Fast, accurate construction of multiple sequence alignments from protein language embeddings

Minh Hoang[1[0000−0002−1053−6744]†], Isabel Armour-Garb[1,2[0000−0002−6332−3750]†], and Mona Singh[1,2]

[1] Lewis-Sigler Institute of Integrative Genomics, Princeton University, Princeton, NJ 08544, USA
[2] Department of Computer Science, Princeton University, Princeton, NJ 08544, USA
[†]These authors contributed equally.
minhhoang@princeton.edu, iarmourgarb@princeton.edu, mona@cs.princeton.edu

Abstract. Multiple sequence alignment (MSA) is a foundational task in computational biology, underpinning protein structure prediction, evolutionary analysis, and domain annotation. Traditional MSA algorithms rely on pairwise amino acid substitution matrices derived from conserved protein families. While effective for aligning closely related sequences, these scoring schemes struggle in the low-identity "twilight zone." Here, we present a new approach for constructing MSAs leveraging amino acid embeddings generated by protein language models (PLMs), which capture rich evolutionary and contextual information from massive and diverse sequence datasets. We introduce a windowed reciprocal-weighted embedding similarity metric that is surprisingly effective in identifying corresponding amino acids across sequences. Building on this metric, we develop ARIES (**A**lignment via **R**ec**I**procal **E**mbedding **S**imilarity), an algorithm that constructs a PLM-generated template embedding and aligns each sequence to this template via dynamic time warping in order to build a global MSA. Across diverse benchmark datasets, ARIES achieves higher accuracies than existing state-of-the-art approaches, especially in low-identity regimes where traditional methods degrade, while scaling almost linearly with the number of sequences to be aligned. Together, these results provide the first large-scale demonstration of the power of PLMs for accurate and scalable MSA construction across protein families of varying sizes and levels of similarity, highlighting the potential of PLMs to transform comparative sequence analysis.

Code availability: `https://github.com/Singh-Lab/ARIES`

Keywords: Protein Language Model; Deep Learning; Multiple Sequence Alignment

1 Introduction

Comparative analysis of protein sequences provides fundamental insights into their structure, function, and evolution. A central tool for such analysis is multiple sequence alignment (MSA), which arranges evolutionarily-related residues across homologous sequences to reveal conserved positions and motifs. The quality of an MSA directly influences a wide range of downstream applications, including protein structure prediction [32], phylogenetic reconstruction [56], and functional annotation [9]. In modern structure prediction systems such as AlphaFold [21], co-variation patterns extracted from MSAs provide essential constraints for learning residue–residue contacts and 3D geometry, making accurate alignments a prerequisite for many advances in computational proteomics.

Over the years, numerous MSA algorithms have been developed (e.g., [53, 23, 12, 47, 28, 34]). Most of these approaches follow a progressive alignment strategy where the most similar sequences are aligned first, and more distant sequences are iteratively added according to a guide tree. Alignment scores are typically computed using substitution matrices such as PAM [10] and BLOSUM [15], which are comprised of log-odds scores reflecting how often one residue substitutes for another [3], as estimated from conserved protein families. MSAs built using these matrices are highly accurate when sequence identity is high, but lose reliability in the low sequence identity "twilight zone" of alignment [43]. A major limitation stems from the context-independent nature of traditional substitution matrices, which assign each residue the same fixed similarity score for a given substitution, regardless of its biochemical or structural environment within the sequence.

The advent of protein language models (PLMs) [6, 1, 40, 7, 14, 42] offers a transformative opportunity to overcome this limitation. PLMs such as ESM-2 [27] and ProtT5 [14] are trained on massive protein sequence databases to learn context-aware embeddings that implicitly capture underlying evolutionary relationships. PLM embeddings have been used for BLAST-like database searches, where homologs are identified based on sequence-level embedding similarity [45, 25, 22, 19]. In recent years, several studies have extended the use of PLM embeddings to construct MSAs, with varying degrees of success.

The first such approach, vcMSA [33], employs a clustering-based strategy to recursively assemble aligned columns from embedding representations, but its accuracy and scalability remain limited for large sequence sets. Meanwhile, methods such as EBA [38] and PEbA [20] incorporate embedding-based similarity measures directly into classical pairwise alignment dynamic programming frameworks, but they lack mechanisms to reconstruct full MSAs. More recently, learnMSA2 [5] integrates PLM embeddings into a hidden Markov model-based framework for aligning large, well-conserved families. However, its statistical estimates become unstable for small or highly divergent sets, leading to substantial drops in accuracy. Collectively, these PLM-based methods have shown that sequences can be aligned via PLM-derived substitution scores, but none have yet consistently and efficiently yielded high-quality alignments across both small and large protein families.

In this paper, we present a robust and scalable PLM-based technique for constructing MSAs and demonstrate its excellent performance across diverse benchmarks of varying sizes and degrees of heterogeneity. Our approach introduces three key innovations: (1) a context-aware, reciprocal similarity measure that enhances the robustness of residue comparisons; (2) a highly scalable two-phase strategy, based on "star" alignment [2], that builds MSAs from pairwise alignments with a template sequence without requiring explicit gap penalty functions; and (3) a template-construction procedure that leverages PLMs to generate a well-informed central sequence for this star alignment strategy.

Our similarity measure is motivated by the observation that prior residue-level PLM-derived similarities [33, 20] can struggle to identify aligned residue pairs within low-identity regions of the sequences. To reduce sensitivity to local context perturbations, we incorporate a windowing step that aggregates similarity over the respective local windows of the residue pair being compared. To further distinguish true evolutionary correspondences from superficial contextual matches, we introduce a reciprocal weighting mechanism that prioritizes mutually recognized residue pairs over asymmetric matches. This reciprocality reward sharpens the alignment signal and proves effective in the low-identity regime (Section 3.1 and 3.4).

Using our similarity measure, we leverage dynamic time warping (DTW) as the core alignment primitive in our workflow [44]. DTW offers a key advantage over traditional alignment algorithms: it naturally handles insertions and deletions without requiring explicit gap penalties, instead representing them as many-to-one or

one-to-many mappings along the alignment path. This feature is especially useful in the context of PLM-based alignment, since gap embeddings cannot be generated without knowing their locations in advance. Building on this primitive, we construct MSAs of protein families based on a star alignment framework (Fig. 1, top), where each protein is pairwise-aligned against a representative template sequence. The resulting pairwise alignments are merged into a global MSA by grouping residues that map to the same template position, with many-to-one cases disambiguated by selecting the correspondences with highest embedding similarities.

For extremely large sequence sets, however, multiple subgroups (or clusters) of highly similar sequences are often observed, and simply selecting a single sequence from the input set favors certain subgroups, and hence degrades overall performance (Section 3.4). To overcome this shortcoming of classical star alignment [2], we employ PLMs to synthesize a balanced and informative template that incorporates features from the top-K "medoid" sequences. Specifically, in the first phase of our alignment approach, we identify the K sequences that are the closest to all other sequences, and then perform a miniature star alignment, using the global medoid as an initial template and obtaining a coarse alignment for these K sequences (Fig. 1, bottom). In the second phase, we then re-embed the aligned medoid sequences individually with a PLM and compute the position-wise average of their embeddings to generate a synthesized template that ideally captures shared evolutionary signals across subgroups. This refined template is subsequently used to align all sequences in the input set.

We evaluate our PLM-based alignment framework, ARIES (**A**lignment via **R**ec**I**procal **E**mbedding **S**imilarity), on three benchmark datasets: HOMSTRAD [51], QuanTest2 [46], and BAliBASE 3.0 [54], which span moderate to low sequence identity ranges and range in size from 2 to 1000 sequences. Compared with leading MSA tools [53, 47, 24, 13, 37, 11, 12, 5, 33, 49, 55], including Clustal Omega, MAFFT, MUSCLE, T-Coffee, MAGUS, TWILIGHT, and PLM-based baselines such as learnMSA2 and vcMSA, ARIES consistently achieves higher alignment accuracies. The greatest improvements are for low-sequence identity sequence sets, which constitute the cases where traditional algorithms struggle. Furthermore, ARIES scales nearly linearly with the number of sequences, making it suitable for modern large-scale analyses. Altogether, our results establish that PLM-based alignment offers a powerful, scalable, and accurate alternative to traditional substitution-matrix-based methods, bridging the gap between deep learning representations and classical sequence analysis.

2 Methodology

Overview. ARIES builds on the principle of embedding-based pairwise alignment. Amino acid embeddings are obtained for each sequence (Section 2.1), and pairwise correspondences between residues are computed using their contextual similarity in the learned embedding space of PLMs (Section 2.2). Pairs of sequences are aligned using the DTW algorithm (Section 2.3). To scale this pairwise procedure to multiple sequences, we adopt a two-phase star alignment MSA building approach (Section 2.4, Fig. 1), which first generates a template sequence (Section 2.4.2), and then aligns each remaining sequence to this template using DTW. From these pairwise alignments, we reconstruct the global MSA by iteratively building column blocks centered around each template residue (Section 2.4.1). As DTW can produce many-to-one mappings, we further introduce a disambiguation step to infer residue correspondences and consistent gap placements (Section 2.4.1).

2.1 Embedding generation

We obtain embeddings of protein sequences using PLMs such as ESM-2 [27] and ProtT5 [14]. Given an amino acid sequence $(a_1, a_2, \ldots, a_L)$, the PLM produces a set of hidden states $\left\{Z^{(i)} = \left(z_1^{(i)}, z_2^{(i)}, \ldots, z_L^{(i)}\right)\right\}_{i=1}^{\ell_{\max}}$, where $\ell_{\max}$ is the number of layers in the PLM and $z_j^{(i)} \in \mathbb{R}^h$ denotes the embedding of a_j at the layer i. By convention, $Z^{(1)}$ denotes the output of the final layer. Special tokens added by the PLM (e.g., CLS or EOS) are not considered in our framework. Since different layers of PLMs capture complementary structural and functional information [30], we represent each residue a_j as the concatenation of its hidden embeddings from the last ℓ layers: $x_j = [z_j^{(1)} z_j^{(2)} \ldots z_j^{(\ell)}]$, resulting in a vector with $d \triangleq \ell h$ dimensions. Details of PLMs used in our experiments are given in Appendix C, and the effect of varying ℓ is evaluated in Section 3.4.

2.2 Reciprocal-weighted similarity score between windows of PLM embeddings

Unlike traditional substitution matrices such as BLOSUM [15] or PAM [10], PLM embeddings capture dependencies between residues through self-attention pre-training, which enables them to reflect subtle contextual

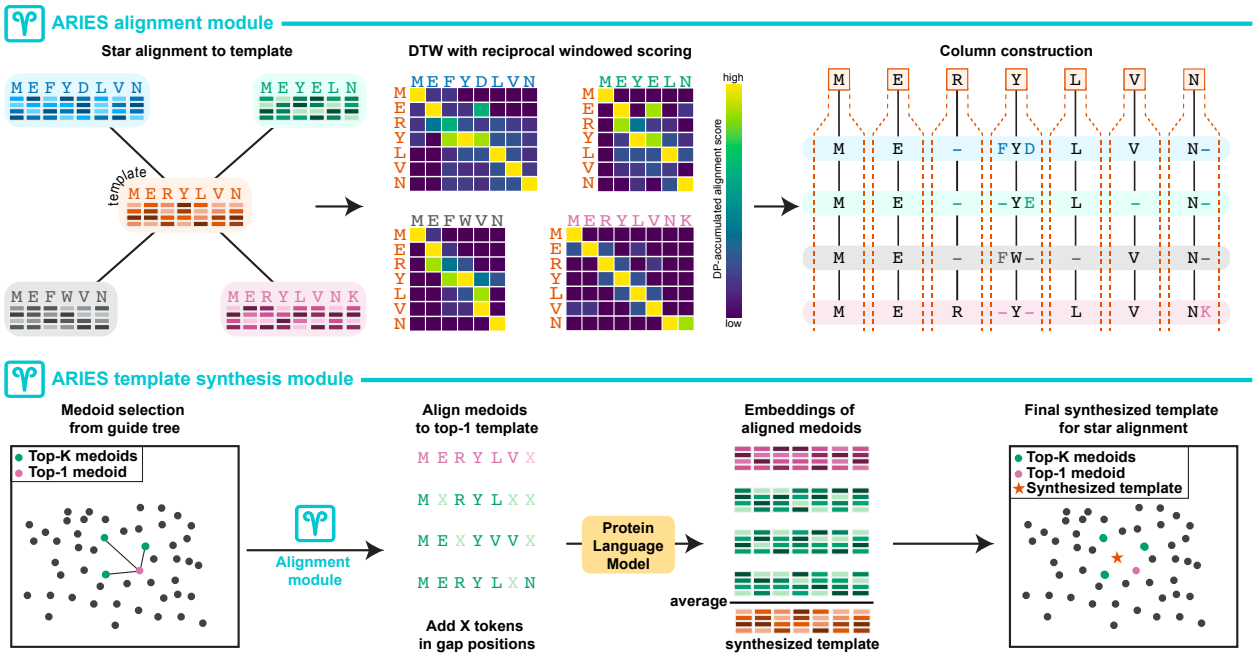

Fig. 1. The algorithmic core of ARIES consists of two modules, the *star-alignment* module (top) and the *template synthesis* module (bottom). In the alignment module, all sequences are aligned to a single template via dynamic time warping (DTW) using a reciprocal, window-based embedding similarity score. Per-template-position column blocks are constructed by selecting anchor residues (denoted by vertical black lines), placing remaining residues around these anchors, and adding gaps to form coherent alignment columns. In the template synthesis module, the K sequences most similar to all other sequences (i.e., the top-K medoids) are aligned using the ARIES alignment module with the top-1 medoid as the template. Gaps in the alignment are replaced with 'X' tokens, and the sequences with 'X' tokens are individually re-embedded with the PLM. A synthesized consensus template is produced by position-wise averaging of the resulting embeddings across all K sequences, including the top-1 medoid.

information such as residue co-evolution, local structure, and domain-specific motifs [30]. Deriving similarity scores directly from these embeddings thus allows the alignment process to operate in a semantically meaningful representation space, where geometric proximity reflects functional or structural correspondence.

Let $A = (a_1, a_2, \ldots, a_{L_1})$ and $B = (b_1, b_2, \ldots, b_{L_2})$ be two protein sequences with their respective PLM embeddings $X = (x_1, x_2, \ldots, x_{L_1}) \in \mathbb{R}^{L_1 \times d}$ and $Y = (y_1, y_2, \ldots, y_{L_2}) \in \mathbb{R}^{L_2 \times d}$. Their residue-level similarity can be encoded as a matrix $S \in \mathbb{R}^{L_1 \times L_2}$, where each entry S_{ij} quantifies the similarity between the pair of amino acids a_i and b_j. We can measure this similarity using the negative Euclidean distance (NED) of their embeddings, i.e., $S_{ij} \triangleq -\|x_i - y_j\|_2$. While the NED metric can by itself, with no alignment step, help identify amino acid pairs across sequences that should be aligned (see Section 3.1), to obtain a more stable and discriminative similarity function, we extend NED in two ways: first by aggregating information over local windows of amino acids, and second by applying a reciprocal weighting scheme that prioritizes mutually consistent matches. These augmentations are described in what follows.

Window-based similarity measure. Since the NED metric can sometimes lead to high similarities between amino acid pairs that should not be aligned, we first generalize NED to capture similarity between windows of embeddings, as window-based similarity has been shown to improve precision in prior work [17]. For some choice of window span w, we encode each residue a_i in A by concatenating all embeddings in the $(2w+1)$-length context window centered around it, i.e., $\mathbf{x}_i^w \triangleq (x_{i-w}, \ldots, x_i, \ldots, x_{i+w})$. Likewise, each residue b_j in B is encoded by $\mathbf{y}_j^w \triangleq (y_{j-w}, \ldots, y_j, \ldots, y_{j+w})$. To ensure that each residue is associated with a sufficiently large context window, we pad both ends of the original sequence with w contiguous "unknown amino acid" tokens, represented by the token "X" in most PLM vocabularies. The window-level similarity matrix between A and B is then given by the matrix $W \in \mathbb{R}^{L_1 \times L_2}$, where each entry W_{ij} is obtained as a weighted aggregation over corresponding token-pair similarities between $\mathbf{x}_i^w$ and $\mathbf{y}_j^w$. Specifically, we define:

$$W_{ij} \triangleq -\sum_{t=-w}^{w} \gamma_t \|x_{i+t} - y_{j+t}\|_2 \ , \tag{1}$$

where $\gamma_t \triangleq \exp\left(-\frac{t^2}{2\sigma^2}\right)$ follows a discrete Gaussian kernel centered at zero, assigning greatest importance to the central residue and progressively discounting the contributions of neighboring positions.

Reciprocal-weighted similarity measure. While window averaging reduces noise, contextual differences across the embedding space of X and Y can give rise to bias in their pairwise similarity scores. In practice, we observe that some residues produce broad, non-specific similarity patterns, showing moderate similarity to many positions in the other sequence. Conversely, other residues align strongly to only a few positions. Such asymmetry can mislead the alignment process, as residues with non-specific similarities may dominate the scoring matrix and cause incorrect matches. To counteract this phenomenon, we further introduce a reciprocal-weighted similarity term that adjusts entries in W to reward mutually consistent correspondences.

The mutual consistency of two residues a_i and b_j can be assessed by weighing their alignment signal W_{ij} against other entries of W in the same row and column. Specifically, we compute the softmax normalizations of W_{ij} along row i and column j of W:

$$U_{ij} = \frac{\exp(W_{ij})}{\sum_{k=1}^{L_2} \exp(W_{ik})} \qquad \text{and} \qquad V_{ij} = \frac{\exp(W_{ij})}{\sum_{k=1}^{L_1} \exp(W_{kj})} \ . \tag{2}$$

Here, U_{ij} represents how strongly a_i favors b_j over other residues in B, and likewise V_{ij} reflects the reciprocal preference in the opposite direction. If both perspectives agree, i.e., each residue strongly favors the other compared to all alternatives, then the alignment is considered mutually consistent. Building on this intuition, we define the *reciprocal consistency score* as $R_{ij} \triangleq 0.5 \times \log(U_{ij}V_{ij})$, which corresponds to the log of the geometric mean between the two directional affinities. We note that the log transformation guarantees that R_{ij} is negative, with R_{ij} becoming more negative with decreasing specificity. Since W_{ij} is also negative and becomes more so when amino acids are less similar to each other in embedding space, the W and R matrices can naturally be combined via addition. Our proposed pairwise similarity matrix is thus defined as $S = W + \lambda R$, where the hyperparameter λ controls the strength of reciprocal reinforcement. The effect of the λ hyperparameter value is further investigated in Section 3.4.

2.3 Dynamic Programming for Pairwise Sequence Alignment

Dynamic programming (DP) forms the computational backbone of most sequence alignment algorithms, with the Needleman-Wunsch algorithm [36] finding the best global alignment of two pairs of sequences given a substitution matrix and gap penalty score. Most traditional MSA approaches use heuristic approaches to assign gap penalty scores, though some learn position-specific gap penalties directly from sequence data (e.g., the HHalign method [41] used by Clustal Omega [47]). In the embedding-based alignment paradigm, since embeddings for gaps cannot be generated without prior knowledge of gap locations, it is not clear how to set gap penalty scores. Indeed, existing embedding-based alignment approaches either avoid explicit pairwise alignment altogether (e.g., vcMSA [33] and learnMSA2 [5]) or depend on ad-hoc gap penalty schemes that do not generalize well across diverse alignment scenarios (e.g., PeBA [20]). To address this challenge, we adopt the DTW formulation [44] that is frequently used in signal processing domains to align time series data. The DTW framework aligns two time series by allowing local stretching and compression along the time axis to maximize the overall correspondence between aligned time points. When applied to biological sequence alignment, this relaxes the strict one-to-one or one-to-gap constraint of classical alignment methods, yielding gap-free alignment maps in which a single residue may correspond to multiple contiguous residues in the other sequence. In this work, we show that gap placement can subsequently be inferred through a post-hoc refinement procedure. The DP recursion underlying DTW is described below.

Definition 1 (Dynamic Time Warping on Protein Embedding Space). *Given embeddings $X = (x_1, x_2, \ldots, x_{L_1})$ and $Y = (y_1, y_2, \ldots, y_{L_2})$, and a pairwise similarity matrix $S \in \mathbb{R}^{L_1 \times L_2}$, DTW constructs an optimal alignment path $P_* = \{(i_k, j_k)\}_{k=1}^{K}$ of some arbitrary length K that maximizes the total path similarity score. That is, $P_* = \arg\max_P \sum_{(i,j)\in P} S_{ij}$.*

A valid alignment path is constrained to (1) start and end at the sequence boundaries, i.e., $(i_1, j_1) = (1, 1)$*,* $(i_K, j_K) = (L_1, L_2)$*, and (2) advance one step at a time in either sequence, i.e.,* $(i_{k+1} - i_k, j_{k+1} - j_k) \in \{(0, 1), (1, 0), (1, 1)\}$*. The optimal cumulative alignment score up to residue* i *of* X *and residue* j *of* Y *is then computed via the following DP recursion:*

$$D(i, j) = S_{ij} + \max\{D(i-1, j), D(i, j-1), D(i-1, j-1)\} \ , \tag{3}$$

where we initialize $D(0, 0) = 0$ *and* $D(i, 0) = D(0, j) = -\infty$ *for all* $i \in [L_1]$ *and* $j \in [L_2]$*. The final alignment path is recovered by backtracking from* (L_1, L_2) *to* $(1, 1)$*, analogous to traditional sequence alignment [44].*

2.4 Constructing MSAs from pairwise alignments with template sequence

Building on the pairwise DTW alignment primitive above, we extend our pipeline to the MSA setting. Let $A^k = (a_1^k, a_2^k, \ldots, a_{L_k}^k)$ be the kth sequence in a protein family with N sequences, and $X^k = (x_1^k, x_2^k, \ldots, x_{L_k}^k) \in \mathbb{R}^{L_k \times d}$ be its PLM embeddings. The conventional strategy for building an MSA is to employ a progressive alignment scheme, where pairwise alignments are merged hierarchically along a guide tree [53, 47]. However, in our framework, DTW requires embeddings as input and thus one must construct an embedding-level representation for each intermediate aligned cluster before proceeding to the next merge. This necessitates repeatedly committing to an alignment between a subset of sequences and forming a corresponding consensus embedding. As with progressive alignments within traditional MSA approaches, errors introduced early in the process are amplified in subsequent steps, potentially degrading the overall alignment quality.

To circumvent the issue of repeatedly generating embeddings for intermediate alignments, we instead adopt a two-phase star-alignment strategy, in which a single template is synthesized from the K most central sequences, and then this template serves as a common reference frame to align all sequences. By anchoring each DTW alignment to the same coordinate system, we prevent the propagation of incorrect correspondences across multiple pairwise alignments (see Appendix E, Fig. 19, for a comparison between our star-alignment approach and some progressive variants). In what follows, we detail our strategies for merging the resulting set of pairwise DTW alignments into a coherent MSA, and constructing the template sequence to maximize the effectiveness of the star alignment scheme.

2.4.1 Star alignment procedure. Our star-alignment scheme can be described independently of the procedure used to construct the template sequence. Let this arbitrary template sequence be of length L_τ, and let $T = (t_1, t_2, \ldots, t_{L_\tau}) \in \mathbb{R}^{L_\tau \times d}$ denote the embeddings of its amino acids. Note that T may be constructed directly in the embedding space without requiring a semantically valid protein sequence. We further let $S^k \in \mathbb{R}^{L_\tau \times L_k}$ denote the pairwise similarity matrix between the template sequence and the kth sequence, and P^k denote the DTW alignment path obtained for this pair. Without loss of generality, we assume that for each $(i, j) \in P^k$, i refers to the index of the template sequence.

To construct a coherent MSA from the set of alignment paths $\{P^k\}_{k=1}^N$, we iterate over each template position i and generate a corresponding column block Q_i for the final alignment. Each sequence X^k then contributes to Q_i all residues that associate with t_i in the alignment path P^k, i.e., $\{x_j^k \mid (i, j) \in P^k\}$ forming the column block's k^{th} row. As DTW permits multi-to-one assignments, a single column block may receive several contiguous residues from the same sequence. Conversely, a single residue in a sequence may be assigned to multiple column blocks. To resolve ambiguities in column assignment, whenever a residue is mapped to multiple column blocks, we assign it the most likely column based on its embedding similarity scores with the template residues. Letting c_j^k denote the column block that x_j^k is assigned to:

$$c_j^k \ \triangleq \ \underset{j' : (t_i, x_{j'}^k) \in P_k}{\operatorname{argmax}} \ S_{ij'}^k \ . \tag{4}$$

Following this disambiguation step, the kth row of column block Q_i is given by $Q_{ik} = \{x_j^k \mid c_j^k = i\}$. By construction, all column blocks are mutually disjoint across the template positions. However, not all of them are trivially aligned because each sequence can contribute a different number of residues to the same column block. In order to obtain a coherent alignment, we need to infer appropriate gap placements within each block. Our strategy is to identify an anchor residue for each row Q_{ik} that acts as a true alignment point between X_k and the template. The remaining residues on Q_{ik} will be positioned on either side of this point,

forming a contiguous stretch to avoid fragmented gaps. Finally, we add the required number of gaps to both ends of each stretch to produce a cleanly aligned column block.

In particular, the anchor residue on row Q_{ik} is again determined by comparing embedding similarity scores. Letting m_i^k denote the index in X^k that is inferred to align to t_i, we have

$$m_i^k \triangleq \underset{j':x_{j'}^k \in Q_{ik}}{\operatorname{argmax}} \quad S_{ij'}^k \ . \tag{5}$$

Residues preceding m_i^k in Q_{ik} are treated as a single, contiguous insertion to the left of the alignment column, and are appended to the left of the anchor column. Analogously, residues following m_i^k in Q_{ik} are treated as a single, contiguous insertion and are appended to the right of the anchor column. Our MSA construction procedure is detailed in Alg. 1, Appendix A.

2.4.2 Constructing a template sequence. To obtain a representative template sequence for our star-alignment procedure, we first construct a guide tree using Clustal Omega's mBed algorithm [47]. The key idea behind mBed is to avoid computing all N^2 pairwise distances. Instead, it selects $\log N$ random anchor sequences from the input set and encodes each sequence as a vector of its distances to these anchors. The distance between any two sequences can then be implicitly approximated by the Euclidean distance between their encoded vectors, enabling construction of a distance-based tree in $\mathcal{O}(N \log N)$ time [50].

From this tree, we select the medoid sequence (i.e., the leaf with minimum total distance to all other leaf nodes) as the star-alignment template. We compute this efficiently via an $\mathcal{O}(N)$ rerooting dynamic programming procedure that performs two depth-first traversals [48]: the first pass computes, for each node, the sum of distances to its descendant leaves; the second recursively "reroots" the tree, updating each child's total distance based on how distances change when the root shifts across an edge (see Alg. 2 in Appendix A).

For larger sequence sets, a single medoid can bias the alignment, especially when sequences are highly divergent or span multiple remote subfamilies; this is a known limitation of standard star-alignment, which assumes one representative sequence can anchor all others. To mitigate this, we construct a PLM-generated consensus template that integrates information from multiple medoids rather than relying on a single one.

We select the top-K medoids using the same procedure as before at no extra cost, then align these medoids using our star-alignment method (with the top-1 medoid as the initial template). Gap characters in this MSA are replaced with 'X' tokens, and the modified aligned medoids are passed through the PLM to obtain updated embeddings. We then average embedding vectors position-wise across the K aligned sequences to form a consensus template, which is used to align the full input set. Empirically, $K = \lceil \ln(N) \rceil$ works well and adds negligible runtime. Pseudocode is provided in Appendix A, and the effect of varying K in Section 3.4.

2.5 Benchmarking

Datasets. We evaluate ARIES on BAliBASE 3.0 [54], HOMSTRAD [51], and QuanTest2 [46]. BAliBASE is built from structurally-derived reference alignments and defines "core blocks" that identify reliably aligned regions; we evaluate performance on these core blocks. BAliBASE is divided into sequence sets covering diverse challenging scenarios such as low sequence identity, divergent outliers, remote subfamilies, and sequences with large terminal or internal extensions. HOMSTRAD consists of protein families whose alignments are based on structural superposition, making it a reliable gold standard for assessing accuracy within conserved regions. QuanTest2 includes sequence sets composed of three HOMSTRAD reference sequences with known structures, augmented with 997 background sequences obtained from Pfam [35]; we align all 1,000 sequences but evaluate accuracy only on the reference trio, enabling large-scale assessment while retaining structural ground truth. Our evaluation consists of 218 BAliBASE alignments, 571 HOMSTRAD families, and 151 QuanTest2 sets. In some cases, other methods do not output an alignment for some sets; if this happens, we ignore these sets when comparing the method to ARIES. Further information about these three benchmarking datasets are given in Appendix C.

Parameter settings. For ARIES, we use a default window span $w = 9$ (or window size 19), reciprocal weight $\lambda = 400$, and sequence embeddings formed by concatenating outputs from the last $\ell = 9$ layers of the 650M parameter ESM-2 model [27]. The star alignment template is generated using the top $\lceil \ln N \rceil$

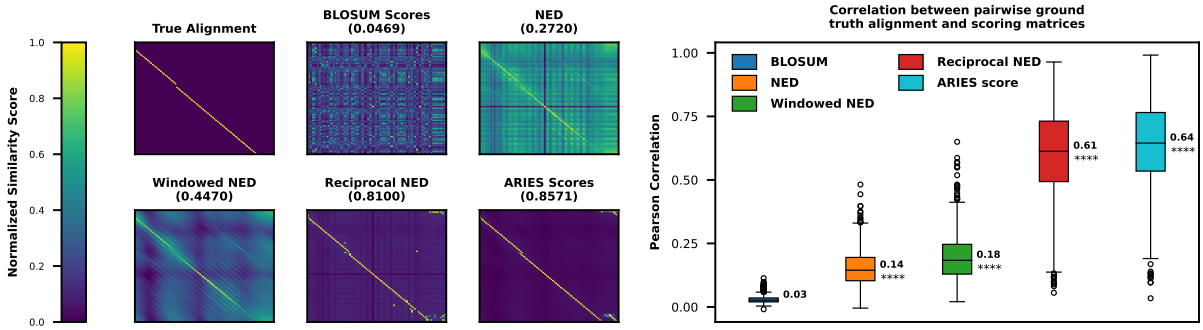

Fig. 2. *Left:* Binary true alignment matrix of two BAliBASE RV11 sequences, followed by heatmaps of five different pairwise residue-similarity matrices for these sequences with Pearson correlation of these similarity scores with the true alignment matrix given. *Right:* Box plots of Pearson correlations between the true alignment and the same five scoring schemes across all sequence pairs in BAliBASE RV11. Median correlation is annotated next to each box. Stars denote statistically significant higher values over than the method to the left ($p \leq 1e^{-4}$, one-sided Wilcoxon signed-rank test with Holm-Bonferroni correction).

medoids. Section 3.4 shows ablation studies to analyze the effect of these parameters. For sequences that are longer than 1022, the maximum amino acid context length seen during training of ESM-2, ARIES uses a tiling approach [8] to approximate the amino acid embeddings (see description and performance analysis in Appendix E, Fig. 14). This affects 46 alignment sets in BAliBASE benchmark, and none of the alignment sets in the other two benchmarks.

Clustal Omega [47], ClustalW [53], MAFFT Auto, MAFFT L-INS-i, MAFFT G-INS-i [24], MAGUS [49], TWILIGHT [55], MUSCLE [13], ProbCons [12], T-Coffee [37], and FAMSA [11] are run on CPU with their default parameters. Methods that allow the specification of substitution matrices are evaluated using the default BLOSUM-62 matrix. See Appendix E, Fig. 16, 17, 18 for ablation studies examining the relative performance of alternative substitution matrices. The PLM-based methods learnMSA2 [5] and vcMSA [33] are run on GPU with recommended default settings. We use ProtT5-XL-Half for learnMSA2 as it is the default PLM for that method. For vcMSA, we pad 10 "X" tokens at each sequence beginning/end and use the last 16 layers of ProtT5-XL. To ensure a fair comparison where PLM-based baselines use the same underlying model, we conduct additional experiments comparing ARIES with ProtT5-XL-Half to learnMSA2, and ARIES with ProtT5-XL to vcMSA (see Appendix E, Fig. 15). Details of the employed baselines and PLMs are given in Appendices B and C.

Evaluation metrics. We evaluate alignments using the Total Column (TC) and Sum-of-Pairs (SP) scores. The TC score is the fraction of columns in the output alignment that exactly match columns in the gold-standard reference alignment, while the SP score is the fraction of residue pairs in the output that are also paired in the reference. Formal definitions are provided in Eq. (7) and Eq. (6) in Appendix C.

Runtime evaluation. For our runtime evaluations on the QuanTest2 dataset, each benchmark is allotted 10 CPU cores with 10 GB of memory. GPU-accelerated methods (ARIES and learnMSA2) are run on a single NVIDIA A100 GPU (80 GB) while maintaining the same CPU and memory allocation.

3 Results

3.1 Reciprocal, window-based embedding similarity enables detection of aligned residues

We assess our similarity scores (Section 2.2) under low-identity conditions with sequence pairs from the BAliBASE RV11 subset (i.e., alignments within BAliBASE with the lowest average pairwise sequence identity). The left panel of Fig. 2 compares the gold-standard alignment of one pair (top-left heatmap) against its pairwise BLOSUM-derived similarity matrix and four embedding-based scoring variants: (a) NED; (b) windowed NED ($w = 9$); (c) reciprocal-weighted NED ($\lambda = 400$); and (d) ARIES scoring scheme with our default parameters ($w = 9$, $\lambda = 400$). All embeddings are obtained using the 650M parameter ESM-2 model [27] and all similarity scores are normalized to $[0, 1]$.

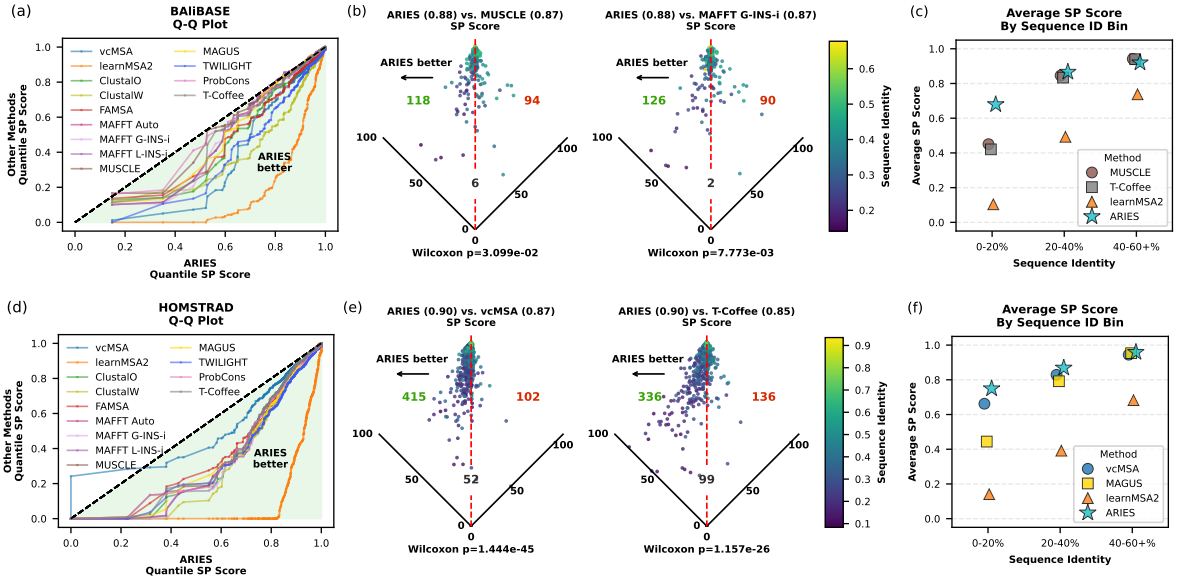

Fig. 3. Comparison of ARIES with state-of-the-art MSA methods on (a) BAliBASE and (b) HOMSTRAD. (a, d): Q-Q plots comparing SP scores for ARIES and alternative methods across quantiles; shaded region indicates where ARIES achieves better performance. (b, e): Volcano plots comparing SP scores for ARIES and the two top-performing previous methods, as determined by mean SP score. Each point corresponds to a sequence set that is aligned; those to the left of the vertical line indicate better performance by ARIES. Colored counts show the number of sequence sets for which ARIES performs better than (green), worse than (red), or equal to (gray) the competitor method. Number in bracket next to each method name shows its mean SP score on the entire dataset. Points are colored by mean sequence identity of the MSA set; darker color indicates lower identity. Holm-Bonferroni corrected p-values from one-sided Wilcoxon signed-rank tests are annotated at the bottom of each plot. (c, f): Mean SP scores of representative methods in three sequence identity bins.

On the exemplar pair, the BLOSUM matrix, lacking contextual information, shows negligible correlation with the true alignment (Fig. 2, $\rho = 0.0469$). NED by itself increases this 5.8-fold ($\rho = 0.272$). Windowed NED ($\rho = 0.447$) and reciprocal-weighted NED ($\rho = 0.810$) substantially improve correlation, while the ARIES scoring scheme, which combines both, achieves the highest correlation ($\rho = 0.857$). The right panel of Fig. 2 shows Pearson correlation distributions across all RV11 pairs. Trends mirror the exemplar: each augmentation significantly improves correlation over the previous ($p \leq 1e^{-4}$, one-sided Wilcoxon signed-rank test with Holm-Bonferroni correction). Across alignments, the combination of both augmentations (i.e., ARIES' design) yields a median correlation 4.57 times greater than that of the original NED score. The effects of window size and the reciprocal weight λ on alignment quality are studied in Section 3.4.

3.2 ARIES outperforms existing methods across diverse benchmarks

We first compare the alignment accuracy of ARIES against a suite of widely used MSA tools on the BAliBASE and HOMSTRAD benchmarks. We present our results in terms of the SP score metric in Fig. 3 and defer the TC score results to Appendix D, Fig. 9, 11, 13.

We showcase the superior performance of ARIES as compared to other methods using Q-Q plots (Fig. 3a,d). Each curve compares ARIES (x-axis) to another single method (y-axis), with points showing SP scores at the same quantile. The shaded region below the diagonal highlights quantiles where ARIES outperforms the competitor method. Across both datasets, most curves lie below the diagonal, indicating that ARIES consistently exceeds competing methods. On BAliBASE, some methods perform similarly to ARIES at upper quantiles, while lower quantiles—representing more challenging alignments—show pronounced margins, demonstrating ARIES' robustness where other approaches degrade (see Appendix E, Tables 2, 3 for average SP and TC scores in each of the six BAliBASE subsets). On HOMSTRAD, ARIES exhibits substantially higher SP scores than all other methods, as reflected by curves lying well below the diagonal line of equal performance; vcMSA is second-best (see Appendix D, Fig. 7), but considerably slower.

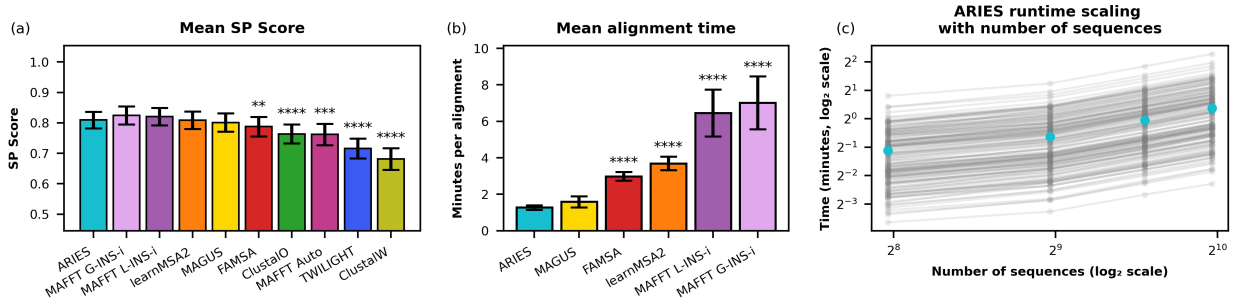

Fig. 4. Comparison of ARIES with other MSA methods on the QuanTest2 dataset. (a) For each scalable method considered, the mean SP score across all QuanTest2 alignments, shown with 95% confidence intervals. (b) Mean alignment runtime (minutes per alignment) for methods that ARIES does not significantly outperform, shown with 95% confidence intervals. ARIES has the lowest mean runtime of these methods, and its runtime is significantly lower than that of all of these methods excepting MAGUS. Stars in panels (a) and (b) denote Holm-Bonferroni corrected one-sided paired Wilcoxon signed-rank tests across alignments ($^{*}\, p \leq 0.05$, $^{**}\, p \leq 1e^{-2}$, $^{***}\, p \leq 1e^{-3}$, $^{****}\, p \leq 1e^{-4}$). (c) log-log plot of ARIES runtimes for each QuanTest2 family across varying subset sizes. Lines show individual families, with blue points indicating the mean runtime at each size tested.

Volcano plots (Fig. 3b,e) provide a detailed view of how ARIES compares to the top two performing other methods (as judged by their average SP score across sequence sets in BAliBASE and HOMSTRAD); analogous comparisons for all competing methods are given in Appendix D, Fig. 8-11. Each point represents a protein family, and is plotted based on its SP score for ARIES (left axis) and the competing method (right axis). Points to the left of the vertical line represent cases where ARIES produces more accurate alignments. Across all comparisons (Appendix D, Fig. 8), the plots show a strong leftward skew, with far more points favoring ARIES than the other methods. On BAliBASE, ARIES outperforms the next-best approach, MUSCLE, on 54.1% of sequence sets. On HOMSTRAD, ARIES performs better than vcMSA, the next best method, in 72.9% of cases. Against the worst performing baseline, learnMSA2, ARIES outperforms it in more than 85% sequence sets on both datasets (see Appendix D, Fig. 8, 10). Using one-sided Wilcoxon signed-rank tests with Holm–Bonferroni correction, we confirm that ARIES' SP score improvements over all competing baselines are statistically significant on BAliBASE (largest corrected $p = 0.031$), and HOMSTRAD (largest corrected $p = 1.157e^{-26}$). The TC score improvements over other baselines are also all statistically significant (see volcano plots for TC scores in Appendix D, Fig. 9, 11, largest corrected values of $p = 0.024$ and $p = 1.454e^{-22}$ on BAliBASE and HOMSTRAD, respectively).

Further, across both datasets, the alignment sets with lowest sequence identity (depicted with darker colors) tend to fall to the left of the vertical line, reinforcing that ARIES performs particularly well in this "twilight zone" of very low sequence identity. We quantify this trend by evaluating the average SP scores of ARIES and three representative baselines (best, median, and worst performers) on three coarse sequence-identity bins (Fig. 3c,f). ARIES achieves the highest mean SP scores in lower-identity bins (0–20% and 20–40% on HOMSTRAD, and 0-20% on BAliBASE) and matches the top and median methods otherwise. In contrast, learnMSA2, which was designed to align large sequence families, consistently performs worst across all bins, suggesting it is indeed poorly suited for the small- to medium-sized families in BAliBASE and HOMSTRAD.

3.3 ARIES scales efficiently and accurately to large datasets

We further evaluate ARIES in large-scale settings using the QuanTest2 benchmark, which contains 151 alignments with 1000 sequences each. We compare ARIES to the nine baseline approaches that scale well to this input size, defined as those taking less than 10 minutes per set on average with default parameters (Fig. 4a). Volcano plots for all pairwise baseline comparisons are given in Appendix D, Fig. 12 (SP score) and Fig. 13 (TC score).

ARIES obtains higher mean SP scores than all methods other than MAFFT G-INSI and MAFFT L-INSI. However, the two MAFFT variants do not perform significantly better than ARIES (Fig. 12, one-sided Wilcoxon Holm-Bonferroni corrected p-values of 0.0949 and 0.142, respectively), and likewise ARIES does not significantly outperform learnMSA2 and MAGUS (the latter of which merges alignments obtained by

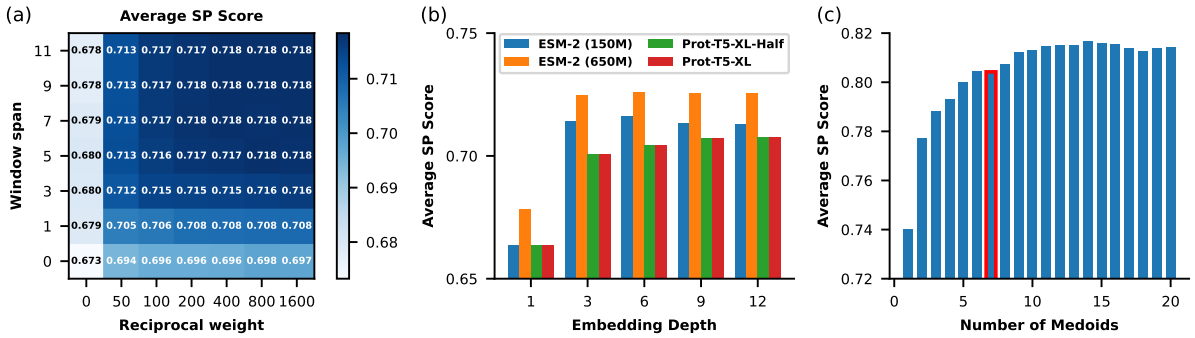

Fig. 5. Ablation study of key components of ARIES. (a) Average SP score on BAliBASE dataset across 49 combinations of window sizes and reciprocal-weight parameters; (b) Average SP score on BAliBASE dataset across varying embedding depth using four different PLM backbones; and (c) Average SP score on QuanTest 2.0 dataset with varying number K of medoid sequences used in the template-building stage. The red-outlined bar marks $K = \lceil \ln(N) \rceil$.

MAFFT L-INSI). On the other hand, ARIES significantly outperfoms ClustalO, ClustalW, FAMSA, TWILIGHT and MAFFT Auto (Fig. 12, one-sided Wilcoxon test with Holm-Bonferroni correction).

ARIES provides substantial runtime improvements as compared to the other MSA methods whose performances are neither significantly worse or significantly better than it (Fig. 4b). Within this set, ARIES is significantly faster than learnMSA2, FAMSA, MAFFT L-INS-i, and MAFFT G-INS-i under the same resource budget (see Section 2.5 for hardware description). ARIES also has a lower mean runtime than MAGUS, though this difference is not statistically significant. We note that both ARIES and learnMSA2 are GPU-accelerated, and thus most directly comparable, while the other methods run on CPU only. Full runtime results are in Fig. 21.

To further evaluate scaling, we measure runtime of ARIES on progressively larger subsets of each family in QuanTest2 (250, 500, 750, and 1000 sequences). Fig. 4c shows a log-lot plot of the runtime (in minutes) versus sequence count, revealing approximately linear scaling with the number of sequences. When excluding the smallest input size, the slope in log-scale is 1.01 (95% confidence interval: 0.798, 1.216), consistent with approximately linear scaling over the tested range. When all input sizes are included, the observed slope is 0.74 (95% confidence interval: 0.644, 0.843); the sublinear slope observed when including the smallest input size is likely due to the presence of fixed startup costs (e.g., initialization and model loading), which contribute a constant overhead. In theory, our approach can scale even further with multi-GPU parallelism and larger accelerator clusters. This establishes ARIES as a highly efficient PLM-based alignment approach for large sequence families.

3.4 Ablation studies

We next examine how ARIES' hyperparameters affect performance. Specifically, we conduct controlled experiments to probe the impact of (a) window and reciprocal-weight parameters in the similarity metric, (b) embedding depth across different PLM backbones, and (c) the number of medoids used for template construction. Results of these ablations are shown in Fig. 5a–c, with experimental details provided below.

Effect of varying window and reciprocal-weighting parameters. Fig. 5a presents a heatmap summarizing SP scores on the BAliBASE dataset across 49 hyperparameter settings. We vary the window span $w \in \{0, 1, 3, 5, 7, 9, 11\}$, corresponding to window lengths in $\{1, 3, 7, 11, 15, 19, 23\}$), and the reciprocal weight parameter $\lambda \in \{0, 50, 100, 200, 400, 800, 1600\}$. Here, setting $w = 0$ and $\lambda = 0$ corresponds to using just NED. Across these combinations, we observe a significant jump in SP score moving from a residue-based metric ($w = 0$) to any window-based metric ($w \geq 1$) at all λ values (one-sided Wilcoxon p-values $\leq 1e^{-4}$ for each of the seven comparisons). This pattern demonstrates that incorporating local context indeed improves the robustness of alignment signals, which agrees with our earlier finding on metric fitness (Section 3.1). Performance remains relatively consistent as we further increase w. We attribute this stability to the smoothing effect of the discrete Gaussian weights: residues further away from the center naturally receive lower influence, allowing the method to maintain robustness even as the window size increases. We also find that increas-

ing λ consistently leads to meaningful improvements in alignment quality, with improvements saturating by $\lambda = 800$, confirming that mutual agreement between residues provides a robust and reliable alignment signal.

Choice of PLM and embedding depth. Fig. 5b examines the effect of embedding depth on alignment quality using four PLM backbones: ESM-2 (150M), ESM-2 (650M) [27], ProtT5-XL-Half, and ProtT5-XL [14]. For each model, ARIES SP scores on BAliBASE are computed when representing amino acids using concatenations of the last ℓ PLM layer embeddings, with $\ell \in [1, 3, 6, 9, 12]$. Across all models, the largest gain occurs from $\ell = 1$ to $\ell = 3$, likely because the final layer emphasizes residue identity over structural context. Runtime increases with embedding dimension (see Appendix D, Fig. 22). Relative PLM rankings remain consistent, with ESM-2 (650M) performing best, indicating ARIES is model-agnostic yet benefits from stronger embeddings.

Template sequence construction. Fig. 5c illustrates how the number of medoid sequences K influences the template quality on the QuanTest2 dataset. Incorporating more medoid sequences naturally allows the template to incorporate patterns from a broader range of structural subgroups, which is especially beneficial for large and diverse families such as those in QuanTest2. As expected, ARIES shows sharp improvement in SP scores going from one medoid sequence to larger values of K, which demonstrates the weakness of a simple star-alignment stragety and the importance of well-informed template generation. The improvements seem to saturate beyond $K = 14$, which suggests that a relatively small number of representative sequences already capture most of the diversity in the MSA set. The red-outlined bar highlights the configuration $K = \lceil \ln(N) \rceil$, which achieves near-optimal performance. This principled, dataset-adaptive way to set the number of medoids also guarantees an efficient runtime, as it only adds a negligible $\mathcal{O}(\log(N))$ overhead to our pipeline (see empirical runtime analysis in Appendix D, Fig. 21).

4 Discussion

We have developed ARIES, a new framework for constructing MSAs that leverages contextual representations from PLMs. ARIES integrates a windowed reciprocal-weighted embedding similarity metric with a DTW-based star-alignment strategy driven by a PLM-generated template, moving beyond the constraints of fixed substitution matrices and explicit gap penalties. Across large-scale evaluations on BAliBASE, HOMSTRAD, and QuantTest2, ARIES consistently delivers substantial, statistically-significant performance gains over state-of-the-art methods while maintaining favorable scaling with growing sequence sets. These results demonstrate that PLM-derived signals can materially advance MSA accuracy for evolutionarily distant proteins and provide a scalable foundation for large-scale alignment.

Although ARIES performs well, several directions could further advance embedding-based alignment. Classical MSA ideas, such as iterative refinement [13, 24], consistency transformations [12, 37], and profile–profile alignment [26], could be incorporated to improve accuracy. BLAST-style seeding [4] and related seed selection techniques [31, 16, 18] could also be incorporated to improve runtime scalability. Another opportunity is to learn the similarity metric directly, rather than relying on the zero-shot use of pretrained PLM embeddings; given the abundance of protein families with high-quality MSAs, such training should be feasible. Additionally, recent methods such as SMURF [39] and DEDAL [29] (Appendix B), although not focused on full MSA construction, highlight the benefits of integrating alignment learning into end-to-end workflows; ARIES could serve as an alignment module in such systems, offering improved performance in low-identity settings and scalability to large families.

Overall, given the central role of MSAs in the structural, evolutionary, and functional analyses of proteins, and the clear improvements in scalability and accuracy demonstrated by ARIES, we anticipate that PLM-based approaches will become a new foundation for large-scale protein alignment.

References

1. Alley, E., Khimulya, G., Biswas, S., AlQuraishi, M., Church, G.: Unified rational protein engineering with sequence-based deep representation learning. Nature Methods **16**, 1315–1322 (2019)
2. Altschul, S., Lipman, D.: Trees, stars, and multiple biological sequence alignment. SIAM Journal of Applied Mathematics **49**, 197–209 (1989)
3. Altschul, S.F.: Amino acid substitution matrices from an information theoretic perspective. Journal of Molecular Biology **219**(3), 555–565 (1991)

4. Altschul, S.F., Gish, W., Miller, W., Myers, E.W., Lipman, D.J.: Basic local alignment search tool. Journal of molecular biology **215**(3), 403–410 (1990)
5. Becker, F., Stanke, M.: learnMSA2: deep protein multiple alignments with large language and hidden Markov models. Bioinformatics **40**(Supplement_2), ii79–ii86 (Sep 2024)
6. Bepler, T., Berger, B.: Learning the protein language: Evolution, structure, and function. Cell Systems **12**(6), 654–669 (Jun 2021)
7. Bepler, T., Berger, B.: Learning protein sequence embeddings using information from structure. Proceedings of International Conference on Learning Representations (2019), `http://arxiv.org/abs/1902.08661`
8. Brandes, N., Goldman, G., Wang, C.H., Ye, C.J., Ntranos, V.: Genome-wide prediction of disease variant effects with a deep protein language model. Nat. Genet. **55**(9), 1512–1522 (Sep 2023)
9. Capra, J.A., Singh, M.: Predicting functionally important residues from sequence conservation. Bioinformatics **23**(15), 1875–1882 (2007)
10. Dayhoff, M., Schwartz, R., Orcutt, B.: A model of evolutionary change in proteins. Atlas of protein sequence and structure **5**, 345–352 (1978)
11. Deorowicz, S., Debudaj-Grabysz, A., Gudyś, A.: FAMSA: Fast and accurate multiple sequence alignment of huge protein families. Scientific Reports **6**(1), 33964 (Sep 2016)
12. Do, C.B., Mahabhashyam, M.S., Brudno, M., Batzoglou, S.: ProbCons: Probabilistic consistency-based multiple sequence alignment. Genome Research **15**(2), 330–340 (Feb 2005)
13. Edgar, R.C.: MUSCLE: a multiple sequence alignment method with reduced time and space complexity. BMC Bioinformatics **5**(1), 113 (Aug 2004)
14. Elnaggar, A., Heinzinger, M., Dallago, C., Rehawi, G., Yu, W., Jones, L., Gibbs, T., Feher, T., Angerer, C., Steinegger, M., Bhowmik, D., Rost, B.: ProtTrans: Towards cracking the language of life's code through self-supervised learning. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021)
15. Henikoff, S., Henikoff, J.G.: Amino acid substitution matrices from protein blocks. Proceedings of the National Academy of Sciences of the United States of America **89**(22), 10915–10919 (Nov 1992)
16. Hoang, M., Marçais, G., Kingsford, C.: Masked minimizers: Unifying sequence sketching methods. bioRxiv pp. 2022–10 (2022)
17. Hoang, M., Singh, M.: Locality-aware pooling enhances protein language model performance across varied applications. Bioinformatics **41**(Supplement_1), i217–i226 (2025)
18. Hoang, M., Zheng, H., Kingsford, C.: Deepminimizer: A differentiable framework for optimizing sequence-specific minimizer schemes. In: International Conference on Research in Computational Molecular Biology. pp. 52–69. Springer (2022)
19. Iovino, B.G., Tang, H., Ye, Y.: Protein domain embeddings for fast and accurate similarity search. Genome Research **34**(9), 1434–1444 (2024)
20. Iovino, B.G., Ye, Y.: Protein embedding based alignment. BMC Bioinformatics **25**(1), 85 (Feb 2024)
21. Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S.A.A., Ballard, A.J., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J., Back, T., Petersen, S., Reiman, D., Clancy, E., Zielinski, M., Steinegger, M., Pacholska, M., Berghammer, T., Bodenstein, S., Silver, D., Vinyals, O., Senior, A.W., Kavukcuoglu, K., Kohli, P., Hassabis, D.: Highly accurate protein structure prediction with AlphaFold. Nature **596**(7873), 583–589 (2021)
22. Kaminski, K., Ludwiczak, J., Pawlicki, K., Alva, V., Dunin-Horkawicz, S.: pLM-BLAST: distant homology detection based on direct comparison of sequence representations from protein language models. Bioinformatics **39**(10), btad579 (2023)
23. Katoh, K., Misawa, K., Kuma, K., Miyata, T.: MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform. Nucleic Acids Research **30**(14), 3059–3066 (Jul 2002)
24. Katoh, K., Standley, D.M.: MAFFT Multiple Sequence Alignment Software Version 7: Improvements in Performance and Usability. Molecular Biology and Evolution **30**(4), 772–780 (Apr 2013)
25. Kilinc, M., Jia, K., Jernigan, R.L.: Improved global protein homolog detection with major gains in function identification. Proceedings of the National Academy of Sciences **120**(9), e2211823120 (2023)
26. Krogh, A., Brown, M., Mian, I.S., Sjölander, K., Haussler, D.: Hidden markov models in computational biology: Applications to protein modeling. Journal of molecular biology **235**(5), 1501–1531 (1994)
27. Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., Dos Santos Costa, A., Fazel-Zarandi, M., Sercu, T., Candido, S., Rives, A.: Evolutionary-scale prediction of atomic-level protein structure with a language model. Science **379**(6637), 1123–1130 (Mar 2023)
28. Liu, K., Warnow, T.J., Holder, M.T., Nelesen, S.M., Yu, J., Stamatakis, A.P., Linder, C.R.: SATe-II: very fast and accurate simultaneous estimation of multiple sequence alignments and phylogenetic trees. Systematic Biology **61**(1), 90–106 (Jan 2012)
29. Llinares-López, F., Berthet, Q., Blondel, M., Teboul, O., Vert, J.P.: Deep embedding and alignment of protein sequences. Nature Methods **20**(1), 104–111 (2023)

30. Lupo, U., Sgarbossa, D., Bitbol, A.F.: Protein language models trained on multiple sequence alignments learn phylogenetic relationships. Nature Communications **13**(1), 6298 (2022)
31. Marçais, G., Pellow, D., Bork, D., Orenstein, Y., Shamir, R., Kingsford, C.: Improving the performance of minimizers and winnowing schemes. Bioinformatics **33**(14), i110–i117 (2017)
32. Marks, D.S., Hopf, T.A., Sander, C.: Protein structure prediction from sequence variation. Nature Biotechnology **30**(11), 1072–1080 (2012)
33. McWhite, C.D., Armour-Garb, I., Singh, M.: Leveraging protein language models for accurate multiple sequence alignments. Genome Research **33**(7), 1145–1153 (Jul 2023)
34. Mirarab, S., Nguyen, N., Guo, S., Wang, L.S., Kim, J., Warnow, T.: PASTA: Ultra-Large Multiple Sequence Alignment for Nucleotide and Amino-Acid Sequences. Journal of Computational Biology **22**(5), 377–386 (May 2015)
35. Mistry, J., Chuguransky, S., Williams, L., Qureshi, M., Salazar, G., Sonnhammer, E., Tosatto, S., Paladin, L., Raj, S., Richardson, L., Finn, R., Bateman, A.: Pfam: The protein families database in 2021. Nucleic Acids Research **49**, D412–D419 (2021)
36. Needleman, S.B., Wunsch, C.D.: A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of Molecular Biology **48**(3), 443–453 (Mar 1970)
37. Notredame, C., Higgins, D.G., Heringa, J.: T-coffee: a novel method for fast and accurate multiple sequence alignment1. Journal of Molecular Biology **302**(1), 205–217 (Sep 2000)
38. Pantolini, L., Studer, G., Pereira, J., Durairaj, J., Tauriello, G., Schwede, T.: Embedding-based alignment: combining protein language models with dynamic programming alignment to detect structural similarities in the twilight-zone. Bioinformatics **40**(1), btad786 (Jan 2024)
39. Petti, S., Bhattacharya, N., Rao, R., Dauparas, J., Thomas, N., Zhou, J., Rush, A.M., Koo, P., Ovchinnikov, S.: End-to-end learning of multiple sequence alignments with differentiable Smith–Waterman. Bioinformatics **39**(1), btac724 (2023)
40. Rao, R., Bhattacharya, N., Thomas, N., Duan, Y., Chen, P., Canny, J., et al.: Evaluating protein transfer learning with TAPE. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 32 (2019)
41. Remmert, M., Biegert, A., Hauser, A., , Söding, J.: HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment. Nature Methods **9**, 73–175 (2012)
42. Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., Guo, D., Ott, M., Zitnick, C.L., Ma, J., et al.: Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. Proceedings of the National Academy of Sciences **118**(15), e2016239118 (2021)
43. Rost, B.: Twilight zone of protein sequence alignments. Protein Engineering **12**(2), 85–94 (1999)
44. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Transactions on Acoustics, Speech, and Signal Processing **26**(1), 43–49 (1978)
45. Schütze, K., Heinzinger, M., Steinegger, M., Rost, B.: Nearest neighbor search on embeddings rapidly identifies distant protein relations. Frontiers in Bioinformatics **2** (2022)
46. Sievers, F., Higgins, D.G.: Quantest2: benchmarking multiple sequence alignments using secondary structure prediction. Bioinformatics **36**(1), 90–95 (07 2019)
47. Sievers, F., Wilm, A., Dineen, D., Gibson, T.J., Karplus, K., Li, W., Lopez, R., McWilliam, H., Remmert, M., Söding, J., Thompson, J.D., Higgins, D.G.: Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. Molecular Systems Biology **7**, 539 (Oct 2011)
48. Sleator, D.D., Tarjan, R.E.: A data structure for dynamic trees. In: Proceedings of the Thirteenth Annual ACM Symposium on Theory of Computing. pp. 114–122 (1981)
49. Smirnov, V., Warnow, T.: MAGUS: multiple sequence alignment using graph clustering. Bioinformatics **37**(12), 1666–1672 (2021)
50. Sokal, R.R., Michener, C.D., et al.: A statistical method for evaluating systematic relationships. University of Kansas Scientific Bulletin **38**, 1409–1438 (1958)
51. Stebbings, L.A., Mizuguchi, K.: Homstrad: recent developments of the homologous protein structure alignment database. Nucleic Acids Research **32**, D203–D207 (01 2004)
52. Tay, Y., Dehghani, M., Abnar, S., Shen, Y., Bahri, D., Pham, P., Rao, J., Yang, L., Ruder, S., Metzler, D.: Long range arena: A benchmark for efficient transformers (2020), `https://arxiv.org/abs/2011.04006`
53. Thompson, J.D., Higgins, D.G., Gibson, T.J.: 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**(22), 4673–4680 (Nov 1994)
54. Thompson, J.D., Koehl, P., Ripp, R., Poch, O.: Balibase 3.0: Latest developments of the multiple sequence alignment benchmark. Proteins. **61**(1) (2005)
55. Tseng, Y.H., Walia, S., Turakhia, Y.: Ultrafast and ultralarge multiple sequence alignments using twilight. Bioinformatics **41**(Supplement_1), i332–i341 (2025)
56. Whelan, S., Goldman, N.: A general empirical model of protein evolution derived from multiple protein families using a maximum-likelihood approach. Molecular Biology and Evolution **18**(5), 691–699 (2001)

A ARIES multiple sequence alignment module

Alg. 1 describes the star-alignment procedure introduced in Section 2.4.1. Within this procedure, the subroutine ReciprocalScore implements our embedding-based similarity metric, parameterized by the window size w and reciprocal weight λ (Section 2.2), while DynamicTimeWarping performs the DTW-based pairwise alignment described in Section 2.3.

Alg. 2 details the linear-time rerooting dynamic programming algorithm we use to extract the top-K medoids from the Clustal guide tree. This tree is constructed using the mBed method, which embeds sequences into a reduced-dimensional space and clusters them efficiently. The rerooting DP procedure enables us to compute medoid scores at every internal node in $\mathcal{O}(N)$ time with two depth-first traversals.

Alg. 3 outlines our template generation strategy based on these K medoids. After aligning the medoids with our method, we convert their alignment into a generative template by replacing intermediate gaps with 'X' tokens, re-embedding the resulting pseudo-sequences, and taking a position-wise average of their embeddings.

Finally, Alg. 4 formalizes the complete ARIES MSA pipeline, integrating medoid extraction, template construction, and star alignment into a unified end-to-end framework.

Algorithm 1 ARIES Star-Alignment Procedure (StarAlign)

Require: Raw sequences and PLM embeddings $\{A^k = (a_1^k, a_2^k, \ldots, a_{L_k}^k), X^k = (x_1^k, x_2^k, \ldots, x_{L_k}^k)\}_{k=1}^N$
Require: Template embeddings $T = (t_1, \ldots, t_{L_\tau})$, window span w, reciprocal weight λ

for $k = 1$ to N **do**
  $S^k \leftarrow$ ReciprocalScore(T, X^k, w, λ) ▷ Section 2.2
  $P^k \leftarrow$ DynamicTimeWarping(S^k) ▷ Section 2.3
end for
for $k = 1$ to N **do**
  for $j = 1$ to L_k **do**
    $c_j^k \leftarrow \underset{i \in \{i \mid (t_i, x_j^k) \in P_k\}}{\operatorname{argmax}} S_{ij}^k$ ▷ Disambiguating column block assignment (Eq. 4)
  end for
  for $i = 1$ to L_τ **do**
    $Q_{ik} \leftarrow \{x_j^k \mid c_j^k = i\}$
    $m_i^k \leftarrow \underset{j \in Q_{ik}}{\operatorname{argmax}} S_{ij}^k$ ▷ Inferring true alignment site in column block (Eq. 5)
    $Q_{ik}^- \leftarrow \{x_j^k \in Q_{ik} \mid j < m_i^k\}$
    $Q_{ik}^+ \leftarrow \{x_j^k \in Q_{ik} \mid j > m_i^k\}$
  end for
end for
Initialize empty final MSA
for $i = 1$ to L_τ **do**
  $G_i^- \leftarrow \max_k |Q_{ik}^-|$ ▷ Maximum length of left sub-block
  $G_i^+ \leftarrow \max_k |Q_{ik}^+|$ ▷ Maximum length of right sub-block
  for $k = 1$ to N **do**
    Initialize an empty k^{th} row of the i^{th}column block
    Add $G_i^- - |Q_{ik}^-|$ gap characters to row k ▷ Pad left
    Add a_j^k for all j s.t. $x_j^k \in Q_{ik}^-$ to row k ▷ Residues to the left of inferred alignment site
    Add $a_{m_i^k}^k$ to row k ▷ Anchor residue
    Add a_j^k for all j s.t. $x_j^k \in Q_{ik}^+$ to row k ▷ Residues to the right of inferred alignment site
    Add $G_i^- - |Q_{ik}^+|$ gap characters to row k ▷ Pad right
    Append row k of column block to row k of alignment
  end for
end for
return final MSA

Algorithm 2 Top-K Medoid Selection (TOP-K-MEDOIDS)

Require: Raw sequences $\{A^k = (a_1^k, a_2^k, \ldots, a_{L_k}^k)\}_{k=1}^N$, number of medoids k
Guide tree $\mathfrak{T}$ in adjacency list form $\leftarrow$ MBED $\left(A^1, A^2, \ldots, A^N\right)$
$\mathfrak{T}[u]$ is list of (v, d) tuples where v is a child node of u, and d is distance from u to v
Initialize SIZE$[u_i] \leftarrow 0$, DISTSUM$[u_i] \leftarrow 0$ for all nodes u_i

function DFS1($u, parent$) ▷ **First DFS:** subtree sizes and partial distance accumulation
  SIZE$[u] \leftarrow 1$, DISTSUM$[u] \leftarrow 0$
  for each $(v, d) \in \mathfrak{T}[u]$ **do**
    if $v \neq parent$ **then**
      DFS1(v, u)
      SIZE$[u] \leftarrow$ SIZE$[u]$ + SIZE$[v]$
      DISTSUM$[u] \leftarrow$ DISTSUM$[u]$ + DISTSUM$[v] + d \cdot$ SIZE$[v]$
    end if
  end for
end function

function DFS2($u, parent$) ▷ **Second DFS:** rerooting DP for total leaf distances
  for each $(v, d) \in \mathfrak{T}[u]$ **do**
    if $v \neq parent$ **then**
      DISTSUM$[v] \leftarrow$ DISTSUM$[u] + d \cdot (|\mathfrak{T}| - 2 \cdot$ SIZE$[v])$
      DFS2(v, u)
    end if
  end for
end function

DFS1(ROOT, NULL)
DFS2(ROOT, NULL)
return TOP-K($\{(u,$ DISTSUM$[u]) \mid$ ISLEAF$(u)\}$) ▷ Return indices of top-K medoids

Algorithm 3 ARIES Template Selection (CONSTRUCTTEMPLATE)

Require: Raw seqs and embeddings $\{A^k, X^k\}_{k=1}^N$, PLM $\mathfrak{L}$, no. medoids K, window span w, reciprocal weight λ
Medoid indices $M = \{m_1, m_2, \ldots, m_K\} \leftarrow$ TOP-K-MEDOIDS $\left((A^1, A^2, \ldots, A^N), K\right)$ ▷ Alg. 2
$V^1, V^2, \ldots, V^K \leftarrow$ STARALIGN($\{A^m, X^m \mid m \in M\}$, TEMPLATE $= X^{m_1}$) ▷ Alg. 1
for $k = 1$ to K **do**
  $V^k \leftarrow \mathfrak{L}$(REPLACE($V^k$, "$-$", "$X$")) ▷ Replace gaps with unknown token and re-embed
end for
return $T \leftarrow \frac{1}{K} \sum_{i=1}^K V^k$ ▷ Compute new template from aligned medoid embedding

Algorithm 4 ARIES Multiple Sequence Alignment (ARIES)

Require: Raw sequences $\{A^k\}_{k=1}^N$, PLM $\mathfrak{L}$, no. medoids K, window span w, reciprocal weight λ
for $k = 1$ to N **do**
  $X^k \leftarrow \mathfrak{L}(A^k)$
end for
$T \leftarrow$ CONSTRUCTTEMPLATE($\{A^j, X^k\}_{j=1}^N, \mathfrak{L}, K$)
return STARALIGN($\{A^j, X^j\}_{j=1}^N$, TEMPLATE $= T$)

B Related work

ClustalW [53] (version 1.2.4) is a classic progressive alignment algorithm that constructs a guide tree from pairwise distances and performs alignment using position-specific gap penalties and sequence-weighting schemes.

ClustalO (Clustal Omega) [47] (version 2.1) is the successor to **ClustalW** and scales to large protein families using mBed sequence embeddings, guide-tree clustering, and optimized progressive alignment.

MUSCLE [13] (version 5.1) is an iterative refinement method combining progressive alignment with tree-based realignment cycles, offering a strong balance between computational efficiency and accuracy.

MAFFT [23] (version 7.526) is a scalable MSA framework that accelerates pairwise alignment using fast Fourier transforms and offers multiple iterative refinement strategies for improved accuracy. We evaluate three configurations: **MAFFT L-INS-i**, which performs iterative refinement using local pairwise alignment information, **MAFFT G-INS-i**, which performs iterative refinement using global pairwise alignment information, and **MAFFT Auto**, (the `-auto` flag), which "automatically selects an appropriate strategy from L-INS-i, FFT-NS-i and FFT-NS-2, according to data size." For L-INS-i and G-INS-i, we allow up to 1000 refinement iterations.

T-Coffee [37] (version 13.46.2) is a consistency-based method that integrates heterogeneous pairwise alignment libraries to guide progressive alignment through a unified scoring matrix.

FAMSA [11] (version 2.4.1) is a fast progressive alignment tool optimized for very large datasets, using efficient clustering, SIMD-accelerated scoring, and memory-efficient data structures.

ProbCons [12] (version 1.12) is a consistency-based method employing pair-HMM posterior probabilities and maximum expected accuracy optimization.

TWILIGHT [55] (version 0.2.3) is a progressive alignment method designed for very large datasets, using parallelization and memory-efficiency strategies.

MAGUS [49] (version 0.2.0) is a multiple sequence alignment method for large datasets that aligns sequence subsets independently and merges them through graph-based clustering, rather than relying on traditional guide tree-based progressive alignment.

vcMSA [33] (version 1.2.0) is the first PLM-based MSA framework. This method infers MSA through recursive vector clustering. However, the recursive clustering is not scalable for large protein families.

learnMSA2 [5] (version 2.0.14) integrates PLM embeddings with a deep HMM architecture. This method is optimized for large protein families, though performance can be unstable on smaller or highly divergent input sets.

SMURF [39] introduces a differentiable formulation of the Smith–Waterman algorithm that produces a soft alignment tensor approximating a discrete MSA, enabling end-to-end integration with downstream models such as AlphaFold2. **DEDAL** [29], **EBA** [38], and **PEbA** [20] all leverage PLM embeddings to perform pairwise sequence alignment. These methods cannot reconstruct a coherent multiple sequence alignment.

C Other experiment details

Protein Language Models. Table 1 below gives the specifications of 4 open-source language models used in our experiments. All PLMs are publicly released on HuggingFace.

PLM	ESM-2 (150M)	ESM-2 (650M)	Prot-T5-XL	Prot-T5-XL-Half
No. params	150M	650M	1.2B	1.2B
Output dims	480	640	1024	1024
Precision	32 bits	32 bits	32 bits	16 bits

Table 1. No. parameters, output dimension, and floating point precision of various PLMs used in our experiments.

Datasets. BAliBASE is a structure-based benchmark comprised of reference alignment sets derived from three-dimensional structural superpositions, followed by manual curation and refinement of the resulting alignments. Each reference set is designed to isolate a specific class of difficult alignment scenarios, including low-identity sequences, highly divergent outliers, remote subfamilies, long terminal expansions, and large internal insertions. These categories (RV11, RV12, RV20, RV30, RV40, and RV50) provide controlled tests of alignment robustness under diverse evolutionary and structural conditions. BAliBASE annotates core blocks, which are subsets of alignment columns corresponding to structurally conserved regions that are reliably aligned. Performance is assessed with respect to these core blocks.

HOMSTRAD (Homologous Protein Structure Alignment Database) is comprised of protein families grouped by sequence and structural similarity, for which high-quality experimentally-determined structures are available. For each family, alignments are generated using structural superposition rather than sequence similarity alone, and are then curated to ensure consistency within conserved structural regions. This construction makes HOMSTRAD a reliable benchmark for evaluating whether sequence-based methods recover structurally-meaningful residue correspondences in well-characterized protein folds.

QuanTest2 is a large-scale structure-based benchmark designed to assess alignment performance in high-depth and potentially noisy settings. Each alignment contains exactly 1000 sequences: three reference sequences drawn from HOMSTRAD with known three-dimensional structures, supplemented with 997 additional homologous sequences sampled from Pfam to span broader sequence diversity. Methods are applied to all 1000 sequences, but accuracy is evaluated only on the induced alignment of the three reference sequences. This design enables large-scale testing on diverse sequence collections while retaining a reliable structural ground truth for scoring.

Fig. 6 shows several key statistics of the three datasets used throughout our empirical studies. Across three datasets, HOMSTRAD has the smallest input size (on average 3.52 ± 3.72 sequences per set), followed by BAliBASE (28.54 ± 26.23 sequences per set), whereas every set in the QuanTest2 dataset has exactly 1000 sequences. BAliBASE has both longest maximum sequence length and largest sequence length variance among all 3 datasets, while QuanTest2 has the lowest average sequence identity.

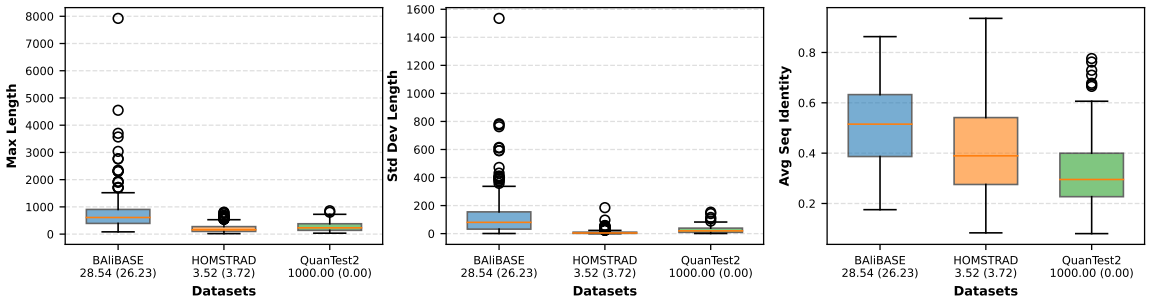

Fig. 6. Key statistics of the BAliBASE, HOMSTRAD, and QuanTest2 datasets. We report the mean number of sequences per family, along with its standard deviation (in parentheses), beneath each dataset label on the x-axis. *Left:* Box plots of maximum sequence length in each protein family. *Middle:* Box plots of standard deviation of sequence lengths in each protein family. *Right:* Box plots of average sequence identity in each protein family. For QuanTest2 we measured the average sequence identity among the three reference sequences.

Metrics. Let $\mathcal{R}$ be the set of tuples (i, j, u, v) specifying that residue u of sequence i aligns with residue v in sequence j in the reference alignment, and $\mathcal{P}$ be the corresponding set for the predicted alignment. The **Sum-of-Pairs (SP) score** is defined as:

$$\mathrm{SP}(\mathcal{P}, \mathcal{R}) \triangleq \frac{\sum_{(i,j,u,v)\in P} \mathbb{I}((i,j,u,v) \in \mathcal{R})}{|\mathcal{R}|} . \tag{6}$$

Further let L_r and L_p be the number of columns in $\mathcal{R}$ and $\mathcal{P}$; P_c be the set of all tuples (i, u) specifying that residue u of sequence i is assigned to column c of P; and $\mathrm{col}(R, i, u)$ denote the column in R that residue u of sequence i is assigned to. The **Total Column (TC) score** is defined as:

$$\mathrm{TC}(\mathcal{P}, \mathcal{R}) = \frac{\sum_{c=1}^{L_p} \sum_{c'=1}^{L_r} \prod_{(i,u)\in P_c} \mathbb{I}(\mathrm{col}(R, i, u) = c')}{L_r} . \tag{7}$$

D Additional evaluation of ARIES

ARIES achieves better SP and TC scores than existing methods across diverse benchmarks. To complement the summarized SP results reported in Section 3.2, we additionally provide the summarized TC results (Fig. 7) and the complete set of volcano plots comparing ARIES against all other baselines with respect to both SP and TC scores (Figs. 8 and 13).

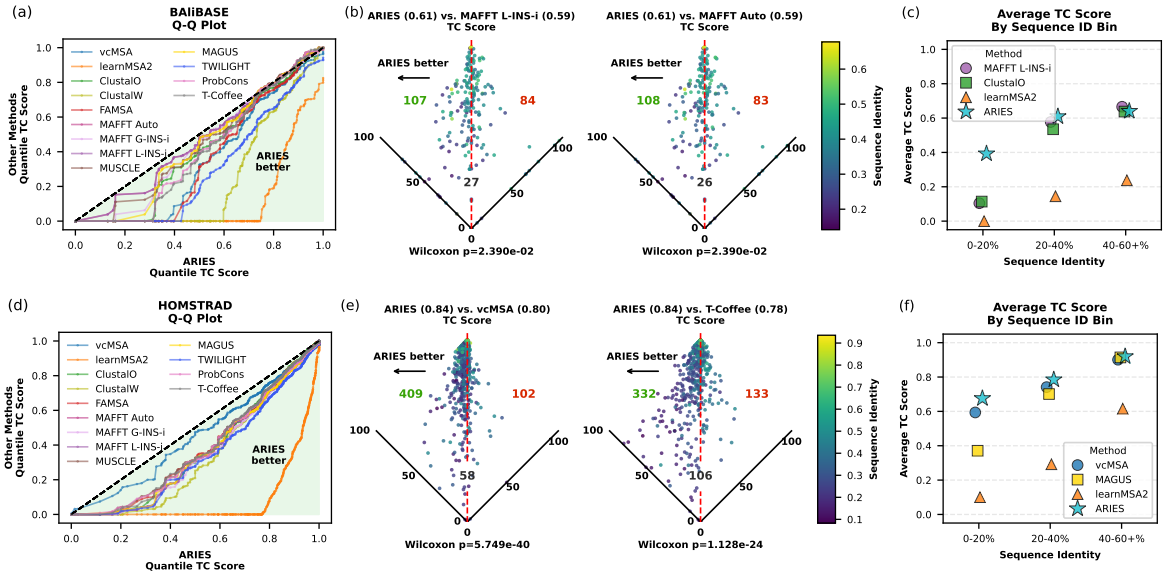

Fig. 7. Comparison of ARIES with state-of-the-art MSA methods on (a) BAliBASE and (b) HOMSTRAD. (a, d): Q-Q plots comparing SP scores for ARIES and alternative methods across quantiles; shaded region indicates where ARIES achieves better performance. (b, e): Volcano plots comparing TC scores for ARIES and the two top-performing previous methods, as determined by mean TC score. Each point corresponds to a sequence set that is aligned; those to the left of the vertical line indicate better performance by ARIES. Colored counts show the number of sequence sets for which ARIES performs better than (green), worse than (red), or equal to (gray) the competitor method. Number in bracket next to each method name shows its mean TC score on the entire dataset. Points are colored by mean sequence identity of the MSA set; darker color indicates lower identity. Holm-Bonferroni corrected p-values from one-sided Wilcoxon signed-rank tests are annotated at the bottom of each plot. (c, f): Mean TC scores of representative methods in three sequence identity bins.

Head-to-head SP score comparison with all baselines on BAliBASE dataset.

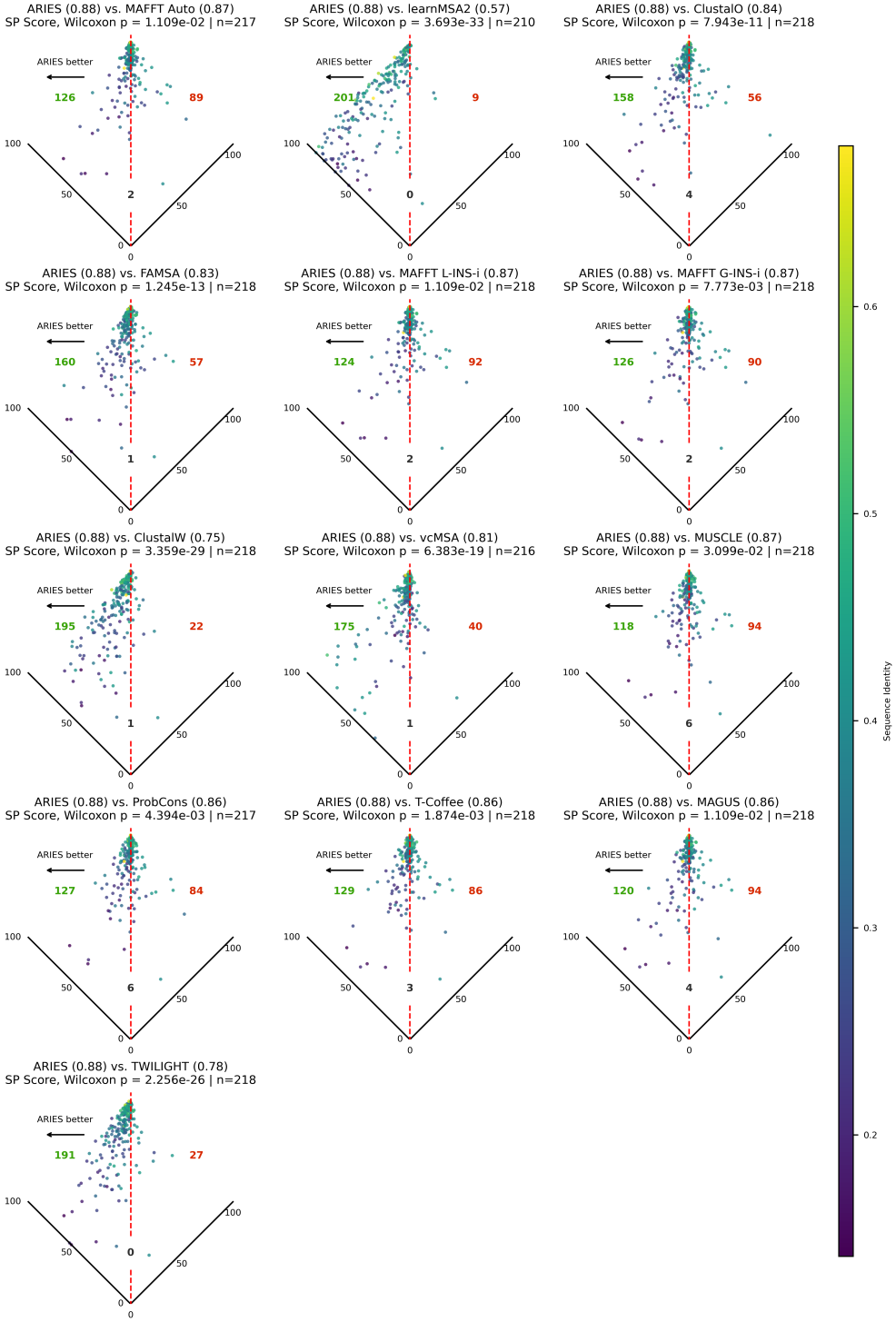

Fig. 8. Comparison of ARIES with state-of-the-art MSA methods on BAliBASE using SP scores. Mean SP scores over the entire dataset are given next to method names; Holm-Bonferroni-corrected p-value of one-sided Wilcoxon signed-rank test is reported for each baseline method.

Head-to-head TC score comparison with all baselines on BAliBASE dataset.

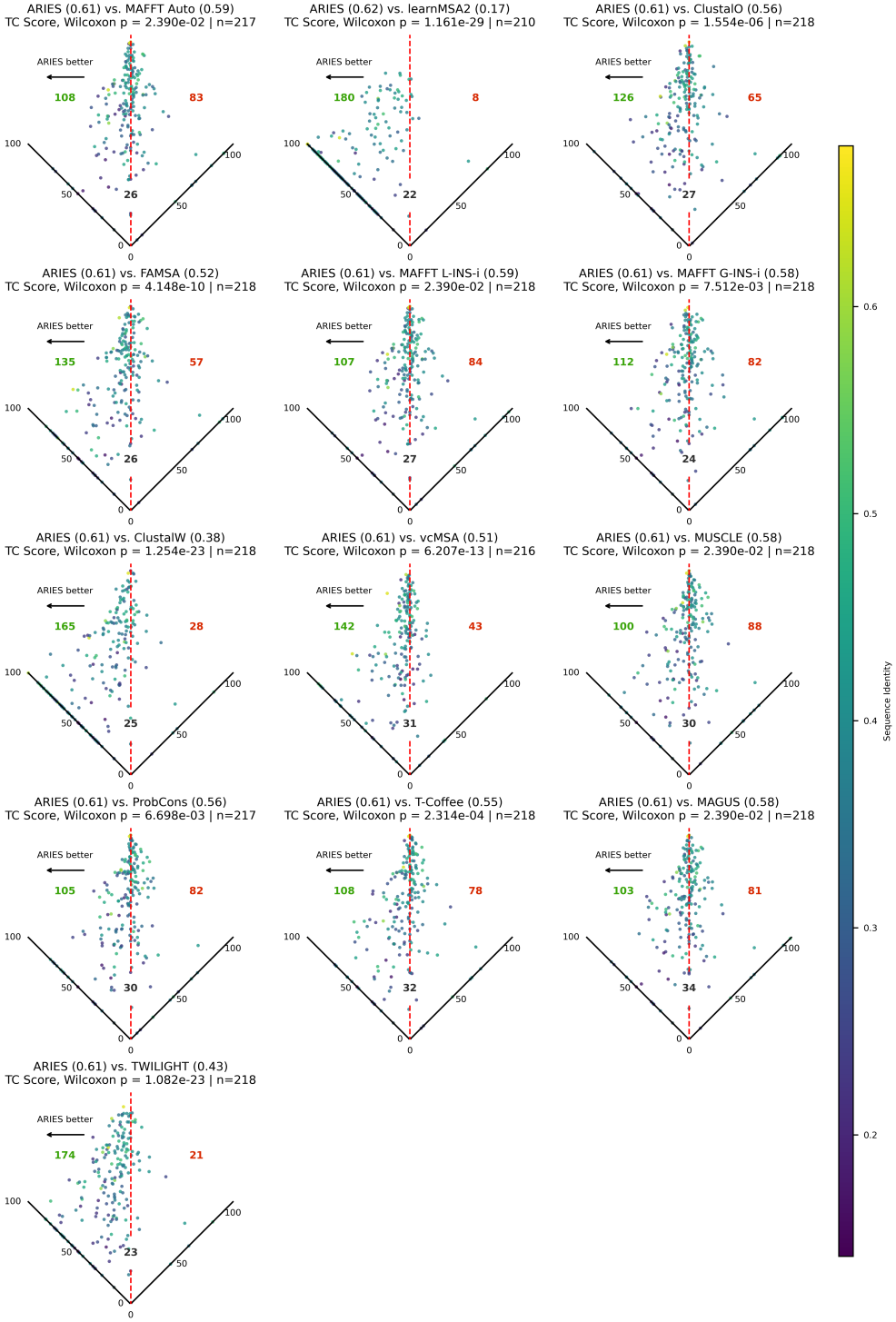

Fig. 9. Comparison of ARIES with state-of-the-art MSA methods on BAliBASE using TC scores. Mean TC scores over the entire dataset are given next to method names; Holm-Bonferroni-corrected p-value of one-sided Wilcoxon signed-rank test is reported for each baseline method.

Head-to-head SP score comparison with all baselines on HOMSTRAD dataset.

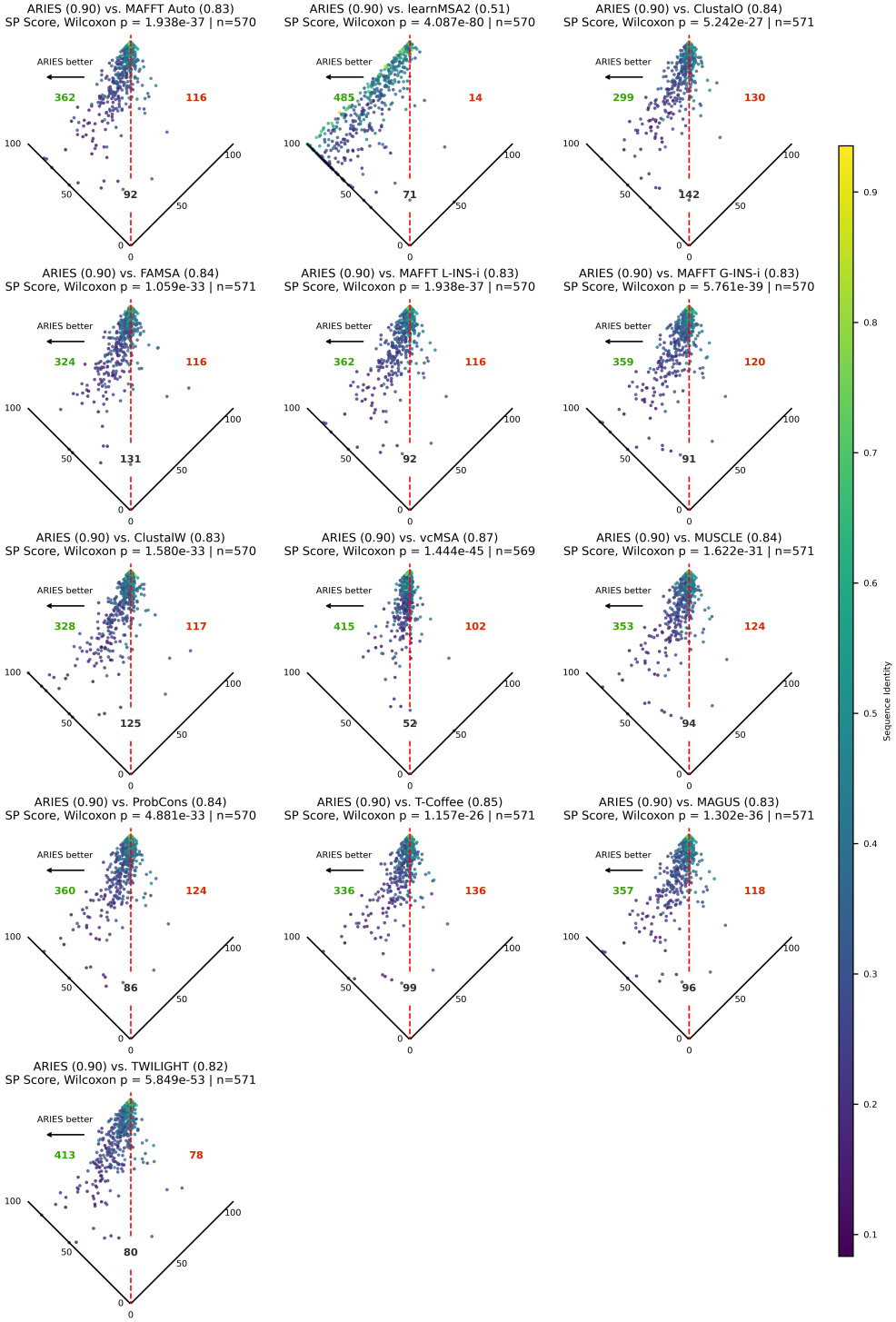

Fig. 10. Comparison of ARIES with state-of-the-art MSA methods on HOMSTRAD using SP scores. Mean SP scores over the entire dataset are given next to method names; Holm-Bonferroni-corrected p-value of one-sided Wilcoxon signed-rank test is reported for each baseline method.

Head-to-head TC score comparison with all baselines on HOMSTRAD dataset.

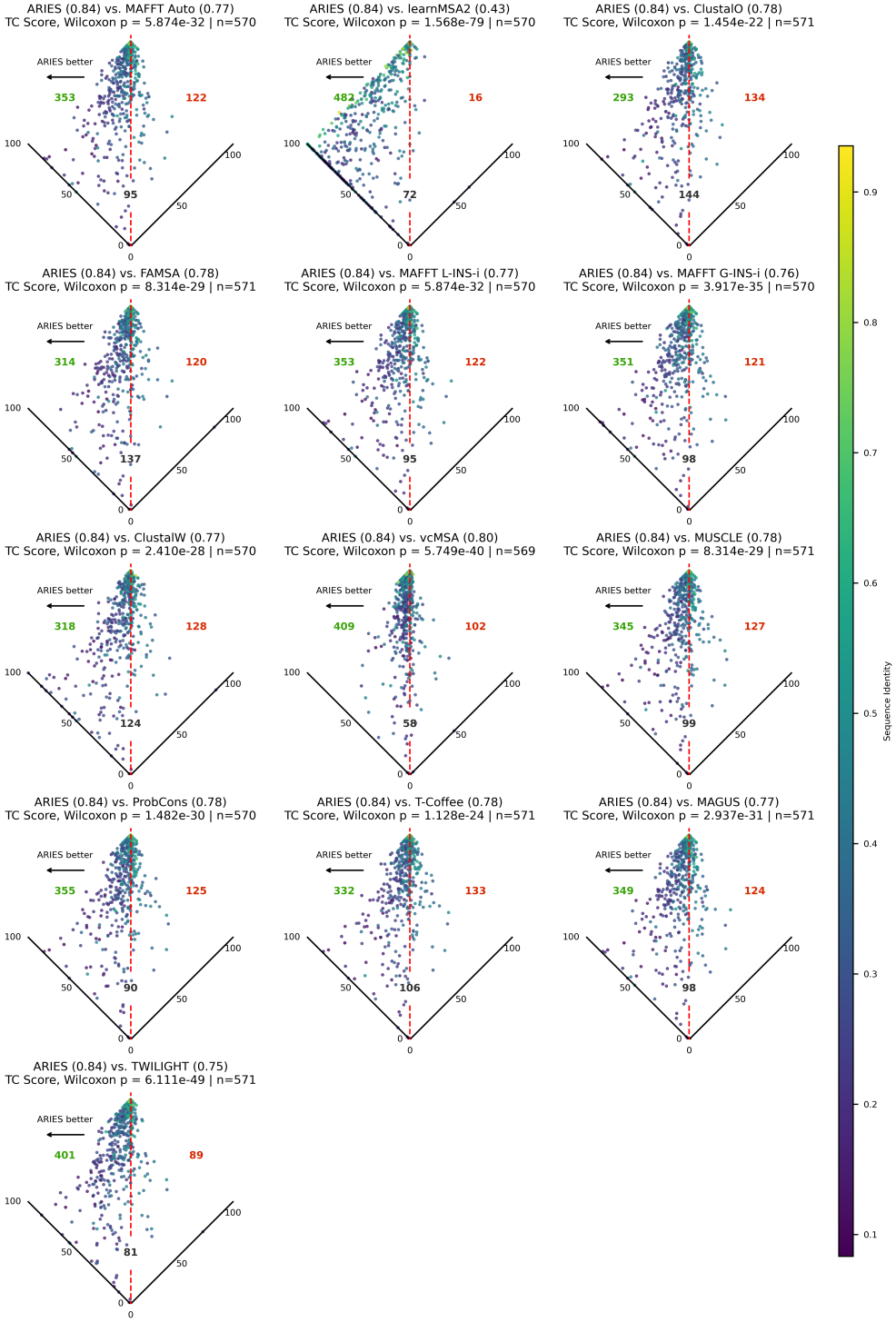

Fig. 11. Comparison of ARIES with state-of-the-art MSA methods on HOMSTRAD using TC scores. Mean TC scores over the entire dataset are given next to method names; Holm-Bonferroni-corrected p-value of one-sided Wilcoxon signed-rank test is reported for each baseline method.

Head-to-head SP score comparison with all scalable baselines on QuanTest2 dataset.

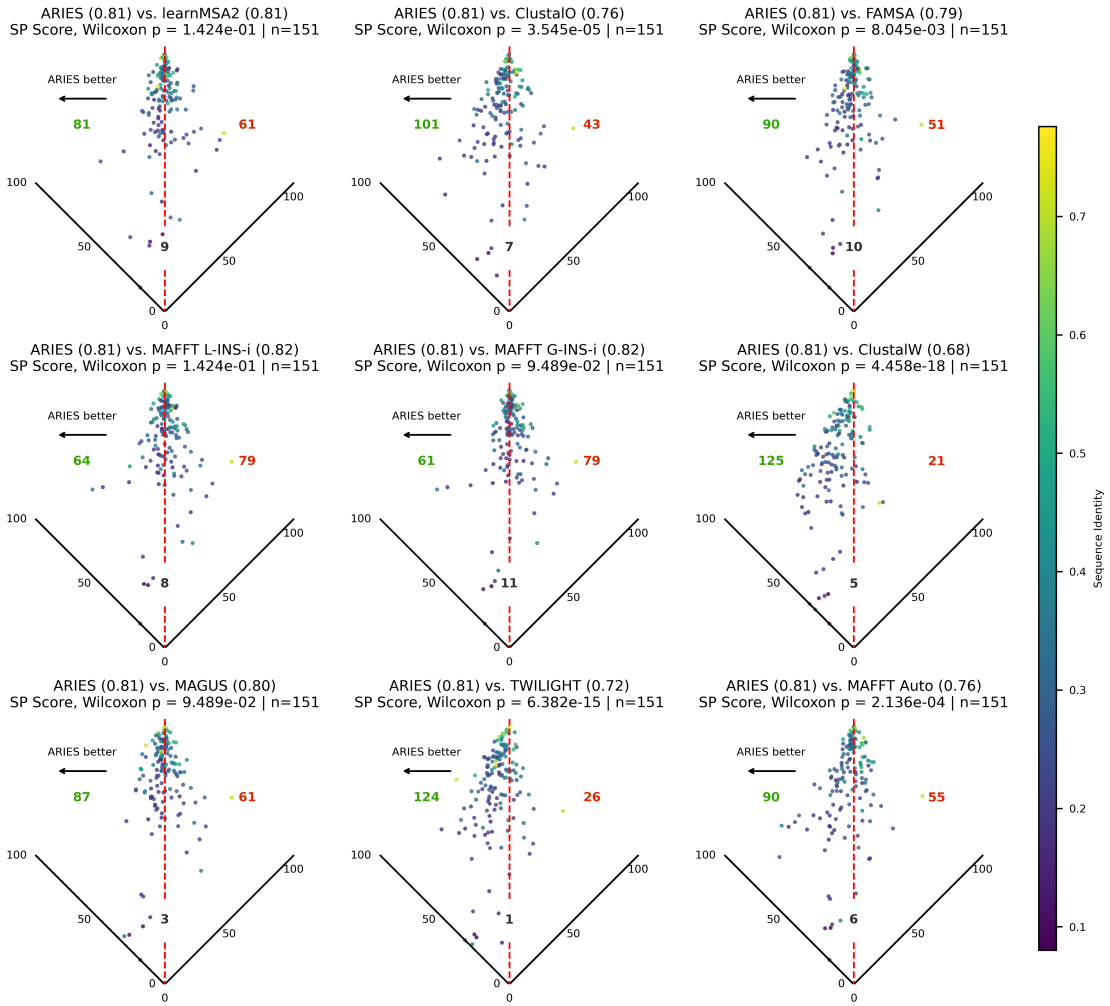

Fig. 12. Comparison of ARIES with state-of-the-art MSA methods on QuanTest2 using SP scores; Holm-Bonferroni-corrected p-value of one-sided Wilcoxon signed-rank test conducted in the direction of the method with the higher mean SP score is reported for each baseline method.

Head-to-head TC score comparison with all scalable baselines on QuanTest2 dataset.

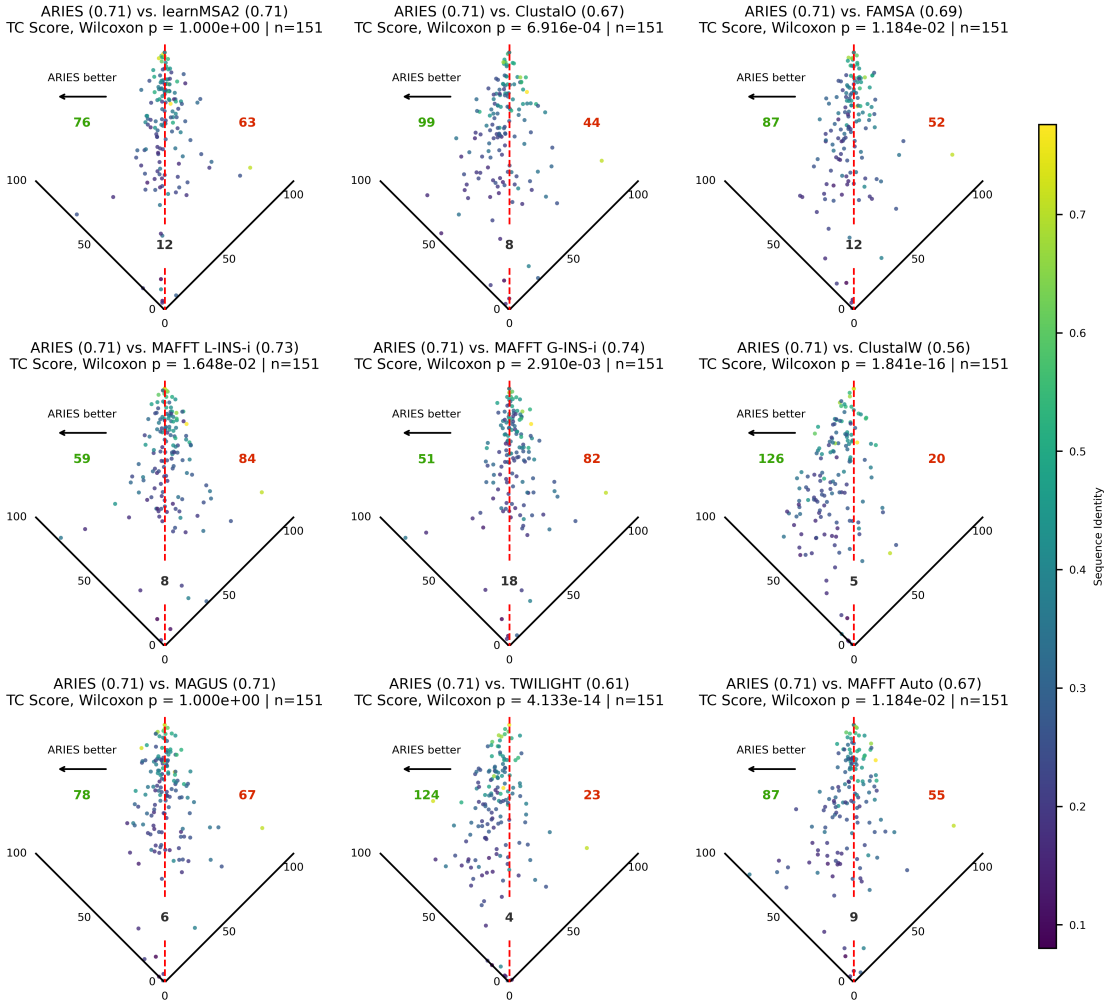

Fig. 13. Comparison of ARIES with state-of-the-art MSA methods on QuanTest2 using TC scores; Holm-Bonferroni-corrected p-value of one-sided Wilcoxon signed-rank test conducted in the direction of the method with the higher mean TC score is reported for each baseline method.

SP and TC score comparisons with all baselines on each BAliBASE subset.

Tables 2 and 3 summarize comparisons between ARIES and previous methods on each individual BAliBASE reference set (RV11-RV50). Entries denote the mean SP or TC score for each method within a given RV. The sample size (n) shown in each column header indicates the number of alignment sets included in that RV. Cells marked with † indicate that one or more alignments failed for that method in the corresponding RV (for example, some methods failed in instances where a sequence contains a non-standard character). In such cases, the comparison with ARIES is performed using only the subset of alignments for which both methods produced valid results. ARIES did not have any failure cases.

Across reference sets, ARIES is strongest on RV11 where it attains higher mean SP and TC scores than every comparison method and wins on a large majority of alignments. ARIES also performs well on RV20, achieving higher mean TC than all comparison methods, and competitive SP performance. On RV12, all methods aside from learnMSA2 achieve uniformly high SP and TC scores, indicating a high-accuracy regime in which performance differences are small in absolute magnitude. Although several refinement-based baselines slightly exceed ARIES in mean score, these differences occur near the performance ceiling and do not reflect large practical gaps. In contrast, RV30 appears more challenging for ARIES: several iterative refinement methods (notably MAFFT variants and MUSCLE) achieve higher mean SP and TC scores and win a larger fraction of alignments. RV40 shows a more balanced pattern, with small mean differences and many non-significant comparisons, suggesting near-parity among methods in that regime. Overall, ARIES demonstrates strong gains in the most divergent setting (RV11), competitive performance in high-accuracy regimes (RV12, RV40), and identifiable opportunities for improvement in insertion-heavy scenarios such as RV30.

Method	**RV11 (n=38)**	**RV12 (n=44)**	**RV20 (n=41)**	**RV30 (n=30)**	**RV40 (n=49)**	**RV50 (n=16)**
ARIES	**0.790**	0.929	0.917	0.839	0.900	0.905
ClustalO	0.590	0.906	0.902	0.862	0.902	0.862
ClustalW	0.501	0.864	0.852	0.725	0.789	0.743
FAMSA	0.620	0.922	0.893	0.822	0.874	0.843
MAFFT G-INS-i	0.647	0.938	0.926	**0.868**	0.919	0.904
MAFFT L-INS-i	0.651	0.937	**0.927**	0.860	**0.919**	0.902
MAFFT auto	0.651	0.937	0.927	0.860	0.918†	0.902
MAGUS	0.651	0.937	0.920	0.847	0.915	0.902
MUSCLE	0.685	0.942	0.912	0.855	0.910	**0.911**
ProbCons	0.670	0.941	0.917	0.845	0.904†	0.894
T-Coffee	0.658	**0.944**	0.915	0.838	0.906	0.895
TWILIGHT	0.504	0.878	0.855	0.803	0.832	0.818
learnMSA2	0.152	0.588	0.836†	0.653†	0.656†	0.535
vcMSA	0.745†	0.920	0.857	0.798	0.733†	0.822

Table 2. BAliBASE: ARIES vs competitors (SP score) across each RV set in BAliBASE. Entries are mean SP scores. Cells marked † indicate that one or more alignments failed for that method in that RV, so the ARIES comparison used only the subset of alignments that succeeded. The highest score per RV is in bold. In the event of a tie, the method with the lowest p-value (one-sided Wilcoxon signed-rank test) is chosen.

Method	RV11 (n=38)	RV12 (n=44)	RV20 (n=41)	RV30 (n=30)	RV40 (n=49)	RV50 (n=16)
ARIES	**0.602**	0.832	**0.546**	0.516	0.562	0.577
ClustalO	0.362	0.794	0.453	0.579	0.583	0.537
ClustalW	0.230	0.717	0.222	0.276	0.398	0.312
FAMSA	0.374	0.824	0.343	0.487	0.524	0.479
MAFFT G-INS-i	0.406	0.843	0.444	**0.599**	0.588	0.599
MAFFT L-INS-i	0.424	0.845	0.468	0.586	**0.607**	0.597
MAFFT auto	0.424	0.845	0.468	0.584	0.607†	0.595
MAGUS	0.415	0.846	0.454	0.570	0.593	0.596
MUSCLE	0.434	**0.862**	0.424	0.580	0.558	**0.627**
ProbCons	0.420	0.860	0.412	0.547	0.536†	0.579
T-Coffee	0.410	0.859	0.406	0.474	0.554	0.587
TWILIGHT	0.273	0.724	0.296	0.446	0.417	0.407
learnMSA2	0.010	0.287	0.222†	0.235†	0.152†	0.095
vcMSA	0.504†	0.824	0.436	0.387	0.405†	0.442

Table 3. BAliBASE: ARIES vs competitors (TC score) across each RV set in BAliBASE. Entries are mean TC scores. Cells marked † indicate that one or more alignments failed for that method in that RV, so the ARIES comparison used only the subset of alignments that succeeded. The highest score per RV is in bold. In the event of a tie, the method with the lowest p-value (one-sided Wilcoxon signed-rank test) is chosen.

E Additional ablation studies

Comparing tiled and untiled BAliBASE alignments.

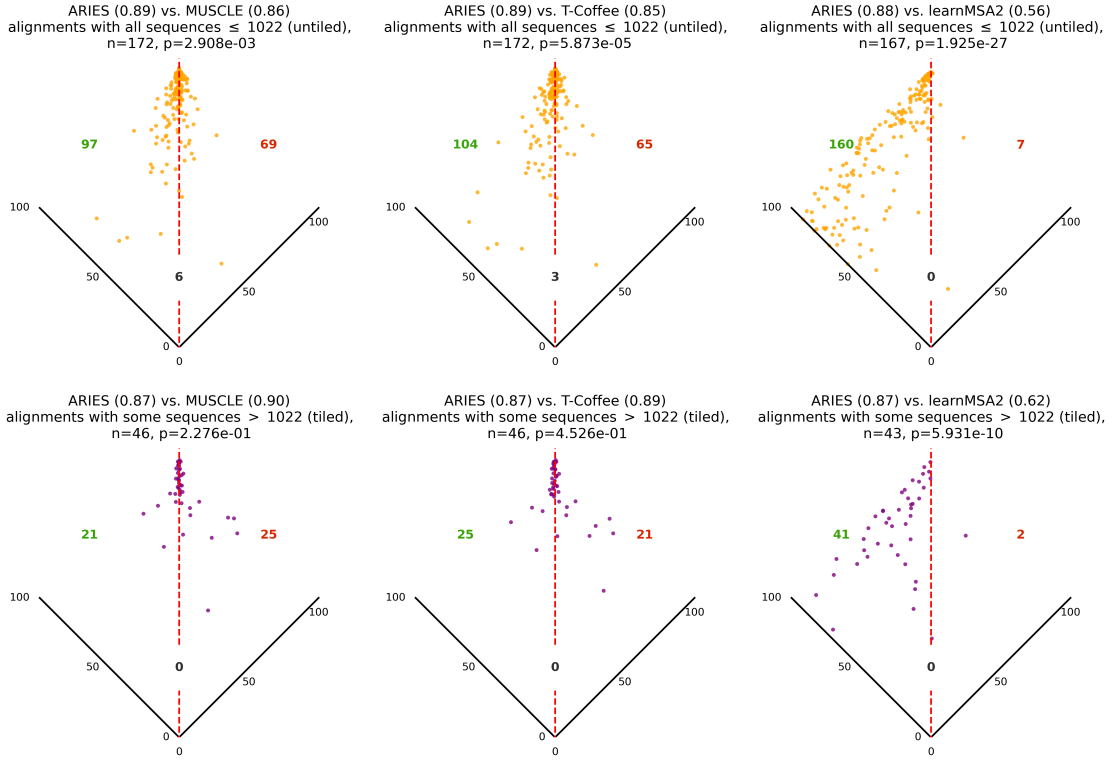

Fig. 14. Comparing alignment results for the best-, median-, and lowest-performing competitor methods on BAliBASE, for untiled (shown in orange) and tiled (shown in purple) alignments. Any alignment with any sequence longer than 1022 amino acid is considered tiled, regardless of how many sequences in the alignment are actually tiled. We report the Holm–Bonferroni-corrected p-value from a one-sided Wilcoxon signed-rank test, where the tested direction corresponds to the method with the higher mean SP score.

Most existing PLMs are trained with a maximum context length due to the scalability constraints of their training process [52]. For example, ESM-2 is trained with a maximum context length of 1024 tokens, two of which are reserved for beginning- and end-of-sequence tokens [42, 27]. Embeddings for amino acids in longer sequences are typically of lower quality, and running long sequences through PLMs can be prohibitively slow. To work around this constraint, we adopt a tiling strategy that partitions long protein sequences into overlapping segments, thereby preserving contextual continuity across tile boundaries [8]. Each tile is processed independently by the PLM, and the resulting representations are subsequently aggregated using a position-weighted average.

This practical trick enables us to align sequence sets with maximum lengths exceeding 1022 residues, covering an additional 46 BAliBASE sets. However, we observe that embeddings derived from tiled sequences tend to be of lower quality than those from untiled sequences, which in turn leads to reduced alignment performance. Figure 14 compares ARIES with the best-, median-, and lowest-performing competitor methods separately on the 172 untiled and 46 tiled BAliBASE sets. While ARIES significantly outperforms all three competitors on the untiled subset, it underperforms the strongest competitor (MUSCLE) on the tiled subset. These findings suggest that ARIES would likely benefit from protein language models trained with inherently longer context windows, which could mitigate the need for tiling.

Comparing ARIES with vcMSA and learnMSA with the same underlying PLMs.

To ensure a fair comparison in which PLM-based baselines employ the same underlying model, we conducted additional experiments comparing ARIES using ProtT5-XL-Half with learnMSA2 (which utilizes ProtT5-

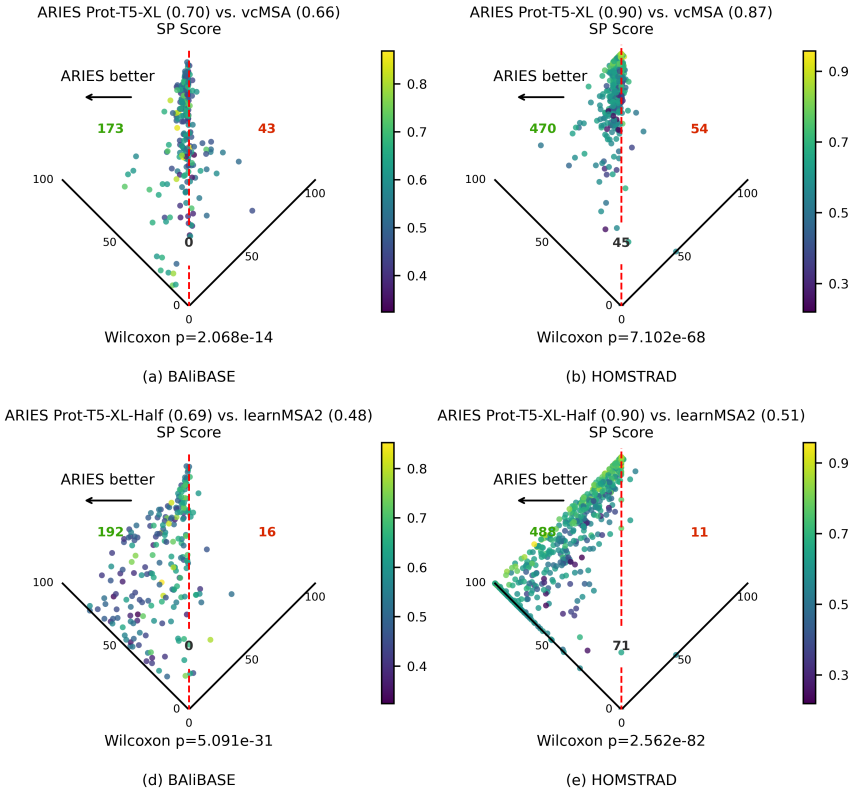

Fig. 15. Comparison of ARIES (ProtT5-XL) against vcMSA on (a) BAliBASE and (b) HOMSTRAD datasets, and comparison of ARIES (ProtT5-XL-Half) against learnMSA2 on (c) BAliBASE and (d) HOMSTRAD datasets.

XL-Half) on BAliBASE (Fig. 15a) and HOMSTRAD (Fig. 15b), as well as ARIES using ProtT5-XL with vcMSA (which utilizes ProtT5-XL) on BAliBASE (Fig. 15c) and HOMSTRAD (Fig. 15d). In both cases, ARIES significantly outperforms the corresponding baseline when using the same underlying PLM.

Trying different substitution matrices for competitor methods.

In the main manuscript, baseline methods were run using their default settings as provided by the authors. For classical protein alignment tools, this corresponds to the BLOSUM62 substitution matrix. For the top three comparison methods by average SP score on each evaluation dataset, we also explored the use of different substitution matrices where supported by the current versions of those tools. These methods included MUSCLE (BAliBASE, HOMSTRAD), MAFFT L-INS-i (BAliBASE, QuanTest2), MAFFT G-INS-i (BAliBASE, QuanTest2), vcMSA (HOMSTRAD), T-Coffee (HOMSTRAD), and learnMSA2 (QuanTest2).

Among these, only MAFFT L-INS-i, MAFFT G-INS-i, and T-Coffee provide support in their current releases for specifying alternative amino acid substitution matrices. Other methods either do not expose this option (e.g., MUSCLE, MAGUS), or rely on fixed internal scoring schemes or downstream alignment components that do not directly admit user-specified matrices in practice. Accordingly, on datasets where one of these matrix-configurable methods ranked among the top three, we evaluated performance under two additional substitution matrices.

For BAliBASE, both MAFFT L-INS-i and MAFFT G-INS-i ranked among the top three and allow substitution matrix changes. We tested BLOSUM45 and JTT250 in addition to the default BLOSUM62. Across matrices, ARIES significantly outperforms both MAFFT G-INS-i and MAFFT L-INS-i, shown in Fig. 16.

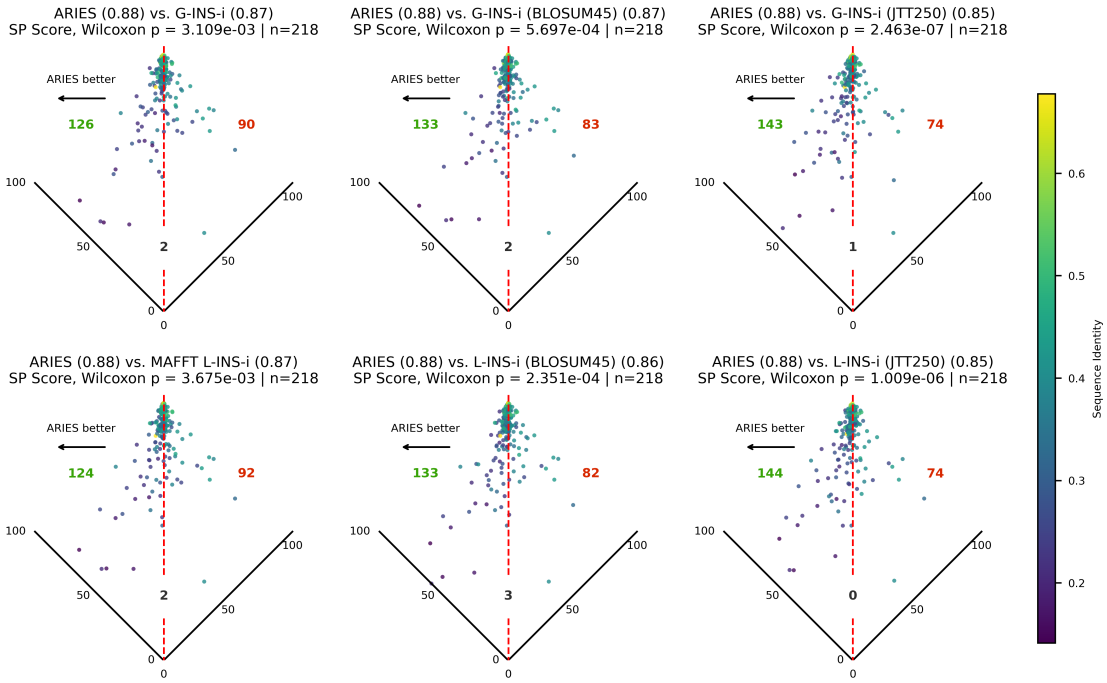

Fig. 16. SP scores across all BAliBASE sets of ARIES, MAFFT L-INS-i, and MAFFT G-INS-i with different substitution matrices. ARIES significantly outperforms both MAFFT variants regardless of substitution matrix.

For HOMSTRAD, T-Coffee was the only top-three method supporting matrix selection. We evaluated BLOSUM45 and PAM250 (as JTT matrices are not supported in this configuration) in addition to the default BLOSUM62. ARIES significantly outperforms T-Coffee regardless of the substitution matrix used (Wilcoxon signed-rank test corrected p-values all $< 1e^{-25}$), shown in Fig. 17.

For QuanTest2, both MAFFT L-INS-i and MAFFT G-INS-i ranked among the top three and allow substitution matrix changes. We again tested BLOSUM45 and JTT250 in addition to the default BLOSUM62. Across all three matrices, performance differences were small and inconsistent. In some configurations, ARIES produced higher SP scores on more datasets, while in others, the MAFFT variants did. However, none of these differences were statistically significant (corrected one-sided Wilcoxon signed-rank test p-values all > 0.1), shown in Fig. 18. Thus, varying the substitution matrix did not materially affect the relative performance of the methods.

Across all tested configurations, changing substitution matrices did not alter the conclusions of our study.

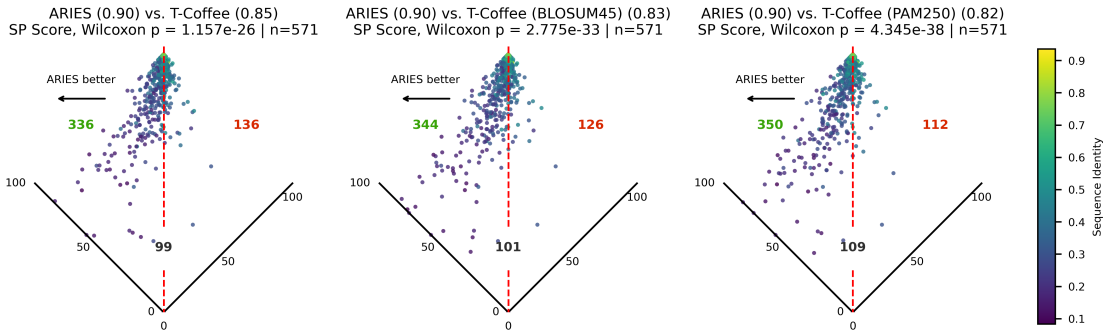

Fig. 17. SP scores across all HOMSTRAD sets of ARIES and T-Coffee with different substitution matrices. ARIES significantly outperforms T-Coffee in all tested configurations.

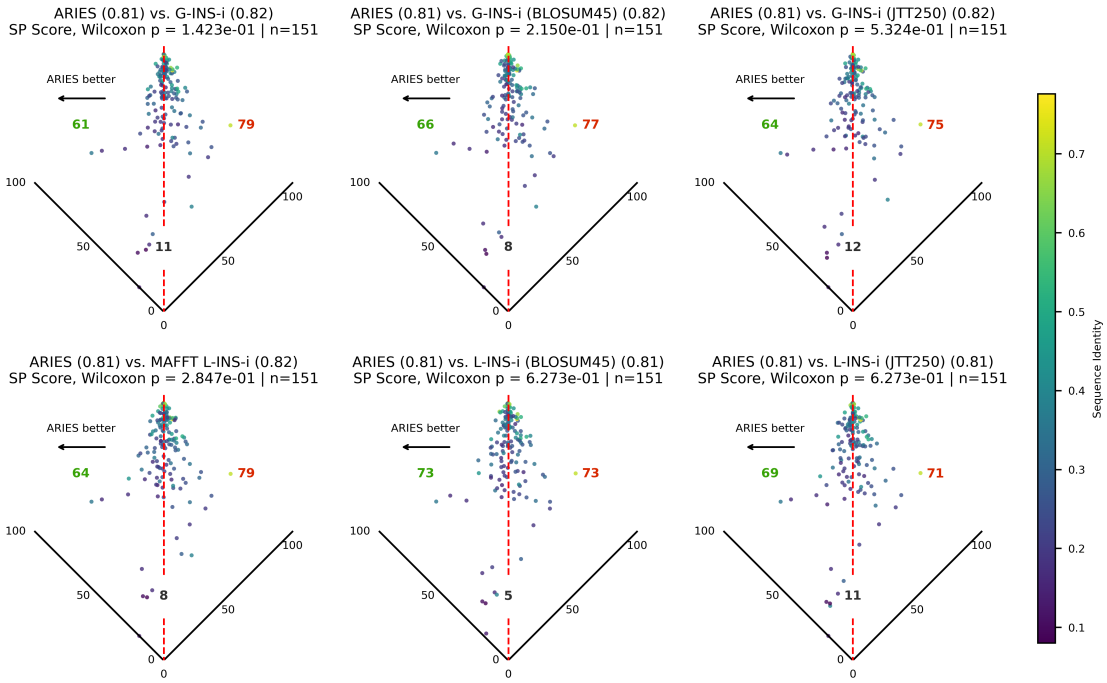

Fig. 18. SP scores across all QuanTest2 sets of ARIES, MAFFT L-INS-i, and MAFFT G-INS-i with different substitution matrices. p-values shown correspond to one-sided Wilcoxon signed-rank tests revisefor the method with the higher average SP score outperforming the other, with a Holm-Bonferroni correction.

Comparing star-alignment ARIES with progressive-alignment ARIES.

In our PLM-based progressive alignment framework, one must construct an embedding-level representation for each intermediate aligned cluster before proceeding to the next merge; this is due to the DTW procedure taking as input amino acid embeddings. Thus, this requires generating a new embedding representation at each stage in the progressive alignment. We implemented two progressive alignment variants of ARIES. Specifically, we used the ClustalW guide tree to determine the order of alignment. Initially, each sequence is treated as an individual cluster. At every progressive step, a pair of clusters is aligned via dynamic time warping (DTW) and merged into a new cluster. The embedding of the newly formed cluster is computed in one of two ways: (1) re-embedding the aligned protein sequences with the PLM, where gap characters are replaced by 'X' tokens, similar to how ARIES synthesizes its template embeddings (we refer to this progressive alignment version as "Re-Embed"), or (2) simply averaging the embeddings of aligned residues, while keeping insertion embeddings unchanged from their original sequence (we refer to this version as "Mean-Merge"). When aligning clusters, gaps are handled conservatively: once introduced, a gap is preserved according to its earliest occurrence.

Fig. 19 presents volcano plots comparing the SP scores achieved by star-alignment ARIES compared to these variants on all BAliBASE alignment sets. ARIES with star-alignment achieves an average SP score of 0.72, significantly outperforming both the Re-Embed progressive alignment (average SP score 0.68, $p = 5.20e^{-10}$) and the Mean-Merge progressive alignment (average SP score 0.69, $p = 1.32e^{-2}$). We stress that we do not claim that progressive alignment strategies are inherently inferior. Alternate approaches for cluster embedding generation, or incorporating strategies such as iterative refinement, may improve progressive alignment performance and would be a promising direction for future work.

On the optimal number of medoids for ARIES template synthesis.

To assess whether $K = \lceil \ln(N) \rceil$ serves as an effective heuristic that frequently yields near-optimal performance, we reproduce Fig. 5 for all 151 QuanTest MSA sets, subsampling each alignment to the first $N = 250, 500, 750, 1000$ sequences. The y-axis is normalized against the mean SP score of the best K to compare across different values of N. Although $K = \lceil \ln(N) \rceil$ (highlighted in red) does not always attain the

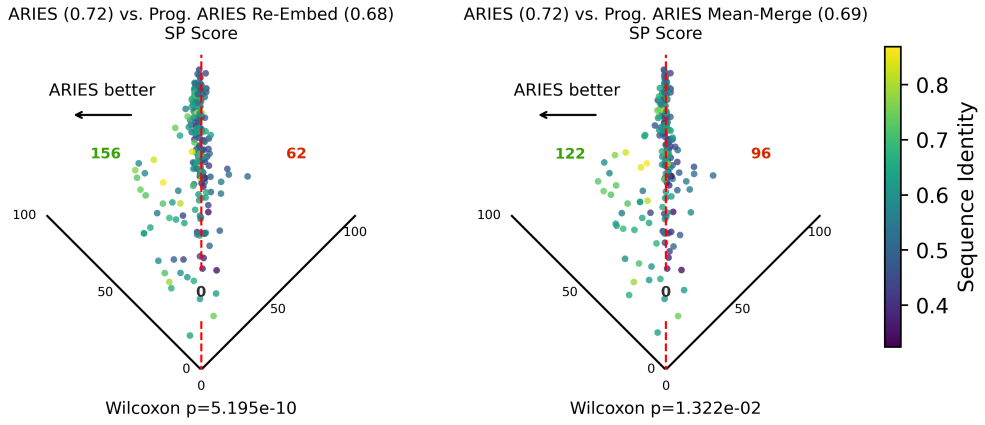

Fig. 19. Average SP score (across all BAliBASE sets) of ARIES with star- vs. two progressive alignment strategies.

absolute best performance, it is within 1.6% of the best performance for $K \in \{1, 2, \ldots, 20\}$ across all values of N.

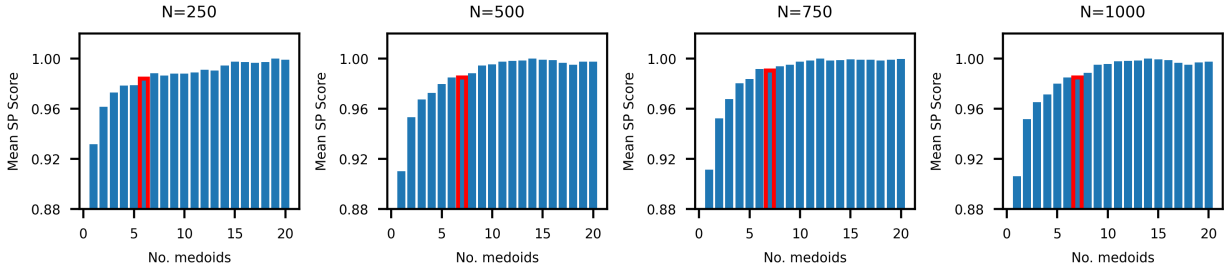

Fig. 20. Normalized average SP score (across all BAliBASE sets) of ARIES with different no. top-K medoids ($K \in \{1, 2, \ldots, 20\}$) for template construction. $K = \lceil \ln(N) \rceil$ is higlighted in red.

Runtime comparison at different input sizes.

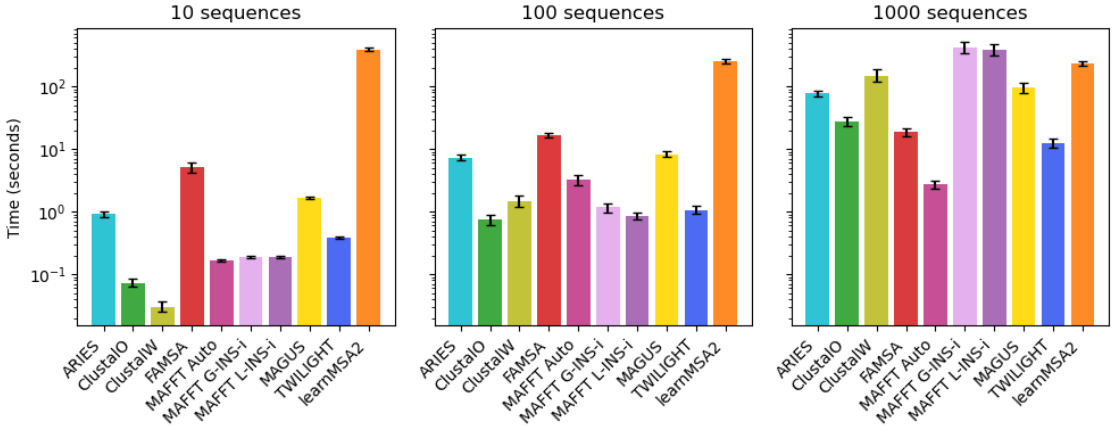

Fig. 21. Runtime comparison of 10 MSA methods on QuanTest2 sets of 10, 100, and 1000 sequences.

Among the methods able to produce alignments of 1000 sequences in under 10 minutes on average, we measured runtime on sequence sets of size 10, 100, and 1000 (Fig. 21). These sets are the first N sequences in each QuanTest2 reference alignment set. We note again that each baseline was tested using the default arguments to maintain consistency. Some tools (e.g., MUSCLE) have faster modes, but we did not enable these non-default modes to ensure fair comparison across tools. All CPU-based baseline methods were run with 10 CPU cores and 10 GB of memory per-core. GPU-accelerated methods (ARIES and learnMSA2) were executed on a single NVIDIA A100 GPU with 80 GB of memory and the same CPU limits. For ARIES, we select the ESM-2 model with 650M parameters with embedding depth 9 based on our ablation results, despite the higher runtime compared to smaller or shallower configurations. Importantly, even with this more expensive setting, our method remains substantially faster than all other GPU-based competitor methods.

ARIES scales approximately linearly with N, as do several of the CPU-based baselines. LearnMSA2, which is the only other GPU-enabled method we evaluate, has a runtime that scales linearly with N but requires roughly the same runtime on small sequence sets as it does on sets of 1000 sequences likely because it struggles to converge when only a few samples are available.

Runtime comparison using ARIES for varying embedding depths and dimensions. Fig. 22 examines the impact of model size and embedding depth on the runtime of ARIES. We compare two ESM variants, with 150M and 650M parameters, respectively. The results indicate that, within the same class of underlying PLM, ARIES runtime scales linearly with both the number of model parameters and the embedding depth.

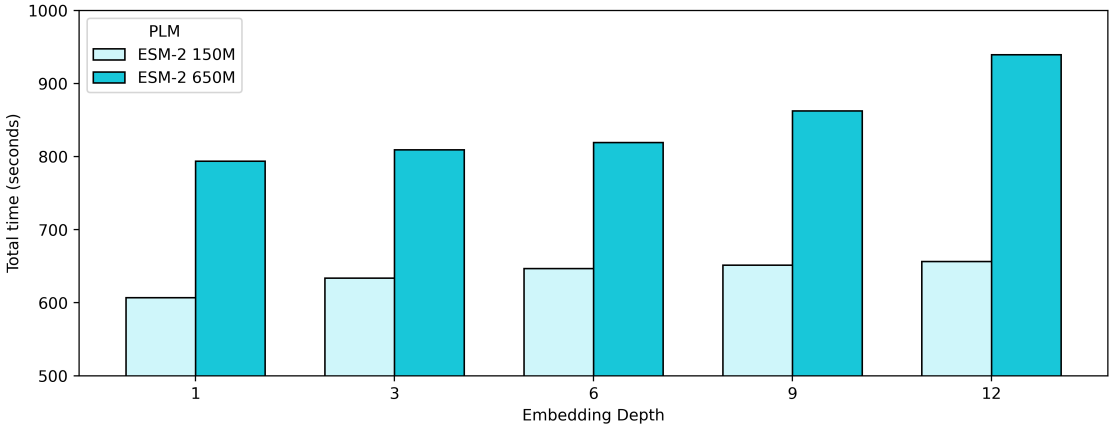

Fig. 22. Runtime comparison for the entire BAliBASE dataset. Larger PLMs are substantially slower, and runtime increases further with embedding depth, though to a lesser extent.