59 / 0.63 / 0.53	0.38 / 0.44 / 0.30
DIALIGN	0.68 / 0.63 / 0.68	0.68 / 0.62 / 0.63	0.48 / 0.41 / 0.34
FSA	0.71 / 0.62 / 0.85	**0.75** / 0.62 / 0.84	**0.59** / 0.38 / 0.52
FSA (--maxsn)	0.73 / 0.68 / 0.76	0.74 / 0.68 / 0.72	0.52 / 0.45 / 0.39
MAFFT	0.74 / 0.71 / 0.71	0.68 / 0.71 / 0.61	0.44 / 0.49 / 0.35
MUMMALS	0.74 / 0.70 / 0.73	0.69 / 0.70 / 0.64	0.49 / 0.52 / 0.38
MUSCLE	0.70 / 0.67 / 0.66	0.63 / 0.66 / 0.57	0.40 / 0.46 / 0.32
Probalign	**0.76** / 0.72 / 0.73	0.71 / 0.71 / 0.65	0.49 / 0.50 / 0.37
ProbCons	0.74 / 0.70 / 0.72	0.69 / 0.70 / 0.64	0.47 / 0.50 / 0.37
T-Coffee	0.72 / 0.67 / 0.71	0.67 / 0.67 / 0.63	0.45 / 0.46 / 0.35
SeqAn::T-Coffee	0.73 / 0.69 / 0.70	0.67 / 0.69 / 0.61	0.43 / 0.47 / 0.34

Table 1: Comparisons of the accuracies (Acc), sensitivities (Sn) and positive predictive values (PPV) of FSA and other alignment methods on the BAliBASE 3 [15] and SABmark 1.65 [39] databases. FSA in default mode has superior accuracy on the BAliBASE 3 + fp and SABmark 1.65 datasets and comparable performance in maximum-sensitivity mode on the BAliBASE 3 dataset to the most accurate method, MAFFT. Note that FSA has significantly higher positive predictive values than any other program on all datasets. The BAliBASE 3 + fp dataset was designed to test the robustness of alignment programs to incomplete homology. Traditional alignment programs, designed to maximize sensitivity, suffer greatly-increased mis-alignment when even a single non-homologous sequence is introduced; in contrast, FSA is robust to the non-homologous sequence and has an unchanged positive predictive value. Remarkably, FSA was the only tested program with a mis-alignment rate of $< 50\%$ on the SABmark 1.65 dataset; the majority of the homology statements made by other programs were incorrect. The BAliBASE 3 dataset consisted of full-length sequences in all reference sets RV11, RV12, RV20, RV30, RV40 and RV50; we created the BAliBASE 3 + fp dataset from the same reference sets by adding a single false-positive, a random sequence, to each alignment. The SABmark 1.65 dataset consisted of the Twilight Zone and Superfamilies datasets.

Program	BRAliBase 2.1 (Acc / Sn / PPV)	Consan mix80 (Acc / Sn / PPV)
ClustalW	0.85 / 0.86 / 0.86	0.65 / 0.65 / 0.68
DIALIGN	0.82 / 0.83 / 0.85	0.76 / 0.75 / 0.82
FSA	0.90 / 0.91 / 0.94	0.77 / 0.74 / 0.92
FSA (--maxsn)	**0.91** / 0.92 / 0.92	**0.78** / 0.78 / 0.86
MAFFT	0.90 / 0.91 / 0.91	0.77 / 0.78 / 0.77
MUSCLE	0.90 / 0.91 / 0.90	0.74 / 0.76 / 0.74
ProbConsRNA	0.91 / 0.92 / 0.92	(failed to align)
T-Coffee	0.81 / 0.82 / 0.84	0.38 / 0.33 / 0.40
SeqAn::T-Coffee	0.89 / 0.90 / 0.90	(failed to align)

Table 2: Comparisons of the accuracies (Acc), sensitivities (Sn) and positive predictive values (PPV) of FSA and other alignment methods on the the BRAliBase 2.1 dataset of small RNAs [40] and the Consan mix80 dataset of Small and Large Subunit ribosomal RNAs [41]. FSA dominated all other tested programs on both benchmarks, producing alignments with both better sensitivity and specificity. The BRAliBase 2.1 dataset consisted of all alignments with 15 sequences (the largest alignments). The mix80 dataset provided difficult alignment problems: The four alignments each contain from 107 to 254 sequences of approximately 1-4 kilobases in length, with average percentage identity less than $< 50\%$. Two program, ProbConsRNA and SeqAn::T-Coffee, were incapable of aligning these large datasets. When run in --fast mode, FSA considers only a subset ($\sim 20\%$ in this case) of all sequence pairs. Note that because the mix80 dataset consists of long sequences, FSA automatically uses anchoring for speed. FSA does not use anchoring on the short sequences of BRAliBase 2.1.

Program	Protein	DNA
AMAP	14%	n/a
ClustalW	97%	95%
DIALIGN	24%	17%
FSA	**4**%	**5**%
FSA (--maxsn)	21%	17%
MAFFT	83%	93%
MUMMALS	63%	n/a
MUSCLE	89%	80%
Probalign	44%	n/a
ProbCons	51%	77%
T-Coffee	63%	75%
SeqAn::T-Coffee	74%	78%

Table 3: The large-scale random sequence tests conducted in [42] indicate that for most alignment programs, aligned sequences are not necessarily homologous. Even when run in maximum-sensitivity mode (--maxsn), FSA aligned only a small fraction of the random sequence. We generated 50 datasets, each with 10 random sequences, and ran all programs with default parameters. Protein sequences were 300 aa in length and DNA sequences were 1,000 nt in length. Results reported for ProbCons on DNA sequences were obtained with ProbConsRNA. These results are taken from [42].

Program	Alignment similarity (average)
ClustalW	0.914
DIALIGN	0.912
FSA (--noindel2)	**0.943**
FSA (--noindel2 --noanchored)	**0.945**
MAFFT	0.932
MUSCLE	0.915
ProbCons	0.902
T-Coffee	0.897
SeqAn::T-Coffee	0.905

Table 4: We assessed the concordance between alignments obtained in nucleotide and amino acid space by aligning all 1,502 genes in *Saccharomyces cerevisiae* which have orthologs in the six related yeast species *S. paradoxus*, *S. mikatae*, *S. kudriavzevii*, *S. bayanus*, *S. castellii*, and *S. kluyveri* (this dataset was analyzed in [5]). Alignments produced by FSA, in both anchored and unanchored (--noanchored) modes, had the highest concordance. Alignment similarity between alignments computed in nucleotide and amino acid space was assessed by converting the amino acid alignment to the implied nucleotide alignment and computing the alignment similarity (the proportion of identical homology statements made by the alignments; see Section 2.4 for details) between them. Alignments for ProbCons on nucleotide sequences were obtained with ProbConsRNA.

FSA options	BAliBASE 3 (Acc / Sn / PPV)	BAliBASE 3 + fp (Acc / Sn / PPV)	SABmark 1.65 (Acc / Sn / PPV)
(default)	0.71 / 0.62 / 0.85	**0.75** / 0.62 / 0.84	**0.59** / 0.38 / 0.52
--fast	0.70 / 0.61 / 0.85	0.74 / 0.62 / 0.84	0.59 / 0.37 / 0.52
--nolearn	0.72 / 0.65 / 0.81	0.75 / 0.65 / 0.79	0.56 / 0.44 / 0.44
--refinement 0	0.70 / 0.61 / 0.85	0.74 / 0.61 / 0.84	0.59 / 0.37 / 0.52
--noindel2	0.70 / 0.61 / 0.85	0.74 / 0.60 / 0.84	0.59 / 0.38 / 0.52
--maxsn	**0.73** / 0.68 / 0.76	0.74 / 0.68 / 0.72	0.52 / 0.45 / 0.39
--fast --maxsn	0.73 / 0.67 / 0.76	0.73 / 0.67 / 0.71	0.52 / 0.44 / 0.39
--nolearn --maxsn	0.73 / 0.68 / 0.74	0.70 / 0.68 / 0.67	0.49 / 0.47 / 0.37
--refinement 0 --maxsn	0.72 / 0.66 / 0.78	0.73 / 0.66 / 0.73	0.53 / 0.43 / 0.39
--noindel2 --maxsn	0.73 / 0.68 / 0.76	0.72 / 0.68 / 0.70	0.51 / 0.45 / 0.39

Table 5: Ablation analysis of FSA on the protein benchmarks of Table 1: Comparisons of the accuracies (Acc), sensitivities (Sn) and positive predictive values (PPV) of FSA with different components enabled or disabled. From top to bottom, FSA was run in default mode, --fast mode, with learning disabled, with iterative refinement disabled, and with 1 set (rather than 2 sets) of indel states; these options were then repeated for maximum-sensitivity mode (--maxsn). As made evident by the results (PPV) on the BAliBASE 3 + fp and SABmark 1.65 datasets, query-specific learning helps FSA to distinguish homologous and non-homologous sequences.

FSA options	BRAliBase 2.1 (Acc / Sn / PPV)	FSA options	Consan mix80 (Acc / Sn / PPV)
(default)	0.90 / 0.91 / 0.94		
--fast	0.90 / 0.91 / 0.94	--fast	0.77 / 0.74 / 0.92
--nolearn	0.91 / 0.92 / 0.93	--nolearn --fast	0.77 / 0.74 / 0.93
--refinement 0	0.90 / 0.91 / 0.93	--refinement 0 --fast	0.73 / 0.69 / 0.94
--noindel2	0.91 / 0.92 / 0.93	--noindel2 --fast	0.73 / 0.69 / 0.91
		--noanchored --fast	0.77 / 0.74 / 0.93
--maxsn	**0.91** / 0.92 / 0.92		
--fast --maxsn	0.91 / 0.92 / 0.92	--fast --maxsn	0.78 / 0.78 / 0.86
--nolearn --maxsn	0.91 / 0.92 / 0.92	--nolearn --fast --maxsn	0.78 / 0.78 / 0.85
--refinement 0 --maxsn	0.90 / 0.91 / 0.93	--refinement 0 --fast --maxsn	0.74 / 0.70 / 0.92
--noindel2 --maxsn	0.91 / 0.92 / 0.92	--noindel2 --fast --maxsn	0.74 / 0.73 / 0.84
		--noanchored --fast --maxsn	**0.79** / 0.79 / 0.85

Table 6: Ablation analysis of FSA on the RNA benchmarks of Table 2: Comparisons of the accuracies (Acc), sensitivities (Sn) and positive predictive values (PPV) of FSA with different components enabled or disabled. From top to bottom, FSA was run in default mode, --fast mode, with learning disabled, with iterative refinement disabled, with 1 set (rather than 2 sets) of indel states, and with anchored disabled; these options were then repeated for maximum-sensitivity mode (--maxsn). Iterative refinement is important for the large alignments of the mix80 dataset.

FSA options	Protein	DNA
(default)	4%	5%
--fast	4%	5%
--nolearn	13%	8%
--refinement 0	**3**%	**5**%
--noindel2	5%	10%
--maxsn	21%	17%
--fast --maxsn	22%	17%
--nolearn --maxsn	30%	16%
--refinement 0 --maxsn	19%	15%
--noindel2 --maxsn	27%	21%

Table 7: Ablation analysis of FSA on the unrelated sequence benchmarks of Table 3: Comparisons of the accuracies (Acc), sensitivities (Sn) and positive predictive values (PPV) of FSA with different components enabled or disabled. From top to bottom, FSA was run in default mode, --fast mode, with learning disabled, with iterative refinement disabled, and with 1 set (rather than 2 sets) of indel states; these options were then repeated for maximum-sensitivity mode (--maxsn). Query-specific learning helps to make FSA robust to non-homologous sequence.

FSA options	100	200	300	500	1,000 seqs
FSA	6,407 s	27,534 s	—	—	—
FSA --parallelize 10	819 s	5,713 s	22,113 s	—	—
FSA --fast	1,650 s	3,781 s	6,207 s	12,249 s	—
FSA --fast --parallelize 10	201 s	513 s	924 s	2,511 s	15,179 s

Table 8: Runtimes for FSA in regular, --fast and --parallelize modes when aligning 16S sequences with a 3-state HMM (--noindel2) and refinement disabled (--refinement 0). When running in --fast mode on a cluster with 10 processors (3.00 and 3.20 GHz; 8 GB of RAM), FSA can align 500 16S sequences in under 45 minutes. The parallelized FSA was run on a cluster managed by the Condor batch queueing system [63]; nodes were connected by a 100 Mbps Ethernet network. 16S sequences were obtained as a random slice of prokMSA from Greengenes [64] and had an average length of 1,450 nt.

	1	5	10	15	20 processors
100 seqs	1,650 s	365 s	214 s	135 s	105 s
200 seqs	3,781 s	889 s	506 s	385 s	355 s

Table 9: Runtimes for FSA in --fast --parallelize P mode as a function of the number of processors P in the computer cluster with a 3-state HMM (--noindel2) and refinement disabled (--refinement 0). Sequences and cluster specifications are same as for Table 8.

Figures

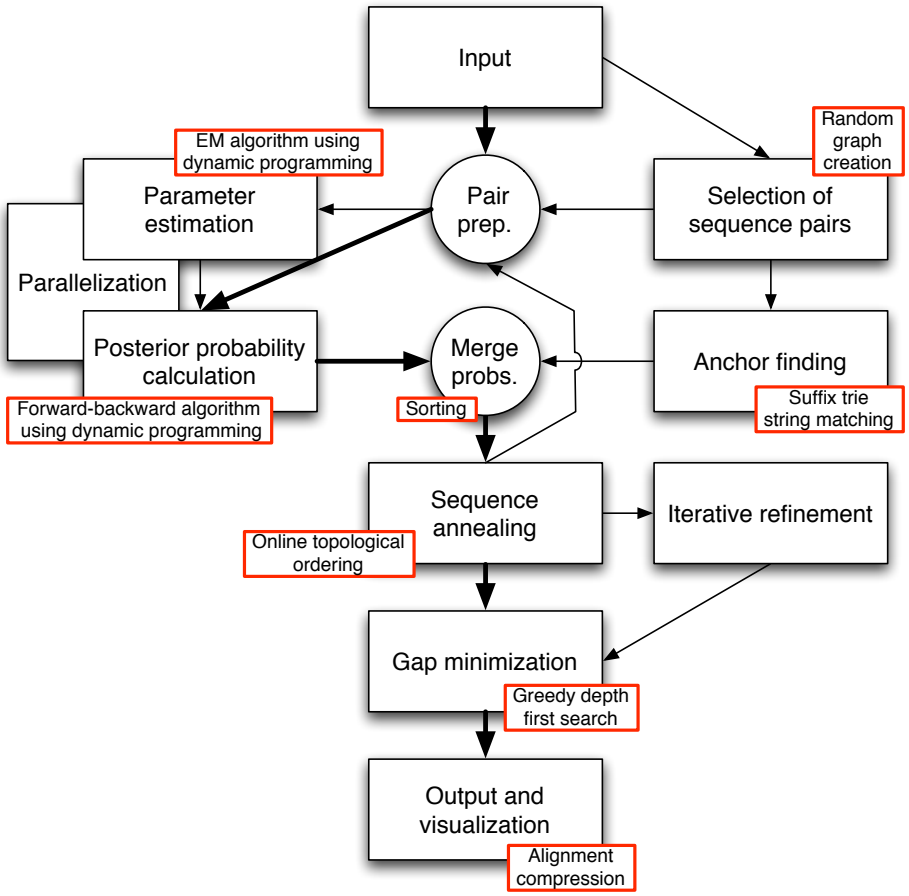


Figure 1: Overview of the components constituting the FSA alignment program. The algorithms that are used in each component are highlighted in the accompanying boxes. The bold arrows show the simplest mode of use for FSA, where posterior probabilities are calculated directly using default parameters for all pairs of sequences and the optional steps of anchor finding and iterative refinement are omitted.

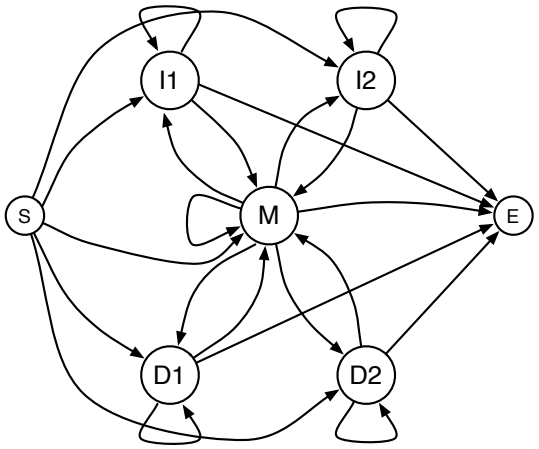


Figure 2: The default Pair HMM used by `FSA` has two sets of Insert (I) and Delete (D) states to generate a two-component geometric mixture distribution. `FSA` can optionally use a three-state HMM, which has only one set of Insert and Delete states. M is a Match state emitting aligned characters.

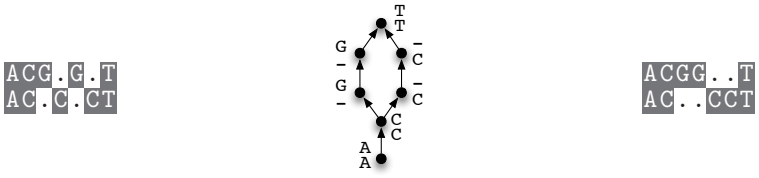


Figure 3: Two alignments (left and right) which make the same homology statements and therefore are both represented by the same POSET (center); see Section 2.4 for a discussion of this view of alignments as POSETs. The alignment on the right minimizes the number of gap-open events, and as such is appropriate for analyses such as inferring parsimonious indel frequencies across a clade. Alignments are displayed with TeXshade [65].

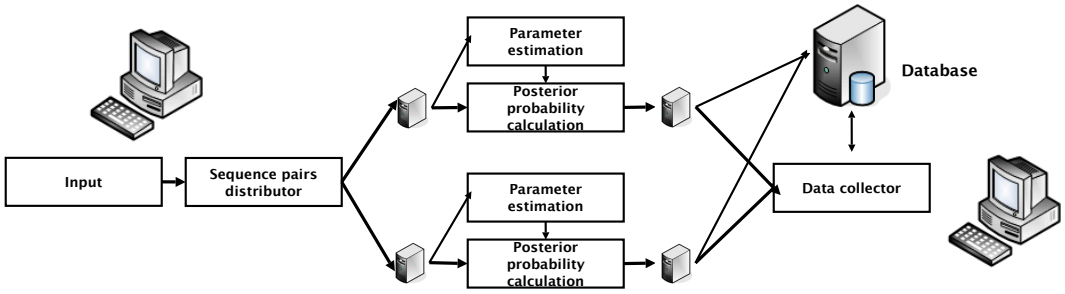
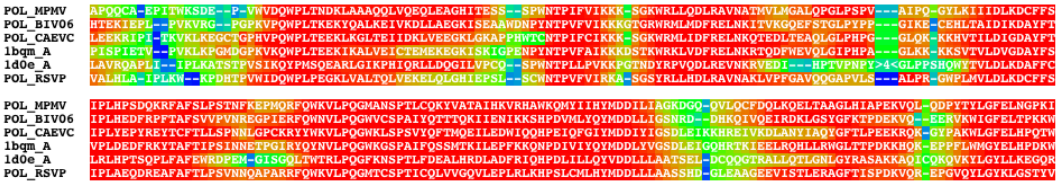


Figure 4: Schematic overview of FSA's parallelization strategy on a computer cluster. For large input sizes, a disk-based database may be used to store some of the primary data structures and reduce memory usage.

Alignment accuracy according to FSA's statistical model:



Alignment accuracy according to the reference structural alignment:

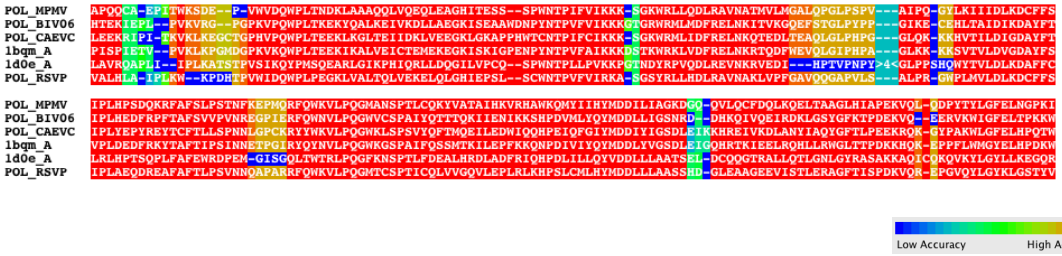


Figure 5: The Java GUI allows users to visualize the estimated alignment accuracy under FSA's statistical model. FSA's alignment is colored according the expected accuracy under FSA's statistical model (top) as well as according to the "true" accuracy (bottom) given from a comparison between FSA's alignment and the reference structural alignment. It is clear from inspection that accuracies estimated under FSA's statistical model correspond closely to the true accuracies. Sequences are from alignment BBS12030 in the RV12 dataset of BAliBASE 3 [15].

References

[1] Thompson JD, Higgins DG, Gibson TJ (1994) CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position specific gap penalties and weight matrix choice. Nucleic Acids Research 22:4673–4680.

[2] Larkin M, Blackshields G, Brown N, Chenna R, McGettigan P, et al. (2007) Clustal W and Clustal X version 2.0. Bioinformatics 23:2947–2948.

[3] Edgar R, Batzoglou S (2006) Multiple sequence alignment. Curr Opin Struct Biol 16:368–373.

[4] Lunter G, Rocco A, Mimouni N, Heger A, Caldeira A, et al. (2007) Uncertainty in homology inferences: assessing and improving genomic sequence alignment. Genome Research .

[5] Wong K, Suchard M, Huelsenbeck J (2008) Alignment uncertainty and genomic analysis. Science 319:473–476.

[6] Clamp M, Cuff J, Searle S, Barton G (2004) The Jalview Java alignment editor. Bioinformatics 20:426–427.

[7] Worobey M, Gemmel M, Teuwen D, Haselkorn T, Kunstman K, et al. (2008) Direct evidence of extensive diversity of HIV-1 in Kinshasa by 1960. Nature 455:661–664.

[8] Holmes I (2003) Using guide trees to construct multiple-sequence evolutionary HMMs. Bioinformatics 19 Suppl. 1:i147–157.

[9] Suchard MA, Redelings BD (2006) BAli-Phy: simultaneous Bayesian inference of alignment and phylogeny. Bioinformatics 22:2047–2048.

[10] Bradley RK, Holmes I (2007) Transducers: an emerging probabilistic framework for modeling indels on trees. Bioinformatics 23:3258–3262.

[11] Ádám Novák, Miklós I, Lyngsø R, Hein J StatAlign: An extendable software package for joint Bayesian estimation of alignments and evolutionary trees. Preprint.

[12] Miller W (2001) Comparison of genomic DNA sequences: solved and unsolved problems. Bioinformatics 17:391–397.

[13] Schwartz AS (2007) Posterior Decoding Methods for Optimization and Accuracy Control of Multiple Alignments. Ph.D. thesis, EECS Department, University of California, Berkeley. URL http://www.eecs.berkeley.edu/Pubs/TechRpts/2007/EECS-2007-39.html.

[14] Schwartz AS, Pachter L (2007) Multiple alignment by sequence annealing. Bioinformatics 23:e24–9.

[15] Thompson J, Koehl P, Ripp R, Poch O (2005) BAliBASE 3.0: latest developments of the multiple sequence alignment benchmark. Proteins 61:127–136.

[16] Lunter G (2007) HMMoC–a compiler for hidden Markov models. Bioinformatics 23:2485–2487.

[17] Kurtz S, Phillippy A, Delcher A, Smoot M, Shumway M, et al. (2004) Versatile and open software for comparing large genomes. Genome Biol 5:R12.

[18] Bray N, Pachter L (2004) MAVID: Constrained ancestral alignment of multiple sequences. Genome Research 14:693–699.

[19] Brudno M, Do C, Cooper G, Kim M, Davydov E, et al. (2003) LAGAN and Multi-LAGAN: efficient tools for large-scale multiple alignment of genomic DNA. Genome Res 13:721–731.

[20] Brudno M, Steinkamp R, Morgenstern B (2004) The CHAOS/DIALIGN WWW server for multiple alignment of genomic sequences. Nucleic Acids Res 32:W41–44.

[21] Paten B, Herrero J, Beal K, Fitzgerald S, Birney E (2008) Enredo and Pecan: Genome-wide mammalian consistency-based multiple alignment with paralogs. Genome Res 18:1814–1828.

[22] Slater G, Birney E (2005) Automated generation of heuristics for biological sequence comparison. BMC Bioinformatics 6:31.

[23] Dewey CN (2006) Whole-Genome Alignments and Polytopes for Comparative Genomics. Ph.D. thesis, EECS Department, University of California, Berkeley. URL http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-104.html.

[24] Kruskal CP, Weiss A (1985) Allocating independent subtasks on parallel processors. IEEE Trans Software Eng 11:1001–1016.

[25] Morgenstern B, Stoye J, Dress A (1999) Consistent equivalence relations: a set-theoretical framework for multiple sequence alignment. Technical report, University of Bielefeld, FSPM.

[26] Lee C, Grasso C, Sharlow M (2002) Multiple sequence alignment using partial order graphs. Bioinformatics 18:452–464.

[27] Eriksson N, Pachter L, Mitsuya Y, Rhee S, Wang C, et al. (2008) Viral population estimation using pyrosequencing. PLoS Comput Biol 4:e1000074.

[28] Morgenstern B, Dress A, Werner T (1996) Multiple DNA and protein sequence alignment based on segment-to-segment comparison. Proceedings of the National Academy of Sciences of the USA 93:12098–12103.

[29] Abdeddaïm S (1997) On incremental computation of transitive closure and greedy alignment. In: CPM. pp. 167–179.

[30] Abdeddaïm S, Morgenstern B (2001) Speeding up the dialign multiple alignment program by using the 'greedy alignment of biological sequences library' (gabios-lib). In: JOBIM '00: Selected papers from the First International Conference on Computational Biology, Biology, Informatics, and Mathematics. London, UK: Springer-Verlag, pp. 1–11.

[31] Elias I (2006) Settling the intractability of multiple alignment. J Comput Biol 13:1323–1339.

[32] Pearce DJ, Kelly PHJ (2007) A dynamic topological sort algorithm for directed acyclic graphs. ACM Journal of Experimental Algorithmics 11:1.7. doi: http://doi.acm.org/10.1145/1187436.1210590.

[33] Ajwani D, Friedrich T (2007) Average-case analysis of online topological ordering. In: Tokuyama T, editor, Proceedings of the 18th International Symposium on Algorithms and Computation, (ISAAC 2007). Springer, volume 4835 of *Lecture Notes in Computer Science*, pp. 464–475.

[34] Saitou N, Nei M (1987) The neighbor-joining method: a new method for reconstructing phylogenetic trees. Mol Biol Evol 4:406–425.

[35] Gascuel O (1997) BIONJ: an improved version of the NJ algorithm based on a simple model of sequence data. Mol Biol Evol 14:685–695.

[36] Knudsen B, Miyamoto M (2003) Sequence alignments and pair hidden Markov models using evolutionary history. J Mol Biol 333:453–460.

[37] Thorne JL, Kishino H, Felsenstein J (1991) An evolutionary model for maximum likelihood alignment of DNA sequences. Journal of Molecular Evolution 33:114–124.

[38] Gascuel O, Steel M (2006) Neighbor-joining revealed. Mol Biol Evol 23:1997–2000.

[39] Van Walle I, Lasters I, Wyns L (2005) Sabmark—a benchmark for sequence alignment that covers the entire known fold space. Bioinformatics (Oxford, England) 21:1267–1268.

[40] Wilm A, Mainz I, Steger G (2006) An enhanced RNA alignment benchmark for sequence alignment programs. Algorithms for molecular biology 1:19.

[41] Dowell RD, Eddy SR (2006) Efficient pairwise RNA structure prediction and alignment using sequence alignment constraints. BMC Bioinformatics 7:400.

[42] Bradley RK, Bray N, Dewey CN, Pachter L, Schwartz A (2008) Finding the trees in Darwin's forest. Submitted.

[43] Katoh K, Toh H (2008) Recent developments in the MAFFT multiple sequence alignment program. Brief Bioinformatics 9:286–298.

[44] Pei J, Grishin N (2006) MUMMALS: multiple sequence alignment improved by using hidden Markov models with local structural information. Nucleic Acids Res 34:4364–4374.

[45] Edgar RC (2004) Muscle: a multiple sequence alignment method with reduced time and space complexity. BMC bioinformatics 5:113.

[46] Roshan U, Livesay D (2006) Probalign: multiple sequence alignment using partition function posterior probabilities. Bioinformatics 22:2715–2721.

[47] Do CB, Mahabhashyam MSP, Brudno M, Batzoglou S (2005) ProbCons: Probabilistic consistency-based multiple sequence alignment. Genome Res 15:330–340. Comparative Study.

[48] Notredame C, Higgins D, Heringa J (2000) T-Coffee: A novel method for fast and accurate multiple sequence alignment. J Mol Biol 302:205–217.

[49] Rausch T, Emde A, Weese D, Döring A, Notredame C, et al. (2008) Segment-based multiple sequence alignment. Bioinformatics 24:i187–192.

[50] Griffiths-Jones S, Moxon S, Marshall M, Khanna A, Eddy SR, et al. (2005) Rfam: annotating non-coding RNAs in complete genomes. Nucleic Acids Research 33:D121–4.

[51] Wuyts J, Perrière G, Van De Peer Y (2004) The European ribosomal RNA database. Nucleic Acids Res 32:D101–103.

[52] Do CB, Mahabhashyam MSP, Brudno M, Batzoglou S ProbConsRNA. URL `http://probcons.stanford.edu/download.html`. Unpublished.

[53] Kellis M, Patterson N, Endrizzi M, Birren B, Lander ES (2003) Sequencing and comparison of yeast species to identify genes and regulatory elements. Nature 423:241–254. Comparative Study.

[54] Sinha S, He X (2007) MORPH: probabilistic alignment combined with hidden Markov models of cis-regulatory modules. PLoS Comput Biol 3:e216.

[55] Bradley R, Pachter L, Holmes I (2008) Specific alignment of structured RNA: stochastic grammars and sequence annealing. Bioinformatics 24:2677–2683.

[56] Hein J, Wiuf C, Knudsen B, Moller MB, Wibling G (2000) Statistical alignment: computational properties, homology testing and goodness-of-fit. Journal of Molecular Biology 302:265–279.

[57] Hein J (2001) An algorithm for statistical alignment of sequences related by a binary tree. In: Altman RB, Dunker AK, Hunter L, Lauderdale K, Klein TE, editors, Pacific Symposium on Biocomputing. Singapore: World Scientific, pp. 179–190.

[58] Lunter GA, Miklós I, Song YS, Hein J (2003) An efficient algorithm for statistical multiple alignment on arbitrary phylogenetic trees. Journal of Computational Biology 10:869–889.

[59] Durbin R, Eddy S, Krogh A, Mitchison G (1998) Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids. Cambridge, UK: Cambridge University Press.

[60] Holmes I, Durbin R (1998) Dynamic programming alignment accuracy. Journal of Computational Biology 5:493–504.

[61] Hirosawa M, Totoki Y, Hoshida M, Ishikawa M (1995) Comprehensive study on iterative algorithms of multiple sequence alignment. Comput Appl Biosci 11:13–18.

[62] Goux JP, Kulkarni S, Yoder M, Linderoth J (2000) An enabling framework for master-worker applications on the computational grid. In: HPDC '00: Proceedings of the 9th IEEE International Symposium on High Performance Distributed Computing. Washington, DC, USA: IEEE Computer Society, p. 43.

[63] Condor. URL `http://www.cs.wisc.edu/condor/`.

[64] DeSantis T, Hugenholtz P, Larsen N, Rojas M, Brodie E, et al. (2006) Greengenes, a chimera-checked 16S rRNA gene database and workbench compatible with ARB. Appl Environ Microbiol 72:5069–5072.

[65] Beitz E (2000) TEXshade: shading and labeling of multiple sequence alignments using LATEX2 epsilon. Bioinformatics 16:135–139.