The *bits* between proteins

Dinithi Sumanaweera[1], Lloyd Allison[1], and Arun S. Konagurthu[1,*]

[1] Faculty of Information Technology, Monash University, Clayton VIC 3800, Australia.
*Corresponding author: Arun S. Konagurthu (arun.konagurthu@monash.edu)

Abstract

Comparison of protein sequences via alignment is an important routine in modern biological studies. Although the technologies for aligning proteins are mature, the current state of the art continues to be plagued by many shortcomings, chiefly due to the reliance on: (i) naive objective functions, (ii) fixed substitution scores independent of the sequences being considered, (iii) arbitrary choices for gap costs, and (iv) reporting, often, one optimal alignment without a way to recognise other competing sequence alignments. Here, we address these shortcomings by applying the compression-based Minimum Message Length (MML) inference framework to the protein sequence alignment problem. This grounds the problem in statistical learning theory, handles directly the complexity-vs-fit trade-off without ad hoc gap costs, allows unsupervised inference of all the statistical parameters, and permits the visualization and exploration of competing sequence alignment landscape.

1 Introduction

A protein is characterised as a sequence over the amino acid alphabet. An alignment between two amino acid sequences specifies a one-to-one correspondence between their subsequences. Inferring meaningful alignment relationships is central to deciphering how, and to what extent, proteins evolve from their common ancestor.

Protein sequence alignment programs are geared towards finding an alignment that optimises a specified objective function using two key criteria. The first is a user-specified choice of a *substitution scoring matrix* that associates a numeric score for each pair of matched (corresponding) amino acid letters. The second is the user-specified *gap function parameters* to penalise any unmatched gap runs.

The first criterion is defined in the form of an empirically-derived substitution scoring matrix. Dayhoff [1] computed the original PAM (Point Accepted Mutation) family of log-odds-ratio-based substitution matrices that model frequencies of substitutions across varying levels of evolutionary relatedness. For instance, PAM-10 matrix is used when comparing sequences that are 90% similar, PAM-30 when they are 75% similar, PAM-70, PAM-120, and PAM-250 when sequences are 55%, 37% and 20% similar respectively [2]. Another commonly used family of scoring matrices is the BLOSUM series designed by Henikoffs [3]. These computations deal with the substitution frequencies derived directly from ungapped blocks of conserved protein sequences, constrained within varying thresholds of sequence similarity. BLOSUM-50 and BLOSUM-62 are commonly used by many aligners for sequences that are 25% and 30% similar. For a comprehensive summary of substitution matrices, refer [2].

The *affine gap* penalty function, of the form $\gamma(l) = g_o + (l-1)g_e$, is commonly used for the second criterion, where l is the run-length of any unmatched stretch (gap) of

amino acid symbols in an alignment, while g_o and g_e are gap penalty parameters that penalise opening and extending a gap respectively. Unlike the substitution scores, (g_o, g_e) parameters are not based on any evolutionary model, but rather tuned by the user to suit the scoring matrix and sequences being aligned. BLOSUM-50, for instance, is commonly used with (g_o, g_e) parameters of $(10, 2)$, BLOSUM-62 commonly uses $(11, 1)$, PAM-70 uses $(10, 1)$ and PAM-30 uses $(9, 1)$.

The choices of substitution matrix and gap parameters affect the global protein alignment accuracy. The field continues to lack a reliable and *unsupervised method* of aligning protein sequences across the spectrum of evolutionary relationships. This motivates the research presented in this paper.

We apply the information-theoretic Minimum Message Length (MML) method to address the protein sequence alignment problem. MML is a Bayesian method of inferring promising *hypotheses* that best explain observed data [4]. A sequence alignment is a hypothesis that explains the relationship between two observed sequences. Formulating the sequence alignment problem in MML is best understood as an imaginary communication between a *transmitter-receiver* pair. The transmitter's goal is to communicate to the receiver the observed amino acid sequence pair, $\langle \mathbf{S}, \mathbf{T} \rangle$. One approach is to assume $\mathbf{S}$ and $\mathbf{T}$ are *unrelated* and, thus, transmit them independent of each other. We refer to this as the *null model* message, and denote its length as:

$$NULL(\langle \mathbf{S}, \mathbf{T} \rangle) = NULL(\mathbf{S}) + NULL(\mathbf{T}) \qquad \text{bits.} \tag{1}$$

In contrast, an *alignment model* attempts to economise the message length using any *shared* information between $\langle \mathbf{S}, \mathbf{T} \rangle$ via an alignment $\mathcal{A}$. This model yields a two-part message. In the first, the transmitter communicates the alignment $\mathcal{A}$. In the second part, the amino acid symbols of $\mathbf{S}$ and $\mathbf{T}$ are encoded using $\mathcal{A}$. Denote this two-part message length as:

$$\mathrm{I}(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle) = \underbrace{\mathrm{I}(\mathcal{A})}_{\text{First part}} + \underbrace{\mathrm{I}(\langle \mathbf{S}, \mathbf{T} \rangle \,|\mathcal{A})}_{\text{Second part}} \qquad \text{bits.} \tag{2}$$

The above equation estimates the *Shannon information content* [5] of various terms as the *negative logarithm of the probability* used to state that term: $I(\cdot) = -\log(\Pr(\cdot))$.

The message length term $\mathrm{I}(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$, linked with the joint probability of $\mathcal{A}$ and $\langle \mathbf{S}, \mathbf{T} \rangle$, allows the comparison of competing alignment hypotheses, and gauge their statistical significance. Formally, the difference of alignment model message lengths between any two alignments, $\mathcal{A}_1$ and $\mathcal{A}_2$, gives the log-odds ratio of their posterior probabilities:

$$I(\mathcal{A}_1, \langle \mathbf{S}, \mathbf{T} \rangle) - I(\mathcal{A}_2, \langle \mathbf{S}, \mathbf{T} \rangle) = \log \left(\frac{\Pr(\mathcal{A}_2) \Pr(\langle \mathbf{S}, \mathbf{T} \rangle \,|\mathcal{A}_2)}{\Pr(\mathcal{A}_1) \Pr(\langle \mathbf{S}, \mathbf{T} \rangle \,|\mathcal{A}_1)} \right) = \underbrace{\log \left(\frac{\Pr(\mathcal{A}_2 |\, \langle \mathbf{S}, \mathbf{T} \rangle)}{\Pr(\mathcal{A}_1 |\, \langle \mathbf{S}, \mathbf{T} \rangle)} \right)}_{\text{log-odds posterior ratio}}.$$

It follows from above that the best alignment hypothesis is one with the shortest value of $\mathrm{I}(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$. Furthermore, $NULL(\langle \mathbf{S}, \mathbf{T} \rangle) - \mathrm{I}(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$ provides a natural test for the statistical significance of the alignment hypothesis. Alignments whose estimated $\mathrm{I}(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$ is longer than $NULL(\langle \mathbf{S}, \mathbf{T} \rangle)$ are rejected.

Previous applications of MML to aligning DNA (nucleotide) sequences include [6, 7, 8]. The work here applies MML to the protein sequence alignment problem

that requires new statistical models for estimation of terms involved in $NULL(\langle \mathbf{S}, \mathbf{T} \rangle)$ (see Equation 1) and $\mathrm{I}(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$ (see Equation 2) for any given protein sequence pair $\langle \mathbf{S}, \mathbf{T} \rangle$ and their alignment $\mathcal{A}$. We develop methods that find the best $\mathcal{A}$ under this compression framework, and visualise the whole landscape of competing alignment hypotheses.

2 Methods

2.1 Estimation of $NULL(\cdot)$

A string generated from a fixed alphabet $\aleph = \{x_1, \ldots, x_{|\aleph|}\}$ with $|\aleph|$ symbols can be modelled by a multi-state model. A multi-state model is defined using the corresponding set of multinomial state probability parameters $\{\Pr(x_1), \ldots, \Pr(x_{|\aleph|})\}$. The space of possible probability values these parameters can take defines the standard $(|\aleph| - 1)$-simplex with $|\aleph| - 1$ free parameters [9]. Given a string of length N from $\aleph$, where each $x_i \in \aleph$ is observed $|x_i|$ times, the MML (Wallace-Freeman [10]) estimate of $\Pr(x_i)$ is $\hat{\Pr}(x_i) = \frac{|x_i|+1/2}{N+|\aleph|/2}$ [4, 9].

Any protein sequence is a string from the amino acid alphabet. There are 20 naturally occurring amino acids. The Universal Protein resource (UniProt) [11] provides a large corpus of protein sequences on which the MML null probability estimate, $\hat{\Pr}(x_i)$, for each amino acid symbol $x_i \in \aleph$ can be computed.

In this work, we model the observed protein sequences as order $k = \{0, 1, 2\}$ Markov chains. For any protein sequence $\mathbf{Y}$, denoted as a string of amino acid codes $(y_1 y_2 \ldots y_n)$, the null statement of $\mathbf{Y}$ involves (i) stating the number of amino acids in $\mathbf{Y}$ over a Wallace-tree code over positive integers [12], taking $I_{\text{integer}}(n)$ bits, followed by (ii) the null encoding of each $y_i \in \mathbf{Y}$ (using the order-k MML estimates), taking $I_{\text{null}}(\mathbf{Y}) = \sum_{i=1}^{n} I_{\text{null}}(y_i) = \sum_{i=1}^{n} -\log(\hat{\Pr}(y_i | y_{i-1} \cdots y_{i-k}))$. Therefore:

$$NULL(\mathbf{Y}) = I_{\text{integer}}(n) + I_{\text{null}}(\mathbf{Y}) = I_{\text{integer}}(n) + \sum_{i=1}^{n} I_{\text{null}}(y_i) \qquad \text{bits.} \qquad (3)$$

The null probability estimates need *not* be transmitted as the computation of these and the data on which they are computed is common knowledge. Equation 3 in turn gets used for computing the null model encoding length $NULL(\langle \mathbf{S}, \mathbf{T} \rangle)$ of any given sequence pair $\langle \mathbf{S}, \mathbf{T} \rangle$ in Equation 1, taking $O(|\mathbf{S}| + |\mathbf{T}|)$ time.

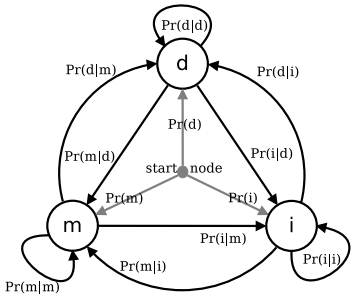


Figure 1: Alignment between two sequences is a string generated by a three-state machine with `match`, `delete` and `insert` states.

2.2 Estimation of $\mathrm{I}(\mathcal{A})$:

An alignment between any sequence pair $\langle \mathbf{S}, \mathbf{T} \rangle$ can be represented as a three-state string over `m` (match), `d` (delete), or `i` (insert) states. Communicating an alignment $\mathcal{A}$ involves sending its length, $|\mathcal{A}|$, using the Wallace-Tree code, followed by the three-state string modelled as an order 1 Markov chain (see Fig. 1).

The transition probabilities out of each state in this machine add up to 1 (see Fig. 1): $\sum \Pr(*|\mathtt{m}) = \sum \Pr(*|\mathtt{d}) = \sum \Pr(*|\mathtt{i}) = 1, \forall * \in \{\mathtt{m},\mathtt{d},\mathtt{i}\}$. Furthermore, the following transition symmetry between `delete` and `insert` states are applied, reducing the number of free parameters: $\Pr(\mathtt{m}|\mathtt{i}) \equiv \Pr(\mathtt{m}|\mathtt{d})$; $\Pr(\mathtt{i}|\mathtt{m}) \equiv \Pr(\mathtt{d}|\mathtt{m})$; $\Pr(\mathtt{i}|\mathtt{d}) \equiv \Pr(\mathtt{d}|\mathtt{i})$; $\Pr(\mathtt{i}|\mathtt{i}) \equiv \Pr(\mathtt{d}|\mathtt{d})$.

The MML estimation of parameters on a multistate distribution (Section 2.1) is used here to estimate the transition probabilities (denoted as $\vec{\Theta}_{\mathcal{A}}^{\mathrm{MML}}$) with the alphabet size of 3, and the Markov order of 1. We note that the parameters $\vec{\Theta}_{\mathcal{A}}^{\mathrm{MML}}$ have to be stated explicitly as a part of the message that encodes and transmits the string using the stated parameters. Wallace and Freeman [10, 4] approximated the precision of stating the MML estimates of any d-dimensional parameter vector $\vec{\Theta}^{\mathrm{MML}}$ to $\kappa_d^{d/2}/\det(\mathcal{F}(\vec{\Theta}^{\mathrm{MML}}))$, where κ_d is the d-dimensional lattice constant [13], and $\det(\mathcal{F}(\vec{\Theta}^{\mathrm{MML}}))$ is the determinant of the *Fisher Information matrix* (the $d \times d$ matrix of the *expected* second derivatives of the negative log likelihood function).

The computation of determinant of the Fisher information for multistate distribution is described at `www.allisons.org/ll/MML/Discrete/Multistate/`. Using this approach, we compute the length, $\mathrm{I}(\vec{\Theta}_{\mathcal{A}}^{\mathrm{MML}})$, of stating the parameters. The statement length of $\mathcal{A}$ given $\vec{\Theta}_{\mathcal{A}}^{\mathrm{MML}}$ is denoted as $\mathrm{I}(\mathcal{A}|\vec{\Theta}_{\mathcal{A}}^{\mathrm{MML}})$. Combining all these terms, the total statement length of transmitting $\mathcal{A}$ takes:

$$\mathrm{I}(\mathcal{A}) = \underbrace{I(|\mathcal{A}|)}_{\text{length}} + \underbrace{\mathrm{I}(\vec{\Theta}_{\mathcal{A}}^{\mathrm{MML}})}_{\text{params}} + \underbrace{\mathrm{I}(\mathcal{A}|\vec{\Theta}_{\mathcal{A}}^{\mathrm{MML}})}_{\text{3-state string}} \qquad \text{bits.} \tag{4}$$

This computation takes $O(|\mathcal{A}|)$ time.

2.3 Estimation of $\mathrm{I}(\langle \mathbf{S}, \mathbf{T} \rangle | \mathcal{A})$*:*

The second part of Equation 2 involves the message-length term required to explain the sequence pair $\langle \mathbf{S}, \mathbf{T} \rangle$ using the knowledge of a given alignment $\mathcal{A}$, communicated in the first part. In Section 2.2 we have seen that this alignment statement comes in the form of a three-state string. On receiving $\mathcal{A}$, the receiver can determine the length of $\mathbf{S}$ (= number of '`m`'s plus number of '`d`'s) and the length of $\mathbf{T}$ (= number of '`m`'s plus number of '`i`'s). To complete the communication under the alignment model, the details of the amino acid symbols that make up $\langle \mathbf{S}, \mathbf{T} \rangle$ are sent. To achieve this, the transmitter can progressively encode, left-to-right, the amino acid symbols associated with each alignment state. Specifically: `m` state implies some $s_i \in \mathbf{S}$ is aligned with some $t_j \in \mathbf{T}$; `d` state implies some $s_i \in \mathbf{S}$ remains unaligned; `i` state implies some $t_j \in \mathbf{T}$ remains unaligned.

The statement of any s_i or t_j when the alignment is in the '`d`' or the '`i`' state is carried out using the null encoding (Section 2.1), taking $I_{\mathrm{null}}(s_i)$ and $I_{\mathrm{null}}(t_j)$ bits respectively.

However, in the '`m`' state, there is a pair of amino acids $\langle s_i, t_j \rangle$ that are hypothesised to be related, and can potentially be stated concisely compared to their null encodings. To encode such matched pairs, we aim for a probabilistic model over pairs of amino acids that is capable of handling extrapolations of their pairwise relationships across varying evolutionary distances of their underlying sequences.

2.3.1 Transition probability matrix for matched amino acid pairs.

Relying on alignments of 71 families of closely-related proteins, Dayhoff and colleagues [1] modelled the transition probabilities of each amino acid under evolutionary pressure. Their approach derived a unit Point Accepted Mutation matrix (PAM-1). PAM-1 (denoted as M^1) is a transition probability matrix over amino acid symbols, where any $M^1(x, y)$ gives the transition probability of the amino acid y mutating into x, $\Pr(x|y)$, in one PAM time-step. This is same as the probability that amino acid y will mutate into x in sequences that have diverged by one unit in PAM distance terms. (Note, $M^1(x, y) \equiv \Pr(x|y) \neq \Pr(y|x) \equiv M^1(y, x)$.)

Generalizing, a PAM-n matrix ($M^n, \forall n > 0$) gives the probability of any amino acid y mutating into any x in n PAM steps. PAM-n (M^n) can be derived from PAM-1 (M^1) via *matrix exponentiation*: $(M^1)^n = M^n$. Efficient matrix exponentiation is derived by storing the eigen-decomposition of M^1 into $S\Lambda S^{-1}$, where S is the matrix of eigenvectors of M^1, and Λ is the diagonal matrix of M^1's eigenvalues. Thus, $M^n = S\Lambda^n S^{-1}$. Furthermore, as $n \to \infty$, M^n should approach the stationary (null probability) distribution characterised in Section 2.1. (See Table 2(a) in Section 3.)

2.3.2 Encoding the matched amino acid pairs using a given PAM-n.

Given an alignment $\mathcal{A}$ of a sequence pair $\langle \mathbf{S}, \mathbf{T} \rangle$, let $\mathcal{A}_{\mathtt{m}}$ denote the subsequence of $\mathcal{A}$ that defines all its matches. Denote the ordered set of matches between amino acid pairs contained in the subsequence $\mathcal{A}_{\mathtt{m}}$ to be of the form: $\langle \mathbf{S}, \mathbf{T} \rangle_{\mathtt{m}} = (\langle s_{i_1}, t_{j_1} \rangle, \langle s_{i_2}, t_{j_2} \rangle, \ldots)$. We describe two approaches to encode these amino acid pairs using a given PAM-n (M^n) transition probability matrix:

(i) Amino acid mutation-generation (asymmetric) machine: In this approach, each matched pair of symbols $\langle s_{i_k}, t_{j_k} \rangle \in \langle \mathbf{S}, \mathbf{T} \rangle_{\mathtt{m}}$, is stated as follows: First s_{i_k} is stated using the null model probability of $\Pr(s_{i_k})$, taking $I_{\text{null}}(s_{i_k}) = -\log(\Pr(s_{i_k}))$ bits. Then, t_{j_k} is stated using s_{i_k} and PAM-n, taking $I(t_{j_k}|s_{i_k}, M^n) = -\log(\Pr(t_{j_k}|s_{i_k}, M^n)) = -\log(M^n(t_{j_k}, s_{i_k}))$ bits. Thus, over all the set of matches defined by $\mathcal{A}_{\mathtt{m}}$, the length $\mathrm{I}(\langle \mathbf{S}, \mathbf{T} \rangle_{\mathtt{m}} | \mathcal{A}_{\mathtt{m}}, M^n)$ using this machine is:

$$\mathrm{I}(\langle \mathbf{S}, \mathbf{T} \rangle_{\mathtt{m}} | \mathcal{A}_{\mathtt{m}}, M^n) \;=\; \sum_{k=1}^{|\mathcal{A}_{\mathtt{m}}|} -\log(\Pr(s_{i_k}, t_{j_k}|M^n)) = \sum_{k=1}^{|\mathcal{A}_{\mathtt{m}}|} I_{\text{null}}(s_{i_k}) + I(t_{j_k}|s_{i_k}, M^n). \quad (5)$$

This approach is asymmetric, as swapping the order of sequences from $\langle \mathbf{S}, \mathbf{T} \rangle$ to $\langle \mathbf{T}, \mathbf{S} \rangle$ can yield (slightly) different encoding lengths. This is because PAM-n transition probability matrix, as we noted earlier, is not symmetric.

(ii) Amino acid pair-generation (symmetric) machine: This approach overcomes the problem of asymmetry encountered in the mutation-generation machine by constructing the average of the (asymmetric) joint probabilities of $\Pr(s_{i_k}, t_{j_k}|M^n)$ and $\Pr(t_{j_k}, s_{i_k}|M^n)$ as $\Pr^{\text{avg}}(s_{i_k}, t_{j_k}|M^n) = (\Pr(s_{i_k}) \Pr(t_{j_k}|s_{i_k}, M^n) + \Pr(t_{j_k}) \Pr(s_{i_k}|t_{j_k}, M^n))/2$ that remains invariant to the order of evaluation of the two sequences. Thus:

$$\mathrm{I}(\langle \mathbf{S}, \mathbf{T} \rangle_{\mathtt{m}} | \mathcal{A}_{\mathtt{m}}, M^n) = \sum_{k=1}^{|\mathcal{A}_{\mathtt{m}}|} -\log(\overset{\text{avg}}{\Pr}(s_{i_k}, t_{j_k}|M^n)) = \sum_{k=1}^{|\mathcal{A}_{\mathtt{m}}|} I(s_{i_k}, t_{j_k}|M^n) \qquad \text{bits.} \quad (6)$$

2.3.3 Estimating the optimal PAM-n (M^n) for a given alignment:

Given an alignment $\mathcal{A}$ over a sequence pair $\langle \mathbf{S}, \mathbf{T} \rangle$, we want to select the best n of PAM-n (M^n) such that, $\mathrm{I}(\langle \mathbf{S}, \mathbf{T} \rangle_{\mathtt{m}} | \mathcal{A}_{\mathtt{m}}, M^n)$ computed using either Equation 5 or Equation 6, is minimised. Here, we restrict n to be positive integers, and attempt to find the extremum of $\mathrm{I}(\langle \mathbf{S}, \mathbf{T} \rangle_{\mathtt{m}} | \mathcal{A}_{\mathtt{m}}, M^n)$ using an iterative quaternary search algorithm over the domain $lo = 1 \leq n \leq 1000 = hi$. For a *fixed* set of matches $\mathcal{A}_{\mathtt{m}} \subseteq \mathcal{A}$, in each iteration, this algorithm truncates the search domain $[lo \ldots hi]$ to $[lo + \lfloor \frac{(hi-lo)}{4} \rfloor \ldots hi]$ or $[lo \ldots hi - \lfloor \frac{(hi-lo)}{4} \rfloor]$ by removing either the first or the last quarter, after evaluating $\mathrm{I}(\langle \mathbf{S}, \mathbf{T} \rangle_{\mathtt{m}} | \mathcal{A}_{\mathtt{m}}, M^n)$ at those two quarter points. The value of lo upon termination is taken to be the optimal estimate of n. This computation takes $O(|\mathcal{A}_{\mathtt{m}}|)$ time and the overall computation of $\mathrm{I}(\langle \mathbf{S}, \mathbf{T} \rangle | \mathcal{A})$ takes $O(|\mathcal{A}|)$ time.

2.4 Search for an alignment that minimises $\mathrm{I}(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$

Equation 2 in Section 1 gives the length of the two-part message, $\mathrm{I}(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$, using the alignment model. The objective now is to find an alignment $\mathcal{A}^*$, and its associated parameters $(\vec{\Theta}^{\mathrm{MML}}_{\mathcal{A}^*}, M^{n^*})$, that minimises $\mathrm{I}(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$. First, a dynamic programming strategy to find an optimal alignment for any fixed set of parameters is discussed, followed by an Expectation-Maximization method to find the optimal across the parameter space.

2.4.1 Dynamic programming algorithm (DPA)

Substituting Equation 4 in Equation 2 gives the objective function: $\mathrm{I}(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle) = I(|\mathcal{A}|) + \mathrm{I}(\vec{\Theta}^{\mathrm{MML}}_{\mathcal{A}}) + \mathrm{I}(\mathcal{A}|\vec{\Theta}^{\mathrm{MML}}_{\mathcal{A}}) + \mathrm{I}(\langle \mathbf{S}, \mathbf{T} \rangle | \mathcal{A})$. This objective function is sub-additive, especially due to the two terms $I(|\mathcal{A}|)$ and $\mathrm{I}(\vec{\Theta}^{\mathrm{MML}}_{\mathcal{A}})$. To permit the application of a DPA, we move these terms outside the objective, and optimise the rest:

$$\mathrm{I}(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle) = \underbrace{I(|\mathcal{A}|) + \mathrm{I}(\vec{\Theta}^{\mathrm{MML}}_{\mathcal{A}})}_{\text{overlooked during DPA}} + \underbrace{\mathrm{I}(\mathcal{A}|\vec{\Theta}^{\mathrm{MML}}_{\mathcal{A}}) + \mathrm{I}(\langle \mathbf{S}, \mathbf{T} \rangle | \mathcal{A})}_{\text{optimised during DPA}}. \tag{7}$$

This is justifiable because, for closely competing alignment hypotheses, $I(|\mathcal{A}|) + \mathrm{I}(\vec{\Theta}^{\mathrm{MML}}_{\mathcal{A}})$ yields similar values, and hence can be treated as constant during the DPA, whose precise values can be added at the end. Moreover, for the terms involved in the DPA, we have $\mathrm{I}(\mathcal{A}|\vec{\Theta}^{\mathrm{MML}}_{\mathcal{A}}) + \mathrm{I}(\langle \mathbf{S}, \mathbf{T} \rangle | \mathcal{A}) \gg I(|\mathcal{A}|) + \mathrm{I}(\vec{\Theta}^{\mathrm{MML}}_{\mathcal{A}})$.

Based on this, we can implement a DPA similar to the one proposed by Gotoh [14]. Our approach takes $O(|\mathbf{S}||\mathbf{T}|)$ time, at the expense of storing three DPA *history matrices*, denoted as $\mathrm{Hist}_{\mathtt{m}}$, $\mathrm{Hist}_{\mathtt{i}}$ and $\mathrm{Hist}_{\mathtt{d}}$. Any $\mathrm{Hist}_{\mathtt{m}}(\mathtt{i},\mathtt{j})$ stores the optimal alignment derived for the prefixes $[s_1 \ldots s_i : t_1 \ldots t_j]$ ending in the '$\mathtt{m}$' state. Similarly, $\mathrm{Hist}_{\mathtt{i}}(\mathtt{i},\mathtt{j})$ and $\mathrm{Hist}_{\mathtt{d}}(\mathtt{i},\mathtt{j})$ store the alignments of those prefixes ending in the '$\mathtt{i}$' and the '$\mathtt{d}$' state respectively. Fig. 2 gives the DPA recurrences on these 3 matrices using any *fixed* set of alignment parameters, $(\vec{\Theta}, M^n)$. $\mathrm{I}(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$ is computed as $\min(\mathrm{Hist}_{\mathtt{m}}(|\mathbf{S}|, |\mathbf{T}|), \mathrm{Hist}_{\mathtt{i}}(|\mathbf{S}|, |\mathbf{T}|), \mathrm{Hist}_{\mathtt{d}}(|\mathbf{S}|, |\mathbf{T}|)) + I(|\mathcal{A}|) + I(\vec{\Theta})$. $\mathcal{A}$ is derived by tracing back on these history matrices.

$$
\begin{aligned}
\mathrm{Hist}_{\mathtt{m}}(\mathtt{i,j}) &= \min \begin{cases} \mathrm{Hist}_{\mathtt{m}}(\mathtt{i\text{-}1,j\text{-}1}) + I_{\vec{\Theta}}(\mathtt{m}|\mathtt{m}) + I(s_i, t_j|M^n) \\ \mathrm{Hist}_{\mathtt{i}}(\mathtt{i\text{-}1,j\text{-}1}) + I_{\vec{\Theta}}(\mathtt{m}|\mathtt{i}) + I(s_i, t_j|M^n) \\ \mathrm{Hist}_{\mathtt{d}}(\mathtt{i\text{-}1,j\text{-}1}) + I_{\vec{\Theta}}(\mathtt{m}|\mathtt{d}) + I(s_i, t_j|M^n) \end{cases} \\
\mathrm{Hist}_{\mathtt{i}}(\mathtt{i,j}) &= \min \begin{cases} \mathrm{Hist}_{\mathtt{m}}(\mathtt{i,j\text{-}1}) + I_{\vec{\Theta}}(\mathtt{i}|\mathtt{m}) + I_{\mathrm{null}}(t_j) \\ \mathrm{Hist}_{\mathtt{i}}(\mathtt{i,j\text{-}1}) + I_{\vec{\Theta}}(\mathtt{i}|\mathtt{i}) + I_{\mathrm{null}}(t_j) \\ \mathrm{Hist}_{\mathtt{d}}(\mathtt{i,j\text{-}1}) + I_{\vec{\Theta}}(\mathtt{i}|\mathtt{d}) + I_{\mathrm{null}}(t_j) \end{cases} \\
\mathrm{Hist}_{\mathtt{d}}(\mathtt{i,j}) &= \min \begin{cases} \mathrm{Hist}_{\mathtt{m}}(\mathtt{i\text{-}1,j}) + I_{\vec{\Theta}}(\mathtt{d}|\mathtt{m}) + I_{\mathrm{null}}(s_i) \\ \mathrm{Hist}_{\mathtt{i}}(\mathtt{i\text{-}1,j}) + I_{\vec{\Theta}}(\mathtt{d}|\mathtt{i}) + I_{\mathrm{null}}(s_i) \\ \mathrm{Hist}_{\mathtt{d}}(\mathtt{i\text{-}1,j}) + I_{\vec{\Theta}}(\mathtt{d}|\mathtt{d}) + I_{\mathrm{null}}(s_i) \end{cases}
\end{aligned}
$$

Figure 2: DPA recurrences for optimizing $\mathrm{I}(\mathcal{A}|\vec{\Theta}_{\mathcal{A}}^{\mathrm{MML}}) + \mathrm{I}(\langle \mathbf{S}, \mathbf{T} \rangle | \mathcal{A})$ (see Equation 7).

2.4.2 Expectation-Maximization for optimal values of $(\vec{\Theta}, M^n)$

We use the Expectation-Maximization (EM) strategy to iteratively optimise parameters $(\vec{\Theta}, M^n)$ that minimise $\mathrm{I}(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$ across the parameter space. In the *expectation step* (E-step), we run the DPA described in Section 2.4.1 for some fixed set of parameters $(\vec{\Theta}, M^n)$. This yields an alignment $\mathcal{A}$. In the *maximization step* (M-step) we maximise these parameters, keeping the $\mathcal{A}$ fixed (Section 2.2 and Section 2.3.3). This updates the parameters, and the process is repeated until convergence (alignment does not change). That is, using an initial set of fixed parameters $(\vec{\Theta}_0, M^{n_0})$, E-step returns an alignment $\mathcal{A}_1$. The M-step updates these initial parameters to $(\vec{\Theta}_1, M^{n_1})$. In the next iteration using the updated parameters $(\vec{\Theta}_2, M^{n_2})$, E-step return $\mathcal{A}_2$, and M-step updates to $(\vec{\Theta}_3, M^{n_3})$ and so on, until $\mathcal{A}_{t-1} \equiv \mathcal{A}t$.

2.5 Visualizing the competing alignment landscapes

Existing sequence alignment methods output a single alignment, while ignoring a host of other closely-competing alignments. It is possible using this MML approach to visualise the whole landscape of alignments transitioning through every (`i,j`) cell in our DPA. This is achieved by running the EM strategy as discussed and memoizing the resulting history matrices with their associated alignment parameters. Then, the sequences are each *reversed* and the DPA is run on these *reversed sequences* using the remembered parameters inferred in the forward direction (while accounting for the fact that the DPA is now running in reverse). By consistently collating the DPA histories at each (`i,j`) from forward and backward directions of the sequences, we can visualise the best DPA transitioning through each (`i,j`) in the combined histories.

3 Results

We applied the method described in Section 2.1 to infer the MML null probability estimates on data downloaded from UniProt [11]. The full data contains 130,603 protein sequences with 59,441,063 amino acids. We also infer the same estimates separately using the proteomes of eight organisms covering the three distinct domains

Table 1: MML (order=0) estimates of amino acids codes from the extended alphabet, inferred using: All=whole UniProt, and full proteomes of: HS=*H. sapiens*; EC=*E. coli*; PW=*P. wallikeri*; AT=*A. thaliana*; MM=*M. musculus*; DM=*D. melanogaster*; SC=*S. cerevisiae*; MJ=*M. jannaschii*.

	A	B	C	D	E	F	G	H	I	K	L	M	N
All	7.3e-02	1.4e-07	1.7e-02	5.3e-02	6.7e-02	3.9e-02	6.5e-02	2.4e-02	5.4e-02	5.9e-02	9.6e-02	2.3e-02	4.3e-02
HS	7.0e-02	4.5e-08	2.3e-02	4.8e-02	7.1e-02	3.7e-02	6.6e-02	2.6e-02	4.4e-02	5.7e-02	1.0e-01	2.1e-02	3.6e-02
EC	9.7e-02	4.2e-07	1.1e-02	5.1e-02	5.7e-02	3.9e-02	7.5e-02	2.3e-02	6.0e-02	4.4e-02	1.1e-01	2.8e-02	3.8e-02
PW	4.2e-02	1.4e-06	2.1e-02	5.6e-02	7.4e-02	4.4e-02	5.6e-02	2.3e-02	7.8e-02	9.9e-02	8.3e-02	2.2e-02	7.9e-02
AT	6.6e-02	7.6e-08	1.8e-02	5.3e-02	6.6e-02	4.3e-02	6.6e-02	2.2e-02	5.4e-02	6.2e-02	9.6e-02	2.5e-02	4.4e-02
MM	7.0e-02	1.6e-07	2.2e-02	4.9e-02	7.0e-02	3.7e-02	6.5e-02	2.6e-02	4.3e-02	5.6e-02	1.0e-01	2.2e-02	3.6e-02
FF	7.4e-02	2.4e-07	1.8e-02	5.3e-02	6.5e-02	3.6e-02	6.2e-02	2.7e-02	5.0e-02	5.7e-02	9.2e-02	2.4e-02	4.7e-02
SC	5.5e-02	1.8e-07	1.2e-02	5.9e-02	6.6e-02	4.4e-02	5.0e-02	2.2e-02	6.6e-02	7.3e-02	9.5e-02	2.1e-02	6.2e-02
MJ	5.9e-02	1.5e-06	1.3e-02	5.6e-02	8.8e-02	4.0e-02	6.7e-02	1.5e-02	1.0e-01	1.0e-01	9.3e-02	2.3e-02	4.9e-02

Continued...

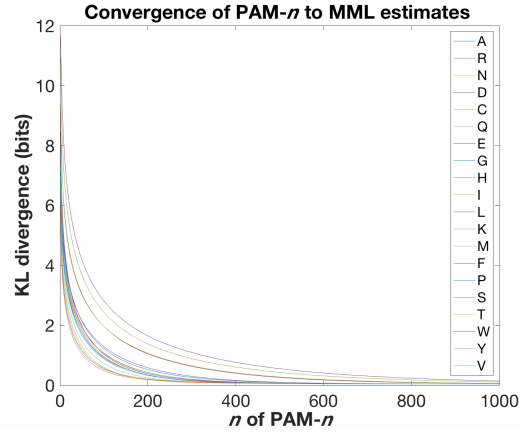
	O	P	Q	R	S	T	U	V	W	X	Y	Z
All	1.9e-07	5.3e-02	4.2e-02	5.3e-02	7.9e-02	5.5e-02	2.3e-06	6.4e-02	1.2e-02	1.4e-05	3.0e-02	4.2e-08
HS	4.5e-08	6.3e-02	4.8e-02	5.6e-02	8.3e-02	5.4e-02	3.4e-06	6.0e-02	1.2e-02	4.5e-08	2.7e-02	4.5e-08
EC	4.2e-07	4.5e-02	4.4e-02	5.5e-02	5.7e-02	5.4e-02	2.9e-06	7.2e-02	1.5e-02	1.2e-06	2.8e-02	4.2e-07
PW	1.4e-06	2.9e-02	2.8e-02	4.2e-02	7.2e-02	4.8e-02	1.4e-06	5.4e-02	6.9e-03	5.0e-05	4.4e-02	1.4e-06
AT	7.6e-08	4.7e-02	3.5e-02	5.3e-02	8.8e-02	5.1e-02	7.6e-08	6.8e-02	1.2e-02	7.6e-08	2.9e-02	7.6e-08
MM	5.3e-08	6.2e-02	4.8e-02	5.6e-02	8.4e-02	5.3e-02	3.6e-06	6.1e-02	1.2e-02	6.3e-06	2.7e-02	5.3e-08
FF	2.4e-07	5.3e-02	5.2e-02	5.5e-02	8.2e-02	5.5e-02	2.7e-06	5.9e-02	9.9e-03	3.7e-06	3.0e-02	2.4e-07
SC	1.8e-07	4.4e-02	4.0e-02	4.4e-02	9.0e-02	5.9e-02	1.8e-07	5.6e-02	1.0e-02	1.8e-07	3.4e-02	1.8e-07
MJ	1.5e-06	3.6e-02	1.5e-02	4.1e-02	4.3e-02	4.0e-02	2.3e-05	7.1e-02	6.8e-03	2.6e-05	4.1e-02	1.5e-06

of life: Eukaryota, Prokaryota and Archea. Table 1 gives the Markov order 0 MML estimates. (Order 1 and 2 Markov parameters are available on request.)

Among the parameters that our MML-based approach automatically infers when aligning any sequence pair is the n of PAM-n. Recall from Section 2.3.3 that PAM-n (M^n) is the amino acid transition probability matrix, giving the probability of one amino acid mutating into another in n PAM (evolutionary) steps. We also noted that as $\lim_{n\to\infty} M^n$, the matrix approaches the stationary (null probability) distribution. Table 2(a) shows the Kullback-Leibler divergence between the distribution of probabilities of each amino acid (i.e., each column of PAM-n) with our inferred MML-estimates of the null probabilities, for the values of $0 \leq n \leq 1000$. The convergence to the null distribution we inferred is evident from this plot.

To compare the effectiveness of our alignment method, we downloaded all the 630 pairwise alignments from the manually-curated HOMSTRAD database [15]. We realigned each sequence pair from this set using our MML-based inference algorithm, under both symmetric and asymmetric machine models (described in Section 2.3). The same pairs were also realigned using popular aligners, ClustalW2 [16] and MUSCLE [17], in pairwise alignment mode. Furthermore, we implemented Gotoh's algorithm [14] using any given scoring matrix under the affine-gap penalty function of the form $\gamma(l) = g_o + (l-1)g_e$, and ran this program using BLOSUM-{30,62,90} and PAM-{10,100,250} scoring matrices with the (g_o, g_e) penalty values of $(10, 1)$. The alignments generated by all these methods are compared in terms of the compression gained. To enable this comparison, for every alignment across various methods, we maximise the parameters using the M-step described in Section 2.4.2. This allows us to compute the compression gained for any given alignment of a pair of sequences as $NULL(\langle \mathbf{S}, \mathbf{T} \rangle) - \mathrm{I}(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$, using both the symmetric and asymmetric alignment models described in Section 2.3.

Table 2: (a) Kullback-Leibler divergence between each amino acid distribution under PAM-n (M^n) and the stationary (null) distribution of amino acids, for varying values of n. Our MML-based method infers the value of n and other alignment parameters automatically (see Section 2.4.2).
(b) Comparison of alignments using 630 HOMSTRAD datasets across various sequence aligners. The alignments generated by various programs are measured in terms of the amount of compression gained ($NULL(\langle \mathbf{S}, \mathbf{T} \rangle) - \mathrm{I}(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$) in bits under both the symmetric and asymmetric models described in Section 2.3. Presented are the 1st (Q1), 2nd (Q2=median) and 3rd (Q3) quartile statistics of compression of alignments generated by various methods. The cells highlighted in bold are the ones that yield maximum compression across all the methods considered.



(a)

	Compression (in bits)					
	Symmetric			Asymmetric		
	Q1	Q2	Q3	Q1	Q2	Q3
MML	**12.6**	**82.7**	**233.6**	**11.5**	**82.4**	**233.1**
ClustalW2	4.8	72.5	231.5	2.2	69.5	229.3
Muscle	5.8	77.5	232.2	4.9	75.1	230.3
Gotoh-BLOSUM-30	-10.5	66.6	221.6	-11.9	62.9	219.6
Gotoh-BLOSUM-62	1.8	74.7	232.2	-0.8	73.4	225.2
Gotoh-BLOSUM-90	-7.5	69.9	225.0	-9.8	66.5	222.9
Gotoh-PAM-10	-35.5	15.1	177.1	-35.3	14.5	175.9
Gotoh-PAM-100	-3.5	71.3	227.4	-5.5	70.3	224.2
Gotoh-PAM-250	3.7	75.5	228.1	2.2	72.9	227.6

(b)

Table 2(b) presents the quartile statistics of compression, using both symmetric and asymmetric models, over all the 630 alignments generated by various methods. This table shows that the alignments inferred using our MML-based approach yields most compression, consistently across all quartile marks.

Our MML approach is distinguished from other methods not only because it is fully unsupervised in its inference of the evolutionary distance (n of PAM-n) and other alignment parameters, but also for its ability to provide powerful visualizations of competing alignment hypotheses in the form of alignment landscapes. Such landscape plots (see Fig. 3) can visually show the relationship between the optimal alignment and other closely-competing alignments in the same context.

Funding: DS's PhD scholarship is funded by Monash University. AK's research is funded by Australian Research Council's Discovery Project (DP150100894).

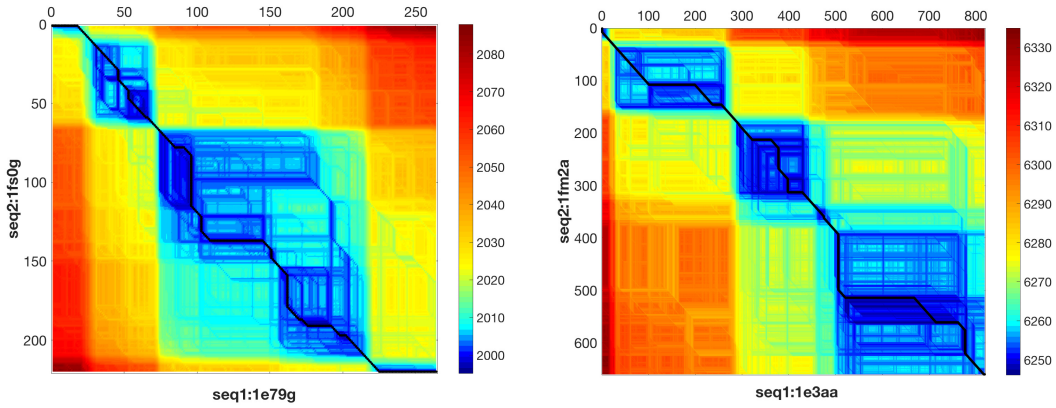
4 References

[1] MO Dayhoff, RM Schwartz, and BC Orcutt, "A model of evolutionary change in proteins," in *Atlas of protein sequence and structure*, vol. 5, pp. 345–352. National Biomedical Research Foundation Silver Spring, MD, 1978.

[2] WR Pearson, "Selecting the right similarity-scoring matrix," *Current protocols in bioinformatics*, pp. 3–5, 2013.

[3] S Henikoff and JG Henikoff, "Amino acid substitution matrices from protein blocks," *Proc Natl Acad Sci USA*, vol. 89, no. 22, pp. 10915–10919, 1992.

[4] CS Wallace, *Statistical and Inductive Inference using Minimum Message Length*, Information Science and Statistics. SpringerVerlag, 2005.



(a) (b)

Figure 3: Example visualizations of competing alignment landscapes. The color map ranges from dark Blue (= smaller $I(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$ alignments) to dark Red (=longer $I(\mathcal{A}, \langle \mathbf{S}, \mathbf{T} \rangle)$ alignments). (a) Competing alignment landscape of amino acid sequences of γ subunits of ATP Synthases from *Bos Taurus* (PDBID: `1E79`, chain G) and *Escherichia Coli* (PDBID: `1FS0`, chain G). (b) Landscape plot for the protein sequences of cephalosporin acylase from *Brevundimonas diminuta* (PDBID: `1FM2` chain A) and penicillin acylase from *Escherichia coli* (PDBID: `1E3A` chain A).

[5] CE Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 1948.

[6] L Allison, CS Wallace, and CN Yee, "Finite-state models in the alignment of macromolecules," *Journal of Molecular Evolution*, vol. 35, no. 1, pp. 77–89, 1992.

[7] CN. Yee and L Allison, "Reconstruction of strings past," *Bioinformatics*, vol. 9, no. 1, pp. 1–7, 1993.

[8] DR Powell, L Allison, and TI Dix, "Modelling-alignment for non-random sequences.," in *Australian Conference on Artificial Intelligence*. Springer, 2004, pp. 203–214.

[9] L Allison, *Coding Ockham's Razor*, Springer, 2018 (to appear).

[10] Chris S Wallace and Peter R Freeman, "Estimation and inference by compact coding," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 240–265, 1987.

[11] UniProt Consortium et al., "Uniprot: the universal protein knowledgebase," *Nucleic acids research*, vol. 45, no. D1, pp. D158–D169, 2017.

[12] CS Wallace and JD Patrick, "Coding decision trees," *Machine Learning*, vol. 11, no. 1, pp. 7–22, 1993.

[13] JH Conway and NJA Sloane, *Sphere packings, lattices and groups*, vol. 290, Springer Science & Business Media, 2013.

[14] O Gotoh, "An improved algorithm for matching biological sequences," *Journal of molecular biology*, vol. 162, no. 3, pp. 705–708, 1982.

[15] K Mizuguchi, CM Deane, TL Blundell, and JP Overington, "HOMSTRAD: a database of protein structure alignments for homologous families," *Protein science*, vol. 7, no. 11, pp. 2469–2471, 1998.

[16] MA et al. Larkin, "Clustal W and Clustal X version 2.0," *Bioinformatics*, vol. 23, no. 21, pp. 2947–2948, 2007.

[17] RC Edgar, "MUSCLE: multiple sequence alignment with high accuracy and high throughput," *Nucleic acids research*, vol. 32, no. 5, pp. 1792–1797, 2004.