Chapter 2
Method

Abstract This chapter describes a novel MRF-based method for homology detection and fold recognition. In particular, it covers how to build an MRF model for a protein sequence, how to score the similarity of two MRF models and the similarity between one MRF model and one native structure, and finally an alternating direction method of multipliers (ADMM) method that can optimize the scoring function.

Keywords Markov random fields (MRF) · Hidden Markov models (HMM) · Alternating direction method of multipliers (ADMM) · Mutual information · Residue co-evolution

2.1 Modeling a Protein Family Using Markov Random Fields

Given a protein sequence, we run PSI-BLAST [1] with 5 iterations and E-value cutoff 0.001 to find its sequence homologs and then build their multiple sequence alignment (MSA). We can use a multivariate random variable $X = (X_1, X_2, \ldots, X_N)$, where N is the number of columns (or the MSA length), to model the MSA. Here each X_i is a finite discrete random variable representing the amino acid at column i in the MSA, taking values from 1 to 21, corresponding to 20 amino acids and gap. The occurring probability of the whole MSA can be modeled by an Markov Random Field (MRF), which is a function of X. MRF is an undirected graph that can be used to model a set of correlated random variables. As shown in Fig. 2.1, an MRF node represents one column in the MSA and an edge represents the correlation between two columns. Here we ignore very short-range residue correlation since it is not very informative. An MRF consists of two types of functions: $\phi(X_i)$ and $\psi(X_i, X_k)$, where $\phi(X_i)$ is an amino acid preference function for node i and $\psi(X_i, X_k)$ is a pairwise amino acid preference function for edge (i, k)

J. Xu et al., *Protein Homology Detection Through Alignment of Markov Random Fields*, SpringerBriefs in Computer Science,
DOI 10.1007/978-3-319-14914-1_2

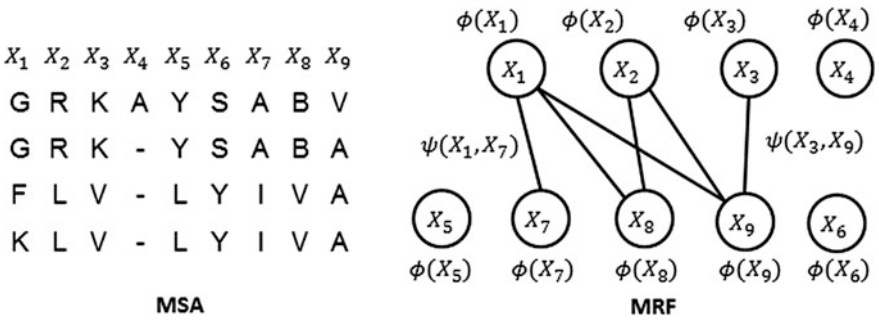

Fig. 2.1 Modeling a multiple sequence alignment (*left*) by Markov random fields (*right*)

that reflects interaction between two nodes. Then, the probability of observing a particular protein sequence X can be calculated as follows.

$$P(X) = \frac{1}{Z}\Pi_i \phi(X_i)\Pi_{(i,k)}\psi(X_i, X_k), \tag{2.1}$$

where Z is the normalization factor (i.e., partition function). If P(X) is Gaussian, this MRF is called Gaussian Graphical Model (GGM). In this case, X_i shall be interpreted as a 21-dimension binary vector, in which each element corresponds to one amino acid type or gap.

We use two kinds of information in MRFs for their alignment. One is the occurring probability of 20 amino acids and gap at each node (i.e., each column in MSA), which can also be interpreted as the marginal probability at each node. The other is the correlation between two nodes, which can be interpreted as interaction strength of two MSA columns.

2.2 Estimating the Parameters of Markov Random Fields

In this section, we describe how to estimate the parameters in the ψ function of the MRF. As stated above, ψ is a pairwise amino acid preference function for edge (i, k) reflecting interactions between two nodes (residues). A couple of strategies can be used to estimate residue interaction strength. One is to estimate residue co-evolution strength from the MSA and then use this co-evolution strength as residue interaction strength. Mutual information (MI) is a simple and popular way to estimate residue co-evolution strength. However, mutual information (MI) is only a local statistics (i.e., the MI of 2 positions is calculated independent of the other positions), so MI is not very accurate in estimating residue co-evolution strength. Instead of using MI, we can use direct information (DI), which is a global statistics (i.e., measuring the residue co-evolution strength of two positions considering other positions). DI can be calculated by some contact prediction programs

such PSICOV, Evfold, plmDCA [2–4] as residue co-evolution. PSICOV assumes that $P(X)$ is a Gaussian distribution function and calculates the correlation between two columns by inverse covariance matrix. By contrast, plmDCA does not assume a Gaussian distribution and is more efficient and also slightly more accurate. Generally speaking, these programs are time-consuming.

The reliability of mutual information (MI) or direct information (DI) [2] depends on the number of non-redundant sequence homologs. When there are few sequence homologs, the resulting MI or DI is not very accurate. Therefore, it is not enough to only use residue co-evolution strength to estimate residue interaction strength. We can use other contact prediction programs such as PhyCMAP [4] which integrates both residue col-evolution information, PSI-BLAST sequence profile and others to predict the probability of two residues in contact. PhyCMAP works much better than PSICOV and Evfold when proteins under study have a small number of sequence homologs [4].

In this work, we use predicted inter-residue Euclidean distance to reflect interaction strength of two residues. This is based upon an assumption that two spatially-close residues tend to have strong interaction. We predict the inter-residue distance using sequence information such as mutual information and its power series, PSI-BLAST sequence profile and other protein features. See [5] for more details. Below we briefly describe how to predict inter-residue distance from sequence information using probabilistic neural networks (PNN).

We discretize $C_\alpha - C_\alpha$ distance into 13 bins (3–4, 4–5, 5–6, …, 14–15, and >15 Å). Each bin is also called a label. Given a protein and a pair of two residues i and j, let d_k denote the bin into which their distance falls, and x_k denote the protein feature vector consisting of some position-specific sequence profile information and also mutual information between two positions. We would like to estimate the probability of observing d_k given the feature vector x_k. That is, instead of only considering the most possible distance labels assigned to each pair of nodes (residues), we would like to estimate the probability distribution of d_k. The reason is that the predicted distance probability distribution is more informative than a single predicted value.

Formally, let $p_\theta(d_k|x_k)$ be the probability of the distance label d_k conditioned on the feature vector x_k Meanwhile, θ is the model parameter vector. We estimate $p_\theta(d_k|x_k)$ as follows:

$$p_\theta(d_k|x_k) = \frac{\exp(L_\theta(d_k, x_k))}{Z_\theta(x_k)} \tag{2.2}$$

where $Z_\theta(x_k) = \sum_d \exp(L_\theta(d, x_k))$, is the partition function and $L_\theta(d, x_k)$ is a two-layer neural network. Figure 2.2 shows an example of the neural network with three and five neurons in the first and second hidden layers, respectively. Each neuron is a sigmoid function. The function $L_\theta(d_k, x_k)$ can be calculated as,

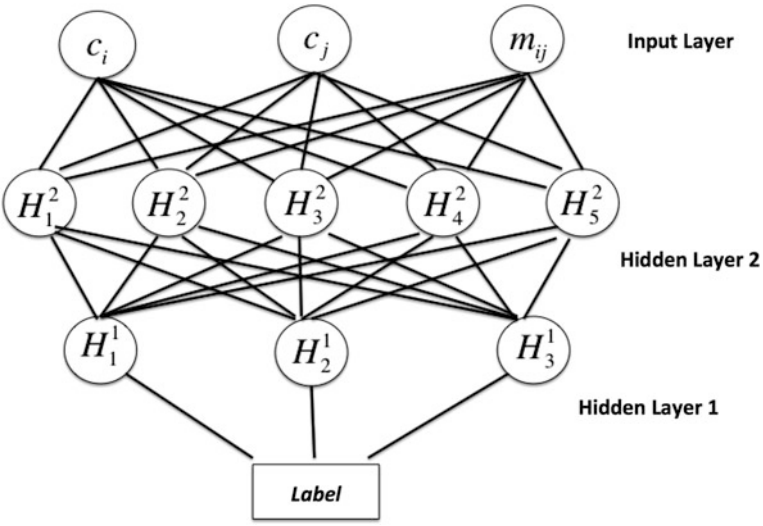

Fig. 2.2 Illustration of probabilistic neural networks (PNN), in which c_i and c_j are the protein sequence features centered at the *i*th and *j*th residues, respectively, and m_{ij} (mutual information) is the feature that must be fetched at the *i*th and *j*th residues simultaneously

$$L_\theta(d_k, x_k) = \sum_{g_1=1}^{G_1} \theta^0_{d_k,g_1} h\left(\sum_{g_1,g_2}^{G_2} \theta^1_{g_1,g_2} h(<\theta^2_{g_2}, x_k>)\right) \tag{2.3}$$

where G_1 and G_2 are the number of gates in the two hidden layers, $<\cdot,\cdot>$ denotes the inner product of two vectors, $\theta^2_{g_2}$ is the weight factor of the g_2th neuron in the second layer; $\theta^1_{g_1,g_2}$ is the weight connecting the first layer and the second layer. $\theta^0_{d_k,g_1}$ is the weight connecting the first layer and the labels.

In current implementation, our neural network contains two hidden layers. The first hidden layer (i.e., the layer connecting to the input layer) contains 100 neurons, and the second hidden layer (i.e., the layer connecting to the output layer) has 40 neurons. This neural network is similar to what is used by the Zhou group [6] for inter-residue contact prediction, which uses 100 and 30 neurons in the two hidden layers, respectively. The Zhou group has shown that using two hidden layers can obtain slightly better performance than using a single hidden layer. The input layer of our network has about 600 features, so in total, our neural network has between 60,000 and 70,000 parameters to be trained. We use the maximum likelihood method to train the model parameter θ and to determine the number of neurons in each hidden layer by maximizing the occurring probability of native $C_\alpha - C_\alpha$ distance in a set of training proteins. It is challenging to maximize the objective function Eq. (2.2) since it is non-convex and a large amount of training data is used. We use a limited-memory BFGS method [7] to fulfill this. We generated an initial

solution randomly and then ran the training algorithm on a supercomputer for about a couple of weeks. Our training algorithm terminated when the probability of either the training set or the validation set did not improve any more. Note that all the model parameters are learned from the training set but not the validation set. The validation set, combined with the training set, is only used to determine when our training algorithm shall terminate. Our training algorithm usually terminates after 3,000 iterations. We also reran our training algorithm starting from nine initial solutions and did not observe explicit performance difference among these runs. See our work on EPAD [5] for more details.

We use two kinds of input features in this neural network model: PSI-BLAST sequence profile and residue co-evolution. One is context-specific sequence profile for a small sequence segment centered at one specific residue in question. The sequence profile is generated by running PSI-BLAST on the NR database with 5 iterations and an E-value of 0.001. The other feature we used is residue co-evolution information. Mutual information is a classical method to measure residue co-evolution strength. However, mutual information cannot differentiate direct from indirect interactions. For example, when residue a has strong interaction with b and b has strong interaction with residue c, it is likely that residue a also has interaction with c. In order to reduce the impact of this kind of indirect information, some global statistical methods such as Graphical Lasso [3] and Pseudo-likelihood [8, 9] methods are proposed to estimate residue co-evolution strength. However, these methods are time-consuming. In this work, to account for chaining effect of residue coupling, we use the powers of the mutual information matrix. In particular, let MI denotes the mutual information matrix, we use MI^k where k ranges from 2 to 11 to estimate the chaining effect.

2.3 Scoring Similarity of Two Markov Random Fields

This section will introduce how to align two proteins by aligning their corresponding MRFs. As shown in the left picture of Fig. 2.3, building an alignment is equivalent to finding a unique path from the left-top corner to the right-bottom corner. For each vertex along the path, we need a score to measure how good it is to transit to the next vertex. That is, we need to measure how similar two nodes of the two MRFs are. We call this kind of scoring function node alignment potential. Second, in addition to measure the similarity two aligned MRF nodes, we want to quantify the similarity between two MRF edges. For example, in the right picture of Fig. 2.3 residues “L” and “S” of the first protein are aligned to residues “A” and “Q” of the 2nd protein, respectively. We would like to estimate how good it is to align the pair (L, S) to the pair (A, Q). This pairwise similarity function is a function of two MRF edges and we call it edge alignment potential. When the edge alignment potential is used to score the similarity of two MRFs, Viterbi algorithm or simple dynamic programming cannot be used to find the optimal alignment. It can

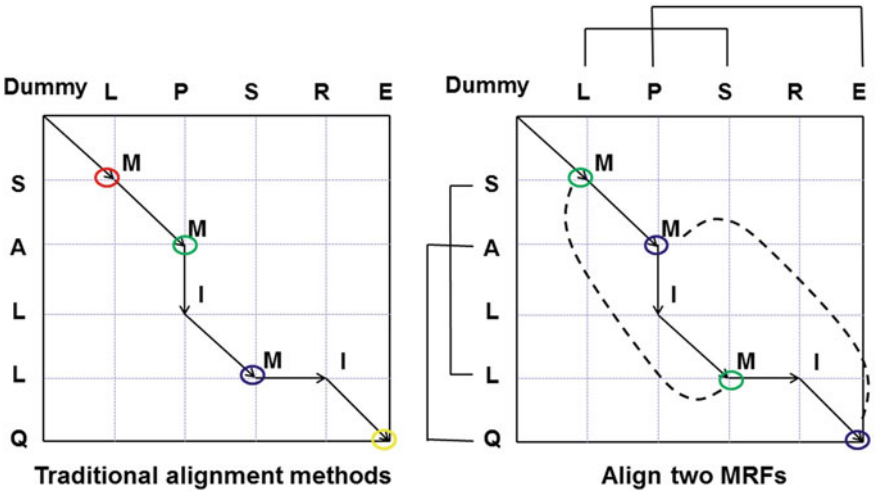

Fig. 2.3 Traditional alignment methods (*left*) and our MRFalign method (*right*)

be proved that when edge alignment potential is considered and gaps are allowed the MRF-MRF alignment problem is NP hard [10]. In this work, we will describe an ADMM [11] algorithm to quickly find a suboptimal alignment of two MRFs. Although suboptimal, empirically the resultant alignment has very good accuracy.

2.4 Node Alignment Potential of Markov Random Fields

Given an alignment path, its node alignment potential is the accumulative potential of all the vertices along the path. In particular, the node alignment potential is a function of two MRF nodes and measures the similarity of two aligned positions using local protein features. We use a Conditional Neural Fields (CNF) [12] method to develop such a node alignment potential, using a procedure very similar to what is described in the protein threading paper [13, 14]. Briefly speaking, we estimate the occurring probability of an alignment A between two proteins T and S as follows.

$$P(A|T,S) = \mathrm{e}^{\sum_{(i,j,u)\in A} E_u\left(T_i,S_j\right)}/Z(T,S) \tag{2.4}$$

where $Z(T,S)$ is a normalization factor summarizing all the possible alignments between T and S, and $E_u\left(T_i,S_j\right)$ is a neural network with one hidden layer that calculates the log-likelihood of a vertex (i,j,u) in the alignment path, where i is a node in T, j a node in S, and u a state. When u is a match state, E_u takes as input the sequence profile context of two nodes i and j, denoted as $\mathrm{T_i}$ and $\mathrm{S_j}$, respectively,

and yields the log-likelihood of these two nodes being matched. When u is an insertion state, it takes as input at the sequence profile context of one node and yields the log-likelihood of this node being an insertion. The sequence profile context of node i is a $21 \times (2w+1)$ matrix where $w = 5$, consisting of the marginal probability of 20 amino acids and gap at $2w+1$ nodes indexed by $i-w$, $i-w+1, \ldots, i, i+1, \ldots, i+w$. In case that one column does not exist (when $i \leq w$ or $i+w > N$), zero is used. See Fig. 2.4 for an illustration of how to use neural networks to estimate local similarity of two MRF nodes.

We train the parameters in E_u by maximizing the occurring probability of a set of reference alignments, which are generated by a structure alignment tool DeepAlign [15]. That is, we optimize the model parameters so that the structure alignment of one training protein pair has the largest probability among all possible alignments. Notice that by using neural networks the objective function of Eq. (2.4) is neither concave nor log-concave, so it is challenging to find globally optimal solution. Here we use the Limited memory BFGS (L-BFGS) algorithm to solve it to suboptimal. To obtain a good solution, we run L-BFGS several times starting from different initial solutions and return the best suboptimal solution. A L_2-norm regularization factor, which is determined by fivefold cross validation, is used to restrict the search space of model parameters to avoid over-fitting.

Let $\theta^u_{i,j}$ denote the local alignment potential of a vertex (i,j,u) in the alignment path. We calculate $\theta^u_{i,j}$ from E_u as follows.

$$\theta^u_{i,j} = E_u\left(T_i, S_j\right) - \mathrm{Exp}(E_u) \tag{2.5}$$

where $\mathrm{Exp}(E_u)$ is the expected value of E_u, which depends only on the alignment state but not any specific protein pair. It is used to offset the effect of the background, which is the log-likelihood yielded by E_u for any randomly chosen node

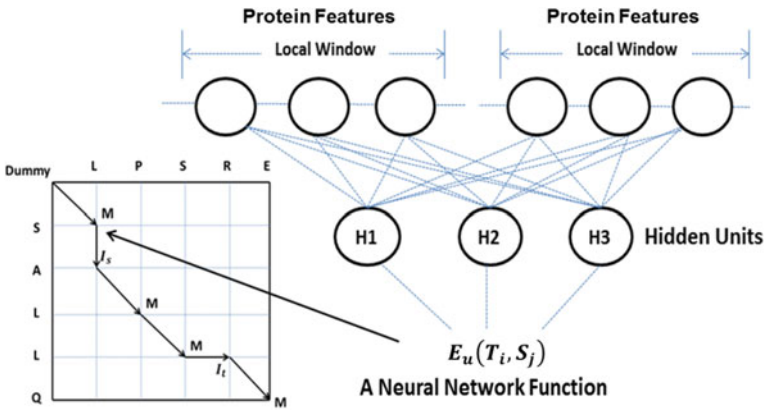

Fig. 2.4 An example neural network for calculating node alignment potential, in which there is one hidden layer. The function takes features from both proteins as input and yields one log-likelihood score

pairs (or nodes). We can calculate the reference alignment probability E_u in Eq. (2.5) by randomly sampling a set of protein pairs, each with the same lengths as the sequence S and template T, respectively, and then estimating the probability of alignment A based upon these randomly sampled protein pairs. As long as we generate sufficient number of samplings, we shall be able to approximate $\text{Exp}(E_u)$ very well.

To perform protein homology detection, only sequence and profile based features are included. For the sequence similarity features, we use BLOSUM 80 and BLOSUM 62 matrix [16], structure-derived amino acid substitution matrix [17], and statistical potential derived amino acid similarity matrix [18]. We also use sequence-profile similarity score and profile-profile similarity score as described in section Introduction. We do not use an affine gap penalty, but a context-specific gap penalty which is parameterized by amino acid type, predicted secondary structure type and position-specific profile information.

2.5 Edge Alignment Potential of Markov Random Fields

The edge alignment potential calculates the similarity of two edges, one from each MRF, based upon the interaction strength of two ends in one edge. As mentioned above, we use predicted distance probability distribution based on the features of two nodes to estimate their interaction strength. Let d_{ik}^T be a random variable for the Euclidean distance between two residues at i and k and d_{jl}^S is defined similarly. Let $\theta_{i,k,j,l}$ denote the alignment potential between edge (i,k) in T and edge (j,l) in S. As shown in Fig. 2.5 we calculate $\theta_{i,k,j,l}$ as follows.

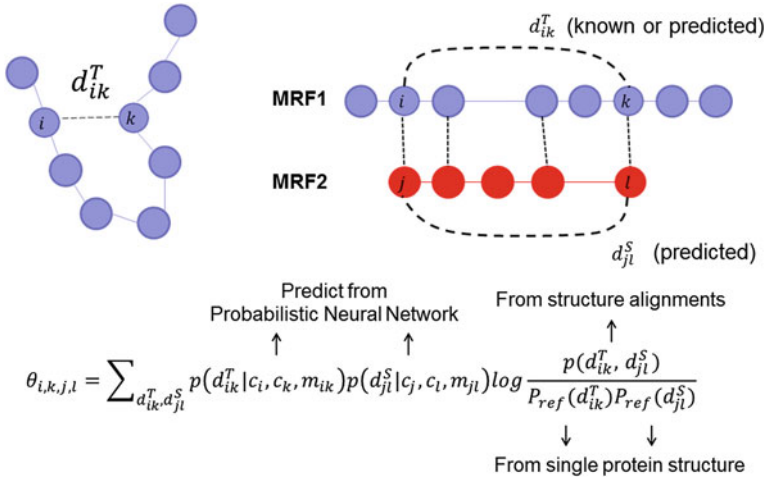

Fig. 2.5 Illustration of edge alignment potential for MRF-MRF alignment

$$\theta_{i,k,j,l} = \sum_{d_{ik}^T, d_{jl}^S} p\left(d_{ik}^T | c_i, c_k, m_{ik}\right) p\left(d_{jl}^S | c_j, c_l, m_{jl}\right) \log \frac{p(d_{ik}^T, d_{jl}^S)}{P_{ref}(d_{ik}^T) P_{ref}(d_{jl}^S)} \tag{2.6}$$

where $p\left(d_{ik}^T | c_i, c_k, m_{ik}\right)$ is the probability of two nodes i and k in T interacting at distance d_{ik}^T; $p\left(d_{jl}^S | c_j, c_l, m_{jl}\right)$ is the probability of two nodes j and l in S interacting at distance d_{jl}^S; $p\left(d_{ik}^T, d_{jl}^S\right)$ is the probability of one distance d_{ik}^T being aligned to another distance d_{jl}^S in reference alignments; and $P_{ref}\left(d_{ik}^T\right)$ $\left(P_{ref}\left(d_{jl}^S\right)\right)$ is the background probability of observing d_{ik}^T (d_{jl}^S) in a protein structure. Meanwhile x_i and x_k are position-specific features centered at the ith and kth residues, respectively, and m_{ik} represents the mutual information between the ith and kth columns in the multiple sequence alignment.

Compared to contact-based potentials, here we use interaction at a given distance to obtain a higher-resolution description of the residue interaction pattern, as shown in Fig. 2.5. Therefore, this edge alignment potential is more informative and thus, may lead to better alignment accuracy and homology detection rate.

Now we explain how to calculate each term in Eq. (2.6). $P_{ref}\left(d_{ik}^T\right)$ $\left(P_{ref}\left(d_{jl}^S\right)\right)$ can be calculated by simple counting on a set of non-redundant protein structures, e.g., PDB25. Similar to $P_{ref}\left(d_{ik}^T\right)$, $P\left(d_{ik}^T, d_{jl}^S\right)$ can also be calculated by simple counting on a set of non-redundant reference alignments. That is, we randomly choose a set of protein pairs such that two proteins in each pair are similar at least at the fold level. Then we generate their reference alignment (i.e., structure alignments) using a structure alignment tool DeepAlign [15] and finally do simple counting to estimate $p\left(d_{ik}^T, d_{jl}^S\right)$. In order to use simple counting, we discretize inter-residue distance into 12 intervals: <4, 4–5, 5–6, …, 14–15, and >15 Å.

As explained in the previous section, we predict $p\left(d_{ik}^T | c_i, c_k, m_{ik}\right)$ using a probabilistic neural network (PNN) implemented in our context-specific distance-dependent statistical potential package EPAD. EPAD takes as input sequence profile contexts and mutual information and then yields inter-residue distance probability distribution. See the EPAD paper [5] for the technical details. The EPAD package has been blindly tested in CASP10 for template free modeling. The CASP10 results show that EPAD can successfully fold some targets with unusual fold (according to the CASP10 Free Modeling assessor Dr. BK Lee). Our large-scale experimental test also indicates EPAD is much better than those context-independent distance-based pairwise potentials such as DOPE [19], RW [20] and DFIRE [21] in ranking protein decoys [5].

2.6 Scoring Similarity of One Markov Random Fields and One Template

This MRF-based alignment method can also be applied to protein threading. In this scenario, one of the two proteins under alignment has solved 3D structure. Of course we can just simply use the node and edge alignment potentials described in previous sections to align one MRF to one solved structure. In order to use the native structure information in the protein with solved 3D structure, we may revise the alignment potentials as follows.

1. Instead of using predicted secondary structure and solvent accessibility, we may use their native information for the protein with solved 3D structure, which can be generated by DSSP [22].
2. Let T denote the protein with solved 3D structure. We can directly calculate the inter-residue distance for any residue pairs in T. That is, $p\left(d_{ik}^T|c_i, c_k, m_{ik}\right)$ reduces to a simple distribution that has probability 1 for the native distance between residues i and k and 0 otherwise. So, the edge alignment potential can be simplified as follows.

$$\theta_{i,k,j,l} = P\left(d_{ij}^T|c_i, c_j\right) = \sum_{d_{ij}^S} P(d_{ij}^T|d_{ij}^S)P(d_{ij}^S|c_i, c_j) \tag{2.7}$$

where d_{ij}^S represents the distance of the two sequence residues at the two aligned positions, $P\left(d_{ij}^T|d_{ij}^S\right)$ is the conditional probability of d_{ij}^T on d_{ij}^S and $P\left(d_{ij}^S|c_i, c_j\right)$ is the conditional probability of d_{ij}^S estimated from the contexts (denoted x_i and x_j) of the two sequence residues. In Eq. (2.7), $P\left(d_{ij}^T|d_{ij}^S\right) = \frac{P(d_{ij}^T, d_{ij}^S)}{P(d_{ij}^S)}$ where $P\left(d_{ij}^S\right)$ $\left(= P_{ref}\left(d_{ij}^T\right)\right)$ is the background probability, and $P\left(d_{ij}^T, d_{ij}^S\right)$ is the joint probability of the pairwise distances of two aligned residue pairs and can be calculated by simple statistics using a set of non-redundant protein structure alignments generated by a structure alignment tool such as DeepAlign.

2.7 Algorithms for Aligning Two Markov Random Fields

As mentioned before, an alignment can be represented as a path in the alignment matrix, which encodes an exponential number of paths. We can use a set of $3N_1N_2$ binary variables $z_{i,j}^u$ to define a path, where N_1 and N_2 are the lengths of the two MSAs, (i, j) is an entry in the alignment matrix and u the associated state. Meanwhile, $z_{i,j}^u$ is equal to 1 if the alignment path passes (i, j) with state u. Therefore, the

problem of finding the best alignment between two MRFs can be formulated as the following quadratic optimization problem.

$$\text{(P1)} \quad \max_z \sum_{i,j,u} \theta^u_{i,j} z^u_{i,j} + \frac{1}{L} \sum_{i,j,k,l,u,v} \theta^{uv}_{i,j,k,l} z^u_{i,j} z^v_{k,l} \tag{2.8}$$

where $\theta^u_{i,j}$ and $\theta^{uv}_{i,j,k,l}$ are node and edge alignment potentials as described in previous section. Meanwhile, $\theta^{uv}_{i,j,k,l}$ is equal to 0 if either u or v is not a match state. L is the alignment length and $1/L$ is used to make the accumulative node and edge potential have similar scale. Note that L is unknown and we will describe how to determine it later in this section. Finally, the solution of P1 is also subject to the constraint that all those $z^u_{i,j}$ with value 1 shall form a valid alignment path. This constraint shall be enforced to all the optimization problems described in this section.

It is computationally intractable to find the optimal solution of P1. Below we present an Alternating Direction Method of Multipliers (ADMM) method that can efficiently solve this problem to suboptimal. See [11] for a tutorial of the ADMM method. To use ADMM, we rewrite P1 as follows by making a copy of Z to y, but without changing the solution space.

$$\begin{aligned} \text{(P2)} \quad \max_{z,y} & \sum_{i,j,u} \theta^u_{i,j} z^u_{i,j} + \tfrac{1}{L} \sum_{i,j,k,l,u,v} \theta^{uv}_{i,j,k,l} z^u_{i,j} y^v_{k,l} \\ & s.t.\ \forall k,l,v, \quad z^v_{k,l} = y^v_{k,l} \end{aligned} \tag{2.9}$$

Problem P2 can be augmented by adding a term to penalize the difference between z and y.

$$\begin{aligned} \text{(P3)} \quad \max_{z,y} & \sum_{i,j,u} \theta^u_{i,j} z^u_{i,j} + \tfrac{1}{L} \sum_{i,j,k,l,u,v} \theta^{uv}_{i,j,k,l} z^u_{i,j} y^v_{k,l} - \tfrac{\rho}{2} \sum_{i,j,u} \left(z^u_{i,j} - y^u_{i,j} \right)^2 \\ & s.t.\ \forall i,j,u, \quad z^u_{i,j} = y^u_{i,j} \end{aligned} \tag{2.10}$$

P3 is equivalent to P2 and P1, but converges faster due to the penalty term. Here ρ is a hyper-parameter influencing the convergence rate of the algorithm. Empirically, setting ρ to a constant (=0.5) enables our algorithm to converge within 10 iterations for most protein pairs.

Adding the constraint $z^u_{i,j} = y^u_{i,j}$ using a Lagrange multiplier λ to Eq. (2.10), we have the following Lagrangian dual problem:

$$\begin{aligned} \text{(P4)} \quad \min_\lambda \max_{z,y} & \sum_{i,j,u} \theta^u_{i,j} z^u_{i,j} + \frac{1}{L} \sum_{i,j,k,l,u,v} \theta^{uv}_{i,j,k,l} z^u_{i,j} y^v_{k,l} \\ & + \sum_{i,j,u} \lambda^u_{i,j} \left(z^u_{i,j} - y^u_{i,j} \right) - \frac{\rho}{2} \sum_{i,j,u} \left(z^u_{i,j} - y^u_{i,j} \right)^2 \end{aligned} \tag{2.11}$$

It is easy to prove that P3 is upper bounded by P4. Now we will solve P4 and use its solution to approximate P3 and thus, P1.

Since both z and y are binary variables, the last term in Eq. (2.11) can be expanded as follows.

$$\frac{\rho}{2}\sum_{i,j,u}\left(z_{i,j}^{u}-y_{i,j}^{u}\right)^{2}=\frac{\rho}{2}\sum_{i,j,u}\left(z_{i,j}^{u}+y_{i,j}^{u}-2z_{i,j}^{u}y_{i,j}^{u}\right) \tag{2.12}$$

For a fixed λ, we can split P4 into the following two sub-problems.

$$\text{(SP1)}\quad y^{*}=\operatorname{argmax}\sum_{k,l,v}y_{k,l}^{v}C_{k,l}^{v} \tag{2.13}$$

where $C_{k,l}^{v}=\frac{1}{L}\sum_{i,j,u}\theta_{i,j,k,l}^{uv}z_{i,j}^{u}-\lambda_{k,l}^{v}-\frac{\rho}{2}\left(1-2z_{k,l}^{v}\right)$

$$\text{(SP2)}\ z^{*}=\operatorname{argmax}\sum_{i,j,u}z_{i,j}^{u}D_{i,j}^{u} \tag{2.14}$$

where $D_{i,j}^{u}=\theta_{i,j}^{u}+\sum_{k,l,v}\frac{1}{L}\theta_{i,j,k,l}^{uv}y_{k,l}^{v*}+\lambda_{i,j}^{u}-\frac{\rho}{2}(1-y_{i,j}^{u*})$

The sub-problem SP1 optimizes the objective function with respect to y while fixing z, and the sub-problem SP2 optimizes the objective function with respect to z while fixing y. SP1 and SP2 do not contain any quadratic term, so they can be efficiently solved using the classical dynamic programming algorithm for sequence or HMM-HMM alignment.

In summary, we solve P4 using the following procedure

Step 1 Initialize z by aligning the two MRFs without the edge alignment potential, which can be done by dynamic programming. Accordingly, initialize L as the length of the initial alignment.

Step 2 Solve (SP1) first and then (SP2) using dynamic programming, each generating a feasible alignment.

Step 3 If the algorithm converges, i.e., the difference between z and y is very small or zero, stop here. Otherwise, we update the alignment length L as the length of the alignment just generated and the Lagrange multiplier λ using subgradient descent as in Eq. (2.15), and then go back to Step 2.

$$\lambda^{n+1}=\lambda^{n}-\rho(z^{*}-y^{*}) \tag{2.15}$$

Due to the quadratic penalty term in Eq. (2.10), this ADMM algorithm usually converges much faster and also yields better solutions than without this term. Empiric ally, it converges within 10 iterations for most protein pairs. See [11] for the convergence proof of a general ADMM algorithm.

References

1. Altschul, S.F., et al.: Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. **25**(17), 3389–3402 (1997)
2. Marks, D.S., et al.: Protein 3D structure computed from evolutionary sequence variation. PLoS ONE **6**(12), e28766 (2011)
3. Jones, D.T., et al.: PSICOV: precise structural contact prediction using sparse inverse covariance estimation on large multiple sequence alignments. Bioinformatics **28**(2), 184–190 (2012)
4. Wang, Z., Xu, J.: Predicting protein contact map using evolutionary and physical constraints by integer programming. Bioinformatics **29**(13), i266–i273 (2013)
5. Zhao, F., Xu, J.: A position-specific distance-dependent statistical potential for protein structure and functional study. Structure **20**(6), 1118–1126 (2012)
6. Faraggi, E., Xue, B., Zhou, Y.: Improving the prediction accuracy of residue solvent accessibility and real-value backbone torsion angles of proteins by guided-learning through a two-layer neural network. Proteins Struct. Funct. Bioinf. **74**(4), 847–856 (2009)
7. Malouf, R.: A comparison of algorithms for maximum entropy parameter estimation. In: Proceedings of the 6th Conference on Natural Language Learning, vol. 20. Association for Computational Linguistics (2002)
8. Ekeberg, M., Hartonen, T., Aurell, E.: Fast pseudo likelihood maximization for direct-coupling analysis of protein structure from many homologous amino-acid sequences. arXiv preprint arXiv:1401.4832 (2014)
9. Kamisetty, H., Ovchinnikov, S., Baker, D.: Assessing the utility of coevolution-based residue–residue contact predictions in a sequence-and structure-rich era. Proc. Natl. Acad. Sci. **110** (39), 15674–15679 (2013)
10. Lathrop, R.H.: The protein threading problem with sequence amino acid interaction preferences is NP-complete. Protein Eng. **7**(9), 1059–1068 (1994)
11. Boyd, S., et al.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends® Mach. Learn. **3**(1), 1–122 (2011)
12. Peng, J., Bo, L., Xu, J.: Conditional neural fields. In: Advances in Neural Information Processing Systems (2009)
13. Ma, J., et al.: Protein threading using context-specific alignment potential. Bioinformatics **29** (13), i257–i265 (2013)
14. Ma, J., et al.: A conditional neural fields model for protein threading. Bioinformatics **28**(12), i59–i66 (2012)
15. Wang, S., et al.: Protein structure alignment beyond spatial proximity, vol. 3. Science Report (2013)
16. Henikoff, S., Henikoff, J.G.: Amino acid substitution matrices from protein blocks. Proc. Natl. Acad. Sci. **89**(22), 10915–10919 (1992)
17. Prlić, A., Domingues, F.S., Sippl, M.J.: Structure-derived substitution matrices for alignment of distantly related sequences. Protein Eng. **13**(8), 545–550 (2000)
18. Tan, Y.H., Huang, H., Kihara, D.: Statistical potential-based amino acid similarity matrices for aligning distantly related protein sequences. Proteins Struct. Funct. Bioinf. **64**(3), 587–600 (2006)
19. Shen, M.Y., Sali, A.: Statistical potential for assessment and prediction of protein structures. Protein Sci. **15**(11), 2507–2524 (2006)
20. Zhang, J., Zhang, Y.: A novel side-chain orientation dependent potential derived from random-walk reference state for protein fold selection and structure prediction. PLoS ONE **5** (10), e15386 (2010)

21. Zhou, H., Zhou, Y.: Distance-scaled, finite ideal-gas reference state improves structure-derived potentials of mean force for structure selection and stability prediction. Protein Sci. **11**(11), 2714–2726 (2002)
22. Kabsch, W., Sander, C.: Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. Biopolymers **22**(12), 2577–2637 (1983)

Springer

http://www.springer.com/978-3-319-14913-4

Protein Homology Detection Through Alignment of Markov Random Fields
Using MRFalign

Xu, J.; Wang, S.; Ma, J.

2015, VIII, 51 p. 13 illus., 1 illus. in color., Softcover
ISBN: 978-3-319-14913-4