Improved Multiple Sequence Alignments using Coupled Pattern Mining

K. S. M. Tozammel Hossain[†], Debprakash Patnaik[#], Srivatsan Laxman[‡], Prateek Jain[‡], Chris Bailey-Kellogg[§],and Naren Ramakrishnan[†]

[†]Department of Computer Science, Virginia Tech, Blacksburg, VA 24061, USA
[#]Amazon Inc., Seattle, WA 98109, USA
[‡]Microsoft Research, Bangalore 560080, India
[§]Department of Computer Science, Dartmouth College, Hanover, NH 03755, USA

tozammel@vt.edu, patnaik@vt.edu, slaxman@microsoft.com, prajain@microsoft.com, cbk@cs.dartmouth.edu, naren@cs.vt.edu

ABSTRACT

We present ARMiCoRe, a novel approach to a classical bioinformatics problem, viz. multiple sequence alignment (MSA) of gene and protein sequences. Aligning multiple biological sequences is a key step in elucidating evolutionary relationships, annotating newly sequenced segments, and understanding the relationship between biological sequences and functions. Classical MSA algorithms are designed to primarily capture conservations in sequences whereas couplings, or correlated mutations, are well known as an additional important aspect of sequence evolution. (Two sequence positions are coupled when mutations in one are accompanied by compensatory mutations in another). As a result, better exposition of couplings is sometimes one of the reasons for handtweaking of MSAs by practitioners. ARMiCoRe introduces a distinctly pattern mining approach to improving MSAs: using frequent episode mining as a foundational basis, we define the notion of a coupled pattern and demonstrate how the discovery and tiling of coupled patterns using a max-flow approach can yield MSAs that are better than conservation-based alignments. Although we were motivated to improve MSAs for the sake of better exposing couplings, we demonstrate that our MSAs are also improvements in terms of traditional metrics of assessment. We demonstrate the effectiveness of ARMiCoRe on a large collection of datasets.

Categories and Subject Descriptors

J.3 [**Life and Medical Sciences**]; I.5.2 [**Design Methodology**]: Pattern analysis

General Terms

Algorithms

Keywords

Multiple sequence alignment, coupled residues, pattern set mining, coupled patterns, max-flow problems, bioinformtics.

ACM-BCB '12, October 7-10, Orlando, FL, USA

1. INTRODUCTION

Evolutionary pressures on genes and proteins have constrained their (DNA and protein) sequences over generations. As organisms evolve through sequence modifications, mutations that have been evolutionarily selected for survival would be preserved in the sequence record. It is hence of intrinsic biological interest to inspect the sequence record and to unravel those mutations that have withstood the test of time and have been beneficial to the species.

Multiple sequence alignment (MSA) of biological sequences is a classical approach to understand evolutionary constraints. It has been said that "one or two homologous sequences whisper, ..., a full [MSA] shouts out loud" [13]. A plethora of MSA algorithms exists with origins ranging from discrete algorithms [26] to probabilistic models, such as HMMs [15].

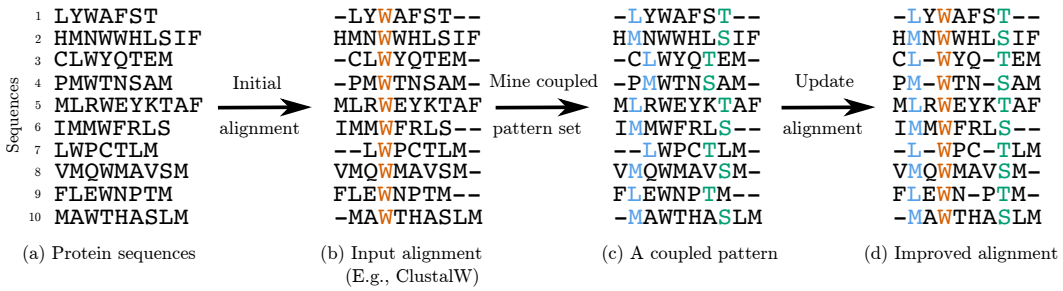
Isn't MSA a Solved Problem?

Although sequence alignment has become a widely deployed tool in bioinformatics, practically every MSA algorithm (e.g., ClustalW [26], Muscle [8], T-Coffee [18], and more) is designed to model and expose conservation, which although being a key evolutionary constraint, does not capture the richness of how sequences evolve and diverge. As seen in Fig. 1, two key forms of constraints are conservation and coupling. Column 4 of Fig. 1 (d) illustrates a conserved column, i.e., all residues are 'W.' Columns 2 and 8 of Fig. 1 (d) illustrate coupling, or compensatory mutations: whenever column 2 is 'L,' column 8 is 'T'; similarly whenever column 2 is 'M,' column 8 is 'S.' In a typical alignment (e.g., Fig. 1 (b)), conservations are manifest and couplings are obscured; in fact, it is often accepted practice for biologists to 'hand tweak' such an alignment to incorporate structural information about sequences and thus obtain a better alignment.

Such tweaking is still somewhat of a black art and requires significant domain expertise. We were motivated to design an automated approach to better expose couplings in an MSA; but in doing so, our approach also improves MSAs according to traditional measures of assessment.

Contributions

- We present Alignment Refinement by Mining Coupled Residues (ARMiCoRe), a pattern mining approach to the problem of multiple sequence alignment. Using frequent episode mining as a foundational basis, we define the notion of a *coupled pattern* that elucidates covarying residues.



(a) Protein sequences (b) Input alignment (E.g., ClustalW) (c) A coupled pattern (d) Improved alignment

Figure 1: Realignment of a hypothetical MSA using coupled pattern mining. Panel (b) is input to ARMiCoRe and (d) is the improved alignment.

Coupled patterns are inferred using a levelwise approach and subsequently 'tiled' using a max-flow algorithm. The tiling is then used to direct the adjustment of a conservation based alignment to capture covarying residues.

- ARMiCoRe can be viewed as a novel application of *pattern set* discovery [2] where the goal is not just to mine interesting patterns (which is the purview of pattern discovery) but to select among them to optimize a set-based measure. ARMiCoRe can be used to tweak alignments from any existing algorithm, to better expose couplings.
- As MSA is an established topic in bioinformatics, we subject ARMiCoRe to a thorough experimental evaluation involving 108 protein families. We identify selective superiorities of ARMiCoRe and demonstrate situations where it outperforms state-of-the-art MSA algorithms.

2. RELATED WORK

Multiple sequence alignment has been studied extensively for the past several decades (see [5] for reviews). A rich set of features exist to classify MSA algorithms. One way to classify them is as global alignment vs. local alignment algorithms. Global alignment algorithms (e.g., ClustalW [26], MUSCLE [8], T-coffee [18], and ProbCons [6]) match sequences over their full lengths, whereas local alignment algorithms (e.g., DIALIGN [16]) aim to align only the most similar regions between sequences. A second way to classify algorithms is in terms of the objective function (e.g., sum of pairs score, entropy, circular sum) used to identify the highest scoring alignment [5]. Finally, MSA algorithms can be classified based on their underlying optimization scheme: exact algorithms, progressive algorithms, and iterative algorithms. An exact algorithm attempts to simultaneously align all of the sequences and find an optimal alignment using an objective function [3]. The underlying problem has been proved to be NP-complete [28] and, hence, impractical for large numbers of sequences.

Progressive and Iterative Algorithms: Heuristic approaches to MSA are either progressive or iterative algorithms. Progressive alignment algorithms (e.g., ClustalW [26] and T-Coffee [18]), typically more appealing, involve building a guide tree based on sequence similarity and progressively aligning sequences following the order of the guide tree. Variants on progressive alignment typically use guide tree reestimation, modifying objective functions, and/or post-processing [5]. In guide tree reestimation, algorithms compute new distance matrices based on the initial MSA produced by progressive alignment, and the revised distance matrix is used to create a new guide tree. MUSCLE [8] and PROMALS [20] use this approach. Methods that modify the objective function are referred to as consistency-based methods, e.g., T-Coffee [18], DIALIGN [16], ProbCons [6], and PROMALS [20]. The third variant involves post-processing, also known as iterative algorithms. In this approach, an alignment is first produced rapidly and then refined through a series of iterations until no more improvements can be made [5]. Examples are MUSCLE [8] and DIALIGN [16].

Probabilistic Algorithms: Probabilistic algorithms approach MSA by modeling different aspects: evolutionary models of indels , profile models, and hybrid models that combine probabilistic models with progressive alignment techniques. ProbCons [6] is a well known example that uses maximum expected accuracy scoring to infer a model and is especially useful for divergent sequences. A second example [15] uses a pair of HMMs as the scoring strategy.

Constraint-based Algorithms: These approaches (a.k.a. segment-based alignment algorithms) improve alignment quality by searching and incorporating information about homologs, conserved motifs/domains, and expert-supplied feedback about local similarity. Examples are COBALT [19], DIALIGN [16], and PROMALS [20].

As rich as the above landscape of MSA algorithms is, none of the above algorithms use covariation as a property to align sequences. Coupling is often viewed as a feature that 'comes out' of an alignment as opposed to a criterion or driver for computing the alignment. A very recent work, published in 2010 [11], is the lone exception which uses mutual information to detect coupled residues, and uses constraint programming to realign sequences. As we will show, ARMiCoRe captures not just coupled residues but the richer class of coupled patterns that tile the entire set of sequences; this greater expressiveness leads to improved MSAs, both in terms of exposing couplings, and in terms of traditional metrics of assessment (see Section 5).

3. FORMULATION

We are given a collection $\mathcal{S} = \{s_1, \ldots, s_n\}$ of n aligned sequences (or strings), each of length m, over a finite alphabet. As shown in Fig. 1 (b), the sequences in $\mathcal{S}$ are assumed to have been aligned by a standard MSA method that typically favors conservation (and thus might contain gaps). Each sequence s_i, $i = 1, \ldots, n$, can hence be expressed as $s_i = \langle E_1^i, \ldots, E_m^i \rangle$, $E_j^i \in \mathcal{E} \cup \{\varphi\}$, $j = 1, \ldots, m$, where $\mathcal{E}$ denotes a finite alphabet and φ is the gap symbol. In the case of DNA sequences, $\mathcal{E} = \{A, C, T, G\}$, whereas for protein sequences, $\mathcal{E}$ comprises the 20 amino acid residues. We can even for instance denote amino acids by their physicochemical properties so that the set of 20 amino acids can be reduced to a smaller set of properties.

DEFINITION 1. *An* indexed pattern α *(of size* ℓ*) is defined by a pair of* ℓ*-length sequences,* $(\langle A_1^\alpha, \ldots, A_\ell^\alpha \rangle, \langle \delta_1^\alpha, \ldots, \delta_\ell^\alpha \rangle)$,

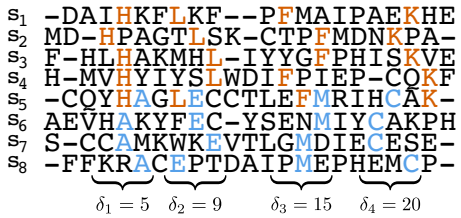


Figure 2: Figure illustrating Example 2.

where each $A_j^\alpha \in \mathcal{E}$, $\delta_j^\alpha \in \mathbb{Z}^+$, $j = 1, \ldots, \ell$, *and* $\delta_{j+1}^\alpha > \delta_j^\alpha$, $j = 1, \ldots, (\ell - 1)$. *We refer to* $\langle \delta_1^\alpha, \ldots, \delta_\ell^\alpha \rangle$ *as the* sequence of positions *over which* α *is defined.*

The semantics of an indexed pattern α is essentially that in a sequence s where α is said to occur, we expect that A_j^α will appear at position δ_j^α (or very close to it) for every $1 \leq j \leq \ell$.

Definition 2. *A sequence* $s = \langle E_1, \ldots, E_m \rangle$ *is said to contain an* ϵ-approximate occurrence *of indexed pattern* α *if there exists a map* $h : \{1, \ldots, \ell\} \rightarrow \{1, \ldots, m\}$, *strictly increasing, such that* $\forall j, 1 \leq j \leq \ell$, $E_{h(j)} = A_j^\alpha$ *and* $|h(j) - \delta_j^\alpha| \leq \epsilon$.

Example 1. $\alpha = (\langle A, E, M, C \rangle, \langle 5, 9, 15, 20 \rangle)$ *is an indexed pattern of size* $\ell = 4$. *An example sequence* s *that contains an* ϵ*-approximate occurrence of* α *is shown below (for* $\epsilon = 1$*). Note that occurrences of symbols A, E, M and C can be found within 1 position of the locations 5, 9, 15 and 20 respectively.*

$$s = \langle \text{KFF}\underbrace{\text{KR A}}_{\delta_1=5}\text{C}\underbrace{\text{E PT}}_{\delta_2=9}\text{DA I}\underbrace{\text{P M E}}_{\delta_3=15}\text{PH}\underbrace{\text{EM C}}_{\delta_4=20}\text{PE} \rangle$$

Definition 3. *The* ϵ-support *of an indexed pattern* α *over the collection* $\mathcal{S}$ *of sequences, denoted* $f_\epsilon(\alpha)$, *is the number of sequences in* $\mathcal{S}$ *that contain* at least *one* ϵ*-approximate occurrence of* α*; the corresponding set of* ϵ-supporting sequences *is denoted by* $\mathcal{U}_\epsilon(\alpha) \subseteq \mathcal{S}$, $f_\epsilon(\alpha) = |\mathcal{U}_\epsilon(\alpha)|$.

Definition 4. *A* coupled pattern, ψ, *of size* k *is defined as a* k*-tuple,* $(\alpha_1, \ldots, \alpha_k)$, *where each* α_i, $i = 1, \ldots, k$ *(referred to as a* constituent *of* ψ*) is an indexed pattern over a* common *sequence of positions* $\langle \delta_1, \ldots, \delta_\ell \rangle$. *The* ϵ-support *of* ψ *over a collection* $\mathcal{S}$ *of sequences, denoted* $F_\epsilon(\psi)$, *is defined as the total number of* ϵ*-supporting sequences of its constituents found in* $\mathcal{S}$, *i.e.,* $F_\epsilon(\psi) = |\cup_{\alpha_i \in \psi} \mathcal{U}_\epsilon(\alpha_i)|$.

Example 2. *Consider the collection of sequences,* $\mathcal{S} = \{s_1, \ldots, s_8\}$, *defined in Figure 2.* $\psi = (\alpha_1, \alpha_2)$ *is an example coupled pattern of size 2, where* $\alpha_1 = (\langle H, L, F, K \rangle, \langle 5, 9, 15, 20 \rangle)$ *and* $\alpha_2 = \langle A, E, M, C \rangle, \langle 5, 9, 15, 20 \rangle$ *are indexed patterns over the same sequence of positions* $\langle 5, 9, 15, 20 \rangle$. *The* ϵ*-support of* ψ *over* $\mathcal{S}$, *for* $\epsilon = 1$, *is* $F_1(\psi) = 8$.

Our main intuition here is that when there is enough evidence for a coupled pattern ψ in a given data set $\mathcal{S}$, the associated sequence of positions $(\delta_1, \ldots, \delta_\ell)$ are *coupled* across multiple sequences of $\mathcal{S}$, in the sense that, mutations in one position are accompanied by corresponding mutations in the others. In *Example 2*, mutations of H to A in position 5, would be accompanied by three other mutations, namely, L to E in position 9, F to M in position 15 and K to C in position 20. To facilitate the detection and measurement of the evidence for a coupled pattern, we define the notion of τ-coverage with respect to the pattern's ϵ-supporting sequences.

Definition 5. *Let* $\mathcal{S}$ *be a given collection of sequences over* $\mathcal{E} \cup \{\varphi\}$. *Consider a coupled pattern* $\psi = (\alpha_1, \ldots, \alpha_k)$ *and its corresponding sets,* $\mathcal{U}_\epsilon(\alpha_i), i = 1, \ldots, k$, *of* ϵ*-supporting sequences. The* τ*-coverage of* ψ *in* $\mathcal{S}$ *with respect to its* ϵ*-supporting sequences, denoted* $\Gamma_\epsilon(\psi, \tau)$, *is defined as follows:*

$$\Gamma_\epsilon(\psi, \tau) = \max_{\mathcal{D}_1, \ldots, \mathcal{D}_k} \sum_{i=1}^{k} |\mathcal{D}_i| \tag{1}$$

where $\mathcal{D}_i \subset \mathcal{S}$, $i = 1, \ldots, k$, *such that the following hold:* $\mathcal{D}_i \subset \mathcal{U}_\epsilon(\alpha_i)$, $\mathcal{D}_i \cap \mathcal{D}_j$ *is* empty *for* $i \neq j$, *and* $|\mathcal{D}_i| \geq \tau$.

Essentially, we want to compute mutually exclusive sets of ϵ-supporting sequences for each of the k constituents of ψ, such that each mutually exclusive set contains at least τ sequences, while the total number of distinct sequences in these sets is maximized.

Example 3. *For the same example as before, with* $\epsilon = 1$, *we get the following sets of* ϵ*-supporting sequences for* α_1 *and* α_2*:* $\mathcal{U}_\epsilon(\alpha_1) = \{s_1, s_2, s_3, s_4, s_5\}$ *(*$f_1(\alpha_1) = 5$*) and* $\mathcal{U}_\epsilon(\alpha_2) = \{s_5, s_6, s_7, s_8\}$ *(*$f1(\alpha_2) = 4$*). Setting* $\mathcal{D}_1 = \{s_1, s_2, s_3, s_4\}$ *and* $\mathcal{D}_2 = \{s_5, s_6, s_7, s_8\}$ *we get the 4-coverage of* ψ *with respect to its 1-supporting sequences to be* $\Gamma_1(\psi, 4) = 8$.

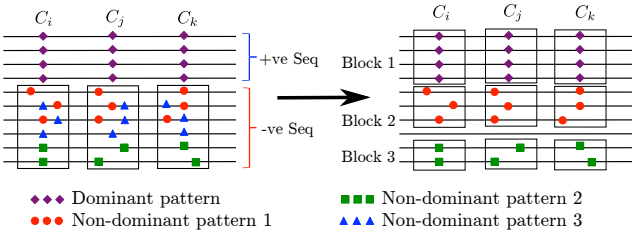


Figure 3: Generating a coupled pattern set from all possible patterns.

There are two main challenges in the detection and use of coupled patterns for improving multiple sequence alignment. First, given a data set $\mathcal{S}$ of (approximately aligned) sequences, we need to find coupled patterns which have high τ-coverage over $\mathcal{S}$. Second, we need to use the high-coverage coupled patterns discovered to improve the MSA relative to the original alignment in $\mathcal{S}$.

Problem 1 (Mining Coupled Patterns). *Consider a data set* $\mathcal{S}$ *of* m*-length sequences over* $\mathcal{E} \cup \{\varphi\}$ *and a fixed sequence of position indices,* $\langle \delta_1, \ldots, \delta_\ell \rangle$. *Given user-defined parameters,* ϵ, K *and* τ *(all non-negative integers) find a coupled pattern of size* $k \leq K$ *over* $\langle \delta_1, \ldots, \delta_\ell \rangle$ *which maximizes* τ*-coverage with respect to its* ϵ*-supporting sequences in* $\mathcal{S}$.

The MSA realignment problem can be stated as follows.

Problem 2 (MSA Realignment). *Given a data set* $\mathcal{S}$ *of* m*-length sequences over* $\mathcal{E} \cup \{\varphi\}$ *and a set of coupled patterns* $\mathbf{\Psi} = \{\psi\}$ *in* $\mathcal{S}$ *each of which has* τ*-coverage of* $\Gamma_\epsilon(\psi, \tau) = \gamma$ *over* ϵ*-supporting sequences, find a* realignment $\mathcal{S}'$ *of the sequences in* $\mathcal{S}$ *where all patterns in* Ψ *have a* τ*-coverage of* $\Gamma_{\epsilon'}(\psi, \tau) \geq \gamma$ *for* $\epsilon' < \epsilon$.

In the above formulation, note that we require coupled patterns discovered in the original (approximate) alignment to still be manifest in the new alignment, but in a more obvious manner. Ideally $\epsilon' = 0$ (which is the situation for the example pattern in Fig. 1 (d)) but in practice we aim to obtain $\epsilon' < \epsilon$.

Algorithm 1 CP-MINER($\mathcal{S}, \mathbf{\Psi}^l, \tau_d, \tau, \epsilon, K$)

Input: A set of aligned sequences $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$, a set of frequent coupled patterns $\mathbf{\Psi}^l$ of size l, dominant residue conservation threshold τ_d, block coverage threshold τ, column-window parameter ϵ, maximum size of a coupled pattern, K.
Output: A set of frequent coupled patterns $\mathbf{\Psi}^{l+1}$ of size $l+1$.

1. $\mathbf{\Psi}^{l+1} \leftarrow \phi$
2. $\mathbf{C}^{l+1} \leftarrow$CANDIDATE-GEN($\mathbf{\Psi}^l$)
3. $\mathbf{\Psi}_1^{l+1} \leftarrow \{\psi : \psi_{dom} = \{\alpha\}, \forall \alpha \in \mathbf{C}^{l+1}\}$
4. **for** $\psi \in \mathbf{\Psi}_1^{l+1}$ **do**
5. $\quad \alpha \leftarrow \psi_{dom}$ $\quad \triangleright$ dominant indexed pattern.
6. $\quad \mathcal{S}^+ \leftarrow \{s_i : s_i$ has an ϵ-approx. occurence of $\alpha\}$
7. $\quad$ **if** $|\mathcal{S}^+| \geq n\tau_d$ **then**
8. $\quad\quad \mathcal{S}^- \leftarrow \mathcal{S} - \mathcal{S}^+$
9. $\quad\quad \mathcal{I} \leftarrow \forall\epsilon$-approximate indexed patterns from $\mathcal{S}^-$
10. $\quad\quad \mathcal{I}' \leftarrow \{\alpha : f_\epsilon(\alpha) \geq \tau, \forall \alpha \in \mathcal{I}\}$
11. $\quad\quad$ **if** $\mathcal{I}' \neq \phi$ and $|\mathcal{I}'| \leq K$ **then**
12. $\quad\quad\quad \psi \leftarrow \psi \cup \mathcal{I}'$
13. $\quad\quad\quad$ **if** ψ is significant **then**
14. $\quad\quad\quad\quad \mathbf{\Psi}^{l+1} \leftarrow \mathbf{\Psi}^{l+1} \cup \psi$
15. **return** $\mathbf{\Psi}^{l+1}$

4. ALGORITHMS

In this section, we present ARMiCoRe, a new method for aligning multiple sequences based on coupling relationships that may exist between residues found in two or more sequence positions. The method consists of two main steps. We start by discovering high-support coupled patterns over various choices of position sequences (described in Sec. 4.1). Finally, in Sec. 4.3, we derive an alternative alignment $\mathcal{S}'$ for $\mathcal{S}$ based on both the original ungapped sequences and the just-discovered coupled patterns.

4.1 Discovering Coupled Patterns

The first step of ARMiCoRe is to choose the sequence positions over which to mine coupled patterns. Then standard level-wise methods (Apriori) are used to discover coupled patterns (restricted to the chosen sequence positions) with sufficient support (cf. Sec. 4.1.1). While level-wise searching for coupled patterns ARMiCoRe looks for patterns that have at most K constituents ignoring τ-coverage (cf. Sec. 4.1.2). Then ARMiCoRe applies a statistical significance test to filter out uninteresting coupled patterns (cf. Sec. 4.1.3). This gives us the pattern set, $\mathbf{\Psi}^l = \{\psi_1, \ldots, \psi_{|\mathbf{\Psi}|}\}$, of ℓ-size indexed patterns, each with support at least τ, each has at most K constituents, and each defined over a common sequence of positions, $\langle \delta_1, \ldots, \delta_\ell \rangle$. Each subset of indexed patterns in ψ can thus be a potential candidate for a τ-coverage coupled pattern. Finally, ARMiCoRe applies a max-flow approach to get the τ-coverage of each ψ (cf. Sec. 4.1.4).

A lower-bound τ on the sizes $|\mathcal{D}_i|$ of the blocks corresponding to each constituent of a coupled pattern (see *Definition 5*) automatically enforces an upper-bound $\lfloor \frac{n}{\tau} \rfloor$ on the size, k, the coupled pattern. At first, it might appear as if the user only needs to prescribe τ to detect interesting patterns (since an upper-bound on k is implied). However, we have observed that in the couplings that are already known in biological data sets, the number of constituents are typically far fewer than $\lfloor \frac{n}{\tau} \rfloor$. Hence, in our framework, the user must specify both an upper-bound K for k as well as a lower-bound τ on the block-sizes $|\mathcal{D}_i|$ of coupled patterns.

We now describe the steps in ARMiCoRe for finding a subset of indexed patterns that implies a coupled pattern, of size at most K, and which maximizes the τ-coverage over its ϵ-supporting sequences. The main hardness in the problem arises from having to maximize coverage with a τ constraint while restricting the number of constituent patterns to no more than K. Hence, we decouple the two problems and show that the individual problems can be solved efficiently. Specifically, we show that by ignoring the τ constraint, the problem of maximizing coverage is a sub-modular function-maximization problem with cardinality constraint. We propose Algorithm 1,2 for generating all possible coupled patterns of size at most K. On the other hand, after selecting coupled patterns of size at most K, maximizing coverage with the τ constraint reduces to a max-flow problem.

4.1.1 Level-wise Coupled Pattern Mining

Our basic idea here is to organize the search for coupled patterns around the (semi) conserved columns of the current alignment. Level 1 patterns are comprised of individual columns, level 2 patterns are comprised of pairs of level 1 patterns, and so on.

For choosing a (semi) conserved column, we employ a *dominant residue conservation threshold* τ_d (see Line 7 of Algorithm 1). We use class-based conservation so that amino acid residues that have similar physico-chemical properties are considered conserved. Class-based conservation can be estimated using the Taylor diagram [23] or by k-means clustering of substitution matrices such as Blosum62 [10]. We have explored both approaches and found the latter to work better (with a setting of 7 non-overlapping clusters).

Amino acids in and around the semi-conserved columns (to within a window length of ϵ) are organized into positive and negative sets of sequences describing the dominant combination and other, non-dominant, ones (see Fig. 3 (left)). As we construct level-2 and greater patterns, we take care to ensure that ϵ does not yield window lengths that cross another semi-conserved column.

Algorithm 2 CANDIDATE-GEN($\mathcal{S}, \mathbf{\Psi}^l$)

Input: A set of frequent coupled patterns $\mathbf{\Psi}^l$ of size l.
Output: A set of indexed patterns $\mathbf{C}^{l+1}$ of size $l+1$.

1. $\mathbf{C}^{l+1} \leftarrow \phi$
2. $\mathcal{A}^l \leftarrow \{\alpha : \alpha = \psi_{dom}, \forall \psi \in \mathbf{\Psi}^l\}$ $\quad \triangleright$ ψ_{dom} denotes an indexed pattern of the most frequent residue.
3. **for all** $\alpha_i, \alpha_j \in \mathcal{A}^l$ **do**
4. $\quad$ **if** there is a prefix match of length $l-1$ between δ^{α_i} and δ^{α_j} **then**
5. $\quad\quad \alpha_k \leftarrow$ Merge(α_i, α_j)
6. $\quad\quad$ **for all** $\alpha_t \in \mathcal{A}^l$ and α_k containing α_t **do**
7. $\quad\quad\quad \alpha_k^{sub} \leftarrow \alpha_t$ $\quad \triangleright$ listing subpatterns
8. $\quad\quad \mathbf{C}^{l+1} \leftarrow \mathbf{C}^{l+1} \cup \alpha_k$
9. **return** $\mathbf{C}^{l+1}$

4.1.2 High ϵ-support using at most K Constituents

We now present the approach taken by ARMiCoRe to solve the problem of maximizing coverage by enforcing only the upper-bound K (user-defined) on the number of constituents of ψ while ignoring the τ constraint. We will test for τ-coverage later as a post-processing step (see Sec. 4.1.4).

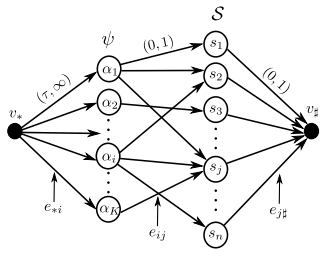


Figure 4: Network $\mathcal{G}$ used in the max-flow step.

Note that at $\tau = 0$, τ-coverage is same as ϵ-support, and this can be shown to be both monotonic and *sub-modular* with respect to its constituents. That is, if $\mathcal{A}$ and $\mathcal{B}$ are two subsets of ψ, such that $\mathcal{A} \subset \mathcal{B}$, then it can be shown that: $\Gamma_\epsilon(\mathcal{A} \cup \alpha, 0) \geq \Gamma_\epsilon(\mathcal{A}, 0)$, and, $\Gamma_\epsilon(\mathcal{A} \cup \alpha, 0) - \Gamma_\epsilon(\mathcal{A}, 0) \geq \Gamma_\epsilon(\mathcal{B} \cup \alpha, 0) - \Gamma_\epsilon(\mathcal{B}, 0)$. Consequently, we can use a greedy algorithm which guarantees a $(1 - \frac{1}{e})$-approximate solution [17]. In other words, we would find a subset of ψ whose ϵ-support (or 0-coverage) is within a factor of $(1 - \frac{1}{e})$ of the optimal subset.

4.1.3 Significance Testing of Coupled Patterns

For level-2 patterns and greater, we perform a 2-fold significance test, the first focusing on the dominant pattern and the second focusing on the non-dominant patterns. For the dominant pattern, we compute the probability, and thus the p-value, of encountering the dominant pattern given the column marginals. For the non-dominant patterns, we conduct a standard enrichment analysis using the hypergeometric distribution to determine if the symbols in the non-dominant pattern are over-represented.

4.1.4 Checking τ-coverage using Max-Flow

Once we have generated ψ with high ϵ-support we proceed to check if a non-zero τ-coverage is feasible (Recall that the coverage will either be zero or the full ϵ-support corresponding to the chosen subset of ψ). This problem reduces to a standard max-flow problem for which efficient (poly-time) algorithms exist. We now present the reduction of this problem to max-flow (see Fig. 4).

Let $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ be a network with $v_*, v_\sharp \in \mathbf{V}$ denoting the source and sink of $\mathcal{G}$ respectively. In addition to v_* and $v_\sharp$, there is a unique node in $\mathbf{V}$ corresponding to each indexed pattern $\alpha_i \in \psi$ and also to each sequence $s_j \in \mathcal{S}$, i.e., $\mathbf{V} = \{v_*, v_\sharp\} \cup \psi \cup \mathcal{S}$. Three kinds of edges are in set $\mathbf{E}$:

1. $e_{*i} \in \mathbf{E}$, representing an edge from the source node v_* to the pattern node, $\alpha_i \in \mathbf{V}$. We will have $e_{*i} \in \mathbf{E}$, $\forall \alpha_i \in \psi$
2. $e_{j\sharp} \in \mathbf{E}$, representing an edge from the sequence node $s_j \in \mathbf{V}$ to the sink node $v_\sharp$. We will have $e_{j\sharp} \in \mathbf{E}$, $\forall s_j \in \mathcal{S}$
3. $e_{ij} \in \mathbf{E}$, representing an edge from pattern node $\alpha_i \in \mathbf{V}$ to the sequence node $s_j \in \mathcal{S}$, whenever the algorithm assigns s_j to $\mathcal{D}_i$ (see *Definition 5*). We will have $e_{ij} \in \mathbf{E}$, $\forall \alpha_i \in \psi, s_j \in \mathcal{S}$ such that s_j is assigned to the block $\mathcal{D}_i$ that corresponds to the i^{th} pattern $\alpha_i \in \psi$.

For any edge $e \in \mathbf{E}$, let $\texttt{LB}(e)$ and $\texttt{UB}(e)$ denote, respectively, the lower and upper bounds on the capacity of edge e. Given a coupled pattern ψ, the computation of its τ-coverage, $\Gamma_\epsilon(\psi, \tau)$, reduces to the computation of max-flow for the network $\mathcal{G}$ under the following capacity constraints:

1. $\texttt{LB}(e_{*i}) = \tau$, $\texttt{UB}(e_{*i}) = \infty$, $\forall \alpha_i \in \psi$
2. $\texttt{LB}(e_{j\sharp}) = 0$, $\texttt{UB}(e_{j\sharp}) = 1$, $\forall s_j \in \mathcal{S}$
3. $\texttt{LB}(e_{ij}) = 0$, $\texttt{UB}(e_{ij}) = 1$, $\forall \alpha_i \in \psi, s_j \in \mathcal{S}$

We can now use any max-flow algorithm, such as [9] to obtain the max-flow in $\mathcal{G}$ subject to the stated capacity constraints. The flow returned will give us $\Gamma_\epsilon(\psi, \tau)$.

4.2 Complexity Analysis

The runtime for finding all possible coupled patterns depends on the number of sequences (n), the alignment length (m), the column-window threshold (ϵ), and the maximum size of the indexed pattern (l). Let p be the number of semi-conserved columns found in level 1 indexed pattern mining. Then the running time for generating all possible coupled patterns is $\mathcal{O}(nm + l(p^3 + lp^2 n\epsilon))$. Since $p \sim \mathcal{O}(m)$, the running time is $\mathcal{O}(l(m^3 + lm^2 n\epsilon))$. Finding a τ-coverage coupled pattern depends on the number of nodes ($\mathcal{O}(n+K)$) and the number of edges (q) in the max-flow network for which the running time is $\mathcal{O}((n+K)q \log((n+K)^2/q))$ [9].

4.3 Updating the Alignment

There are various ways to adjust the given alignment. One strategy is to modify the substitution matrix but this is a global approach and does not lend itself to the local shifting of columns as suggested by coupled pattern sets. We instead adopt a constraint-based alignment strategy, based on COBALT [19], which can flexibly incorporate domain knowledge. Constraints in COBALT are specified in terms of two segments from a pair of sequences that should be aligned with each other in the final result. To convert coupled patterns into constraints, we can adopt various strategies. One approach is to, for each pair of sequences, identify a pair of column positions that should be realigned based on the coupled pattern set. We then map these two positions in the alignment to the corresponding positions in the original (ungapped) sequences. (These two positions in terms of the original sequences thus constitute a segment pair of size one that should be realigned.) Taking all pairs of sequences in this manner would generate a huge number of constraints. We can reduce the number of constraints by considering consecutive pair of sequences. Another approach is to take a subset of sequences, say S_1, for whom the residues match over a column in the coupled pattern. We then take each of the sequences for whom residues do not match over that column in the coupled pattern, and create constraints by pairing the sequence with each of the sequences from S_1. COBALT guarantees a maximal consistent subset of these constraints to be occurred in the final alignment. The runtime for an alignment using COBALT is data-centric [19].

5. EXPERIMENTAL RESULTS

In this section, we assess ARMiCoRe on benchmark datasets. Due to space limitations, we provide only representative results illustrating selective superiorities of ARMiCoRe. Our goals are to answer the following questions:

1. How does ARMiCoRe fare against classical algorithms on benchmark datasets? Here we choose ClustalW and COBALT, two representative MSA algorithms. (see Section 5.3)
2. Can ARMiCoRe extract coupled patterns that capture evolutionary covariation in protein families? (see Section 5.4)
3. Can domain expertise be used to drive the computation of improved alignments? (see Section 5.5)

Table 1: Comparison of ARMiCoRe against ClustalW over all BaliBase datasets (using only core regions). The average scores are shown here. RV20* is curated from RV20 by removing orphan sequences.

Dataset	Q-Score		TC Score		Shift Score		Modeler Score	
	ClustalW	ARMiCoRe	ClustalW	ARMiCoRe	ClustalW	ARMiCoRe	ClustalW	ARMiCoRe
RV12	0.84	**0.89**	0.51	**0.61**	0.73	**0.81**	0.69	**0.79**
RV20	**0.84**	0.79	**0.24**	0.15	**0.83**	0.78	**0.81**	0.78
RV20*	0.88	**0.90**	0.54	**0.57**	**0.88**	**0.88**	0.85	**0.87**
RV30	**0.68**	0.58	**0.23**	0.13	**0.65**	0.58	**0.63**	**0.63**

Table 2: Comparison of ARMiCoRe against ClustalW over the OXBench alignments.

Alignments	Q-Score		TC Score		Shift Score		Modeler Score	
	ClustalW	ARMiCoRe	ClustalW	ARMiCoRe	ClustalW	ARMiCoRe	ClustalW	ARMiCoRe
12s107	**0.99**	0.98	0.80	**0.86**	**0.93**	0.92	**0.87**	**0.87**
12s108	0.97	**0.99**	0.85	**0.94**	0.88	**0.89**	0.79	**0.80**
12t109	**0.96**	**0.96**	0.76	**0.80**	**0.87**	**0.87**	**0.78**	**0.78**
12t113	**0.95**	0.91	**0.82**	0.56	**0.78**	0.76	**0.65**	0.63
12t116	**0.94**	0.87	**0.53**	0.33	**0.76**	0.73	**0.62**	0.61
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
588t28	**1.00**	0.99	**0.97**	**0.97**	**0.89**	**0.89**	0.80	**0.81**
_22s38	**0.95**	**0.95**	**0.82**	0.81	**0.81**	**0.81**	**0.69**	**0.69**
_22t50	**0.96**	0.95	**0.86**	0.83	**0.79**	0.78	**0.64**	**0.64**
__588	**0.98**	**0.98**	**0.83**	0.8	0.88	**0.89**	0.80	**0.81**
___12	0.86	**0.87**	0.00	**0.10**	0.5	**0.53**	0.34	**0.37**

5.1 Datasets

We evaluate our method using three well-known benchmark datasets: BaliBase3 [27], OXBench [21], and SABRE [7]. The BaliBase3 benchmark is created for evaluating both pairwise and MSA algorithms. We use only those alignments from BaliBase that have at least 25 sequences, which yields 48 alignments from three reference sets: RV12, RV20, and RV30. (We chose a threshold of 25 sequences in order to maintain the fidelity of couplings within a sequence family.) The alignments in the reference set RV12 are composed of sequences that are equidistant and have 20-40% identity. The reference set RV20 contains alignments that are composed of highly divergent orphan sequences. The reference set RV30 contains alignments that are composed of sequence groups each of whom have less than 25% identity. OXBench has 3 reference sets and the master set contains 673 alignments that have sequences ranges from 2 to 122. From the master set, we chose a subset that have at least 25 sequences (yields 22 alignments). SABRE contains 423 alignments that have sequences ranges from 3 to 25. We choose a subset of 6 sequences that have at least 20 sequences.

Other than these benchmark datasets, we use families of proteins couplings: GPCR, WW, and PDZ. G-protein coupled receptors are a key demonstrator of allosteric communication and serve to transduce extracellular stimuli into intracellular signals [14]. The entire GPCR family is subdivided into 16 subfamilies (alignments). We use 6 alignments from this set, each of whom involve at least 30 sequences: Amine, Rhodopsin, Peptide, Olfactory, Nucleotide, and Prostanoid. The PDZ family has only one alignment and the WW family has three subfamilies: native, CC, and IC.

5.2 Scoring Criteria

We use four different scoring criteria to assess the quality of a test alignment with respect to a reference alignment.

1. Q-Score [8]: This score, a.k.a. sum-of-pairs score, can be defined as follows. Let T be the number of aligned residue pairs in the reference alignment and L be the number of aligned residues pairs in the reference alignment that are also correctly aligned in the test alignment. Then, Q-score $= \frac{L}{T}$.
2. Total Column Score (TC) [27]: This score is the fraction of columns in a reference alignment that are identical in a test alignment.
3. Modeler Score [22]: Let R be the number of aligned residue pairs in a test alignment and L be the number of aligned residue pairs in the reference alignment that are also correctly aligned in the test alignment. Then, Modeler score $= \frac{L}{R}$.
4. Cline Shift Score [4]: While the above three scores evaluate only correctly aligned residues or residue pairs, the Shift score also penalizes misalignments. See [4] for more details.
5. Coupled Column Score (C-Score): None of the above four scores measure how many of the coupled columns (colums that are participating in the couplings) of a reference alignment are retained in the test alignment. We propose a score to measure the fraction of retained coupled columns based on probabilistic graphical models (PGMs). PGMs can encode couplings of an alignment [24] where nodes denote columns of the alignment and edges denote couplings between two columns. To calculate the C-Score, we create a PGM for a reference alignment, and then count the number of columns (V) that are participating in couplings. For these V coupled columns in the reference alignment, we count how many (V') of them are retained in the test alignment. A column in the reference alignment is considered to be retained in the test alignment if the number of mismatched residues are fewer than 10%. These two counts give us C-Score $= \frac{V'}{V}$.

For all the above measures, higher values are better. The five measures yield a maximum score of 1. Shift Score gives a minimum score -0.2 and the rest give 0 as minimum score.

Table 3: Comparison of ARMiCoRe against ClustalW over the SABRE alignments.

Alignments	Q-Score		TC Score		Shift Score		Modeler Score	
	ClustalW	ARMiCoRe	ClustalW	ARMiCoRe	ClustalW	ARMiCoRe	ClustalW	ARMiCoRe
sup_038	0.82	**0.88**	0	0	0.17	**0.19**	0.10	**0.12**
sup_092	0.20	**0.30**	0	0	0.05	**0.07**	0.03	**0.05**
sup_108	0.89	**0.93**	0	**0.35**	0.46	**0.48**	0.31	**0.32**
sup_126	0.51	**0.59**	0	0	0.19	**0.23**	0.12	**0.14**
sup_167	**0.61**	0.50	0	0	**0.27**	0.23	**0.17**	0.14
sup_215	0.11	**0.18**	0	0	0.00	**0.01**	0.00	**0.01**

Table 4: Comparison of ARMiCoRe against ClustalW and COBALT over the CC subfamily of WW protein family.

Score	ClustalW	COBALT	ARMiCoRe
Q-Score	0.89	0.85	**0.96**
TC Score	**0.51**	0.35	**0.51**
Shift Score	0.93	0.91	**0.97**
Modeler Score	0.90	0.91	**0.97**
C-Score	0.71	0.88	**1.00**

Table 5: Comparison of ARMiCoRe against ClustalW and COBALT over the PDZ family.

Score	ClustalW	COBALT	ARMiCoRe
Q-score	0.85	0.82	**0.87**
TC Score	0	**0.1**	0
Shift score	0.89	0.87	**0.9**
Modeler Score	0.85	**0.89**	0.88
C-Score	0.67	**0.81**	**0.81**

5.3 Comparison with ClustalW

We evaluate ARMiCoRe on the datasets described earlier: benchmark and alignments with couplings. For each of these alignments, we remove gaps and realign with ClustalW in default settings (using the PAM matrix). ARMiCoRe is run on each of the ClustalW alignments to generate coupled patterns and use the coupled patterns to generate constraints, which are used by COBALT to create an improved alignment. We then compare our scores with ClustalW.

Performance of ARMiCoRe on the BaliBase benchmark is given in Table 1. ARMiCoRe shows superior performance over ClustalW on all of the four measures in the RV12 reference set. Note that the performance of ARMiCoRe on RV20 and RV30 is worse than that of ClustalW in all four measures. This is because RV20 and RV30 pool together sequences with poor similarity and thus coupled patterns are not a driver for obtaining good alignments. To test this hypothesis, we removed the orphan sequences from RV20 (RV20*) and as Table 1 shows, the performance of ARMiCoRe is better along three of the four measures. Table 2 describes the results of ARMiCoRe for the OXBench benchmark, once again revealing a mixed performance on a dataset with high sequence diversity. Finally, Table 3 depicts the superior performance of ARMiCoRe over ClustalW in 5 alignments out of 6 alignments in SABRE dataset.

5.4 Modeling Correlated Mutations

We describe the effect of ARMiCoRe on three families that are known to exhibit correlated mutations. We focus on the CC subfamily of the WW domain, the PDZ family, and the Nucleotide subfamily of the GPCR family. Based on C-Score, we evaluate the Performance of ARMiCoRe against ClustalW and COBALT. As shown in Tables 4, 5, and 6, ARMiCoRe is consistently better on at least three measures.

Table 6: Comparison of ARMiCoRe against ClustalW and COBALT over the Nucleotide subfamily of GPCR protein family.

Score	ClustalW	COBALT	ARMiCoRe
Q-Score	0.74	0.68	**0.79**
TC Score	**0.46**	0.34	0.45
Shift Score	0.79	0.74	**0.83**
Modeler Score	0.74	0.77	**0.80**
C-Score	0.52	0.50	**0.63**

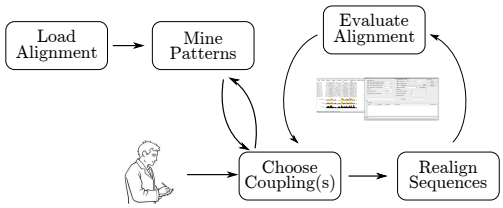
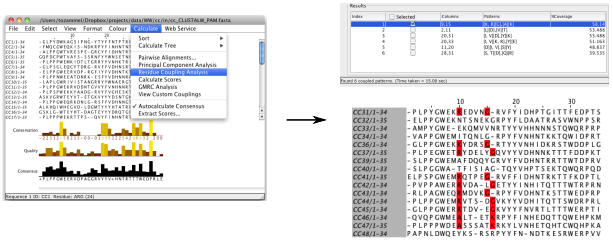


Figure 5: An overview of user interaction with ARMiCoRe.



(a) (b)

Figure 6: Interfaces for mining coupled patterns. (a) An input alignment. (b) Selection of coupled patterns with colored plot of corresponding residues.

5.5 User Interaction in Choosing Couplings

We have developed GUIs for ARMiCoRe that allow users to interactively choose patterns from a set of significant coupled patterns and use them to realign sequences. This enables biologists to bring specific domain knowledge in exposing couplings in the realignment. A typical workflow with ARMiCoRe is illustrated in Fig. 5. A user begins an experiment by loading an initial alignment (see Fig. 6(a)). He or she may decide to improve the alignment using the coupled pattern mining module based on the quality scores measured

in the evaluation module. The coupled pattern mining module facilitates tuning various parameters and gives a set of significant coupled patterns as output. From the pool of coupled patterns, a domain expert can choose meaningful patterns (see Fig. 6(b)) and use them in the realignment module. The realignment module gives a new alignment, which can be evaluated in the evaluation module. A user may repeat the realignment step by choosing different patterns or the mining step by tuning the parameters.

6. DISCUSSION

Evolutionary constraints on genes and proteins to maintain structure and function are revealed as conservation and coupling in an MSA. The advent of cheap, high-throughput sequencing promises to provide a wealth of sequence data enabling such applications, but at the same time requires methods such as ARMiCoRe to improve the alignments and inferred constraints upon which they are based. The alignments obtained by ARMiCoRe can be leveraged to design or classify novel proteins that are stably folded and functional [1, 25], as well as to predict three-dimensional structures from sequence alone [12]. Our work also demonstrates a successful application of pattern set mining where the goal is not just to find patterns but to cover the set of sequences with discovered patterns such that an objective measure is optimized. The ideas developed here can be generalized to other pattern set mining problems in areas like neuroscience, sustainability, and systems biology.

Acknowledgements

This work is supported in part by US NSF grant IIS-0905313.

7. REFERENCES

[1] S. Balakrishnan, H. Kamisetty, J. Carbonell, S. Lee, and C. Langmead. Learning Generative Models for Protein Fold Families. *Proteins: Structure, Function, and Bioinformatics*, 79(4):1061–1078, 2011.

[2] B. Bringmann, S. Nijssen, N. Tatti, J. Vreeken, and A. Zimmermann. Mining Sets of Patterns: A Tutorial at ECMLPKDD 2010, Barcelona, Spain.

[3] H. Carrillo and D. Lipman. The Multiple Sequence Alignment Problem in Biology. *SIAM J. Appl. Math.*, 48(5):1073–1082, 1988.

[4] M. Cline, R. Hughey, and K. Karplus. Predicting Reliable Regions in Protein Sequence Alignments. *Bioinformatics*, 18(2):306–314, 2002.

[5] C. Do and K. Katoh. Protein Multiple Sequence Alignment. In *Functional Proteomics*, volume 484, chapter 25, pages 379–413. 2008.

[6] C. Do, M. Mahabhashyam, M. Brudno, and S. Batzoglou. ProbCons: Probabilistic Consistency-based Multiple Sequence Alignment. *Genome Res.*, 15:330–340, 2005.

[7] R. Edgar. MSA Benchmark Collection. Available at `http://www.drive5.com/bench/`.

[8] R. Edgar. MUSCLE: Multiple Sequence Alignment with High Accuracy and High Throughput. *Nucleic Acids Res.*, 32:1792–1797, 2004.

[9] A. Goldberg and R. Tarjan. A New Approach to the Maximum-Flow Problem. *J. ACM*, 35:921–940, 1988.

[10] R. Gouveia-Oliveira and A. Pedersen. Finding Coevolving Amino Acid Residues using Row and Column Weighting of Mutual Information and Multi-dimensional Amino Acid Representation. *Algorithms for Molecular Biology*, 1:12, 2007.

[11] L. Guasco. *Multiple Sequence Alignment Correction using Constraints*. PhD thesis, Universidade Nova de Lisboa, 2010.

[12] T. Hopf, L. Colwell, R. Sheridan, B. Rost, C. Sander, and D. Marks. Three-Dimensional Structures of Membrane Proteins from Genomic Sequencing. *Cell*, 149(7):1607–1621, 2012.

[13] T. Hubbard, A. Lesk, and A. Tramontano. Gathering Them in to the Fold. *NSMB*, 3(4):313, 1996.

[14] W. Kroeze, D. Sheffler, and B. Roth. G-protein-coupled Receptors at a Glance. *J. Cell Sci.*, 116:4867–4869, 2003.

[15] A. Löytynoja and M. Milinkovitch. A Hidden Markov Model for Progressive Multiple Alignment. *Bioinformatics*, 19(12):1505–1513, 2003.

[16] B. Morgenstern. DIALIGN: Multiple DNA and Protein Sequence Alignment at BiBiServ. *Nucleic Acids Res.*, 32(suppl 2):W33–W36, 2004.

[17] G. Nemhauser, L. Wolsey, and M. Fisher. An Analysis of Approximations for Maximizing Submodular Set Functions-I. *Math. Prog.*, 14:265–294, 1978.

[18] C. Notredame, D. Higgins, and J. Heringa. T-coffee: A Novel Method for Fast and Accurate Multiple Sequence Alignment. *JMB*, 302(1):205–217, 2000.

[19] J. Papadopoulos and R. Agarwala. COBALT: Constraint-based Alignment Tool for Multiple Protein Sequences. *Bioinformatics*, 23(9):1073–1079, 2007.

[20] J. Pei and N. Grishin. PROMALS: Towards Accurate Multiple Sequence Alignments of Distantly Related Proteins. *Bioinformatics*, 23(7):802–808, 2007.

[21] G. P. S. Raghava, S. Searle, P. Audley, J. Barber, and G. Barton. OXBench: A Benchmark for Evaluation of Protein Multiple Sequence Alignment Accuracy. *BMC bioinformatics*, 4(1):47, 2003.

[22] J. Sauder, J. Arthur, and R. Dunbrack. Large-scale Comparison of Protein Sequence Alignment Algorithms with Structure Alignments. *Proteins*, 40:6–22, 2000.

[23] W. Taylor. The Classification of Amino Acid Conservation. *J. Theor. Biol.*, 119:205–218, 1986.

[24] J. Thomas, N. Ramakrishnan, and C. Bailey-Kellogg. Graphical Models of Residue Coupling in Protein Families. *IEEE/ACM TCBB*, 5(2):183–197, 2008.

[25] J. Thomas, N. Ramakrishnan, and C. Bailey-Kellogg. Protein Design by Sampling an Undirected Graphical Model of Residue Constraints. *IEEE/ACM TCBB*, 6(3):506–516, 2009.

[26] J. Thompson, D. Higgins, and T. Gibson. CLUSTAL W: Improving the Sensitivity of Progressive Multiple Sequence Alignment through Sequence Weighting, Position-specific Gap Penalties and Weight Matrix Choice. *Nucleic Acids Res.*, 22:4673–4680, 1994.

[27] J. Thompson, P. Koehl, R. Ripp, and O. Poch. BAliBASE 3.0: Latest Developments of the Multiple Sequence Alignment Benchmark. *Proteins: Structure, Function, and Bioinformatics*, 61(1):127–136, 2005.

[28] L. Wang and T. Jiang. On the Complexity of Multiple Sequence Alignment. *JCB*, 1(4):337–348, 1994.