Aligning biological sequences by exploiting residue conservation and coevolution

Aligning biological sequences by exploiting residue conservation and coevolution

Anna Paola Muntoni,[1,2,3,*] Andrea Pagnani,[1,4,5] Martin Weigt,[3] and Francesco Zamponi[2]

[1]*Department of Applied Science and Technology (DISAT), Politecnico di Torino, Corso Duca degli Abruzzi 24, I-10129 Torino, Italy*

[2]*Laboratoire de Physique de l'Ecole Normale Supérieure, ENS, Université PSL, CNRS, Sorbonne Université, Université de Paris, F-75005 Paris, France*

[3]*Sorbonne Université, CNRS, Institut de Biologie Paris Seine, Biologie Computationnelle et Quantitative LCQB, F-75005 Paris, France*

[4]*Italian Institute for Genomic Medicine, IRCCS Candiolo, SP-142, I-10060 Candiolo (TO), Italy*

[5]*INFN, Sezione di Torino, Via Giuria 1, I-10125 Torino, Italy*

(Received 2 June 2020; accepted 12 November 2020; published 7 December 2020)

Sequences of nucleotides (for DNA and RNA) or amino acids (for proteins) are central objects in biology. Among the most important computational problems is that of sequence alignment, i.e., arranging sequences from different organisms in such a way to identify similar regions, to detect evolutionary relationships between sequences, and to predict biomolecular structure and function. This is typically addressed through profile models, which capture position specificities like conservation in sequences but assume an independent evolution of different positions. Over recent years, it has been well established that coevolution of different amino-acid positions is essential for maintaining three-dimensional structure and function. Modeling approaches based on inverse statistical physics can catch the coevolution signal in sequence ensembles, and they are now widely used in predicting protein structure, protein-protein interactions, and mutational landscapes. Here, we present DCAlign, an efficient alignment algorithm based on an approximate message-passing strategy, which is able to overcome the limitations of profile models, to include coevolution among positions in a general way, and to be therefore universally applicable to protein- and RNA-sequence alignment without the need of using complementary structural information. The potential of DCAlign is carefully explored using well-controlled simulated data, as well as real protein and RNA sequences.

DOI: 10.1103/PhysRevE.102.062409

I. INTRODUCTION

In the course of evolution, biological molecules such as proteins or RNA undergo substantial changes in their amino-acid or nucleotide sequences, while keeping their three-dimensional fold structure and their biological function remarkably conserved. In computational biology, this structural and functional conservation is extensively used: When we can, e.g., establish that two proteins are homologous, i.e., they share some common evolutionary ancestor, properties known for one protein can be translated to its homolog (a process known as *annotation*). As an example, suppose that one is given a sequence of a human protein whose function is unknown. If this sequence can be properly aligned to a protein of known function but from a different, even evolutionarily distant organism, we can expect also the human protein to perform, globally, the same function. Even at the finer amino-acid scale, a given position in two aligned sequences of homologous proteins is expected to have the same physical positioning inside the three-dimensional protein structures of both proteins and to share common functionality (e.g., as active sites or in binding interfaces).

Detecting homology, however, is not an easy task. First, homologous proteins or RNAs may share only 20% or even less of their residues (i.e., amino acids or nucleotides), the others being substituted in evolution, making the detection of similarity rather involved. Even worse, proteins (and RNA) may change their length, as amino acids (or nucleotides) may be inserted into a sequence or deleted from it. Just looking to a single sequence, we have no information on which positions might be insertions or deletions and which positions might be inherited from ancestors, possibly undergoing amino-acid or nucleotide substitutions.

To solve this problem, *sequence alignments* have to be constructed [1]. The objective of sequence alignments is to identify homologous positions, also called matches, along with insertions and deletions, such that the aligned sequences become as similar as possible. In this context, three frequently used but distinct alignment problems can be identified:

(1) *Pairwise alignments* compare two sequences. Under some simplifying assumptions, cf., below, this problem can be solved efficiently using dynamic programming (i.e., an iterative method similar to transfer matrices or message passing in statistical physics) [2,3]. Detecting homology by pairwise alignment is limited to rather close homologs.

(2) More distant homology can be detected using *multiple-sequence alignments* (MSA) of more than two sequences [4], which maximize the global sequence similarities by constructing a rectangular matrix formed by amino acids (or nucleotides) and gaps, representing both insertions and deletions, as illustrated in Fig. 1. The rows of this matrix are the individual sequences; the columns are aligned positions. Besides being able to detect more distant homology, MSA

*Corresponding author: anna.muntoni@polito.it

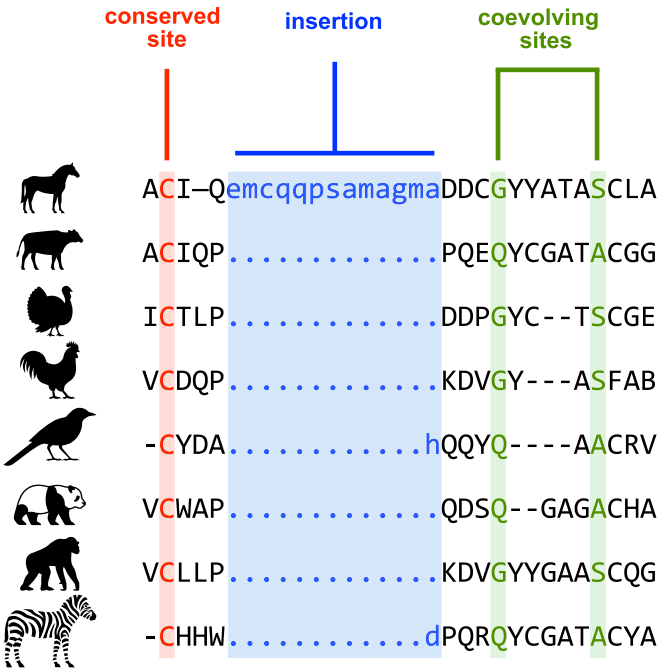

FIG. 1. Example of a multiple sequence alignment. Sequences from different organisms are aligned to maximize their similarity. Some sites are fully conserved; others are variable. Coevolving sites vary in a strongly correlated way. Insertions are indicated by lowercase letters and dots (.) while deletions are shown by dashes (–).

allow for identifying conserved positions, i.e., columns which do not (or rarely) change their symbol. These positions are typically known to be important, either for the functionality (e.g., active sites in proteins) or for the thermodynamic stability of the fold. Although dynamic programming methods can be generalized from pairwise to multiple-sequence alignments, their running time grows exponentially in the number of sequences to be aligned. Many heuristic strategies have been proposed following the seminal ideas of Ref. [5], cf., Refs. [6–9], but the construction of accurate MSA of more than about 10^3 sequences remains challenging.

(3) For larger alignments, a simple strategy is widely used. Instead of constructing a globally optimal sequence alignment, *sequences are aligned one by one to a well-constructed seed MSA* [10,11]. As in the case of MSA, this strategy allows for detecting distant homologs and for exploiting conservation. If the seed MSA is reasonably large ($\gtrsim 10^2$ sequences) and of high quality, very large and rather accurate alignments of up to 10^6 sequences can be constructed easily. This strategy is currently the best choice for constructing large families of homologous proteins [12] and RNAs [13], and is also the subject of our work.

Almost all these sequence-alignment methods are based on the simplifying assumption of *independent-site evolution*. In terms of sequence statistics, this amounts to assuming that the global probability of observing some full-length sequence can be factorized over site-specific but independent single-site probabilities, also known as *profile models* [1]. Profile models are thus able to capture conservation but not coevolution. For an alphabet of only two symbols, hence represented by Ising spins, such a model corresponds to a noninteracting Ising model with site-dependent local fields only; for more than two symbols, a profile model corresponds to a noninteracting Potts model. A successful variant are *profile hidden Markov models* [14], which assume independence of matched positions but take into account that gap stretches are more likely than many individual gaps to reflect the tendency of homologous proteins (or RNAs) to accumulate, in the course of evolution, large-scale modular gene rearrangements. A major advantage of profile models is their computational efficiency, as they allow for determining optimal alignments in polynomial time using dynamic programming (or transfer matrix) methods [1]. They mostly make use of conserved positions, which serve intuitively as anchoring point in aligning new sequences to the seed MSA. Variable positions contribute much less to profile-based sequence alignment, and coevolution is entirely neglected in these models.

An important exception are so-called *covariance models* for functional RNA [15]. Different from proteins, RNA sequences are characterized by low sequence conservation, making alignment via profile models unreliable, but highly conserved secondary structures, due to base pairing inside the single-stranded RNA molecules, and the formation of local helices, the so-called *stems*. Base pairing does not pose constraints on the individual bases, but on the correct pairing in Watson-Crick pairs A:U and G:C, or wobble pairs G:U, which consequently have to be described by a nonfactorizable pair distribution. In the case of RNA, the planar structure of the graph formed by the pairing in the RNA chains still enables the application of exact but computationally efficient dynamic programming [15–18]. However, the construction of covariance models requires the secondary structure to be known *a priori*.

Suppose now that a MSA has been constructed by some alignment method. One would then like to extract as much information as possible from this MSA. In recent decades, amino-acid coevolution has been established as perhaps the most important statistical feature of MSAs beyond conservation [19]. In fact, consider a MSA of M sequences $(a_1, \ldots, a_L)$, each of length L (Fig. 1). The statistics of the MSA are encoded into its one-site frequencies $f_i(a)$, i.e., the frequency of observing $a_i = a$ in a sequence; two-site frequencies $f_{ij}(a, b)$, i.e., the frequency of observing $(a_i = a, a_j = b)$ in a same sequence; three-site frequencies $f_{ijk}(a, b, c)$; and so on. Modeling the MSA statistics by inverse statistical physics [20–23] consists in assuming that each sequence in the MSA is an independent realization from some unknown probability distribution $P(a_1, \ldots, a_L)$, such that all multisite frequencies are reproduced. It has been realized that such a probability can be obtained by a maximum entropy principle (identical to the one used in equilibrium statistical mechanics), under the constraint that the one-site (i.e., conservation) and two-site (i.e., coevolution) frequencies are correctly reproduced. The resulting $P(a_1, \ldots, a_L)$ is then a Potts model with local fields and two-spin interactions only: Interactions involving more than two spins are not present. Remarkably, such models turn out to describe very well the three- and more-site MSA frequencies, even if these have not been included in their construction. This method of analysis, called direct-coupling analysis (DCA) because it provides a

set of two-spin couplings [24–27], has found widespread applications in extracting protein-structure prediction from MSAs [28–31], detecting protein-protein interactions [24,32–38], describing mutational effects [39–43], and even in data-driven sequence optimization and design [40,44,45].

The resulting situation is somewhat paradoxical: MSAs are constructed using profile models (noninteracting Potts models), which neglect coevolution and use conservation only, but once the MSA is available, one can extract relevant information from the coevolution signal via the DCA method, which makes use of pairwise-interacting Potts models. In other words, important structural and functional information is contained in the coevolution of amino acids or nucleotides, but it is neglected in the alignment procedure.

Our work aims at overcoming this paradox, by including the information contained in amino-acid (or nucleotide) coevolution in aligning sequences to a seed MSA. While this idea shows some similarity to that of covariance models and RNA alignment, our method is much more general. In fact, DCA modeling describes coevolution between any possible pair of sites, and as a consequence, it allows for interactions between any possible pair of spins in the Potts model. The resulting interaction graph is fully connected, which makes an application of dynamic programming (or transfer matrix) methods impossible. We cope with this problem by proposing an approximate message passing strategy based on belief propagation [46], further simplified in a high-connectivity mean-field limit for long-range couplings [47]. We show that the resulting DCAlign algorithm outperforms state-of-the-art alignment tools both in well-controlled simulated data and in real protein and RNA sequences.

In parallel to our work, coevolutionary models have been recently used to solve related problems, such as the remote search homology [48] and the alignment of two Potts models [49]. Still, the statistical mechanics-based approach and the formalization of the problem proposed here significantly differ from those in Refs. [48,49].

The plan of the paper is the following: We first formalize the problem and its statistical-physics description. The latter allows us to derive DCAlign, a combined belief-propagation and mean-field algorithm for aligning a sequence to a Potts model constructed by DCA from the seed MSA. The efficiency of our algorithm is first tested in the case of artificial data, which allow us to evaluate the influence of conservation (i.e., single-site statistics) and coevolution (i.e., two-site couplings) in the alignment procedure. Extensive tests and positive results are given for a number of real protein and RNA families. Technical details of the derivation of the algorithm are provided in the Supplemental Material (SM) [50].

II. SETUP OF THE PROBLEM

A. Alignment

The method we are going to describe can be applied to align different types of biological sequences (viz., DNA, RNA, proteins). We discuss here the protein case, but the extension to the other cases is straightforward and it will be considered below. Let us consider an amino-acid sequence $\boldsymbol{A} = (A_1, \ldots, A_N)$ containing a protein domain $\boldsymbol{S} = (S_1, \ldots, S_L)$ of a known family, which we want to identify. Note that $\boldsymbol{S}$ may contain amino acids and gaps, while the original sequence $\boldsymbol{A}$ is composed exclusively by amino acids. We assume that the protein family is well described by a direct coupling analysis (DCA) model, or Potts Hamiltonian, or simply "energy,"

$$\mathcal{H}_{\mathrm{DCA}}(\boldsymbol{S}\,|\,\boldsymbol{J},\boldsymbol{h}) = -\sum_{i<j}^{1,L} J_{ij}(S_i,S_j) - \sum_{i=1}^{L} h_i(S_i)\,. \qquad (1)$$

Here, the sequence $\boldsymbol{S} = (S_1, \ldots, S_L)$ is assumed to be aligned to the MSA of length L of the protein family, and the set of parameters $\boldsymbol{J}$ and $\boldsymbol{h}$ are considered as known, having been learned from some seed alignment [51]. The energy $\mathcal{H}_{\mathrm{DCA}}$ is then considered as a "score" (lower energy corresponds to higher score) for sequence $\boldsymbol{S}$ to belong to the protein family. We address the problem of aligning a sequence $\boldsymbol{A}$ to the model $\mathcal{H}_{\mathrm{DCA}}$ or, in other words, of detecting the domain in $\boldsymbol{A}$ that has the best score within the model $\mathcal{H}_{\mathrm{DCA}}$. In this setting, the solution to our problem is the subsequence (cf., below for the precise definition of a subsequence including insertions and deletions) that, among all the possible subsequences of $\boldsymbol{A}$, minimizes the energy (or, at finite temperature, is a typical sequence sampled from $\mathcal{H}_{\mathrm{DCA}}$). The energy thus serves as a cost function for comparing different candidate alignments. Contrary to profile models, which only take into account conservation of single residues, DCA models also include pairwise interactions related to residue coevolution (and thus in particular at any linear separation along the sequence $\boldsymbol{A}$), which we hope will lead to more accurate alignments in cases where conservation alone is insufficient.

However, the DCA model $\mathcal{H}_{\mathrm{DCA}}$ does not model insertions, because the parameters $\boldsymbol{J}$ and $\boldsymbol{h}$ are inferred from a seed MSA where all columns containing inserts have been removed. A suitable additional cost has thus to be assigned to amino-acid insertions, which are needed in order to find a low-cost alignment. Still, we have to prevent our algorithm from picking up energetically favorable but isolated amino acids out of the (possibly long) input sequence $\boldsymbol{A}$. For modeling this cost, we will explicitly refer to the insertion statistics in the full seed alignment.

Note that the DCA model contains position-specific gap terms in the $\boldsymbol{J}$ and $\boldsymbol{h}$, so the gap statistics of the seed MSA is fully described by the DCA model alone. Nonetheless, the observed statistics deeply depend on how the seed is constructed, and it could *a priori* be nonrepresentative of the gap statistics of the full alignment. To take into account this degree of variability, we allow for the introduction of an additional energy term associated with the presence of gaps. A more detailed discussion is reported in Sec. II B.

Formally speaking, the alignment problem reduces to finding a subsequence $\boldsymbol{S} = (S_1, \ldots, S_L)$ of $\boldsymbol{A} = (A_1, \ldots, A_N)$ such that the following are true:

(1) The subsequence $\boldsymbol{S}$ forms an ordered list of amino acids in $\boldsymbol{A}$ ("match" states) with the possibility of (a) adding gaps, denoted as dashes (–), between two consecutive positions in $\boldsymbol{A}$ and (b) skipping some amino acids of $\boldsymbol{A}$, i.e., interpreting them as insertions.

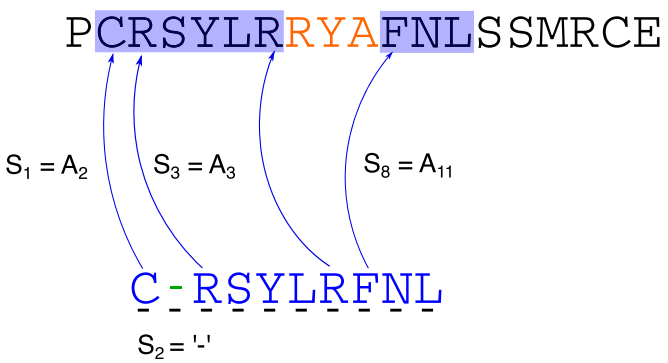

FIG. 2. Example of an alignment. On the top we show a full length sequence $\boldsymbol{A}$, and on the bottom its alignment $\boldsymbol{S}$, in which both gap and insertion events occur. The domain to be aligned is highlighted in blue. We show explicitly three matched states $S_1 = A_2$, $S_3 = A_3$, and $S_8 = A_{11}$ and a gap insertion in $S_2 =$ " $-$ ". In this example, we also show a possible way of skipping some amino acids in the original sequence, that is to assign three insertions, highlighted in red.

(2) The aligned sequence $\boldsymbol{S}$ minimizes

$$E(\boldsymbol{S}|\boldsymbol{J},\boldsymbol{h},\boldsymbol{\lambda},\boldsymbol{\mu}) = \mathcal{H}_{\rm DCA}(\boldsymbol{S}\,|\,\boldsymbol{J},\boldsymbol{h}) + \mathcal{H}_{\rm gap}(\boldsymbol{\mu}) + \mathcal{H}_{\rm ins}(\boldsymbol{\lambda}), \quad (2)$$

with $\mathcal{H}_{\rm gap}(\boldsymbol{\mu})$ being a penalty for adding gaps that depends on the number and position of gaps and on the hyperparameters $\boldsymbol{\mu}$ and $\mathcal{H}_{\rm ins}(\boldsymbol{\lambda})$ being a penalty on insertions, parametrized by the hyperparameters $\boldsymbol{\lambda}$.

We first consider here the case where $N \gtrsim L$, i.e., we are trying to align a domain in a longer sequence, or $N < L$ when we are trying to align a fragment. The case $N \gg L$, i.e., when we search for a hit of the DCA model in a long sequence, may be computationally hard for the approach proposed here, because the alignment time scales roughly as L^2N^2, as discussed below. Current state-of-the-art alignment methods like the Basic local alignment search tool (BLAST) use heuristics to approximately locate possible hits and perform accurate alignment search only in these restricted regions to speed up search. It might be necessary to do this before running DCAlign, but this is not the objective of the current work. In general, it will be better if N is not too different from L.

An example of a full sequence and its alignment is given in Fig. 2. The sequence on the top is a full sequence of $N = 19$ amino acids, and the highlighted part is the target domain. The aligned sequence of length $L = 10$, reported on the bottom, consists in one gap in position 2 and 9 matched amino acids.

B. Gap and insertion penalties

The hyperparameters $\boldsymbol{\mu}$, $\boldsymbol{\lambda}$ determine the cost of adding a gap or an insertion in the aligned sequence. They must be carefully determined to allow for these events without affecting the quality of the alignment. In other words, we would like to reduce, as much as possible, the number of gaps (when the statistics of gaps of the seed we use is biased) and to parsimoniously add insertions when energetically favorable, avoiding to pick up isolated amino acids in the alignment.

To deal with insertions, we use a so-called affine penalty function [1] parametrized by λ_i^o, the cost of adding a first insertion between positions $i-1$ and i, and λ_i^e, the cost of extending an existing insertion, with $\lambda_i^o > \lambda_i^e$. This results in

$$\mathcal{H}_{\rm ins}(\boldsymbol{\lambda}) = \sum_{i=2}^{L} \varphi_i(\Delta n_i),$$
$$\varphi_i(\Delta n_i) = (1-\delta_{\Delta n_i,0})\left[\lambda_o^i + \lambda_e^i(\Delta n_i - 1)\right], \quad (3)$$

where Δn_i is the number of insertions between positions $i-1$ and i. This set of parameters can be learned from a seed alignment through a maximum likelihood (ML) approach as reported in Sec. IV B. Finally, we introduce two types of gap penalties, denoted by $\mu^{\rm int}$ and $\mu^{\rm ext}$, which are associated with an "internal" gap between two matched states and with an "external" gap (at the beginning and at the end of the aligned sequence), respectively. This gives

$$\mathcal{H}_{\rm gap}(\boldsymbol{\mu}) = \sum_{i=1}^{L} \mu_i, \quad (4)$$

where $\mu_i = 0$ for match states, $\mu_i = \mu^{\rm int}$ for internal gaps, and $\mu_i = \mu^{\rm ext}$ for external gaps.

An illustration is given by the aligned sequence in Fig. 2. Insertions are highlighted in red, and the total insertions penalty is then given by $\lambda_o^8 + \lambda_e^8 + \lambda_e^8$. A gap, which increases the total energy by $\mu^{\rm int}$, is highlighted in green at position 2 of $\boldsymbol{S}$.

C. Statistical physics model

We now want to construct a discrete statistical-physics model which defines this alignment. For the positions $1 \leqslant i \leqslant L$, the model has to encode the position of the gaps and of the match states, with their corresponding symbol in the sequence $(A_1, \ldots, A_N)$. We therefore introduce two variables per site $1 \leqslant i \leqslant L$. The first one is a Boolean "spin" $x_i \in \{0, 1\}$, which tells us if S_i is a gap ($x_i = 0$) or an amino-acid match ($x_i = 1$). The second one is a "pointer" $n_i \in \{0, \ldots, N+1\}$, which gives, for the case of match states $x_i = 1$ and $1 \leqslant n_i \leqslant N$, the corresponding position in the original sequence $(A_1, \ldots, A_N)$; note that this allows for insertions if $n_{i+1} - n_i > 1$. If matched symbols start to appear only from a position $i > 1$, we then fill the previous positions $\{j : 1 \leqslant j < i\}$ with gaps having pointer $n_j = 0$. Similarly, if the last matched state appears in $i < L$, we fill a stretch of gaps in positions $\{j : i < j \leqslant L\}$ having $n_j = N+1$. This encoding allows one to distinguish the "external" gaps at the boundary of the aligned sequence, whose total number we denote as $N_{\rm gap}^{\rm ext}$, from the "internal" ones, i.e., between two consecutive matched states, whose total number is $N_{\rm gap}^{\rm int}$. Formally, the number of gaps and insertions are

$$N_{\rm ins} = \sum_{i=1}^{L-1} (n_{i+1} - n_i - 1)\mathbb{I}[N+1 > n_{i+1} > n_i > 0],$$
$$N_{\rm gap}^{\rm int} = \sum_{i=1}^{L} \delta_{x_i,0}\mathbb{I}[0 < n_i < N+1],$$
$$N_{\rm gap}^{\rm ext} = \sum_{i=1}^{L} \delta_{x_i,0}(\delta_{n_i,0} + \delta_{n_i,N+1}), \quad (5)$$

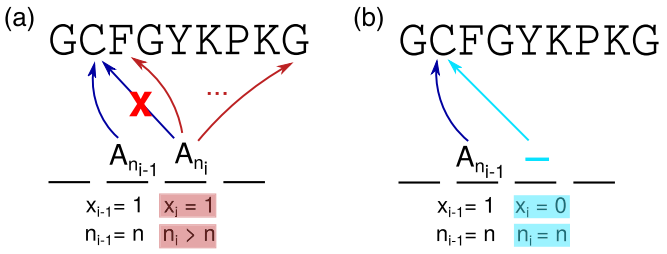

FIG. 3. Short-range constraints. We plot in panel (a) a feasible assignment of two consecutive matched states. If in position $i-1$ we assign $S_{i-1} = A_n$, we can then align the next position i to one of the possible amino acids $S_i \in \{A_{n+1}, \ldots, A_N\}$. As a consequence, $(x_{i-1}, x_i) = (1, 1)$, $n_{i-1} = n$ while $n_i \in \{n+1, \ldots, N\}$. In panel (b), we plot a feasible inclusion of a gap in position i. If the previous site $i-1$ points to $n_{i-1} = n$, we then assign $(x_i, n_i) = (0, n)$ to keep memory of the aligned sequence. In the next position, $i+1$, we can match an amino acid in further positions [according to the constraint in panel (a)] or add another gap with pointer $n_{i+1} = n$.

where $\mathbb{I}[\mathcal{E}]$ is the indicator function of the event $\mathcal{E}$. Introducing the short-hand notation $A_0 =$ "$-$" (gap), a model configuration $(x_1, \ldots, x_L, n_1, \ldots, n_L)$ results in an aligned sequence $(S_1, \ldots, S_L) = (A_{x_1 \cdot n_1}, \ldots, A_{x_L \cdot n_L})$. The auxiliary variables $(\boldsymbol{x}, \boldsymbol{n})$ must be additionally assigned such that the positional constraints illustrated in Fig. 2 are satisfied, i.e., the target subsequence must be ordered, as we now describe.

First of all, in order to correctly set the pointers in presence of gaps in the first and last positions, it is sufficient to set the state of node $i = 1$ as

$$\begin{aligned} n_1 &= 0 \ \text{ if } \ x_1 = 0, \\ N+1 > n_1 &> 0 \ \text{ if } \ x_1 = 1, \end{aligned} \tag{6}$$

and the state of node $i = L$ as

$$\begin{aligned} n_L &= N+1 \ \text{ if } \ x_L = 0, \\ N+1 > n_L &> 0 \ \text{ if } \ x_L = 1 \, . \end{aligned} \tag{7}$$

These properties can be formally expressed by the following two single-position constraints

$$\begin{aligned} \chi_{\text{in}}(x_1, n_1) &= \delta_{x_1,0}\delta_{n_1,0} + \delta_{x_1,1}(1-\delta_{n_1,0})(1-\delta_{n_1,N+1}), \\ \chi_{\text{end}}(x_L, n_L) &= \delta_{x_L,0}\delta_{n_L,N+1} + \delta_{x_L,1}(1-\delta_{n_L,0})(1-\delta_{n_L,N+1}) \, . \end{aligned} \tag{8}$$

Next, we need to locally impose that, for each position $1 < i < L$,

$$\begin{aligned} n_i &= n_{i-1} \ \text{ if } \ x_i = 0 \text{ and } n_i < N+1, \\ n_i &> n_{i-1} \ \text{ if } \ x_i = 1 \text{ or } n_i = N+1 \, ; \end{aligned} \tag{9}$$

i.e., the pointer n_i remains constant when $x_i = 0$, and it jumps to any later position in $n_{i-1}+1, \ldots, N$ if $x_i = 1$. This jump, besides determining the amino acid S_i to be placed in position i, also allows for identifying inserts according to Eq. (5). A pictorial representation of this constraint is shown in Fig. 3. We can formally encode these constraints in a "short-range" function $\chi_{\text{sr}}(x_{i-1}, n_{i-1}, x_i, n_i)$ that, for each pair of consecutive positions $(i-1, i)$, indicates the feasible and unfeasible configurations of the variables $(x_{i-1}, n_{i-1}, x_i, n_i)$ and the associated cost of insertions, as

$$\begin{aligned} \chi_{\text{sr}}(0, n_{i-1}, 0, n_i) &= \mathbb{I}(n_i = n_{i-1}), \\ \chi_{\text{sr}}(1, n_{i-1}, 0, n_i) &= \mathbb{I}(n_i = n_{i-1} \vee n_i = N+1), \\ \chi_{\text{sr}}(0, n_{i-1}, 1, n_i) &= e^{-\varphi_i(\Delta n_i)\mathbb{I}(n_{i-1}>0)} \\ &\quad \times \mathbb{I}(0 \leqslant n_{i-1} < n_i < N+1), \\ \chi_{\text{sr}}(1, n_{i-1}, 1, n_i) &= e^{-\varphi_i(\Delta n_i)} \\ &\quad \times \mathbb{I}(0 < n_{i-1} < n_i < N+1), \end{aligned} \tag{10}$$

where the function $\varphi_i(\Delta n_i)$ is the contribution of the ith position to the affine insertion penalty, as given in Eq. (3) with $\Delta n_i = n_i - n_{i-1} - 1$. Note that by combining the constraints in Eqs. (8) and (10), positions $\{j > 1\}$ ($\{j < L\}$) can either have a gap with $n_j = 0$ ($n_j = N+1$) or the first (last) match at any position $n_j > 0$ ($n_j < N+1$) with no insert penalty.

Finally, gap penalties can be encoded in a single-variable weight,

$$\chi_{\text{gap}}(x_i, n_i) = e^{-(1-x_i)\mu(n_i)}, \tag{11}$$

with

$$\mu(n) = \begin{cases} \mu^{\text{ext}} & n = 0 \vee n = N+1, \\ \mu^{\text{int}} & 1 \leqslant n \leqslant N \, . \end{cases} \tag{12}$$

The DCA Hamiltonian can be rewritten in terms of the auxiliary variables as

$$\mathcal{H}_{\text{DCA}}(\boldsymbol{x}, \boldsymbol{n}|\boldsymbol{J}, \boldsymbol{h}) = -\sum_{i<j} J_{ij}(A_{x_i \cdot n_i}, A_{x_j \cdot n_j}) - \sum_i h_i(A_{x_i \cdot n_i}), \tag{13}$$

while the global cost function E in Eq. (2) is

$$\begin{aligned} &E(\boldsymbol{x}, \boldsymbol{n}|\boldsymbol{J}, \boldsymbol{h}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \\ &\quad = \mathcal{H}_{\text{DCA}}(\boldsymbol{x}, \boldsymbol{n}|\boldsymbol{J}, \boldsymbol{h}) + \sum_i (1-x_i)\mu(n_i) \\ &\qquad + \sum_{i=2}^{L} \varphi_i(\Delta n_i)\mathbb{I}(n_{i-1} > 0)\mathbb{I}(N+1 > n_i) \, . \end{aligned} \tag{14}$$

Collecting all these definitions together, we can associate a Boltzmann weight $W(\boldsymbol{x}, \boldsymbol{n})$ with each possible alignment $(\boldsymbol{x}, \boldsymbol{n})$ of a given sequence $\boldsymbol{A}$, which takes into account all energetic contributions for feasible assignments only,

$$\begin{aligned} W(\boldsymbol{x}, \boldsymbol{n}) = \frac{1}{Z} e^{-\mathcal{H}_{\text{DCA}}(\boldsymbol{x}, \boldsymbol{n})} &\chi_{\text{in}}(x_1, n_1)\chi_{\text{end}}(x_L, n_L) \\ &\times \prod_{i=2}^{L} \chi_{\text{sr}}(x_{i-1}, n_{i-1}, x_i, n_i) \prod_{i=1}^{L} \chi_{\text{gap}}(x_i, n_i), \end{aligned} \tag{15}$$

where Z is the partition function. Note that the "hard constraints" χ_{sr}, which can set the weight to zero, live only on the edges of the linear chain $1, \ldots, L$, while the interactions J_{ij} are in principle fully connected.

Finally, we can map the original minimization problem in Eq. (2) as the statistical physics problem of finding the best assignment of the variables $(\boldsymbol{x}^*, \boldsymbol{n}^*)$ that maximizes the Boltzmann distribution:

$$(\boldsymbol{x}^*, \boldsymbol{n}^*) = \underset{(\boldsymbol{x}, \boldsymbol{n})}{\text{argmax}} \, W(\boldsymbol{x}, \boldsymbol{n}) \, . \tag{16}$$

Alternatively, we could obtain an optimal alignment $(\boldsymbol{x}^*, \boldsymbol{n}^*)$ from an equilibrium sampling of alignments with weight $W(\boldsymbol{x}, \boldsymbol{n})$. Unfortunately, both sampling from $W(\boldsymbol{x}, \boldsymbol{n})$ and identifying the constrained optimal assignment are hard and intractable problems. Note that the space of possible assignments has dimension scaling as $(N+2)^L$, which grows extremely quickly with N and L. For comparison, the DCA problem is defined in a space growing "only" as q^L. However, some approximations inspired by statistical physics can be exploited for seeking an approximate solution.

III. ADVANCED MEAN-FIELD APPROXIMATION

A straightforward approach to make this problem tractable is to use message-passing approximations of the marginal probabilities $P_i(x_i, n_i)$ of Eq. (15), such as belief propagation (BP), which are exact for problems defined on graphs without loops. Note that BP is also exact on linear chains, for which it coincides with the transfer matrix method (equivalent to dynamic programming and to the forward-backward algorithm [1]). One can think to BP as treating exactly the linear chain $1, \ldots, L$, while the longer range interactions are approximated by message passing. In the case of vanishing couplings $J_{ij}(A, B) \equiv 0$ for all i, j such that $|i-j| > 1$ and for all A, B, i.e., in the case of a model with nearest-neighbor interactions only, this formulation is exact; if all couplings vanish, it is quite similar to a profile hidden Markov model, which has the same penalties for opening and extending a sequence of insertions. However, our interactions are instead typically very dense (all couplings are nonzero, and hence the associated graph is very loopy) but weak. We can thus consider a further approximation of BP [47], in which the linear chain $1, \ldots, L$ is still treated exactly, while the contribution of more distant sites is approximated via mean field (MF), in a way similar to Thouless-Anderson-Palmer (TAP) equations [52], also known as approximate message passing (AMP) equations. We refer to this approach simply as MF in the following. In the rest of this section, we derive the BP and MF equations.

A. Transfer matrix equations for the linear chain

Suppose first that only nearest-neighbor couplings $J_{i,i+1}(A, B)$ are nonzero. In this case, the problem is exactly solved by the transfer matrix method, which corresponds to a set of recursive equations for the "forward messages" $F_i(x_i, n_i) = F_{i\to i+1}(x_i, n_i)$, i.e., the probability distribution of site i in absence of the link $(i, i+1)$, and "backward messages" $B_i(x_i, n_i) = B_{i\to i-1}(x_i, n_i)$, i.e., the probability distribution of site i in the absence of the link $(i-1, i)$.

We give here the transfer matrix equations for the forward and backward messages. For compactness, we define the single-site weight contribution to Eq. (15):

$$\begin{aligned}
\mathcal{W}_1(x_1, n_1) &= \chi_{\text{in}}(x_1, n_1) e^{h_1(A_{x_1 \cdot n_1}) - (1-x_1)\mu(n_1)}, \\
\mathcal{W}_i(x_i, n_i) &= e^{h_i(A_{x_i \cdot n_i}) - (1-x_i)\mu(n_i)}, \\
\mathcal{W}_L(x_L, n_L) &= \chi_{\text{end}}(x_L, n_L) e^{h_L(A_{x_L \cdot n_L}) - (1-x_L)\mu(n_L)},
\end{aligned} \tag{17}$$

TABLE I. Schematic summary of the transfer matrix and mean-field equations, which are complemented by the recurrence equations for $\mathcal{F}_i$ and $\mathcal{B}_i$ given in Eqs. (18) and (19). For mean field, one should replace $\mathcal{W}_i \to \mathcal{C}_i$.

$i = 1$	$i = 2, \ldots, L-1$	$i = L$
$P_1 = \frac{1}{z_1}\mathcal{W}_1\mathcal{B}_1$	$P_i = \frac{1}{z_i}\mathcal{W}_i\mathcal{F}_i\mathcal{B}_i$	$P_L = \frac{1}{z_L}\mathcal{W}_L\mathcal{F}_L$
$F_1 = \frac{1}{f_1}\mathcal{W}_1$	$F_i = \frac{1}{f_i}\mathcal{W}_i\mathcal{F}_i$	
	$B_i = \frac{1}{b_i}\mathcal{W}_i\mathcal{B}_i$	$B_L = \frac{1}{b_L}\mathcal{W}_L$

where the second line is for $i = 2, \ldots, L-1$. We then have for the forward messages

$$\begin{aligned}
F_1(x_1, n_1) &= \frac{1}{f_1}\mathcal{W}_1(x_1, n_1), \\
F_i(x_i, n_i) &= \frac{1}{f_i}\mathcal{W}_i(x_i, n_i)\mathcal{F}_i(x_i, n_i), \\
\mathcal{F}_i(x_i, n_i) &= \sum_{x_{i-1}, n_{i-1}} F_{i-1}(x_{i-1}, n_{i-1}) \\
&\quad \times e^{J_{i-1,i}(A_{x_{i-1}\cdot n_{i-1}}, A_{x_i \cdot n_i})} \chi_{\text{sr}}(x_{i-1}, n_{i-1}, x_i, n_i),
\end{aligned} \tag{18}$$

where F_i is defined for $i = 1, \ldots, L-1$ and $\mathcal{F}_i$ for $i = 2, \ldots, L$, and the f_i are normalization constants determined by the requirement that messages are normalized to one. For the backward messages, we have

$$\begin{aligned}
B_L(x_L, n_L) &= \frac{1}{b_L}\mathcal{W}_L(x_L, n_L), \\
B_i(x_i, n_i) &= \frac{1}{b_i}\mathcal{W}_i(x_i, n_i)\mathcal{B}_i(x_i, n_i), \\
\mathcal{B}_i(x_i, n_i) &= \sum_{x_{i+1}, n_{i+1}} B_{i+1}(x_{i+1}, n_{i+1}) \\
&\quad \times e^{J_{i,i+1}(A_{x_i \cdot n_i}, A_{x_{i+1}\cdot n_{i+1}})} \chi_{\text{sr}}(x_i, n_i, x_{i+1}, n_{i+1}),
\end{aligned} \tag{19}$$

where B_i is defined for $i = L, L-1, \ldots, 2$ and $\mathcal{B}_i$ for $i = L-1, \ldots, 1$, and b_i are normalization constants. Finally, the marginal probabilities are given by

$$\begin{aligned}
P_1(x_1, n_1) &= \frac{1}{z_1}\mathcal{W}_1(x_1, n_1)\mathcal{B}_1(x_1, n_1), \\
P_i(x_i, n_i) &= \frac{1}{z_i}\mathcal{W}_i(x_i, n_i)\mathcal{F}_i(x_i, n_i)\mathcal{B}_i(x_i, n_i), \\
P_L(x_L, n_L) &= \frac{1}{z_L}\mathcal{W}_L(x_L, n_L)\mathcal{F}_L(x_L, n_L)\,.
\end{aligned} \tag{20}$$

These equations are summarized in compact form in Table I. They can be easily implemented on a computer and solved in a time scaling as LN.

B. Long-range interactions

We now discuss the inclusion of long-range interactions in the transfer matrix scheme. In order to treat correctly the long-range interaction in BP, it is important to note that the same "light-cone" condition expressed by the constraint χ_{sr} in Eq. (10) holds between any pair (i, j). However, this condition would be violated by the messages of BP due to

their approximate character on loopy graphs. In order to enforce it, we can introduce a new constraint

$$\begin{aligned}\chi_{\rm lr}(x_i, n_i, x_j, n_j) &= \mathbb{I}[i > j+1]\{\delta_{x_i,0}\mathbb{I}[n_i \geqslant n_j] + \delta_{x_i,1}\mathbb{I}[n_i > n_j]\} \\ &+\mathbb{I}[i < j-1]\{\delta_{x_j,0}\mathbb{I}[n_i \leqslant n_j] + \delta_{x_j,1}\mathbb{I}[n_i < n_j]\}\,.\end{aligned} \tag{21}$$

Because this constraint is redundant with respect to Eq. (10), it can be added without changing the weight:

$$W(\boldsymbol{x}, \boldsymbol{n}) = W(\boldsymbol{x}, \boldsymbol{n}) \times \prod_{i<j} \chi_{\rm lr}(x_i, n_i, x_j, n_j)\,. \tag{22}$$

However, adding this constraint ensures that the proper ordering of the pointers is preserved under the BP approximation.

The BP equations can be written straightforwardly (see the SM [50]) and provide an approximation to the marginal probabilities P_i. A naive implementation requires a time scaling as L^3N^2, which can be easily reduced to L^2N^2. In the SM, we discuss a simplification of the BP equations, under the assumption that pairs of sites with $|i-j| > 1$ can be treated in a mean field [47]. We find that the resulting mean-field equations are identical to the transfer matrix ones, with the only replacement of the local weight $\mathcal{W}_i \to \mathcal{C}_i$, with

$$\mathcal{C}_i(x_i, n_i) = \mathcal{W}_i(x_i, n_i)\, \exp\left\{ \sum_{j\notin\{i,i\pm1\}} \sum_{x_j,n_j} \chi_{\rm lr}(x_i, n_i, x_j, n_j) J_{i,j}(A_{x_i\cdot n_i}, A_{x_j\cdot n_j}) P_j(x_j, n_j) \right\}. \tag{23}$$

As a result, the mean-field equations have the same complexity as the transfer matrix equations (LN) with an additional factor LN needed to compute each $\mathcal{C}_i$, resulting in an overall complexity L^2N^2, the same scaling of the BP update. However, while the memory consumption needed by BP to store all the incoming messages at each node scales as $2L^2N$, MF has the advantage of working directly on the $2LN$ approximated marginal probabilities.

C. Assignment

After solution of the MF equations, from the marginal probabilities $\{P_1(x_1, n_1), \ldots, P_L(x_L, n_L)\}$ we have to find the most probable assignment $(\boldsymbol{x}^*, \boldsymbol{n}^*)$, as defined in Eq. (16). The simplest way to do so is to assign to each position i the most probable state according to its marginal, i.e.,

$$(x_i^*, n_i^*) = \underset{x_i,n_i}{\rm argmax}\, P_i(x_i, n_i), \tag{24}$$

which is possible whenever the obtained assignment satisfies all the hard constraints. However, in some cases, the set of locally optimal positions do not satisfy the short-range constraints due to the approximate nature of the MF solution. We then perform a *maximization* step, in which we select the position i^* and the local assignment $(x_{i^*}^*, n_{i^*}^*)$ having the largest probability among all the marginals, i.e.,

$$(i^*, x_{i^*}^*, n_{i^*}^*) = \underset{i,x_i,n_i}{\rm argmax}\{P_1(x_1, n_1), \ldots, P_L(x_L, n_L)\}\,. \tag{25}$$

We then set the state of site i^* in $(x_{i^*}^*, n_{i^*}^*)$, and we proceed with a *filtering* step, in which we set to zero the marginal probabilities of the incompatible states of the first nearest neighbors of i^*. In practice, we multiply the marginals by the short-range constraints computed at $(x_{i^*}^*, n_{i^*}^*)$; i.e., we consider the new marginals on sites $i^* \pm 1$:

$$\begin{aligned}&P_{i^*-1}(x_{i^*-1}, n_{i^*-1})\chi_{\rm sr}(x_{i^*-1}, n_{i^*-1}, n_{i^*}^*, x_{i^*}^*) \\ &P_{i^*+1}(x_{i^*+1}, n_{i^*+1})\chi_{\rm sr}(x_{i^*}^*, n_{i^*}^*, x_{i^*+1}, n_{i^*+1}).\end{aligned} \tag{26}$$

We can now repeat the *maximization* step in order to find the state (x^*, n^*) that maximizes the joint set of probabilities for the positions adjacent to the already aligned part of the sequence (in this case i^*-1 and i^*+1, because we only fixed i^*),

$$(x^*, n^*) = \underset{x,n}{\rm argmax}\{P_{i^*-1}(x, n), P_{i^*+1}(x, n)\}, \tag{27}$$

and we fix this state, in the alignment, in the right position (either i^*+1 or i^*-1). Suppose for simplicity that we have just specified the state in position i^*+1; we now filter the probability of i^*+2 and repeat the choice for the next (x^*, n^*) considering the set of (modified) marginals $\{P_{i^*-1}(x, n), P_{i^*+2}(x, n)\}$. The procedure is repeated until all the L positions are determined.

Note that this scheme is somehow greedy, because the assignments are decided step by step and are constrained by the choices made in the previous positions. Still, the assignment is guided by the marginal probabilities obtained from considering the global energy function and all the hard constraints. Moreover, this assignment procedure is as fast as the "max-marginals" scheme because it does not require us to rerun the update of the equations in Sec. III, and thanks to the step-by-step filtering of the marginals it ensures an outcome that is always compatible with the constraints.

D. Discussion

In this section, we presented a set of approximate equations and an assignment procedure that, together, allow us to solve the alignment problem in polynomial time. In both BP and MF, the equations can be solved at "temperature" equal to one, corresponding to a Boltzmann equilibrium sampling from the weight in Eq. (15), or at zero temperature, corresponding to finding (approximately) the most likely assignment in Eq. (15) (the full set of equations for MF at zero temperature is reported in the SM [50]). Of course, any intermediate temperature could also be considered, but we do not explore other values of temperature in this work.

In all cases, one can compute the free energy associated with the BP or MF solution, which gives a "score" measuring the quality of the alignment. This score could be used, in long sequences with multiple hits, to decide a "best hit." The expression of the free energy is given in the SM.

The MF equations are derived from the BP equations by assuming that all couplings with $|i-j|>1$ are weak enough to be treated in the mean field. However, we know that in (good) protein models, stronger couplings correspond to physical contacts in the three-dimensional structure, while a background of weaker couplings describe other correlations or even just noise. It could be interesting, therefore, to use a mixed BP-MF method, in which weaker couplings with $||J_{ij}|| < K$ are treated in the mean field, while stronger couplings with $||J_{ij}|| \geqslant K$ are treated with BP, for a given threshold K. The case $K=0$ corresponds to pure BP, while the case $K\to\infty$ corresponds to pure MF. One could check whether an optimal value of K exists, but we leave this for future work. We show below the results obtained using MF, which seems preferable to BP in the cases we analyzed. The MF equations converge faster than BP to their fixed point and, additionally, BP is more affected by the loopy character of the graph and often converges to local minima of the energy landscape. We leave a more systematic comparison of the BP and MF schemes to future developments.

IV. LEARNING THE MODEL

We now discuss the learning of the model parameters, namely the couplings and fields of the DCA Hamiltonian, and the hyperparameters $\boldsymbol{\mu}, \boldsymbol{\lambda}$.

A. Potts model

Our alignment method is able to cope with different cost functions, because the implementation of the update equations described in Sec. III B is as general as possible. When we introduce a five-state alphabet, the method is also able to treat RNA alignments.

In this work, we tested several types of maximum entropy models for DCA, which differ in the choice of fitted observables. The usual Potts model, in which all first and second moments of the seed MSA are fitted, is labeled as *potts*, while we also consider a "pseudo" hidden Markov model (*phmm*), with Hamiltonian

$$\mathcal{H}_{\rm phmm}(\boldsymbol{S}|\boldsymbol{J},\boldsymbol{h}) = -\sum_{i,i+1} J_{i,i+1}\delta_{S_i,-}\delta_{S_{i+1},-} - \sum_i h_i(S_i)\,. \quad (28)$$

The *phmm* can be thought of as a profile model playing the role of the emission probabilities of a hidden Markov model (HMM), plus a pairwise interactions $J_{i,i+1}$ between neighboring gaps. This interaction is related, in our mapping to a HMM, to the probability of switching between two consecutive positions, from a "gap" to a "match" state and vice versa. We also considered other variations of the Potts model, such as a model in which we do not fit the second moment statistics of non-neighboring gaps (i.e., long-range gap-gap couplings are set to zero). The motivation behind this choice is that, if DCA couplings are interpreted as predictors for the (conserved) three-dimensional structure, gapped states do not carry any information about coevolution of far-away positions. However, we found that these other variations do not bring additional insight with respect to the *potts* and *phmm* models, so we restrict here the presentation to these two choices.

All these models are learned on a seed alignment using a standard Boltzmann machine DCA learning algorithm [51]. We used a constant learning rate of 5×10^{-2} for most protein families and 10^{-2} for all RNA families and for the longest protein families we used. Because the seed often contains very few sequences, we need to introduce a small pseudocount of 10^{-5} to take into account nonobserved empirical second moments. The Boltzmann machine performs a Monte Carlo sampling of the model using 1000 independent chains and sampling 50 points for each chain (in total the statistics is thus computed using 5×10^4 samples). Equilibration and autocorrelation tests are performed to increase or decrease, if needed, the equilibration or the sampling time of the Monte Carlo.

It is important to keep in mind that the models inferred from the seeds are "nongenerative" because of the reduced number of sequences (samples generated from these models, due to a strong overfitting, are extremely close to seed sequences), but nonetheless they are accurate enough to be used as proper cost functions for our alignment tool. We also mention that models inferred from pseudolikelihood maximization [26,53], which are also known to be nongenerative [51], can be equivalently used for the alignment method described in this work.

B. Insertion penalties

We determine the parameters of the affine insertion penalties using the statistics of the insertions of the seed alignment. Recall that the number of insertions between positions i and $i-1$ is $\Delta n = n_i - n_{i-1} - 1$, as illustrated in Fig. 4. Motivated by the empirical statistics of insertions in true seeds, we model the probability of Δn as

$$P_i(\Delta n) = \begin{cases} \frac{1}{z}\,, & \Delta n = 0, \\ \frac{e^{-\lambda_o^i-\lambda_e^i(\Delta n-1)}}{z}\,, & \Delta n > 0, \end{cases} \quad (29)$$

where

$$z = 1 + \sum_{\Delta n>0} e^{-\lambda_o^i-\lambda_e^i(\Delta n-1)} = 1 + \frac{e^{-\lambda_o^i}}{1-e^{-\lambda_e^i}} \quad (30)$$

is the normalization constant and λ_o^i, λ_e^i are the costs associated with the opening and the extension of an insertion as in the score function defined in Eq. (3). Because the learning of the parameters is done independently for each position i, for the sake of simplicity we will drop the index i in the following.

We determine the values of λ_o, λ_e by maximizing the likelihood $\mathcal{L}(\{\Delta n\}_{a=1}^M \mid \lambda_o, \lambda_e)$ of the data, i.e., the M sequences of the seed, given the parameters, and adding L2-regularization terms in order to avoid infinite or undetermined parameters. Imposing the zero-gradient condition on the likelihood leads to a closed set of nonlinear equations for the maximum likelihood estimators, given in the SI. These equations can be solved, for example, by a gradient ascent scheme in which we iteratively update

$$\lambda^{t+1} \leftarrow \lambda^t + \eta\frac{\partial\mathcal{L}_t}{\partial\lambda} \quad (31)$$

for both λ_o and λ_e, until numerical convergence (more precisely, when the absolute value of the gradient is less then 10^{-4}). The learning rate is $\eta = 10^{-3}$. Note that the empirical

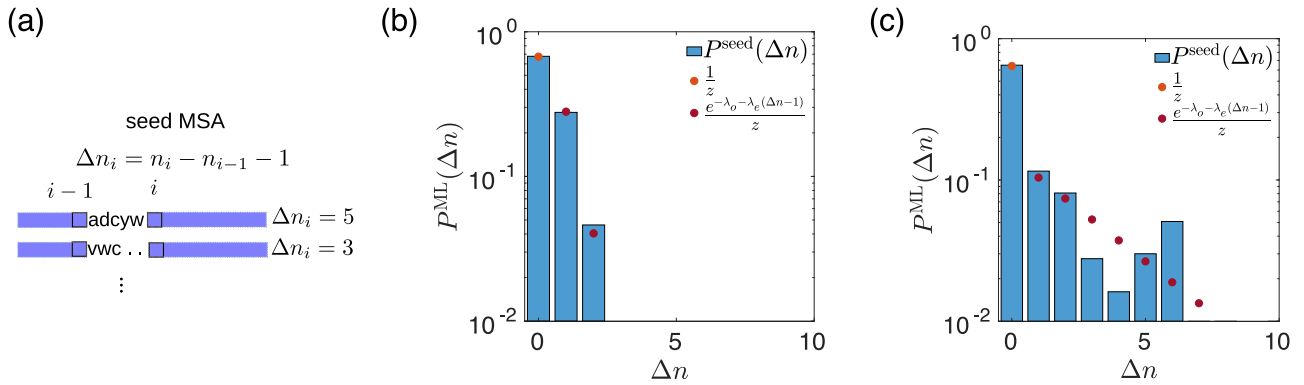

FIG. 4. Inference of insertion penalties. (a) Schematic representation of the Δn variables: The number of insertions between two consecutive positions in the alignment can be computed from the pointer variables $\boldsymbol{n}$. [(b), (c)] Examples of fitting of the empirical probability of Δn using our maximum likelihood approach for (b) position 25 and (c) position 92 for RF00162. In panel (c), the data distribution does not show an exponential profile but our approximation fits well the empirical probability for most of the observed Δn.

distribution can differ from that of our model: For instance, we often encounter positions where either no insertion is present within the seed or the distribution of the positive Δn is not exponential. In the first case, our maximum likelihood approach cannot be applied as it is: In order to apply it, we pretend that the probability of observing at least one insertion is equal to a small parameter ϵ so that we can slightly modify $P^{\rm seed}(\Delta n = 0) = 1 - \epsilon$. In our work, this parameter has been set to $\epsilon = 10^{-3}$. In the second case, we notice that the distribution given by the fit is anyway a nice approximation of the true one. Some examples are reported in Fig. 4.

C. Gap penalties

The gap state is treated in DCA models as an additional amino acid but, by construction of the MSA, it is actually an *ad hoc* symbol used to fill the vacant positions between well-aligned amino acids that are close in the full-length sequence $\boldsymbol{A}$ and should be more distant in the aligned sequence $\boldsymbol{S}$. Thus, the proper number of gaps for each candidate alignment is often sequence dependent and not family dependent: The one-point and two-point statistics of gaps computed from the seed may not be representative of the full alignment statistics. Yet, the couplings and the fields of the DCA models learned from the seed tend to place gaps in the positions mostly occupied by gaps in the seed. This may lead to some bias depending on the seed construction: We notice that if we create seeds using randomly chosen subsets of Pfam [12] alignments produced by HMMer [11], our alignment method, DCAlign, is likely to produce very gapped sequences. In these cases, gaps appear very often, more often than any other amino acid, indicating that our cost function encourages the presence of gaps. Real seeds are instead manually curated and therefore they generally contain few gaps. Even though the Potts models learned from this kind of seeds are less biased, we need to check whether the issue exists anyway and, if needed, treat it. To do so, our idea is to introduce additional penalties to gap states, $\mu_{\rm ext}$ and $\mu_{\rm int}$, as we discussed in Sec. II B in the definition of the cost function. Notice that the distinction between "internal" and "external" gap penalties allows us to differentiate between gaps that are artificially introduced (as in the case of the internal ones) and gaps that reflect the presence of well aligned but shorter domains or fragments, of effective length $L_{\rm frag} < L$. In this last case, some "external" gaps are needed to fill the $L - L_{\rm frag}$ positions at the beginning or at the end of the aligned sequence.

Contrary to the insertion penalties, the gap penalties cannot be directly learned from the seed alignment via an unsupervised training (as their statistics is already included in the Potts model to begin with), but they can be learned in a supervised way. A straightforward procedure consists in realigning the seed sequences using the insertion penalties and the DCA models (*potts* or *phmm*) described in Sec. IV A for several values of $\mu_{\rm ext}$ and $\mu_{\rm int}$. The best values of the gap parameters are those that minimize the average Hamming distance between the realigned seed and the original seed sequences. We performed this supervised learning by setting the values of $\mu_{\rm ext} \in [0.00, 4.00]$ and $\mu_{\rm int} \in [0.00, 4.00]$. These intervals have been chosen after several tests in a larger range of variability, also including negative values (that favor gaps) and very large values compared to the typical parameters of the Potts models. We observed that (i) favoring gaps is always counterproductive and (ii) there exists a threshold, usually around 4, beyond which no gap is allowed in the sequence, which is also counterproductive. For these reasons, and because of the high computational effort required to realign the sequences several times, we decided to use the interval $[0.00, 4.00]$ with sensitivity 0.50 leading to 81 realignments of the seed sequences.

This method works for seeds that contain a large number of sequences (typically $M > 10^3$) but it fails completely when dealing with "small" seeds. In this case, whatever the value of $(\mu_{\rm ext}, \mu_{\rm int})$, the realigned sequences are always identical to the original ones, resulting in an average Hamming distance equal to zero. Indeed, the energy landscape of models learned from few sequences is populated by very isolated and deep local minima centered in the seed sequences. When the algorithm tries to realign an element of the training set, it is able to perfectly minimize the local energy and realign the sequence with no error whatever the additional gap penalty. For short seeds, instead of realigning the seed, we thus extract 1500 sequences from the full set of unaligned

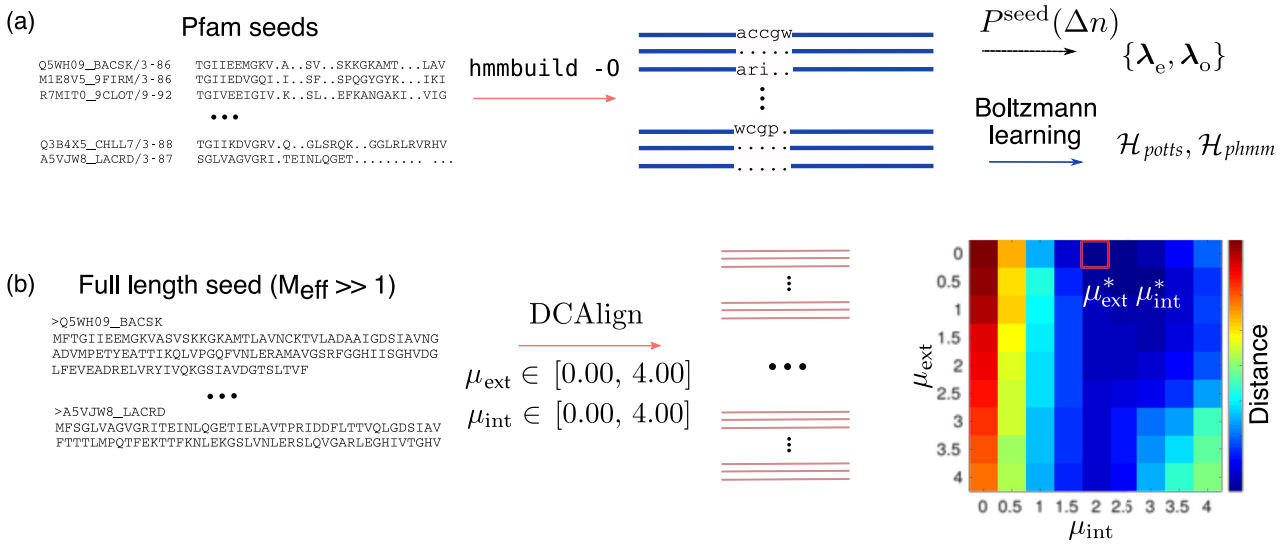

FIG. 5. Scheme of the training process for DCAlign. In panel (a), we show the first step of the learning. We build the aligned seed of a Pfam family using `hmmbuild -O` to detect the matched amino acids (blue line) and the insertions (shown in lowercase letters and "."). From these data, we learn the DCA Hamiltonian and the affine insertion penalties. In panel (b), we pictorially describe the learning of the gap penalties. Here we take into account the seed itself (not only the aligned part but the entire sequences) and we try to align it using the parameters inferred in step (a) using all the 81 possible combinations of $\mu_{\rm ext}$ and $\mu_{\rm int}$, each spanning the interval [0.00, 4.00]. We compare each candidate alignment to the seed alignment, directly using the Hamming distance. The best set of parameters is that minimizing this metric. The plot in panel(b) shows the (average) Hamming distance of the true and realigned seed for the PF00677 family.

sequences (a *validation* set), which we align by varying the gap penalties, always in the range [0.00, 4.00]. We call $\mathcal{H}^0_{\rm seed}$ the DCA Hamiltonian inferred on the seed, and we infer new Hamiltonians $\mathcal{H}^0_{type,\mu_{\rm ext},\mu_{\rm int}}$ (with $type \in \{phmm, potts, \}$) on all the multiple sequence alignments of the validation set. We then choose the best parameters $\mu_{\rm ext}$ and $\mu_{\rm int}$ as those that minimize the symmetric Kullback-Leibler distance between $\mathcal{H}^0_{seed}$ and $\mathcal{H}^0_{type,\mu_{\rm ext},\mu_{\rm int}}$ (the precise definition is given in Sec. V). In other words, we select the best gap penalties as those that produce a validation MSA as statistically close as possible to the seed alignment.

We underline that the values of the penalty parameters also depend on the choice of the gauge for the DCA parameters: In fact, the advanced mean-field equations are not gauge invariant and depending on the choice of the gauge we can have different (optimal) values for the gap penalties. All results shown in this work use the zero-sum gauge for the DCA parameters.

V. COMPUTATIONAL SETUP

A. Pipeline

The computational setting we propose here is the same adopted by state-of-the-art alignment softwares such as HMMer [54] and Infernal [15]. From the seed alignment, we learn all the parameters that characterize our score function and we apply our alignment tool to all the unaligned sequences that contain a domain compatible with the chosen family. More precisely, once a seed is selected (we used the `hmmbuild -O` function of the package HMMer to obtain the aligned seed), we learn the model, the insertions parameters, and the gap penalties as described in Sec. IV. A scheme of our training method is shown in Fig. 5.

After training, our cost function is fully determined. Unaligned sequences are then taken from the full length sequences of the corresponding family in the Pfam database [12] (we do not face here the problem of detecting homologous sequences). Note that, like HMMer, we also include the seed sequences in the sequences to be aligned, in order to obtain a more homogeneous MSA and test the quality of the realignment of the seed. We do not consider the entire sequences, whose length N is often much larger than L, but a "neighborhood" of the hit selected by HMMer. In practice, we add 20 amino acids at the beginning and at the end of the hit, resulting in a final length $N = 20 + L + 20$. We performed the same experiment using $N = 50 + L + 50$ for PF00684 and the resulting sequences were identical to those obtained from a shorter hit. The method seems to be stable for reasonable values of N, i.e., $N \sim L$. For RNA sequences, this preprocessing is not needed, because the full length sequences downloaded from Rfam already have a reasonable length. For most families, we have aligned all the full length sequences (the size of the test sets is specified in Table II) and only in few cases, for particularly large families, we uniformly pick at random $N_{\rm seq} = 10^4$ sequences to align.

We then apply DCAlign using the approximations we discussed in Sec. III, namely the finite-temperature and zero-temperature MF method, to each candidate sequence and we add to our MSA the aligned subsequence that has the lower energy (insertion and gap penalties excluded). Whenever our algorithms do not converge to an assignment of the variables that satisfies all the hard constraints, we apply the "nucleation" procedure explained in Sec. III C that, by means of the approximated marginal probabilities, gives rise to feasible alignments.

TABLE II. Features of the protein and RNA families used in this work. We show here the length L of the sequences for each family, the values of M and M_{eff} [25] for the seed alignment, the number of crystal structures available in the Protein Data Bank (PDB) used to determine the true contact maps based of real observations of the domains structure, the number of the sequences, N_{seq}, to be aligned by our methods, and the value of the gap penalties associated with each family and Hamiltonian. For a subset of 100 uniformly randomly chosen sequences, we show the average length N of the unaligned sequence (for protein domains, this is set to $20 + L + 20$); the last column shows the mean and median values of the computing time.

						μ_{ext}		μ_{int}			Mean t, median t [s]
Identifier	L	M seed	M_{eff} seed	PBD	N_{seq}	$\mathcal{H}_{\text{potts}}$	$\mathcal{H}_{\text{phmm}}$	$\mathcal{H}_{\text{potts}}$	$\mathcal{H}_{\text{phmm}}$	Mean N	$\mathcal{H}_{\text{potts}}$
PF00035	67	81	81	73	19751	2.50	2.00	0.00	2.00	20+L+20	38, 8
PF00677	87	1878	1518	9	14683	0.00	0.50	2.00	2.00	20+L+20	18, 17
PF00684	67	1512	1349	3	10000	0.00	0.00	2.50	2.00	20+L+20	20, 7
PF00763	116	1389	1355	24	10000	1.50	2.50	1.00	1.50	20+L+20	64, 32
RF00162	108	433	241	25	6026	3.50	3.50	3.00	4.00	112	108, 30
RF00167	102	133	105	49	2631	0.50	1.50	2.00	4.00	100	144, 33
RF01734	63	287	287	6	2017	1.00	0.00	2.00	1.50	70	23, 7
RF00059	105	109	83	24	12558	0.00	0.50	1.50	1.50	110	223, 48

B. Observables

To assess the quality of the MSAs generated by DCAlign (that differ in the score function being used to align) and to compare them to the state-of-the-art alignments provided by HMMer (or Infernal), we consider the following observables:

(1) *Sequence-based metrics.* When comparing two candidate MSAs of the same set of sequences (a "reference" and a "target"), it is possible to compute several sequence-wise measures such as the following metrics (normalized by L, the length of the sequences):

(a) The Hamming distance between the two alignments of the same sequence in the reference and target MSAs;

(b) Gap_+: the number of match states in the aligned sequence of the reference MSA that have been replaced by a gap in the target MSA;

(c) Gap_-: the number of gaps in the aligned sequence of the reference MSA that have been replaced by match states in the target MSA;

(d) Mismatch: the number of amino-acid mismatches, that is, the number of times we have match states in both sequences, reference and target, but to different amino acids (positions) in the full sequence $\boldsymbol{A}$.

(2) *Proximity measure.* Consider two different MSAs of the same protein or RNA family. We can compute, for each candidate sequence S_i^1 of the first set, the Hamming distance d_H with respect to all the sequences of the second set. We then collect the minimum attained value, i.e,

$$\hat{d}_i = \min_j d_H\left(S_i^1, S_j^2\right), \tag{32}$$

which gives the distance to the closest sequence in the other MSA. The distribution of $\hat{d}_i$, or some statistical quantity computed from them (such as the average or the median value) provides a good measure of "proximity" between the two sets. We will show below a few examples using the full alignment of a protein family as a first set, and the seed sequences as the second one.

(3) *Symmetric Kullback-Leibler distance*. Another convenient global measure is the symmetric Kullback-Leibler distance between a Boltzmann equilibrium model learned from the seed alignment (the "seed" model) and another model learned from a candidate MSA (the "test" model). In general, the symmetric Kullback-Leibler divergence is a measure of "distance" between two probability distributions and it is defined, for arbitrary densities $\mathcal{P}_1(\boldsymbol{x})$ and $\mathcal{P}_2(\boldsymbol{x})$ of the variables $\boldsymbol{x}$, as

$$D_{\text{KL}}^{\text{sym}}(\mathcal{P}_1, \mathcal{P}_2) = D_{\text{KL}}(\mathcal{P}_1||\mathcal{P}_2) + D_{\text{KL}}(\mathcal{P}_2||\mathcal{P}_1), \tag{33}$$

where D_{KL} is the Kullback-Leibler divergence

$$D_{\text{KL}}(\mathcal{P}_1||\mathcal{P}_2) = \sum_x \mathcal{P}_1(\boldsymbol{x}) \log \frac{\mathcal{P}_1(\boldsymbol{x})}{\mathcal{P}_2(\boldsymbol{x})}\,. \tag{34}$$

In our context, the symmetric KL distance can be efficiently computed through averages of energy differences as

$$\begin{aligned} &D_{\text{KL}}^{\text{sym}}(\mathcal{P}_{\text{seed}}, \mathcal{P}_{\text{test}}) \\ &\quad = D_{\text{KL}}(\mathcal{P}_{\text{test}}||\mathcal{P}_{\text{seed}}) + D_{\text{KL}}(\mathcal{P}_{\text{seed}}||\mathcal{P}_{\text{test}}) \\ &\quad = \langle \mathcal{H}_{\text{seed}} - \mathcal{H}_{\text{test}} \rangle_{\mathcal{P}_{\text{test}}} + \langle \mathcal{H}_{\text{test}} - \mathcal{H}_{\text{seed}} \rangle_{\mathcal{P}_{\text{seed}}}, \end{aligned} \tag{35}$$

where $\mathcal{P}_{(.)}$ is the Boltzmann distribution associated with the energy $\mathcal{H}_{(.)}$, and the brackets $\langle ... \rangle_{\mathcal{P}}$ denote the expectations with respect to $\mathcal{P}$, which can be easily estimated using a Monte Carlo sampling (in contrast to the normal D_{KL}, which depends on the intractable normalization constants, i.e, the partition functions, of the densities $\mathcal{P}_{(.)}$). We run the comparison using two different models for $\mathcal{H}$, that is a Potts model and a profile model.

(4) *Contact map*. The couplings of DCA models can be used to detect the presence of physical interactions between pairs of sites, which are distant in the one-dimensional chain but in close contact in the three-dimensional structure. A good score that indicates a direct contact is the (average-product corrected) Frobenius norm of the coupling matrices, defined as

$$\mathcal{F}_{i,j}^{APC} = \mathcal{F}_{i,j} - \frac{\sum_m \mathcal{F}_{i,m} \sum_n \mathcal{F}_{n,j}}{\sum_{m,n} \mathcal{F}_{m,n}}, \tag{36}$$

where

$$\mathcal{F}_{i,j} = \sqrt{\sum_{A\neq'-',B\neq'-'} J_{i,j}(A,B)^2}, \tag{37}$$

and the couplings are in the zero-sum gauge. We thus compare predicted contact maps obtained using the parameters learned

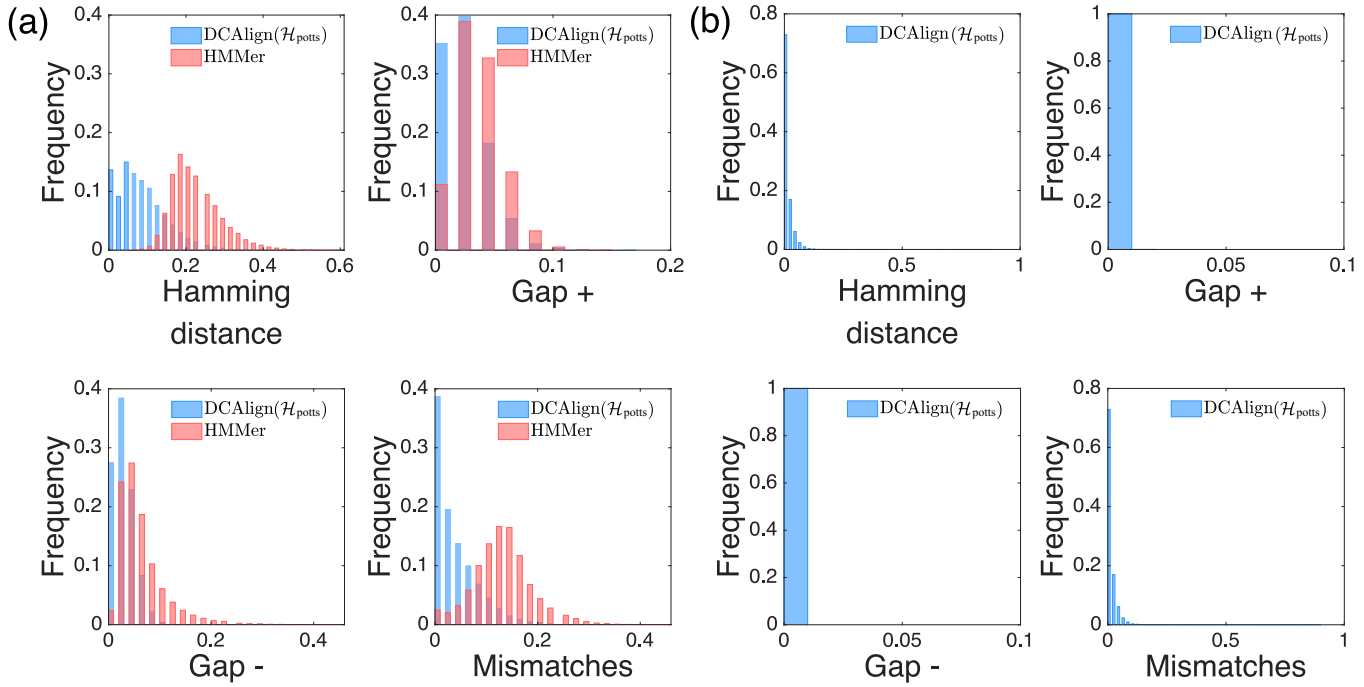

FIG. 6. Comparison between DCAlign and HMMer for synthetic data. Panels (a) and (b) show the histograms of the normalized metrics (Hamming distance, Gap_+, Gap_-, and Mismatches), respectively, in the case of conserved sited only, and of correlated pairwise columns only. Here, the reference is the ground truth and the target is the alignment of DCAlign (blue) or HMMer (red). In panel (b), HMMer results are not shown because `hmmsearch` does not find any relevant hit.

from our alignments and those obtained by HMMer. For this purpose, we use the PlmDCA method to learn the couplings from the alignments, because it is faster than the Boltzmann machine and it is known to be reliable for contact prediction [26]. The ground truths denoting the physical interactions in each protein are obtained running the *Pfam interactions* package [55]. Two sites are said to be in contact if the minimum atomic distance among all the atoms and among all the available protein structures is less than 8 Å.

VI. TEST ON SYNTHETIC DATA

Here we describe two experiments on synthetic data, constructed to compare the performances of DCAlign to state-of-the-art methods in extreme settings: One data set presents conserved but not coevolving sites (i.e., strong variations in amino-acid frequency on each site but no correlation between distinct sites), while the other presents not conserved but coevolving sites (i.e., uniform frequency $1/q$ of amino acids on each site, but strong correlations between distinct sites).

A. Conservation

The first MSA is generated from a nontrivial profile model, in which the empirical probability of observing any of the possible amino acids is position dependent and it is not uniformly distributed among the possible states. The generative model used in this case is the profile model of the PF00018 family, which can be easily learned from the empirical single-site frequencies. From this model, we generate 5×10^4 "aligned" sequences (the "ground truth"), to which we randomly add some insertions, according to the affine insertion penalty distributions learned from the PF00018 seed (Sec. IV B) and 20 uniformly randomly chosen symbols at the beginning and at the end of the aligned sequence. We split this alignment into a training set of 2.5×10^4 sequences, which we use as seed alignment to learn the insertion and gap penalties (using the scheme for abundant seeds) and a Potts model. We align the remaining 2.5×10^4 sequences, used as test set. For comparison, we build a hidden Markov model using `hmmbuild` of the HMMer package on the training set and we align the test sequences through the `hmmsearch` tool.

We show in Fig. 6(a) the histograms of the (normalized) Hamming distances, Gap_+, Gap_-, and mismatches of the MSA obtained by DCAlign and HMMer compared to the ground truth. We observe that DCAlign is able to find the correct hits and to align them in a more precise way if compared to HMMer. In fact, the Hamming distance distribution is shifted to smaller values, suggesting that the number of errors, per sequence, is smaller than that obtained by HMMer. The nature of the mistakes seems to be linked to the presence of mismatches in the case of HMMer, while DCAlign (less often) equally likely inserts more or less gaps, or a match to the wrong symbol. While DCAlign is in principle constructed to exploit coevolution, these results show that even in cases in which, by construction, there is no coevolution signal, DCAlign is able to perform equally good (or even better) than state-of-the-art methods.

B. Covariance

The second experiment is instead focused on correlated data. We *ad hoc* construct an alignment whose first moment statistics resemble those of a uniform distribution; i.e., the

probability of observing any amino acid, in any column of the seed, is $1/q$. In other words, we construct the data in such a way that no conserved sites are present. At the same time, we force the sequences to show coevolving (i.e., correlated) sites, such that the empirical probability of observing a pair of amino acids is different from that obtained in the uniform distribution, i.e., $f_{ij}(S, S') = \overline{\delta_{S_i,S}\delta_{S_j,S'}} \neq \frac{1}{q^2}$ for some (i, j), where the overline indicates the empirical average. To construct a data set with these statistics, we use as generative model a Potts model with four colors (like the RNA alphabet, without the gap state) having nonzero couplings $J_{ij}(S_i, S_j) = -\delta_{S_i,S_j}$ (i.e., an antiferromagnetic Potts model) and no fields. The nonzero couplings are associated with the links of a random regular graph of 50 nodes and degree 5. The presence of the links ensures the appearance of nontrivial second moments while, in order to avoid "polarized" sites, we sample the model (i.e., the Boltzmann distribution associated with this Hamiltonian) at temperature $T = \frac{1}{\beta} = 0.3$, which is deep in the paramagnetic phase of this model [56]. We perform the same training pipeline presented in Sec. VI A except for the learning of the gap penalties: Because there are no gap states, we set $\mu_{\text{ext}} = 0$ and $\mu_{\text{int}} = 0$.

In this case, due to the absence of any conservation, `hmmsearch` does not find any eligible hit. In fact, HMMer tries to align the sequences via a computationally exact recursion on a HMM, but it has no information to exploit while setting up the HMM from the training set, because all amino acids are equally likely to occur in each column. This represents, of course, the worst-case scenario for HMM-based methods. In contrast, the couplings of the learned Potts model allow DCAlign to align this kind of sequences. We remark that in contrast to HMMer, DCAlign has complete information on the statistics of the training alignment, up to second-order covariances. However, being a heuristic method, it sometimes fails to achieve the (global) minimum of the cost function and converges to a local minimum, which depends on the initialization of the target marginals. Reiterating the MF equations using 10 different seeds of the random number generator suffices to reach the proper minimum at least once, for the majority of sequences. We remark that this issue is present only when the MSA does not show any conserved site and thus the algorithm has no easy "anchoring" point, which surely helps lifting the degeneracies in the alignment procedure. For protein and RNA families presented below, the algorithm seems stable and only one minimum emerges upon reinitialization of the marginal probabilities of the algorithm. We quantitatively measure the performance of DCAlign using four sequence-based metrics and the energies associated with the aligned sequences. For this experiment, we refer to the output of our algorithm as the aligned sequence that has the minimum energy among the 10 trial reiterations of the algorithm. We report the distance metrics in Fig. 6(b): The distribution of the Hamming distances suggests that DCAlign can almost perfectly align the majority of the target sequences. Indeed, as shown in Fig. 7, the energies (the Potts Hamiltonian alone or the full cost function which includes the gap and insertion penalties) are identical or very close to the energies of the true sequences. Only 0.08% of the aligned sequences have a Hamming distance density larger than 0.30 (i.e., 15 missed positions over 50). This fraction is so low as to be invisible in the histograms of Fig. 6(b). These extreme cases, in which our algorithm converged to a local minimum in every trial we performed, are represented as purple points in the scatter plots in Fig. 7.

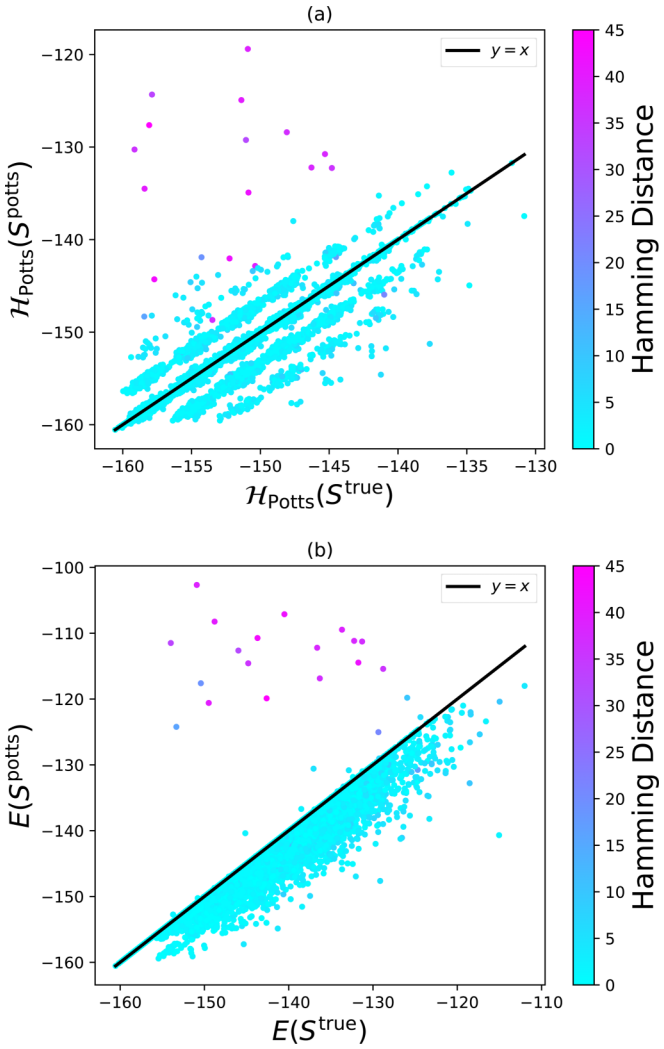

FIG. 7. Energies of synthetic sequences. We show here the scatter plots of the DCA energy (or Potts Hamiltonian) in panel (a) for the true synthetic sequences (x axis) against the ones aligned by DCAlign($\mathcal{H}_{\text{potts}}$) ($y$ axis). (b) Same plot using the total cost function E.

VII. TEST ON PROTEIN AND RNA FAMILIES

A. Choice of families

We show here the performance of our alignment method for several RNA and protein families. In particular, we select the families PF00035, PF00677, PF00684, and PF00763 from the Pfam database (release 32.0) [12,57], and RF00059, RF00162, RF00167, and RF01734 from the Rfam database (release 14.2) [13,58]. The number of sequences, length of the models, and gap penalties used in the simulations are reported in Table II.

We restrict our analysis to "short" families, having L of at most 100 positions, in order to avoid a significant slowing

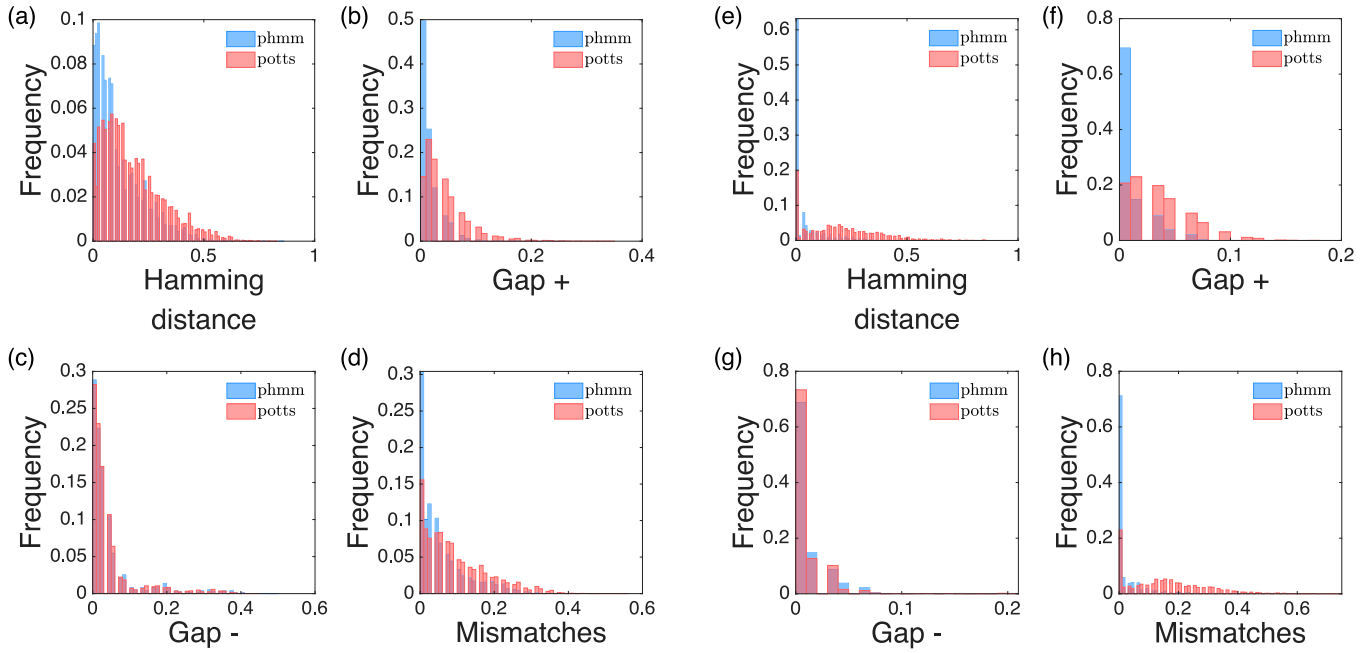

FIG. 8. DCAlign vs state-of-the-art methods for PF00035 and RF00167. We plot here the histograms of the Hamming distances, Gap_+, Gap_-, and mismatches for the protein family PF00035 [(a)–(d)] using as reference the HMMer results and as target the DCAlign results, and for the RNA family RF00167 [(e)–(h)] using as reference the Infernal results and as target the DCAlign results.

down of the alignment process. The seed of PF00035 contains very few sequences, in contrast to PF00677, PF00684, PF00763, which have been chosen because of their large effective number of seed sequences $M_{\mathrm{eff}} > 1000$ (after a standard re-weighting of close-by sequences [25]). We thus always infer the gap penalties according to the abundant seed protocol, except for PF00035. As reference for comparison, we consider the alignments produced by HMMer [54] and already available in the Pfam database. We also perform the alignment of several RNA families, for which secondary-structure knowledge is necessary to obtain good alignments with standard tools. We compare our estimate against that obtained by the state-of-the-art package Infernal [15] which, indeed, employs the secondary structure of the target domains in order to build the so-called covariance model used to align. Note that, in contrast, DCAlign does not use any secondary structure information in the training procedure (but DCA is able to predict the latter [59]). As a further comparison, we also learn a hidden Markov model (using `hmmbuild`) and we apply `hmmalign` to the full-length RNA sequences. We choose precisely these families because of their reasonable length, the abundance of the seed sequences, and the large number of available crystal structures, which are useful for the contact map comparison.

B. Comparison with state-of-the-art methods

As a first comparison, we compute the sequence-based metrics presented in Sec. V B, comparing our full alignment to that achieved by HMMer, for protein sequences, or by Infernal, when dealing with RNA families. We show the results for PF00035 in Figs. 8(a)–8(d) and for RF00167 in Figs. 8(e) and 8(f), which are representative of the typical scenario for protein and RNA sequences. The distribution of all metrics is mostly concentrated in the first bins (the bin width is set here to 0.01) and decays smoothly at larger distances. The peak in the first bin is more prominent when the Hamiltonian used for the alignment is *phmm*, indicating that the sequences aligned by this method are closer to those obtained by HMMer (or Infernal) than the outcomes of DCAlign-*potts*, as one would expect from the similarity between the two alignment strategies (see Sec. IV A).

A notably different behavior emerges for the sequences of the PF00677 family, as shown in Figs. 9(a)–9(d). It is clear from Fig. 9(a) that a large fraction of the sequences aligned by DCAlign differs from those aligned by HMMer by about 40% of the symbols when using $\mathcal{H}_{\mathrm{potts}}$ (the percentage is reduced to about 30% when using $\mathcal{H}_{\mathrm{phmm}}$). The reason seems to be partially linked to the presence of mismatches and to a non-negligible fraction of additional gaps, as indicated by Gap_+. We notice that, unlike the other families, the seed of PF00677 is composed by several clusters of sequences mostly differing in the gap composition: A copious fraction of them have generally few gaps while some other show two long and localized stretches of gaps. The structure of the seed can be better characterized in the principal component space, as depicted in Fig. 9(e), where we plot the projections of the seed sequences in the space of the first two principal components as filled circles. The colors refer to the density of sequences in the (discretized) space. To understand the disagreement between HMMer and our methods, we superimpose the projections of the sequences responsible for the huge peaks in the Gap_+ histogram in the principal component space of the seed sequences, using brown crosses for the sequences aligned by HMMer and pink diamonds for those found by DCAlign-*potts* (note that none of these sequences is a realignment of a seed sequence). Only a small fraction of the HMMer sequences overlap with the central and poorly populated cluster while

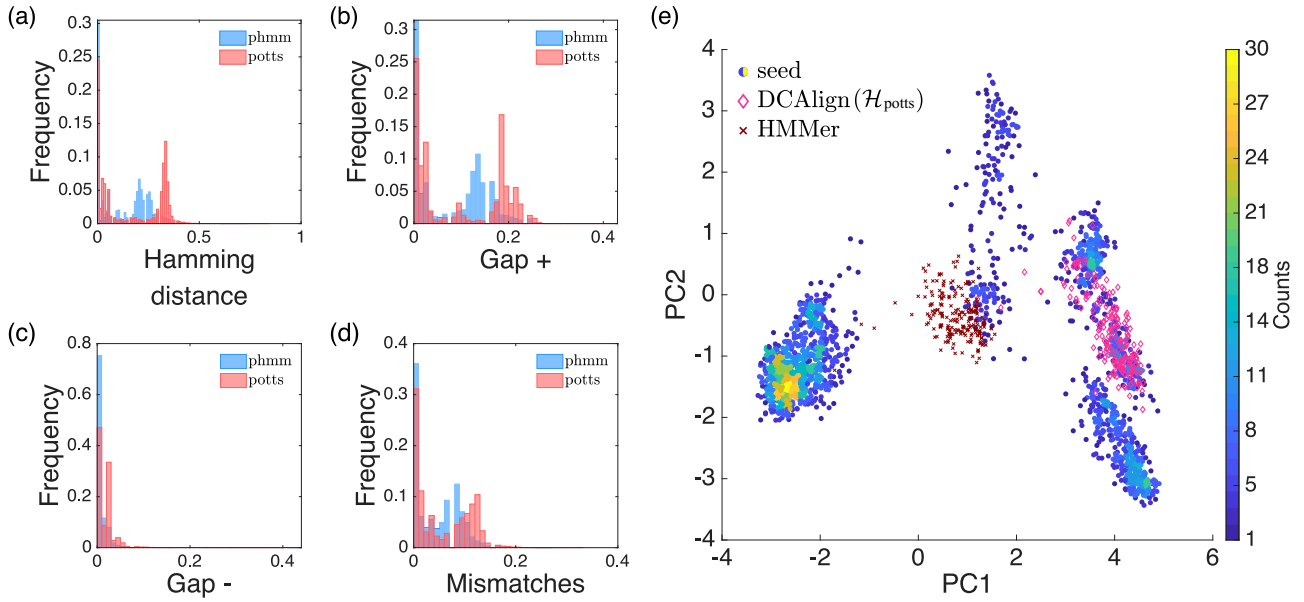

FIG. 9. DCAlign vs HMMer for PF00677. In (a, b, c, d) we plot the Hamming distance, Gap_+, Gap_- and Mismatches using the sequences aligned by HMMer as reference and those obtained by DCAlign as target. In (e) we plot the projections of the seed sequences in the first two principal components of the seed space; the color scale denotes the density of the space. The additional sequences (depicted as pink diamonds if aligned by DCAlign-*potts* or as brown crosses if aligned by HMMer) are responsible for the red peak around 0.2 in panel (b).

the projections obtained from sequences aligned by DCAlign-*potts* lie on a well-defined and populated cluster. We thus conclude that looking at the gap composition of sequences is not sufficient in this case to understand the different behavior of HMMer and DCAlign. A more accurate analysis in the principal components space suggests that the sequences obtained by HMMer are probably miscategorized, at variance with DCAlign sequences that are in agreement with the seed structure.

In summary, although for some of the families analyzed here (the distribution of the four metrics for the remaining families are shown in the SM [50]) the sequences aligned by DCAlign are very similar to those obtained by HMMer or Infernal, the PF00677 family suggests a different scenario, in which DCAlign is able to learn some nontrivial correlations present in the seed and to exploit them in order to achieve a better alignment of the target sequences. DCAlign is then able to reproduce state-of-the-art performance in most cases and to improve them in some cases.

C. Comparison with the seed

In this section, we compare the statistical properties of the MSAs obtained by DCAlign with those of the seed.

1. Kullback-Leibler distances

The statistics of a MSA can be characterized in terms of a statistical (DCA) model. Depending on the complexity of the model, a certain set of observables are fitted from the MSA. For instance, in a profile model only the first moments are fitted, while in a Potts model we can also fit the information about second moments. These statistical models define a probability measure over the space of sequences and thus characterize a given protein/RNA family. We consider here the seed sequences as our ground truth, and we thus consider that a model learned from the seed is the one that better characterizes the protein/RNA family under investigation. We then infer a second model from the full set of aligned sequences, and we ask how different this model is from that learned from the seed. To answer this question, we compute the symmetric Kullback-Leibler divergence $D_{\rm KL}^{\rm sym}$ between the two models (see Sec. V B), which must be intended as a statistical measure of distance between the seed and the set of aligned sequences. In order to fairly compare DCAlign, HMMer, and Infernal, which by construction treat differently the pairwise covariation of the MSA sites, we learn, from a seed and from the test alignment, a profile model $\mathcal{H}^{\rm Prof}$ and a Potts model $\mathcal{H}^{\rm Potts}$.

We show in Figs. 10(a) and 10(b) the results for all families and all methods, when the model learned is a Potts model or a profile model, respectively. We notice that the alignments produced by DCAlign-*potts* always, for the Potts case, and

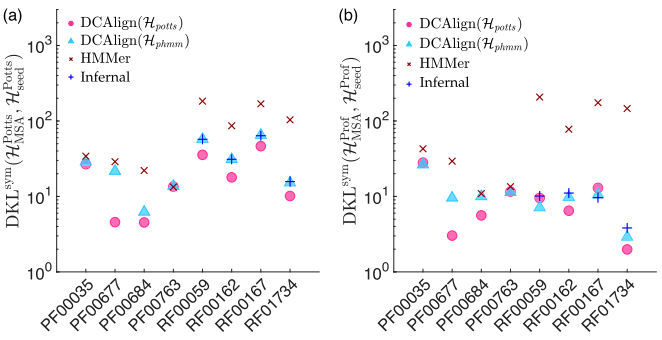

FIG. 10. Symmetric Kullback-Leibler distances. We plot the symmetric KL distance between the MSA and the seed alignment, computed via a Potts model (a) or a profile model (b), for all the families and all the alignment methods we considered.

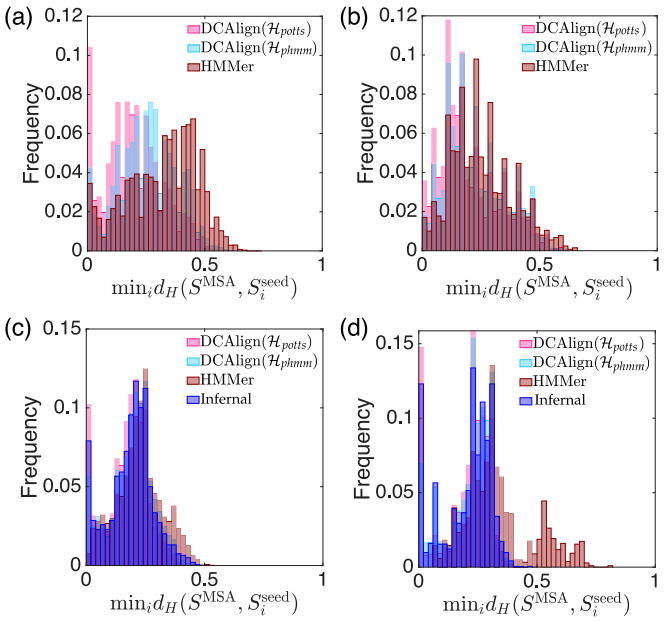

FIG. 11. Distribution of proximity measures. Histograms of the minimum distances computed according to Eq. (32) for the full set of aligned sequences obtained by DCAlign-*potts*, DCAlign-*phmm*, HMMer, and Infernal, against the seed. Panels (a), (b), (c), and (d) refer to the families PF00677, PF00684, RF00162, and RF01734 respectively.

very often, for the profile case, minimize $D_{\rm KL}^{\rm sym}$ with respect to the seed. Infernal is very effective when dealing with RNA sequences but not as good as DCAlign-*potts* for the majority of the cases. HMMer always produces the largest distance (except for PF00763 where basically all methods perform equally well), in particular for RNA families. We mention that alignments produced by `hmmalign` present aligned sequences that always show long concatenated gaps at the beginning and at the end of the sequence, unlike the Rfam full alignment, the seed sequences, and the outputs of DCAlign. This partially explains the difference with respect to the other alignment tools.

These results suggest that the more we use additional information within the alignment process (in particular, when learning $\mathcal{H}_{potts}$ we employ all positions and amino-acid dependent pairwise energy function), the closer the final alignment will be to the seed. Surprisingly, this feature is retrieved even when the model learned from the full set of aligned sequences uses less information than the model used to align, e.g., for the profile model used in Fig. 10(b). Of course, there is a tradeoff, because including additional statistical properties of the seed in the alignment process requires a larger seed.

2. *Proximity measures*

We present here a sequence-based comparison between a candidate alignment, i.e., an alignment obtained by DCAlign-*potts*, DCAlign-*phmm*, or HMMer/Infernal, and the seed that will be considered here as the reference alignment. The metric we use is the proximity measure introduced in Sec. V B. We show in Fig. 11 the distribution of the minimum distances for a representative subset of the families, i.e., PF00677 in Fig. 11(a), PF00684 in Fig. 11(b), RF00162 in Fig. 11(c), and RF01734 in Fig. 11(d). Results for PF00035, PF00762, RF00059, and RF00167 are shown in the SM [50]. We notice that for the majority of the families (protein or RNA) the histograms built from DCAlign-*potts* have a large peak in the first bin (which collects distances from 0 to 0.02), suggesting that there exist more sequences in this alignment which are close to the seed than in any other alignment. A large peak at small distance is also observed for Infernal when dealing with RNA families, as seen from the blue histograms in Figs. 11(c) and 11(d). The Infernal results overlap quite well with those obtained by DCAlign-*phmm*. The histograms produced by HMMer seem to be shifted to larger Hamming distances, thus reflecting a smaller similarity to the seed than all the other methods. Although DCAlign-*phmm* exploits similar information to that encoded in HMMer, the corresponding alignment surprisingly produces, for most of the studied families, results that are more similar to those obtained by DCAlign-*potts* or Infernal.

D. Contact prediction

An important test of the quality of a MSA is related to the interpretability of the DCA parameters learned from it. As mentioned in Sec. V B, the largest couplings are a proxy for the physical contacts in the folded structure of the protein domains. In Table III, we report a summary of the results for three observables associated with the contact prediction: The position of the first false positive in the ranked Frobenius norms, the value of the true positive rate (TPR) at $2L$, and the position at which the TPR is less than 0.80 for the first time. The bold number corresponds to the largest value, and therefore the best performance, among all the methods.

We show in Figs. 12(a)–12(d) the positive predictive value (PPV) curves (left) and the contact maps (right) for the PF00035, PF00684, PF00763, and RF00162 families respectively (results for PF00677 and the other RNA families are shown in the SM [50]). The PPV curves are constructed by plotting the fraction of true positives TP as a function of the number of predictions (TP and FP), i.e., PPV $=$ TP/(TP $+$ FP). The true contact maps are extracted from all the available PDBs and plotted as gray filled squares, while the predicted contact maps are constructed by plotting the Frobenius norms of the DCA couplings that are larger than an arbitrary threshold, here set to 0.20. For RNA sequences, the comparison between the predictions and the ground truth can be performed only using the Frobenius norms associated with the central part of the aligned sequences, because there is no available structural information about the sites on the boundaries. In addition to the predictions obtained from the full set of aligned sequences, we show, for comparison, the predicted contact map obtained from the Potts model inferred from the seed sequences alone. As we can notice from Table III and the plot of the contact maps, there is no strategy that clearly outperforms the others (except the poor results of *seed*, which are easily explained by its limited number of sequences). For RNA families, Infernal seems to accomplish the best predictions in terms of first FP and TPR but nonetheless all the other methods, including HMMer, show comparable results. In fact, although HMMer has the tendency to assign consecutive gaps

TABLE III. Summary of the contact map results. For each protein or RNA family, we show here three metrics computed from the PPV curve retrieved from a set of Potts models. $\mathcal{H}_{\text{seed}}$ is a Potts model learned using the seed sequences alone, while the others are associated with the complete alignments obtained by DCAlign-*potts*, DCAlign-*phmm*, HMMer, and Infernal. The chosen observables give the position of the first false positive (first FP), the value of the true positive rate (TPR) computed after $2L$ predictions and the rank at which the value of the true positive rate is smaller than 0.80 for the first time. A perfect prediction is obtained if all the true positive contacts are associated with the highest value of the Frobenius norm, and thus the higher the value of these metrics, the better the prediction of the contact maps. We show in bold numbers the best performances, for all metrics and among all the methods.

	First FP, TPR(2L), TPR < 0.80				
Identifier	$\mathcal{H}^{\text{Potts}}_{\text{seed}}$	DCAlign($\mathcal{H}_{\text{potts}}$)	DCAlign($\mathcal{H}_{\text{phmm}}$)	HMMer	Infernal
PF00035	9, 0.478, 13	**35**, 0.754, 98	32, 0.799, **134**	28, **0.791**, 119	
PF00677	22, 0.730, 109	**47**, 0.759, 147	28, 0.747, 128	31, **0.793**, **163**	
PF00684	20, 0.582, 28	**29**, 0.672, 101	27, **0.694**, **104**	23, 0.627, 73	
PF00763	80, 0.703, 159	89, 0.828, **288**	85, 0.836, 254	**106**, **0.849**, 272	
RF00059	18, 0.369, 29	29, 0.531, 57	29, 0.519, **64**	**37**, 0.519, 59	33, **0.566**, **64**
RF00162	15, 0.306, 52	17, 0.449, **69**	**25**, 0.398, 61	22, 0.426, 67	19, **0.519**, 61
RF00167	19, 0.324, 27	27, 0.493, 59	25, 0.556, 53	22, 0.577, **62**	**28**, **0.592**, 57
RF01734	**10**, 0.300, 16	**10**, 0.380, 16	**10**, **0.430**, 16	**10**, 0.360, 16	**10**, 0.400, **17**

in the first and last sites of the aligned sequences, these regions are not considered in the comparison, and the core part of the alignment suffices to obtain similar results, in terms of contact prediction, to the other methods. Although in the other metrics presented above there was no clear difference between models learned from large or small seeds, in the contact maps comparison this seems to be an important issue. Indeed, the amount of sequences in the seed slightly affects the quality of the contact map for our methods: For the PF00035 family (whose seed contains only 81 sequences), DCAlign reaches slightly worse performances than HMMer. In contrast, for the PF00763 all methods produce indistinguishable PPV curves and contact maps. Finally, we remark on the results for PF00684 in Fig. 12(b), where DCAlign achieves a better contact prediction, as manifested by the PPV lines. This result, not linked to the way of encoding the seed statistics within the model but shared by both $\mathcal{H}_{\text{potts}}$ and $\mathcal{H}_{\text{phmm}}$, could be caused by a better treatment of the insertions with respect to HMMer.

E. Running time

As mentioned in Sec. II A, the running time of DCAlign scales roughly quadratically on L and N. To give a reference computing time for each family considered here, we aligned 100 uniformly randomly chosen sequences using a laptop computer, and we measure the mean and the median running times as well as the average length of the full-length sequences

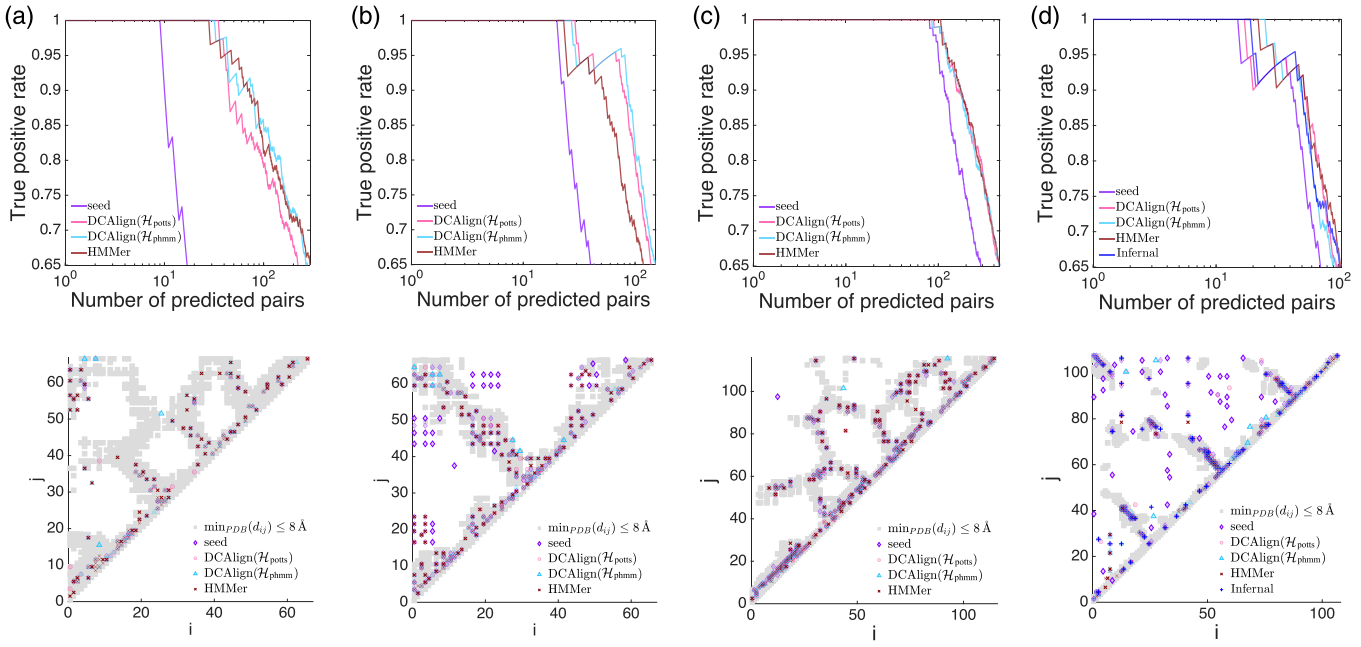

FIG. 12. Contact predictions. We show the positive predictive value curves on the top panels, and on the bottom ones, the contact map retrieved by a set of known crystal structures (gray squares) and the Frobenius norms (computed from the full set of aligned sequences or the seed), for (a) PF00035, (b) PF00684, (c) PF00763, and (d) RF00162.

(note that for protein domains this is fixed to $20 + L + 20$ while for RNA domains it varies). We show these quantities in the last columns of Table II. Remarkably, the running time is, as expected, affected by the length of the unaligned sequence, but also by the number of sequences of the seed. We notice that the more copious the seed is, the less time (more precisely, the less number of iterations of the MF-based algorithm) is needed to converge. For instance, RF00162 and RF00059 unaligned sequences have roughly the same length L (108 and 102 respectively) and average N (112 and 110) but they differ in the M_{eff} of the seed (241 and 83 respectively); this seems to affect the mean computing time, 108 and 223 s for RF00162 and RF00059 respectively. This suggests that accurate models allow for fast alignment, as DCAlign is able to easily detect the target domains while models learned from "small" seeds require more iterations (sometimes the maximum number of iterations, here set to 1000, as in some RNA domains).

VIII. CONCLUSIONS

In this work, we have developed and tested DCAlign, a method to align biological sequences to Potts models of a seed alignment. The set of hyperparameters characterizing the models are inferred by an inverse statistical-physics based method known as direct coupling analysis, which captures both the single- and two-site seed statistics. Single-site statistics often signal residue conservation, i.e., the propensity of some sites to restrict the variation of residue (amino-acid or nucleotide) composition because specific residues are functionally and/or structurally important at certain positions. Two-site statistics are instead related to residue coevolution: For instance, residues in direct contact in a folded protein must preserve biophysically compatible properties, leading to a correlated evolution of pairs of sites.

Most standard alignment algorithms such as HMMer are based on the assumption of independent-site evolution, which is statistically encoded via the so-called profile models, and thus neglect coevolution. In these alignment procedures, strongly conserved residues serve as anchoring points, and a mismatch in these positions surely induces bad alignment scores (i.e., high energies using a physics-like terminology). Variable sites, characterized by high entropy values in the seed MSA, do provide little information for aligning a new sequence to the seed.

However, often residue pairs show a strong degree of coevolution as reflected in two-site statistics, and as a consequence, this important collective information must be taken into account. Up to now, the only example in which this information is exploited in the alignment procedure of RNA sequences, in which the base pairing (Watson-Crick or wobble pairs) of the secondary structure is encoded in the covariance models used to align. Note that this structural information must be given as input to alignment algorithms like Infernal.

In contrast to more specialized alignment algorithms like HMMer (using profile HMM) or Infernal (using covariance models based on secondary RNA structure), DCAlign takes advantage of both conservation and coevolution information contained in the seed alignment and does not require any additional structural input. The most compatible domain among all the possible subsequences of a candidate sequence is determined by maximizing a score, which can be understood as a probability measure of the domain according to a Boltzmann distribution carefully built from the DCA model and gap and/or insertion penalties learned from the seed. We note that the algorithm is formulated in a very general way, and it can thus be applied to any kind of sequence, not necessarily of biological origin.

Using synthetic data at first, we tested the algorithm under extreme conditions, when all information is contained in conservation but none in coevolution, or vice versa. We found that DCAlign performs very well in both cases. This universal applicability is well confirmed in the case of real data; we tested both protein and RNA sequence data, aligning large numbers of sequences to the seed MSA provided by the Pfam and Rfam databases. We find that in most cases, our algorithm performs comparably well to more specialized state-of-the-art methods, while for example profile HMM applied to RNA performs less well. Also, DCAlign does not need any structural information, being based on the seed statistics only.

Remarkably, in one of the studied protein families, we find a large group of sequences, which are aligned differently by HMMer and DCAlign. The sequences aligned by DCAlign show a better coherence with the seed statistics than those aligned by HMMer, as manifested by the principal component analysis in Fig. 9, suggesting that the alignment proposed by DCAlign is to be preferred in this case.

In conclusion, DCAlign provides a general method to solve one of the most important problems in bioinformatics: aligning individual sequences to a reference multiple sequence alignment, taken as a seed. For the first time, this algorithm includes general coevolution information into the alignment process, and it can thus be successfully applied to situations in which conservation is not enough, as we have shown using synthetic data. Furthermore, DCAlign performs equally as well as state-of-the-art methods on protein and RNA data, and outperforms them in some cases. The main limitation of DCAlign is that, even if we have shown how to deal with small seeds, a large enough seed alignment is certainly preferable to obtain more precise coevolution statistics.

The contact map comparison suggests that the models learned from multiple sequences alignments, aligned by DCAlign, are more accurate in predicting real contacts than the models obtained from the seeds. This is probably due to a more precise description of the features of the target family. Therefore, one may think of realigning the full set of unaligned sequences using the model learned from the MSA produced by DCAlign, and iterate this procedure to refine the final MSA. This approach would further slow down the alignment process (because it would require us to redetermine all the terms of the energy function, including the gap and insertion penalties); nonetheless it could have the advantage of fully exploiting the conservation and coevolution signal of the data. We leave this investigation for future work.

On a more technical note, we have not discussed here the problem of detecting homologs from a given long sequence, but we have restricted our application to the neighborhood of the hits selected by HMMer. A possible way of identifying a candidate domain in a long sequence might be running DCAlign for a very few iterations on the full sequence and looking at the marginal probabilities of each site: The

positions associated with the largest probability of matching may correspond to the anchor points of the possible hits. We leave this exploration for future work.

The code for aligning to a given seed model is available in Ref. [60].

ACKNOWLEDGMENTS

The authors thank Sean Eddy, Alessandra Carbone, Francois Coste, and Hugo Talibart for interesting discussions. A.P.M. thanks Edoardo Sarti for interesting discussions and his assistance with the *Pfam interactions* code. A.P.M., A.P., and M.W. acknowledge funding by the EU H2020 research and innovation programme MSCA-RISE-2016 under Grant Agreement No. 734439 INFERNET. A.P.M. and F.Z. acknowledge funding from the Simons Foundation (Grant No. 454955, F.Z.). This work was granted access to the High Performance Computing (HPC) resources of MesoPSL financed by the Region Ile de France and the project Equip@Meso (reference ANR-10-EQPX-29-01) of the program "Investissements d'Avenir" supervised by the "Agence Nationale pour la Recherche".

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

[2] S. B. Needleman and C. D. Wunsch, A general method applicable to the search for similarities in the amino acid sequence of two proteins, J. Mol. Biol. **48**, 443 (1970).

[3] T. F. Smith and M. S. Waterman, Identification of common molecular subsequences, J. Mol. Biol. **147**, 195 (1981).

[4] R. C. Edgar and S. Batzoglou, Multiple sequence alignment, Curr. Opin. Struct. Biol. **16**, 368 (2006).

[5] D.-F. Feng and R. F. Doolittle, Progressive sequence alignment as a prerequisite to correct phylogenetic trees, J. Mol. Evol. **25**, 351 (1987).

[6] J. D. Thompson, D. G. Higgins, and T. J. Gibson, Clustal w: Improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice, Nucl. Acids Res. **22**, 4673 (1994).

[7] C. Notredame, D. G. Higgins, and J. Heringa, T-coffee: A novel method for fast and accurate multiple sequence alignment, J. Mol. Biol. **302**, 205 (2000).

[8] K. Katoh, K. Misawa, K.-i. Kuma, and T. Miyata, Mafft: A novel method for rapid multiple sequence alignment based on fast Fourier transform, Nucl. Acids Res. **30**, 3059 (2002).

[9] R. C. Edgar, Muscle: Multiple sequence alignment with high accuracy and high throughput, Nucl. Acids Res. **32**, 1792 (2004).

[10] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, Gapped blast and psi-blast: A new generation of protein database search programs, Nucl. Acids Res. **25**, 3389 (1997).

[11] S. R. Eddy, Accelerated profile hmm searches, PLoS Comput. Biol. **7**, e1002195 (2011).

[12] S. El-Gebali, J. Mistry, A. Bateman, S. R. Eddy, A. Luciani, S. C. Potter, M. Qureshi, L. J. Richardson, G. A. Salazar, A. Smart, E. L. L. Sonnhammer, L. Hirsh, L. Paladin, D. Piovesan, S. C. E. Tosatto, and R. D. Finn, The Pfam protein families database in 2019, Nucl. Acids Res. **47**, D427 (2018).

[13] I. Kalvari, J. Argasinska, N. Quinones-Olvera, E. P. Nawrocki, E. Rivas, S. R. Eddy, A. Bateman, R. D. Finn, and A. I. Petrov, Rfam 13.0: Shifting to a genome-centric resource for non-coding RNA families, Nucl. Acids Res. **46**, D335 (2017).

[14] S. R. Eddy, Profile hidden Markov models, Bioinformatics (Oxford) **14**, 755 (1998).

[15] E. P. Nawrocki and S. R. Eddy, Infernal 1.1: 100-fold faster RNA homology searches, Bioinformatics **29**, 2933 (2013).

[16] R. Nussinov and A. B. Jacobson, Fast algorithm for predicting the secondary structure of single-stranded RNA, Proc. Natl. Acad. Sci. USA **77**, 6309 (1980).

[17] M. Zuker and D. Sankoff, RNA secondary structures and their prediction, Bull. Math. Biol. **46**, 591 (1984).

[18] W. Fontana, D. A. M. Konings, P. F. Stadler, and P. Schuster, Statistics of RNA secondary structures, Biopolymers **33**, 1389 (1993).

[19] D. De Juan, F. Pazos, and A. Valencia, Emerging methods in protein co-evolution, Nat. Rev. Genet. **14**, 249 (2013).

[20] Y. Roudi, J. Tyrcha, and J. Hertz, Ising model for neural data: Model quality and approximate methods for extracting functional connectivity, Phys. Rev. E **79**, 051915 (2009).

[21] V. Sessak and R. Monasson, Small-correlation expansions for the inverse Ising problem, J. Phys. A: Math. Theor. **42**, 055001 (2009).

[22] A. Decelle and F. Ricci-Tersenghi, Pseudolikelihood Decimation Algorithm Improving the Inference of the Interaction Network in a General Class of Ising Models, Phys. Rev. Lett. **112**, 070603 (2014).

[23] H. C. Nguyen, R. Zecchina, and J. Berg, Inverse statistical problems: From the inverse Ising problem to data science, Adv. Phys. **66**, 197 (2017).

[24] M. Weigt, R. A. White, H. Szurmant, J. A. Hoch, and T. Hwa, Identification of direct residue contacts in protein-protein interaction by message passing, Proc. Natl. Acad. Sci. USA **106**, 67 (2009).

[25] F. Morcos, A. Pagnani, B. Lunt, A. Bertolino, D. S. Marks, C. Sander, R. Zecchina, J. N. Onuchic, T. Hwa, and M. Weigt, Direct-coupling analysis of residue coevolution captures native contacts across many protein families, Proc. Natl. Acad. Sci. USA **108**, E1293 (2011).

[26] M. Ekeberg, C. Lövkvist, Y. Lan, M. Weigt, and E. Aurell, Improved contact prediction in proteins: Using pseudolikelihoods to infer Potts models, Phys. Rev. E **87**, 012707 (2013).

[27] S. Cocco, C. Feinauer, M. Figliuzzi, R. Monasson, and M. Weigt, Inverse statistical physics of protein sequences: A key issues review, Rep. Prog. Phys. **81**, 032601 (2018).

[28] D. S. Marks, L. J. Colwell, R. Sheridan, T. A. Hopf, A. Pagnani, R. Zecchina, and C. Sander, Protein 3d structure computed from evolutionary sequence variation, PLoS One **6**, e28766 (2011).

[29] J. I. Sulkowska, F. Morcos, M. Weigt, T. Hwa, and J. N. Onuchic, Genomics-aided structure prediction, Proc. Natl. Acad. Sci. USA **109**, 10340 (2012).

[30] T. A. Hopf, L. J. Colwell, R. Sheridan, B. Rost, C. Sander, and D. S. Marks, Three-dimensional structures of membrane proteins from genomic sequencing, Cell **149**, 1607 (2012).

[31] S. Ovchinnikov, H. Park, N. Varghese, P.-S. Huang, G. A. Pavlopoulos, D. E. Kim, H. Kamisetty, N. C. Kyrpides, and D. Baker, Protein structure determination using metagenome sequence data, Science **355**, 294 (2017).

[32] A. Procaccini, B. Lunt, H. Szurmant, T. Hwa, and M. Weigt, Dissecting the specificity of protein-protein interaction in bacterial two-component signaling: Orphans and crosstalks, PloS One **6**, e19729 (2011).

[33] C. Baldassi, M. Zamparo, C. Feinauer, A. Procaccini, R. Zecchina, M. Weigt, and A. Pagnani, Fast and accurate multivariate Gaussian modeling of protein families: Predicting residue contacts and protein-interaction partners, PLoS One **9**, e92721 (2014).

[34] C. Feinauer, H. Szurmant, M. Weigt, and A. Pagnani, Interprotein sequence co-evolution predicts known physical interactions in bacterial ribosomes and the TRP operon, PLoS One **11**, e0149166 (2016).

[35] A.-F. Bitbol, R. S. Dwyer, L. J. Colwell, and N. S. Wingreen, Inferring interaction partners from protein sequences, Proc. Natl. Acad. Sci. USA **113**, 12180 (2016).

[36] T. Gueudré, C. Baldassi, M. Zamparo, M. Weigt, and A. Pagnani, Simultaneous identification of specifically interacting paralogs and interprotein contacts by direct coupling analysis, Proc. Natl. Acad. Sci. USA **113**, 12186 (2016).

[37] Q. Cong, I. Anishchenko, S. Ovchinnikov, and D. Baker, Protein interaction networks revealed by proteome coevolution, Science **365**, 185 (2019).

[38] G. Croce, T. Gueudré, M. V. R. Cuevas, V. Keidel, M. Figliuzzi, H. Szurmant, and M. Weigt, A multi-scale coevolutionary approach to predict interactions between protein domains, PLoS Comput. Biol. **15**, e1006891 (2019).

[39] R. S. Dwyer, D. P. Ricci, L. J. Colwell, T. J. Silhavy, and N. S. Wingreen, Predicting functionally informative mutations in escherichia coli bama using evolutionary covariance analysis, Genetics **195**, 443 (2013).

[40] R. R. Cheng, F. Morcos, H. Levine, and J. N. Onuchic, Toward rationally redesigning bacterial two-component signaling systems using coevolutionary information, Proc. Natl. Acad. Sci. USA **111**, E563 (2014).

[41] M. Figliuzzi, H. Jacquier, A. Schug, O. Tenaillon, and M. Weigt, Coevolutionary landscape inference and the context-dependence of mutations in beta-lactamase TEM-1, Mol. Biol. Evol. **33**, 268 (2015).

[42] R. R. Cheng, O. Nordesjö, R. L. Hayes, H. Levine, S. C. Flores, J. N. Onuchic, and F. Morcos, Connecting the sequence space of bacterial signaling proteins to phenotypes using coevolutionary landscapes, Mol. Biol. Evol. **33**, 3054 (2016).

[43] T. A. Hopf, J. B. Ingraham, F. J. Poelwijk, C. P. Schärfe, M. Springer, C. Sander, and D. S. Marks, Mutation effects predicted from sequence co-variation, Nat. Biotechnol. **35**, 128 (2017).

[44] J. M. Reimer, M. Eivaskhani, I. Harb, A. Guarné, M. Weigt, and T. M. Schmeing, Structures of a dimodular nonribosomal peptide synthetase reveal conformational flexibility, Science **366**, eaaw4388 (2019).

[45] W. P. Russ, M. Figliuzzi, C. Stocker, P. Barrat-Charlaix, M. Socolich, P. Kast, D. Hilvert, R. Monasson, S. Cocco, M. Weigt, and R. Ranganathan, An evolution-based model for designing chorismate mutase enzymes, Science **369**, 440 (2020).

[46] M. Mézard and A. Montanari, *Information, Physics, and Computation* (Oxford University Press, Oxford, UK, 2009).

[47] A. Decelle, F. Krzakala, C. Moore, and L. Zdeborová, Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications, Phys. Rev. E **84**, 066106 (2011).

[48] G. W. Wilburn and S. R. Eddy, Remote homology search with hidden Potts models, bioRxiv (2020), doi:10.1101/2020.06.23.168153.

[49] H. Talibart and F. Coste, Compotts: Optimal alignment of co-evolutionary models for protein sequences, bioRxiv (2020), doi:10.1101/2020.06.12.147702.

[50] Please see the Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevE.102.062409 for more details on the mathematical derivation of the approximation scheme and for the additional results.

[51] M. Figliuzzi, P. Barrat-Charlaix, and M. Weigt, How pairwise coevolutionary models capture the collective residue variability in proteins?, Mol. Biol. Evol. **35**, 1018 (2018).

[52] D. J. Thouless, P. W. Anderson, and R. G. Palmer, Solution of "Solvable model of a spin glass," Philos. Mag. **35**, 593 (1977).

[53] M. Ekeberg, T. Hartonen, and E. Aurell, Fast pseudolikelihood maximization for direct-coupling analysis of protein structure from many homologous amino-acid sequences, J. Comput. Phys. **276**, 341 (2014).

[54] R. D. Finn, J. Clements, and S. R. Eddy, HMMER web server: Interactive sequence similarity searching, Nucleic Acids Res. **39**, W29 (2011).

[55] Pfam domain-domain interaction benchmark [https://github.com/infernet-h2020/pfam_interactions], doi: 10.5281/zenodo.4080947.

[56] F. Krzakala and L. Zdeborová, Potts glass on random graphs, EPL **81**, 57005 (2008).

[57] Pfam database, https://pfam.xfam.org/.

[58] Rfam database, http://rfam.xfam.org/.

[59] E. De Leonardis, B. Lutz, S. Ratz, S. Cocco, R. Monasson, A. Schug, and M. Weigt, Direct-coupling analysis of nucleotide coevolution facilitates rna secondary and tertiary structure prediction, Nucl. Acids Res. **43**, 10444 (2015).

[60] DCAlign: Aligning biological sequences using Direct Coupling Analysis models, https://github.com/infernet-h2020/DCAlign.