A simple theory for finding related sequences by adding probabilities of alternative alignments

Martin C. Frith[1,2,3]

[1]Artificial Intelligence Research Center, AIST
[2]Graduate School of Frontier Sciences, University of Tokyo
[3]Computational Bio Big Data Open Innovation Laboratory, AIST

Summary

The main way of analyzing genetic sequences is by finding sequence regions that are related to each other. There are many methods to do that, usually based on this idea: find alignments that are unlikely to occur by chance between unrelated sequences. This approach is suboptimal, and we do not fully understand how to tell if an alignment is likely to occur by chance. A more powerful way is to integrate a similarity measure over alternative ways of aligning the regions. This is rarely done: we lack a clear and simple theory suitable for wide adoption.

Here is described a simplest-possible change to standard sequence alignment, which sums probabilities of alternative alignments. It is easy to calculate the probability of such a similarity occurring by chance. This approach is better than standard alignment at finding distant relationships, at least in a few tests. It fits easily as a component in alignment software, and it generalizes to different kinds of alignment (e.g. DNA-versus-protein with frameshifts). Thus, it can widely contribute to finding subtle relationships between sequences.

Introduction

Many methods have been developed for finding related regions of nucleotide or protein sequences. They usually follow a standard approach: define positive and negative scores for monomer matches, mismatches, and gaps, then seek high-scoring alignments. Why is this bizarre, intricate approach a good way to find related sequences? Its result depends utterly on the score parameters, so how should we choose them?

The answer to both questions is that the approach is equivalent to probability-based alignment [1, 2]. This uses probabilities of matches, mismatches, and gaps, e.g. their average rates in the kinds of related sequences we wish to find, and seeks high-probability alignments. An alignment's score is a log probability ratio [3]:

$$\text{score} = t \ln \left[\frac{\text{prob(alignment)}}{\text{prob(null alignment)}} \right] . \tag{1}$$

t is an arbitrary positive constant, because scaling scores by a constant makes no difference. A "null alignment" is a length-zero alignment between the two sequences. Thus, a high alignment score indicates high probability with these rates.

A more direct way to judge whether two sequence regions are related is to calculate their probability without fixing an alignment [4–7]. In other words, use a score like this:

$$\begin{aligned} \text{score} &= t \ln \left[\frac{\sum_{\text{alignments}} \text{prob(alignment)}}{\text{prob(null alignment)}} \right] \\ &= t \ln \sum_{\text{alignments}} \frac{\text{prob(alignment)}}{\text{prob(null alignment)}} . \end{aligned} \tag{2}$$

The sum is over different possible ways of aligning the sequence regions, so it gives their total probability

with the assumed rates of matches, mismatches, and gaps.

This sum-over-alignments approach seems likely to be better at recognizing subtly related sequences. It has been used with great success in the HMMER software [8, 9]. So it is curious that many sequence search tools developed since HMMER do not use this approach. The reason, perhaps, is that it seems complex. For example, HMMER has an enormous "implicit probabilistic model", and sequence-length–dependent parameters [8].

It is important to know whether a similarity score (eqs. 1, 2) is higher than likely to occur by chance between unrelated sequences. The simplest definition is: the probability of an equal or higher score between two sequences of random i.i.d. (independent and identically distributed) monomers. Even this simplest definition, however, is hard to calculate. There is a theory for gapless alignment [10], but gapped alignment relies on conjecture and simulation. BLAST can calculate it only for a few sets of score parameters, which have been prepared in advance. The ALP library can calculate it for arbitrary parameters [11], by complex simulations. Remarkably, HMMER calculates it in an easier way, based on other conjectures. The scope of those conjectures is unclear, e.g. they seem to hold only after tweaking HMMER to use a "uniform entry/exit distribution" [8].

Here is described a minimal modification of standard alignment, which sums probabilities of alternative alignments (eq. 2). It is easy to use in typical sequence-search software. It is also easy to calculate the probability of an equal or higher score between random sequences, based on a clear conjecture.

Actually, this method is a variant of "hybrid alignment" [12]. Hybrid alignment has been neglected, probably because its description was mathematically fearsome, and it was presented as a strange "hybrid" method. The variant described here is simple and natural.

Methods

Review of standard alignment extension

To cope with big data, sequence search methods use fast heuristics to find potentially related regions, then define the regions precisely by alignment. The heuristics are various: short exact matches, spaced matches, reduced-alphabet matches, groups of nearby matches, etc. [13–16]. The next step is often done by extending alignments to the left and right of a small "core" alignment (fig. 1). This method finds the highest-scoring alignment extension, within a limited range shown in gray in fig. 1. The range is adjusted as the algorithm proceeds, attempting to follow the highest score. The best way to do this is unclear: several ways have been suggested [13, 17–20].

To avoid irrelevant complexity, here is the extension algorithm in a block of predefined size $n \times n$. The two sequences are R (e.g. "reference") and Q (e.g. "query"), with positions numbered as in fig. 1A. The score for aligning monomer x (`a`, `c`, `g`, or `t` for DNA) to y is S_{xy}. Let us use different scores for starting versus extending gaps, because gaps are somewhat rare (low start probability) but often long (high extension probability). So the score for a length-k deletion is $a_D + b_D \times (k-1)$, for a length-k insertion $a_I + b_I \times (k-1)$.

Algorithm 1 Optimum alignment extension in an $n \times n$ block (fig. 1A)

$$X_{00} \leftarrow 0 \qquad X_{i0} \leftarrow -\infty \qquad X_{0i} \leftarrow -\infty \qquad (0 < i \leq n)$$
$$v \leftarrow -\infty \qquad Z_{i0} \leftarrow -\infty \qquad Y_{0i} \leftarrow -\infty \qquad (0 \leq i \leq n)$$

$$\left.\begin{aligned} w &\leftarrow \max(X_{ij}, Y_{ij}, Z_{ij}) \\ v &\leftarrow \max(v, w) \\ X_{i+1\ j+1} &\leftarrow w + S_{R_{i+1} Q_{j+1}} \\ Y_{i+1\ j} &\leftarrow \max(w + a_D,\ Y_{ij} + b_D) \\ Z_{i\ j+1} &\leftarrow \max(w + a_I,\ Z_{ij} + b_I) \end{aligned}\right\} \begin{aligned} 0 \leq i \leq n \\ 0 \leq j \leq n \end{aligned}$$

Several variants of this algorithm are possible. This variant has simple boundary conditions, while min-

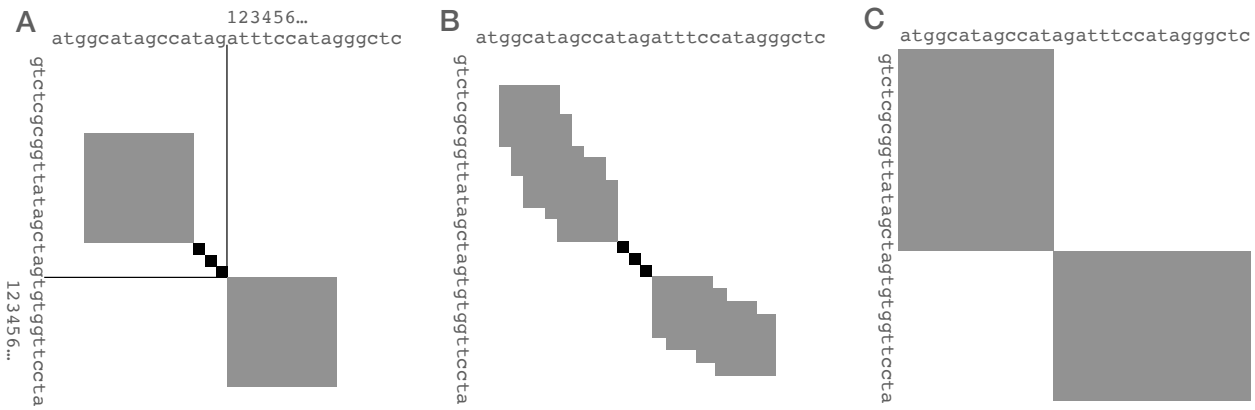

Figure 1: Some ways of extending an alignment in both directions from a small "core" alignment (black). In each direction, an optimum alignment extension is sought from among all possible ones within the gray area. **A** Simple toy method: the gray areas are blocks of predefined size $n \times n$. **B** Realistic example: the gray areas are adjusted as the calculation proceeds, attempting to follow good alignments. **C** Unlimited extension from zero-size core.

imizing add, max, and array access operations [21]. It finds the highest possible score for an alignment ending at R_i and Q_j, with R_i aligned to Q_j (X_{ij}), or R_i aligned to a gap (Y_{ij}), or Q_j aligned to a gap (Z_{ij}). v tracks the best alignment score seen so far, and is the final output. The alignment score after left and right extension is:

$$\text{score} = v_{\text{left}} + (\text{core score}) + v_{\text{right}}\,. \tag{3}$$

New extension algorithm

The new algorithm has two changes. The first, which makes no difference to the result, is to work with exponentiated values. This means that $-\infty \Rightarrow 0$, $0 \Rightarrow 1$, and addition $\Rightarrow$ multiplication. The other change is: maximization $\Rightarrow$ addition. This calculates the sum of probability ratios for all alignment extensions in the gray area (eq. 2), instead of the maximum probability ratio. The score with left and right extension is:

$$\text{score} = t\ln[v'_{\text{left}}] + (\text{core score}) + t\ln[v'_{\text{right}}]\,. \tag{4}$$

Unfortunately, the values in alg. 2 may become too large for the computer to handle. This can be fixed by occasionally rescaling them (see the Supplement).

Algorithm 2 Sum over alignment extensions in an $n \times n$ block (fig. 1A)

$$X'_{00} \leftarrow 1 \qquad X'_{i0} \leftarrow 0 \qquad X'_{0i} \leftarrow 0 \qquad (0 < i \le n)$$
$$v' \leftarrow 0 \qquad Z'_{i0} \leftarrow 0 \qquad Y'_{0i} \leftarrow 0 \qquad (0 \le i \le n)$$

$$\left.\begin{aligned} w' &\leftarrow X'_{ij} + Y'_{ij} + Z'_{ij} \\ v' &\leftarrow v' + w' \\ X'_{i+1\ j+1} &\leftarrow w' \cdot S'_{R_{i+1}Q_{j+1}} \\ Y'_{i+1\ j} &\leftarrow w' \cdot a'_D + Y'_{ij} \cdot b'_D \\ Z'_{i\ j+1} &\leftarrow w' \cdot a'_I + Z'_{ij} \cdot b'_I \end{aligned}\right\} \begin{aligned} 0 \le i \le n \\ 0 \le j \le n \end{aligned}$$

Parameters from probabilities

To use this algorithm, we need to choose values for a'_D, a'_I, b'_D, b'_I, and S'_{xy}. They are related to probabilities of matches, mismatches, and gaps. These probabilities are shown precisely in fig. 2. The main alignment probabilities are shown in the middle: α_D is the deletion start probability, β_D the deletion extension probability, α_I the insertion start probability, and β_I the insertion extension probability. Since we are looking for related *parts* of sequences, there are

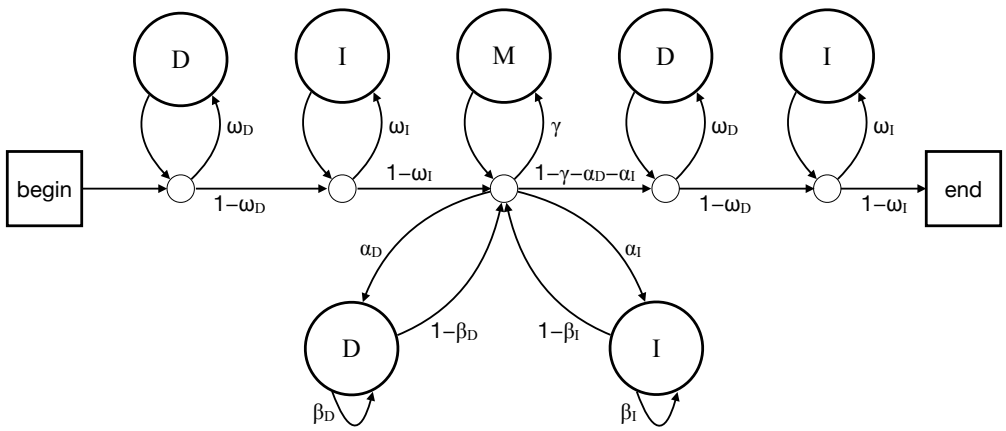

Figure 2: A probability model for two sequences with a related region. The model produces different alignments with different probabilities. Starting at "begin", the arrows are followed according to their probabilities (e.g. ω_D versus $1-\omega_D$). Each pass through a **D** state produces an unaligned letter x in the first sequence, with probabilities ϕ_x. Each pass through an **I** state produces an unaligned letter y in the second sequence, with probabilities ψ_y. Each pass through **M** generates two aligned letters, x and y, with probabilities π_{xy}.

also probabilities for unrelated left and right flanks. A "null alignment" is produced by a path that never traverses the α_D, α_I, or γ arrows. The algorithm parameters are [3]:

$$S'_{xy} = \frac{\gamma}{\omega_D \omega_I} \cdot \frac{\pi_{xy}}{\phi_x \psi_y} \tag{5}$$

$$a'_D = \alpha_D(1-\beta_D)/\omega_D \tag{6}$$

$$b'_D = \beta_D/\omega_D \tag{7}$$

$$a'_I = \alpha_I(1-\beta_I)/\omega_I \tag{8}$$

$$b'_I = \beta_I/\omega_I \tag{9}$$

With these parameters, alg. 2 calculates the sum of probability ratios in eq. 2. The parameters for alg. 1 are defined similarly (see the Supplement).

Choosing the probabilities

The probabilities (π_{xy}, α_D, etc.) should be tuned for the kind of related sequences we wish to find. One way is to use `last-train`, which finds related regions of given sequences and infers these probabilities [22]. Another option is to use BLOSUM or PAM substitution probabilities (π_{xy}), which are in the NCBI C++ toolkit (version 25.2, in `matrix_frequency_data.c`).

The flank probabilities ω_D and ω_I seem less realistic: how should they be chosen? `last-train` chooses them in a special way, explained below.

Short/long alignment bias

The probabilities may be biased towards short or long alignments. For example, if α_D, α_I, and γ are all low (almost 0), but ω_D and ω_I are high (almost 1), shorter alignments will tend to have higher probability. In the converse situation, longer alignments are favored. The probabilities are unbiased when [3]:

$$\frac{\gamma}{\omega_D \omega_I} + \frac{\alpha_D(1-\beta_D)}{\omega_D - \beta_D} + \frac{\alpha_I(1-\beta_I)}{\omega_I - \beta_I} = 1\,. \tag{10}$$

Unbiased probabilities are useful (see below). Therefore, `last-train` chooses ω_D and ω_I so as to satisfy eq. 10. It assumes that $\omega_D = \omega_I$, and finds the unique value that satisfies eq. 10 (see the Appendix).

A non-heuristic similarity score

We wish to know the probability of a similarity score occurring by chance between random sequences. First, we need a non-heuristic definition of similarity score.

We shall use a definition with two useful properties: it is approximated by typical heuristics, and it is easy to calculate the probability of scores for random sequences. The definition is: eq. 2 summed over all alignments that include one pair of positions in the sequences (fig. 1C).

We can calculate this for all possible pairs of positions. Call the sequences $R_1 \dots R_m$ and $Q_1 \dots Q_n$. We first calculate W^F_{ij}, the sum of probability ratios of all alignments that end immediately after i letters of R and j letters of Q:

$$0 \le i \le m, \;\; 0 \le j \le n$$
$$\begin{aligned} W^F_{ij} &= S'_{R_i Q_j} W^F_{i-1\;j-1} + Y^F_{i-1\;j} + Z^F_{i\;j-1} + 1 \\ Y^F_{ij} &= a'_D W^F_{ij} + b'_D Y^F_{i-1\;j} \\ Z^F_{ij} &= a'_I W^F_{ij} + b'_I Z^F_{i\;j-1} \end{aligned}$$

The boundary condition is: when $i < 0$ or $j < 0$, $W^F_{ij} = Y^F_{ij} = Z^F_{ij} = 0$.

We then calculate W^B_{ij}, the sum of probability ratios of all alignments that start immediately after i letters of R and j letters of Q:

$$m \ge i \ge 0, \;\; n \ge j \ge 0$$
$$\begin{aligned} W^B_{ij} &= S_{R_{i+1} Q_{j+1}} W^B_{i+1\;j+1} + Y^B_{i+1\;j} + Z^B_{i\;j+1} + 1 \\ Y^B_{ij} &= a'_D W^B_{ij} + b'_D Y^B_{i+1\;j} \\ Z^B_{ij} &= a'_I W^B_{ij} + b'_I Z^B_{i\;j+1} \end{aligned}$$

The boundary condition is: when $i > m$ or $j > n$, $W^B_{ij} = Y^B_{ij} = Z^B_{ij} = 0$. Finally, the sum of probability ratios of all alignments that include (i, j) is $W^F_{ij} W^B_{ij}$. So the score is $t \ln[W^F_{ij} W^B_{ij}]$.

Similarity scores occurring by chance

We wish to know the probability of getting a similarity score, $s_{\max} = t \ln[\max_{ij}(W^F_{ij} W^B_{ij})]$, between random sequences. We can calculate this for random i.i.d. sequences, with monomer probabilities ϕ_x and ψ_y. Note these must equal the probabilities of the D and I states (fig. 2). The conjecture is that

$$\text{prob}(s_{\max} \ge s) = 1 - \exp(-Kmn/e^{s/t}) \,, \qquad (11)$$

in the limit where the sequence lengths m and n are large, provided that eq. 10 is satisfied. A more general version of this conjecture is shown in the Appendix.

Eq. 11 has one unknown parameter, K. We can estimate K by generating (say) 50 pairs of random i.i.d. sequences, calculating $s_{\max}$ for each pair, and fitting K. The maximum likelihood fit is:

$$K = \frac{\text{harmonic mean}\,[\exp(s_{\max}/t)]}{mn} \,. \qquad (12)$$

The random chance of a similarity score is often reported as an E-value, which is defined as:

$$\begin{aligned} E &= -\ln[1 - \text{prob}(s_{\max} \ge s)] && (13) \\ &= Kmn/\exp(s/t) \,. && (14) \end{aligned}$$

Intuitively, it is the expected number of similarities with score $\ge s$ between two random sequences of length m and n.

Implementation and availability

These methods were implemented in LAST (https://gitlab.com/mcfrith/last). It finds similar regions using either alg. 1 with E-values from ALP, or alg. 2. In both cases, it gets a representative alignment by running alg. 1 and tracing back a way to get the maximum v.

The `last-train` output files from this study are at https://gitlab.com/mcfrith/sum-align. There too is a simple Python script to help use these ideas. It takes as input α_D, α_I, β_D, β_I, and γ, finds $\omega_D = \omega_I$ that satisfies eq. 10, and outputs a'_D, a'_I, b'_D, b'_I, and $\frac{\gamma}{\omega_D \omega_I}$. LAST includes code for calculating the non-heuristic similarity score (used for estimating K), in a self-contained file (`mcf_alignment_path_adder.cc`).

Results

Similarity scores occurring by chance

Let us test whether we can calculate probabilities of similarity scores between random sequences. Three sequence-search scenarios are considered, with typical DNA, `at`-rich DNA, and proteins.

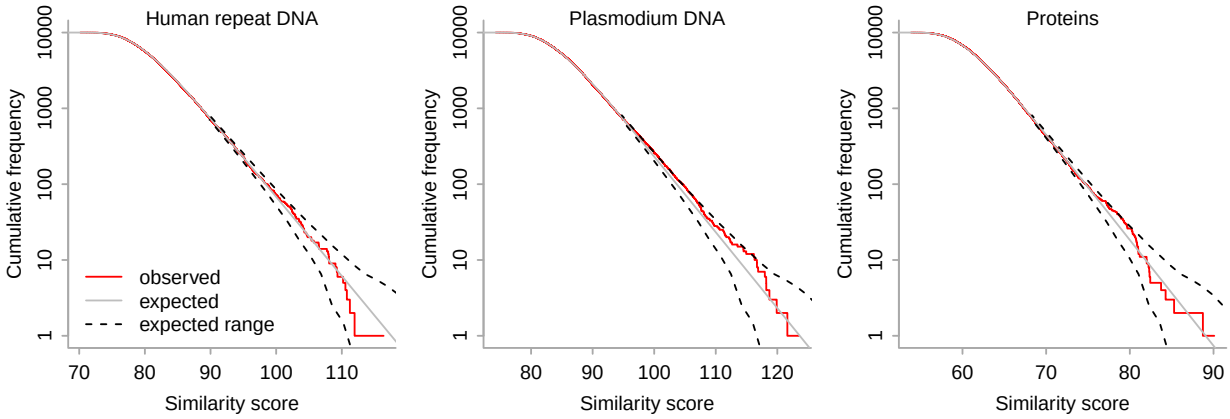

Figure 3: Non-heuristic similarity score ($s_{\max}$) between 10000 pairs of random i.i.d. sequences (length 10000), for three sets of alignment parameters. The frequency has a 5% chance of being outside the dashed lines (2.5% each).

The first scenario is finding ancient (shared with reptiles) repeats in the human genome. This can be done by comparing the genome to repeat consensus sequences from Dfam [23]. Some are relics of transposable elements, others are source unknown. Alignment parameters were found by `last-train`, then similarity scores ($s_{\max}$) were found between random i.i.d. sequences (fig. 3 left). The observed scores are consistent with their probabilities predicted by eq. 11.

The next scenario was to find related parts of the *Plasmodium falciparum* and *Plasmodium yoelii* genomes, which are both about 80% `a+t`. Alignment parameters were found by `last-train`, then similarity scores were found between random `at`-rich sequences (fig. 3 middle). Again, the observed scores are consistent with eq. 11.

The third scenario was to find related parts of proteins from *Aquifex aeolicus* (a heat-loving bacterium) and *Pyrolobus fumarii* (a heat-loving archaeon). Alignment parameters were found by `last-train`, then similarity scores were found between random sequences (fig. 3 right). Yet again, the observed scores are consistent with eq. 11.

Related versus biased sequences

So it seems we can tell if a similarity score is higher than is likely to occur between random sequences. A high score could occur because the sequence regions have a common ancestor, and/or similar composition bias (fig. 4A).

We can investigate this by comparing two sequences after reversing (but not complementing) one of them (but see [24]). Then there are no segments related by evolution, but there may be similarities due to composition bias (fig. 4A).

Thus, the reversed human genome was searched against the repeat consensus sequences, the reversed *P. yoelii* genome against *P. falciparum*, and reversed *P. fumarii* proteins against *A. aeolicus* proteins. In each case, there are more high similarity scores than expected between random sequences (fig. 5 upper row), due to biased regions (fig. 4A).

For standard alignment, these similarities can be avoided by detecting biased regions with `tantan` [25], and "masking" them. An effective masking scheme is to change $S_{xy} \leftarrow \min(S_{xy}, 0)$ when x or y is in a biased region [26]. For the new alignment method, this masking scheme was used: $S'_{xy} \leftarrow \min(S'_{xy}, 1)$.[1] With this masking scheme, the similarity scores between reversed and non-reversed sequences are close to what is expected between random sequences (fig. 5 lower row).

[1] Another untested idea is $S'_{xy} \leftarrow \min(S'_{xy}, \frac{\gamma}{\omega_D \omega_I})$.

```
A tccttcctctcttttcttcccttccctcttctcttcccttcacttccctttctcttctctccctc
  |  || | | ||| || | ||||  ||||||||||| ||  |||  ||||||||| | |||| ||
  tnttttcccccttctcctaccttntctcttctcttccattggcttttcttttctctgccctccttc

B cctgtttggctcctggagtcttagcagagaccaccagcaggggagctaattactgccag-------------tctgtaatgtggacctccgtcc
  ||  ||  |  | |||  |||  || |||||  ||| || ||| || ||||| |||||              | ||| ||||||||  | ||||
  cccattcagtccttggcctctctgctgagactgccaacaagggtgccaattagtgccaattgtgatggtggttttgtgatgtggacacctgtcc

C acttgatcttagccaaaaggccgagaagcgat
  ||||||||||||||||||||||||||||||
  acttgatcttagccaaaaggccgagaagcgat
```

Figure 4: Alignment between parts of: **A** reversed human chromosome Y (upper), and UCON1 (lower). This is the highest-scoring alignment in the top-left panel of fig. 5. **B** Human chromosome 6 (upper) and LFSINE_Vert (lower). **C** Human chromosome 21 (upper) and human U2 (lower).

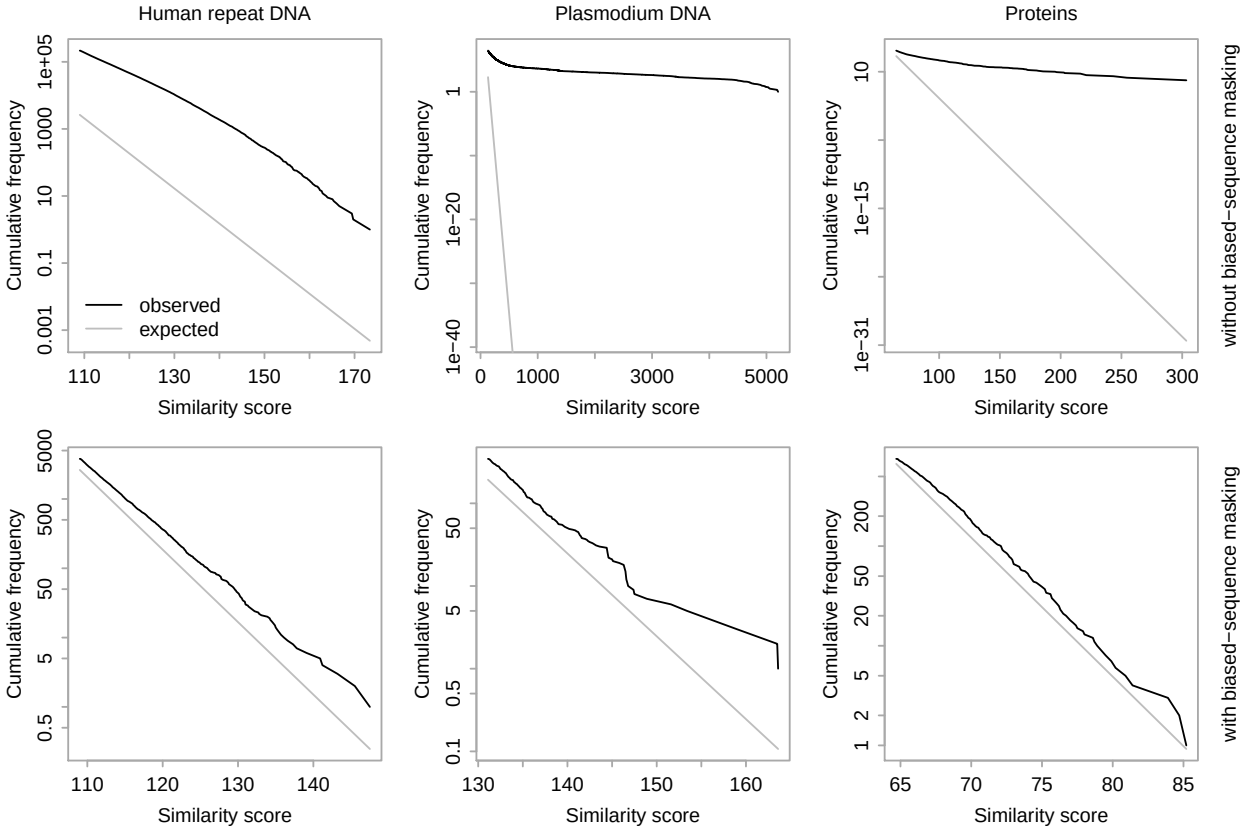

Figure 5: Heuristic similarity scores, between reversed and non-reversed sequences, with or without biased-sequence masking.

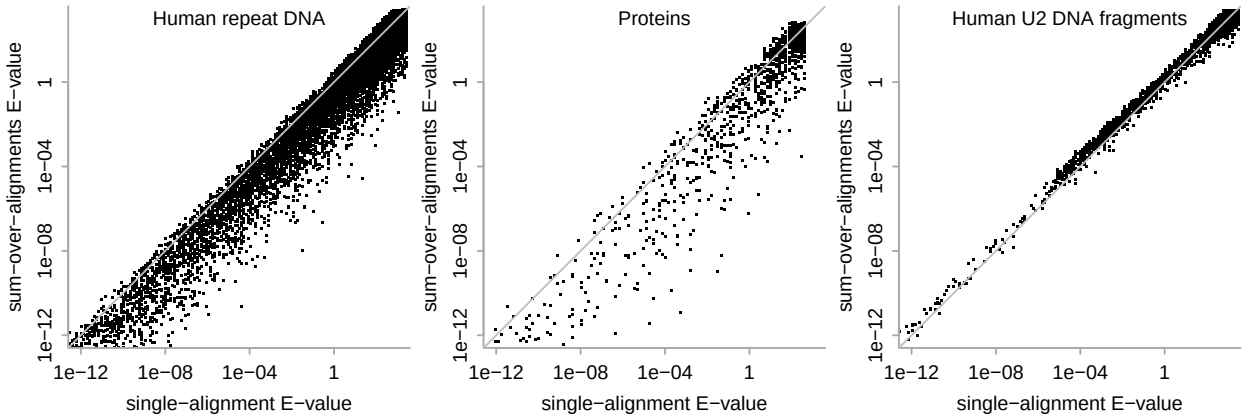

Figure 6: E-values for identical alignments found by alg. 1 (horizontal axis), and alg. 2 (vertical axis). Each point is one alignment. The diagonal gray lines indicate equal E-values. These E-values were calculated with m = total length of all reference sequences (e.g. all *A. aeolicus* proteins), and n = total length of all query sequences (e.g. all *P. fumarii* proteins). For DNA, one query strand was searched against both reference strands, so m was multiplied by 2.

The new method can be more sensitive

Related sequence parts were sought by standard alignment (alg. 1) and by the new method (alg. 2), with biased-sequence masking. The two methods often find identical alignments, but with different scores and E-values. The new method tended to assign lower E-values, for alignments of the human genome to repeat consensus sequences (fig. 6 left), and *P. fumarii* proteins to *A. aeolicus* proteins (fig. 6 middle). For example, both methods found the same alignment between human chromosome 6 and the LF-SINE_Vert consensus (fig. 4B): the alg. 1 E-value is 22, and the alg. 2 E-value is 0.000031. This means the new method can find related sequence parts more confidently.

The new method can be less sensitive

Another sequence search scenario was considered: finding U2 fragments in the human genome. U2 is one of the small nuclear RNAs that splice introns. The genome has many fragmentary copies of it, which are thus considered repeats.

The human genome was searched against a U2 DNA sequence, using the alignment parameters found earlier for the human genome and ancient repeat consensus sequences. This time, alg. 2 tended to assign higher E-values than alg. 1 (fig. 6 right). For example, both methods found the alignment shown in fig. 4C: the alg. 1 E-value is 0.00021, and the alg. 2 E-value is 0.0011.

The likely explanation is as follows. The genome has many short, near-exact copies of the first 30–40 bases of U2. The alignment parameters, however, are tuned for greater sequence divergence (e.g. fig. 4B). These mistuned parameters pessimize both algorithms, but they pessimize alg. 2 more. That is because parameters tuned for greater divergence attach more weight to alternative alignments.

Discussion

We have seen how to find related parts of sequences, by summing probability ratios of alternative alignments between them. This is better than standard alignment at detecting subtle relationships, at least in a few tests. It can be worse than standard alignment, when the probabilities are tuned for high divergence but the sequences have low divergence. The solution is to use appropriate probabilities. The problem with that, though, is there may be related sequences with both high and low divergence. For example,

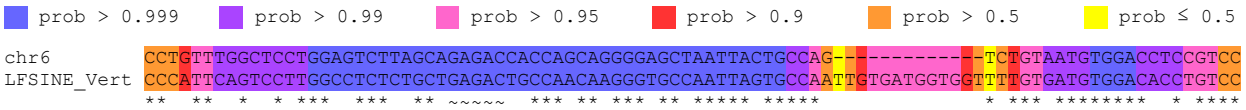

Figure 7: An alignment (the same as fig. 4B) with colors indicating the probability that each column is correct. ~ indicates the "core" (fig. 1), whose column probabilities are assumed to be 1. ⋆ indicates non-core exact matches.

the genome has both old (diverged) and young U2 fragments. The main benefit of the new method is likely for high divergence, where alternative alignments contribute more. It was recently used to find gene-regulating DNA conserved in diverse animals since the Precambrian [27].

A major further benefit is that the new method simplifies E-value calculation. This makes it easier to experiment with different alignment parameters, beyond the BLAST parameter sets. This can be useful for proteins [28], but especially for DNA, because there has been less effort to optimize DNA parameters.

An extension of alg. 2 (see the Supplement) can calculate the probability that each alignment column is correct (fig. 7), based on the assumed rates of matches, mismatches, and gaps [2, 4]. It is also possible to make alignments based on these probabilities: for example, align monomers whose alignment probability is > 0.5 [29].

Some previous sum-over-alignments methods define similarity score differently: their version of eq. 2 has a different denominator [5, 6, 8]. They aim to answer yes/no questions: does this pair of sequences have a related region? In other words, they aim to find relationships for *as many sequences as possible*. Another aim is to find *as many related regions as possible*. The latter aim seems more useful than the former for long sequences (e.g. chromosomes), and perhaps for shorter sequences (e.g. multi-domain proteins). The approach of this paper can be used for either aim, by setting m and n in the E-value formula (eq. 14) to the individual sequence lengths (first aim), or total lengths of all reference and query sequences (second aim). The aim is important when testing sequence search methods, because the test may count the number of sequences or related regions found.

For short sequences, the E-values are inaccurate (too high). The E-value formula (eq. 14) assumes that similar regions have mn possible start points in the two sequences. That is too high, because similar regions must have non-zero length and fit in the sequences. It would be useful to have a correction for this effect, as was done for hybrid alignment [12].

Another powerful way to find related sequences is to use position-specific match, mismatch, and gap rates [2, 30]. This can be done by first finding a group of related sequences, and observing their position-specific rates. This approach could likely be incorporated in the new method, since it was already combined with hybrid alignment [31].

Similar ideas were used in HMMER: it also sums over alternative alignments, and calculates E-values. Moreover, it uses position-specific rates, and a finite-length correction for E-values [8]. The main advantage of the new method is much greater simplicity. It is easy to plug into typical aligners: just replace alg. 1 with alg. 2.

The new method can be generalized to other kinds of alignment. For example, the same approach was used for DNA-versus-protein alignment, with extra probabilities for frameshifts [32]. This revealed relics of Paleozoic mobile elements in vertebrate genomes. A key benefit is the ability to calculate E-values, which was difficult for other kinds of alignment.

Acknowledgments

I am grateful to John Spouge for encouragement that the E-values would be simple, Yi-Kuo Yu for advice about hybrid alignment, and Travis Wheeler for advice about HMMER.

References

1. Dayhoff, M, Schwartz, R & Orcutt, B. A model of evolutionary change in proteins. *Atlas of protein sequence and structure* **5,** 345–352 (1978).
2. Durbin, R, Eddy, SR, Krogh, A & Mitchison, G. *Biological sequence analysis: probabilistic models of proteins and nucleic acids* (Cambridge University Press, 1998).
3. Frith, MC. How sequence alignment scores correspond to probability models. *Bioinformatics* **36,** 408–415 (2020).
4. Allison, L, Wallace, C & Yee, C. Finite-state models in the alignment of macromolecules. *Journal of Molecular Evolution* **35,** 77–89 (1992).
5. Bucher, P & Hofmann, K. *A sequence similarity search algorithm based on a probabilistic interpretation of an alignment scoring system* in *Proc Int Conf Intell Syst Mol Biol.* (1996), 44–51.
6. Webb, BJM, Liu, JS & Lawrence, CE. BALSA: Bayesian algorithm for local sequence alignment. *Nucleic Acids Research* **30,** 1268–1277 (2002).
7. Eddy, SR. *A new generation of homology search tools based on probabilistic inference* in *Genome Informatics 2009: Genome Informatics Series Vol. 23* 205–211 (World Scientific, 2009).
8. Eddy, SR. A probabilistic model of local sequence alignment that simplifies statistical significance estimation. *PLoS Computational Biology* **4,** e1000069 (2008).
9. Wheeler, TJ & Eddy, SR. nhmmer: DNA homology search with profile HMMs. *Bioinformatics* **29,** 2487–2489 (2013).
10. Karlin, S & Altschul, SF. Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proceedings of the National Academy of Sciences* **87,** 2264–2268 (1990).
11. Sheetlin, S, Park, Y, Frith, MC & Spouge, JL. ALP & FALP: C++ libraries for pairwise local alignment E-values. *Bioinformatics* **32,** 304–305 (2016).
12. Yu, YK & Hwa, T. Statistical significance of probabilistic sequence alignment and related local hidden Markov models. *Journal of Computational Biology* **8,** 249–282 (2001).
13. Altschul, SF *et al.* Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research* **25,** 3389–3402 (1997).
14. Ma, B, Tromp, J & Li, M. PatternHunter: faster and more sensitive homology search. *Bioinformatics* **18,** 440–445 (2002).
15. Noé, L & Kucherov, G. Improved hit criteria for DNA local alignment. *BMC Bioinformatics* **5,** 1–9 (2004).
16. Roytberg, M *et al.* On subset seeds for protein alignment. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **6,** 483–494 (2009).
17. Zhang, Z, Berman, P & Miller, W. Alignments Without Low-Scoring Regions. *Journal of Computational Biology* **5,** 197–210 (1998).
18. Suzuki, H & Kasahara, M. Acceleration of nucleotide semi-global alignment with adaptive banded dynamic programming. *BioRxiv,* 130633 (2017).
19. Guidi, G, Ellis, M, Rokhsar, D, Yelick, K & Buluç, A. *BELLA: Berkeley efficient long-read to long-read aligner and overlapper* in *SIAM Conference on Applied and Computational Discrete Algorithms (ACDA21)* (2021), 123–134.
20. Liu, D & Steinegger, M. Block Aligner: an adaptive SIMD-accelerated aligner for sequences and position-specific scoring matrices. *Bioinformatics,* btad487 (2023).
21. Cameron, M, Williams, HE & Cannane, A. Improved gapped alignment in BLAST. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **1,** 116–129 (2004).
22. Hamada, M, Ono, Y, Asai, K & Frith, MC. Training alignment parameters for arbitrary sequencers with LAST-TRAIN. *Bioinformatics* **33,** 926–928 (2017).
23. Storer, J, Hubley, R, Rosen, J, Wheeler, TJ & Smit, AF. The Dfam community resource of transposable element families, sequence models, and genome annotations. *Mobile DNA* **12,** 1–14 (2021).
24. Glidden-Handgis, GE & Wheeler, TJ. WAS IT A MATch I SAW? Approximate palindromes lead to overstated false match rates in benchmarks using reversed sequences. *bioRxiv,* 2023–06 (2023).
25. Frith, MC. A new repeat-masking method enables specific detection of homologous sequences. *Nucleic Acids Research* **39,** e23–e23 (2011).

26. Frith, MC. Gentle masking of low-complexity sequences improves homology search. *PLoS One* **6,** e28819 (2011).

27. Frith, MC & Ni, S. DNA conserved in diverse animals since the Precambrian controls genes for embryonic development. *bioRxiv* (2023).

28. Ng, PC, Henikoff, JG & Henikoff, S. PHAT: a transmembrane-specific substitution matrix. *Bioinformatics* **16,** 760–766 (2000).

29. Miyazawa, S. A reliable sequence alignment method based on probabilities of residue correspondences. *Protein Engineering, Design and Selection* **8,** 999–1009 (1995).

30. Steinegger, M *et al.* HH-suite3 for fast remote homology detection and deep protein annotation. *BMC Bioinformatics* **20,** 1–15 (2019).

31. Yu, YK, Bundschuh, R & Hwa, T. Hybrid alignment: high-performance with universal statistics. *Bioinformatics* **18,** 864–872 (2002).

32. Yao, Y & Frith, MC. Improved DNA-Versus-Protein Homology Search for Protein Fossils. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **20,** 1691–1699 (2023).

Appendix

Conjecture on random similarity scores

We wish to know the probability of getting a similarity score $s_{\max}$ between random sequences. We can calculate this for random i.i.d. sequences, with monomer probabilities Φ_x and Ψ_y. The conjecture is as stated in eq. 11, in the limit where m and n are large, provided that

$$\left(\sum_{x,y} \Phi_x \Psi_y S'_{xy}\right) + \frac{a'_D}{1-b'_D} + \frac{a'_I}{1-b'_I} = 1\,. \qquad (15)$$

In the special case where $\Phi_x = \phi_x$ and $\Psi_y = \psi_y$, eq. 15 is the same as eq. 10.

This conjecture is similar to one for hybrid alignment [12]. Eq. 15 is analogous to eq. 28 in [12]. These equations are different because Yu & Hwa used a slightly different probability model [3, fig. S2]. Eq. 15 can also be viewed as a generalization of the equivalent equation for gapless alignment: eq. 4 in [10].

Finding $\omega_D = \omega_I$ that satisfies eq. 10

First, assume that $\omega_D > \beta_D$ and $\omega_I > \beta_I$. Otherwise, alignment gaps would have non-negative scores, which is not useful for sequence comparison.

As $\omega_D = \omega_I$ increases from $\max(\beta_D, \beta_I)$ to 1, the LHS of eqn. 10 decreases, from a value > 1 to a value < 1. So it equals 1 at a unique point, which can be found by e.g. repeated bisection.