CHAPTER FIVE

Algorithms, Applications, and Challenges of Protein Structure Alignment

Jianzhu Ma, Sheng Wang1
Toyota Technological Institute at Chicago, Chicago, Illinois, USA
1Corresponding author: e-mail address: wangsheng@ttic.edu

Contents

1. Introduction 122
2. Structural Alphabet 124
   2.1 Introduction of structural alphabet 124
   2.2 Different kinds of structural alphabet 126
   2.3 Conformational letters 128
   2.4 Applications of structural alphabet 128
   2.5 Substitution matrix for structural alphabet 130
3. Protein Pairwise Structure Alignment 130
   3.1 Introduction of protein pairwise structure alignment 130
   3.2 The scoring functions to measure the structure similarity 138
   3.3 The search algorithms for pairwise structure alignment 140
   3.4 DeepAlign: The approach beyond spatial proximity 144
   3.5 Result analysis 147
4. Protein MSA 155
   4.1 Introduction of protein MSA 155
   4.2 The current approaches for MSA 158
   4.3 3DCOMB: The scaffold-first approach for MSA 161
   4.4 Result analysis 165
5. Conclusions and Perspectives 167

References 170

Abstract

As a fundamental problem in computational structure biology, protein structure alignment has attracted the focus of the community for more than 20 years. While the pairwise structure alignment could be applied to measure the similarity between two proteins, which is a first step for homology search and fold space construction, the multiple structure alignment could be used to understand evolutionary conservation and divergence from a family of protein structures. Structure alignment is an NP-hard problem, which is only computationally tractable by using heuristics. Three
levels of heuristics for pairwise structure alignment have been proposed, from the representations of protein structure, the perspectives of viewing protein as a rigid-body or flexible, to the scoring functions as well as the search algorithms for the alignment. For multiple structure alignment, the fourth level of heuristics is applied on how to merge all input structures to a multiple structure alignment. In this review, we first present a small survey of current methods for protein pairwise and multiple alignment, focusing on those that are publicly available as web servers. In more detail, we also discuss the advancements on the development of the new approaches to increase the pairwise alignment accuracy, to efficiently and reliably merge input structures to the multiple structure alignment. Finally, besides broadening the spectrum of the applications of structure alignment for protein template-based prediction, we also list several open problems that need to be solved in the future, such as the large complex alignment and the fast database search.

1. INTRODUCTION

The comparison of protein structures is a very important problem in structural and evolutional biology for more than 20 years. Several excellent reviews could be found in Eidhammer, Jonassen, and Taylor (2000) and Koehl (2006). The applications of pairwise structure alignment include the prediction of the new protein’s function (Brylinski & Skolnick, 2008), the homology search against a known database of proteins (Holm & Sander, 1993), the organization and classification of known structures (Holm & Sander, 1994), and the discovery of new structure patterns and their correlation among sequences (Bradley, Kim, & Berger, 2002). The applications of multiple structure alignment (MSA) programs include understanding evolutionary conservation and divergence (Andersen et al., 2001), functional prediction through the identification of structurally conserved active sites in homologous proteins (Irving, Whisstock, & Lesk, 2001), construction of benchmark datasets on which to test multiple sequence alignment programs (Edgar & Batzoglou, 2006), and automatic construction of profiles and threading templates for protein structure prediction (Dunbrack, 2006; Panchenko, Marchler-Bauer, & Bryant, 1999).

Structure alignment is an NP-hard problem, which is only computationally tractable by using heuristics (Koehl, 2006). The first level of heuristics is how to represent the protein structure. Four major classes have been proposed. For example, DALI (Holm & Sander, 1993) represents the protein as a C-alpha-based distance map (C-map); VAST (Gibrat, Madej, Spouge, & Bryant, 1997) represents the protein as the secondary structure
elements (SSEs), PB-Align (Tyagi, De Brevern, Srinivasan, & Offmann, 2008) transforms the protein structure into 1D string (Profile), and most other methods use C-alpha to represent the protein. The second level of heuristics is the perspective to view the protein as a rigid-body or flexible, such as MATT (Menke, Berger, & Cowen, 2008). The third level is the method for the structure alignment, which has two major components: a scoring function to measure the protein similarity and a search algorithm to optimize the scoring function (Hasegawa & Holm, 2009). For example, as a widely used tool, TMalign (Zhang & Skolnick, 2005) employs the TMscore (Zhang & Skolnick, 2004) as the scoring function and applies dynamic programming (DP) as the search algorithm (Levitt & Gerstein, 1998) to optimize the score.

For MSA, besides all three levels, a fourth level of heuristics is applied to solve the problem that how to merge all input structures to a MSA. The current MSA merging method could be classified into two categories: horizontal-first and vertical-first approaches (Wang & Zheng, 2009). For instance, MUSTANG (Konagurthu, Whisstock, Stuckey, & Lesk, 2006) belongs to the horizontal-first method that it progressively merges pairwise alignments into an MSA along a guide tree; MultiProt (Shatsky, Nussinov, & Wolfson, 2004) belongs to the vertical-first method that it identifies some similar fragment blocks (SFBs) among all proteins and then extends these SFBs to an MSA. In addition to these two approaches, recently Ilinkin, Ye, and Janardan (2010) have developed a consensus-first approach, and Wang, Peng, and Xu (2011) and Wang and Zheng (2009) have proposed a scaffold-first approach. These new approaches could efficiently and reliably merge input structures to the MSA.

Although many computer programs have been developed, the alignment accuracy of the pairwise alignment programs is still low when judged by manually curated structure alignments, especially on distantly but functionally related proteins (Wang, Ma, Peng, & Xu, 2013). The reason is that almost all pairwise structure alignment methods only capture protein structure similarity using a spatial proximity of the 3D object (Hasegawa & Holm, 2009). However, it is observed that despite conformational changes, protein structure families exhibit a high local flexibility on a smaller scale called phenotypic plasticity (Csaba, Birzele, & Zimmer, 2008), which is grounded on evolutionary events (i.e., mutation, insertion, and deletion). Under the hypothesis that two protein structures are similar, we assume that they share locally similar substructures that are not necessarily restricted to SSEs. This kind of local conformation changes caused by the evolutionary events
cannot be accurately quantified by spatial proximity of aligned residues. Instead, evolutionary distance shall be a better measure (Wang et al., 2013).

In order to describe the local structures in proteins beyond the traditional SSEs, a variety of discretized sets, that is, the structural alphabets, have been proposed (see Table 5.1). To quantify the evolutionary distance between these local substructures, a substitution matrix for structural alphabets is constructed by a similar way of BLOSUM (Henikoff & Henikoff, 1992). Therefore, by considering the amino acid (AA) and local structure substitution matrices, a new scoring function for pairwise structure alignment has been developed (Wang et al., 2013). Experimental results show that optimizing this score could generate structure alignments much more consistent with manually curated alignments than other automatic tools especially when proteins under consideration are remote homologs.

In this review, we first briefly introduce the basic concept of structural alphabet and the related substitution matrix. Then, we describe protein pairwise structure alignment. The third part of this review focuses on protein MSA. We make a conclusion as well as propose several applications and address some unsolved problems in the last part.

2. STRUCTURAL ALPHABET

2.1. Introduction of structural alphabet

2.1.1 Why we need a structural alphabet

The protein local structures are classically described as three-state SSE that is composed of two regular states, the alpha-helices and the beta-strands, and one nonregular and variable state, the coil (Pauling, Corey, & Branson, 1951). Although the relevant physical meaning of SSE, the limitations are emerged, such as the uncertainties in the assignment of the boundaries around the helix and sheet regions, as well as about half of all residues are assigned to the coil state even though they have significantly different local conformation (Joseph, Agarwal, et al., 2010; Joseph, Bornot, & de Brevern, 2010). These limitations have stimulated a variety of research teams to focus on abstracting the protein backbone conformation in the localized short fragments, that is, the structural alphabets. Consisting of a discretized set of representatives of the local structures, the structural alphabets are generally defined by clustering local fragments in protein structures through a variety of geometric measures, as well as different segment length (Bystroff & Baker, 1998; Camproux et al., 1999; De Brevern et al., 2000; Kolodny et al., 2002; Pandini et al., 2010; Tung et al., 2007). We have developed a description of
<table>
<thead>
<tr>
<th>Name</th>
<th>Len</th>
<th>Size</th>
<th>Features</th>
<th>Available link</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protein blocks (PBs)</td>
<td>5</td>
<td>16</td>
<td>Unsupervised classifier (SOM)</td>
<td><a href="http://www.dsimb.inserm.fr/~debrevern/PBs/index.php">http://www.dsimb.inserm.fr/~debrevern/PBs/index.php</a></td>
<td>[De Brevern, Etchebest, and Hazout (2000)]</td>
</tr>
<tr>
<td>Conformational letter (CLE)</td>
<td>4</td>
<td>17</td>
<td>Gaussian mixture model</td>
<td><a href="http://ttic.uchicago.edu/~majianzhu/PDB_To_CLE_v1.00.tar.gz">http://ttic.uchicago.edu/~majianzhu/PDB_To_CLE_v1.00.tar.gz</a></td>
<td>[Wang and Zheng (2008)]</td>
</tr>
<tr>
<td>FragBag</td>
<td>9,10,11,12</td>
<td>50–600, 100–600, 100–600</td>
<td>Bag of words</td>
<td><a href="http://cs.haifa.ac.il/~ibudowsk/library9_12.html">http://cs.haifa.ac.il/~ibudowsk/library9_12.html</a></td>
<td>[Budowski-Tal, Nov, and Kolodny (2010)]</td>
</tr>
</tbody>
</table>
protein structural alphabet, namely, conformational letter (CLE) (Wang & Zheng, 2008; Zheng, 2008) using pseudo-bond angles of successive four C-alpha atoms. The discretized set of representatives is selected according to the density peaks of probability distribution in the phase space spanned by pseudo-bond angles. Since its development, numerous new research fields using the structural alphabet have been explored, such as fast protein database search (Budowski-Tal et al., 2010; Tung et al., 2007), protein fold recognition (De Brevern et al., 2000), binding-site signature detection (Dudev & Lim, 2007), and most importantly, protein structure alignment (Bornot, Etchebest, & De Brevern, 2009).

2.1.2 Why we need a substitution matrix for structural alphabet
Without a substitution matrix, the use of a structural alphabet is very limited. A substitution matrix called CLESUM (conformational letter substitution matrix) (Zheng & Liu, 2005) is constructed, similar as the way of BLOSUM (Henikoff & Henikoff, 1992), from a representative pairwise aligned structure dataset of the families of structurally similar proteins (FSSPs) (Holm & Sander, 1994). After a protein 3D backbone structure is converted to a 1D sequence of structure alphabet (SA) that is akin to the AA, many tools of sequence analysis can then be applied with certain minor modifications (Wang & Zheng, 2008). As an example of application, fast structure alignment that is based on the CLE is described in the next section. Briefly, structurally similar fragment pairs (SFPs) are found merely by string comparison and a greedy strategy guided by CLESUM similarity scores becomes possible.

2.2. Different kinds of structural alphabet
The principle of a structural alphabet is simple (see Fig. 5.1). A set of average local protein structures is first designed. They approximate every part of the structures. As one residue is associated to one of these prototypes, we can translate the 3D information of the protein structures as a series of prototypes in 1D, as the AA sequence (Zheng, 2008). To see a selected set of structural alphabet, please view Table 5.1. For a more complete review on structural alphabets, see Joseph, Bornot, et al. (2010). Generally, the procedure to deduce finite discrete conformational states from continuous states in a conformational phase space is a clustering analysis. There has been a variety of different ways of clustering using various descriptors for structure. The original descriptor in the structure database Protein Data Bank (PDB) is the Cartesian coordinates of atoms. Many studies to investigate the classification
of protein fragments use the backbone ($\phi, \psi$) dihedral angles, or angles of C-alpha pseudo-bonds or distances derived from the positions of C-alpha atoms (Flocco & Mowbray, 1995; Matsuda, Taniguchi, & Hashimoto, 1997; McCammon, 1977; Robson & Pain, 1974; Rooman, Kocher, & Wodak, 1991). Park and Levitt (1995) represented the polypeptide chain by a sequence of rigid fragments that were chosen from a library of representative fragments and concatenated without any degrees of freedom. De Brevern et al. (2000) proposed a SA called protein blocks (PBs) that are composed of 16 mean protein fragments of 5 residues in length by optimizing 2 goals simultaneously that (i) to obtain a good local structure approximation in ($\phi, \psi$) dihedral angles and (ii) to predict local structures from the sequence. Kolodny et al. (2002) constructed libraries that differ in the fragment length (four to seven residues) and number of representative fragments they contain.

Figure 5.1 Conformational letter (CLE) and one example of assignment. On left shows the 17 CLEs with four contiguous C-alpha in length. They are arranged according to helix-like (in red box), sheet-like (in blue box), and coil-like (in green box). On right shows the example of transforming a 3D protein structure into a 1D CLE string. Each residue $i$ is assigned with a specific CLE by first extracting three angles $x \equiv (\theta, \tau, \theta')$ of four contiguous C-alpha around residue $i$, and then by converting $x$ to a discretized letter with the maximal probability.
which offer a wide range of optimal fragments suited to different accuracies of fit. Tung et al. (2007) have proposed a SA that consists of 23 representative fragments of length 5 by clustering in the Kappa-alpha plot.

### 2.3. Conformational letters

Three contiguous C-alpha atoms determine two pseudo-bonds and a bending angle between them. Four contiguous C-alpha atoms, say \( i-2, i-1, i, \) and \( i+1 \), determine two such bending angles \( \theta, \theta' \) and a torsion angle \( \tau \) which is the dihedral angle between the two planes of triangles \( (i-2, i-1, i) \) and \( (i-1, i, i+1) \), as shown in Fig. 5.1. By using a Gaussian mixture model \( M \) for the density distribution \( x \) of the three angles,

\[
P(x|M) = \sum_{k=1}^{C} \pi_k N(\mu_k, \Sigma_k)
\]

where \( x \equiv (\theta, \tau, \theta') \), \( C \) is the number of the normal distribution categories in the mixture, \( \pi_k \) is the prior for category \( \epsilon_k \), and \( N(\mu, \Sigma) \) is the normal distribution; the local structural states have been clustered as 17 discrete CLEs of a protein structural alphabet. The centers \( \mu \), inverse covariance matrices \( \Sigma^{-1} \), and weights \( \pi \) of the clusters for these CLEs in the phase space spanned by the three angles \( (\theta, \tau, \theta') \) are listed in Table 5.2. When we convert a continuous point \( x \) to its discretized letter \( k^* \), the following equation is applied,

\[
k^* = \arg \max \pi_k x
\]

where \( \pi(x) = \pi_k | \Sigma_k |^{-1/2} \exp(1/2(x - \mu_k)\Sigma_k^{-1}(x - \mu_k)) \).

### 2.4. Applications of structural alphabet

The works on PBs have proved their efficiencies in the description and the prediction of long fragments and short loops (de Brevern, Valadié, Hazout, & Etchebest, 2002; Fourrier, Benros, & De Brevern, 2004), to define a reduced AA alphabet dedicated to mutation design (Etchebest, Benros, Bornot, Camproux, & de Brevern, 2007), in the building of a transmembrane protein (De Brevern et al., 2005) and in the binding-site signature analysis (Dudev & Lim, 2007). Tung et al. (2007) have applied their set of SA to a protein structure database search tool, named 3D-BLAST, which has the features of BLAST (Altschul et al., 1997), for analyzing novel structures and can return a ranked list of alignments. Le, Pollastri, and Koehl (2009) have extended the application of structural alphabets to the problem...
Table 5.2 The 17 conformational states from the mixture model

| State | $\pi$ | $|\Sigma|^{-1/2}$ | $\mu_0$ | $\mu_\tau$ | $\mu_{00}$ | $\mu_{0\tau}$ | $\mu_{\tau\tau}$ | $\mu_{0'0}$ | $\mu_{0'\tau}$ | $\mu_{0'0'}$ |
|-------|-----|----------------|--------|--------|---------|---------|---------|---------|---------|---------|
| I     | 8.2 | 1881           | 1.52   | 0.83   | 1.52    | 275.4   | −28.3   | 84.3    | 106.9   | −46.1   | 214.4   |
| J     | 7.3 | 1797           | 1.58   | 1.05   | 1.55    | 314.3   | −10.3   | 46.0    | 37.8    | −70.0   | 332.8   |
| H     | 16.2| 10,425         | 1.55   | 0.88   | 1.55    | 706.6   | −93.9   | 245.5   | 128.9   | −171.8  | 786.1   |
| K     | 5.9 | 254            | 1.48   | 0.70   | 1.43    | 73.8    | −13.7   | 21.5    | 15.5    | −25.3   | 75.7    |
| F     | 4.9 | 105            | 1.09   | −2.72  | 0.91    | 24.1    | 1.9     | 10.9    | −11.2   | −8.8    | 53.0    |
| E     | 11.6| 109            | 1.02   | −2.98  | 0.95    | 34.3    | 4.2     | 15.2    | −9.3    | −22.5   | 56.8    |
| C     | 7.5 | 100            | 1.01   | −1.88  | 1.14    | 28.0    | 4.1     | 6.2     | 2.3     | −5.1    | 69.4    |
| D     | 5.4 | 78             | 0.79   | −2.30  | 1.03    | 56.2    | 3.8     | 4.2     | −10.8   | −2.1    | 30.1    |
| A     | 4.3 | 203            | 1.02   | −2.00  | 1.55    | 30.5    | 9.1     | 8.7     | 6.0     | 5.7     | 228.6   |
| B     | 3.9 | 66             | 1.06   | −2.94  | 1.34    | 26.9    | 4.6     | 4.9     | 9.5     | −5.0    | 54.3    |
| G     | 5.6 | 133            | 1.49   | 2.09   | 1.05    | 163.9   | 0.6     | 3.8     | 2.0     | −3.7    | 32.3    |
| L     | 5.3 | 40             | 1.40   | 0.75   | 0.84    | 43.7    | 2.5     | 1.4     | −7.0    | −2.9    | 34.5    |
| M     | 3.7 | 144            | 1.47   | 1.64   | 1.44    | 72.9    | 2.1     | 4.8     | 1.9     | −7.9    | 72.9    |
| N     | 3.1 | 74             | 1.12   | 0.14   | 1.49    | 25.3    | 3.2     | 3.1     | 9.9     | 0.9     | 83.0    |
| O     | 2.1 | 247            | 1.54   | −1.89  | 1.48    | 170.8   | −0.7    | 3.7     | −4.1    | 3.1     | 98.7    |
| P     | 3.2 | 206            | 1.24   | −2.98  | 1.49    | 48.0    | 8.2     | 7.3     | −4.9    | −6.6    | 155.6   |
| Q     | 1.7 | 25             | 0.86   | −0.37  | 1.01    | 28.4    | 1.5     | 1.2     | 3.4     | 0.1     | 19.5    |

Angles are in radian. The structural alphabets are sorted according to helix-like, sheet-like, and coil-like.
of protein structure classification that compares with the performance of
other two different sequence representation of proteins: the AA sequence
and the SSE sequence. For a more detailed review on SA application, please

2.5. Substitution matrix for structural alphabet

To use our structural alphabet directly for the structural comparison, a score
matrix similar to BLOSUM for AAs is desired. Using the alignments for rep-
resentative structures in the database FSSP (Holm & Sander, 1994), which
contains 2860 sequence families representing 27,181 protein structures, we
have constructed a substitution matrix called CLESUM for the CLEs. To
the best of our knowledge, CLESUM is the first substitution matrix directly
derived from structure alignments for a structural alphabet (Zheng & Liu,
2005). In particular, the structures of the representative set are converted
to their CLE sequences. All of the pair alignments of FSSP for the proteins
with a sufficient similarity in the representative set are collected for counting
aligned pairs of CLEs. The total number of letter pairs is 1,284,750. An entry
of the matrix is the log ratio of the observed frequency of the aligned
corresponding pair to the expected frequency from a random alignment sim-
ply by chance. The substitution matrix (CLESUM) derived in the same way
as BLOSUM was shown in Table 5.3. For the other structural alphabet’s
substitution matrix, please refer to Tung et al. (2007) and Tyagi et al. (2008).

In order to reveal the relationship between sequence and structure, it is
interesting to consider both AA and SA in a joint space. However, such
space has too many parameters, so the reduction of AA is necessary to bring
down the parameter numbers. As a result, we have developed a simple but
effective approach called entropic clustering based on selecting the best
mutual information between a given reduction of AAs and SAs. The opti-
mized reduction of AA into two groups leads to hydrophobic and hydro-
philic. Combined with our CLE of 17 alphabets, we get a joint alphabet
called hydropathy conformational letter. A joint substitution matrix with
\((17 \times 2) \times (17 \times 2)\) indices is derived (Wang, 2010).

3. PROTEIN PAIRWISE STRUCTURE ALIGNMENT

3.1. Introduction of protein pairwise structure alignment

3.1.1 Why we need the protein pairwise structure alignment

To study a large collection of objects, such as protein structures, we usually
start with classifying them according to a given measure of similarity. Protein
Table 5.3 CLESUM: The conformation letter substitution matrix (in units of 0.5 bit)

<table>
<thead>
<tr>
<th>I</th>
<th>J</th>
<th>H</th>
<th>K</th>
<th>F</th>
<th>E</th>
<th>C</th>
<th>D</th>
<th>A</th>
<th>B</th>
<th>G</th>
<th>L</th>
<th>M</th>
<th>N</th>
<th>O</th>
<th>P</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>-9</td>
<td>-11</td>
<td>-6</td>
<td>-11</td>
<td>-3</td>
<td>-7</td>
<td>-3</td>
<td>-4</td>
<td>0</td>
<td>-1</td>
<td>-3</td>
<td>-5</td>
</tr>
<tr>
<td>J</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>-7</td>
<td>-9</td>
<td>-4</td>
<td>-9</td>
<td>-2</td>
<td>-6</td>
<td>-2</td>
<td>-3</td>
<td>2</td>
<td>0</td>
<td>-2</td>
<td>-3</td>
</tr>
<tr>
<td>H</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>-11</td>
<td>-13</td>
<td>-8</td>
<td>-12</td>
<td>-5</td>
<td>-10</td>
<td>-5</td>
<td>-6</td>
<td>0</td>
<td>-3</td>
<td>-6</td>
<td>-6</td>
</tr>
<tr>
<td>K</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>-7</td>
<td>-8</td>
<td>-3</td>
<td>-8</td>
<td>-2</td>
<td>-5</td>
<td>-1</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>-4</td>
</tr>
<tr>
<td>F</td>
<td>-9</td>
<td>-7</td>
<td>-11</td>
<td>-7</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>-2</td>
<td>1</td>
<td>-1</td>
<td>-2</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>-11</td>
<td>-9</td>
<td>-13</td>
<td>-8</td>
<td>3</td>
<td>4</td>
<td>0</td>
<td>2</td>
<td>-3</td>
<td>2</td>
<td>-2</td>
<td>-2</td>
<td>-5</td>
<td>-4</td>
<td>-5</td>
<td>-1</td>
</tr>
<tr>
<td>C</td>
<td>-6</td>
<td>-4</td>
<td>-8</td>
<td>-3</td>
<td>2</td>
<td>0</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>-11</td>
<td>-9</td>
<td>-12</td>
<td>-8</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>-2</td>
<td>1</td>
<td>-3</td>
<td>-2</td>
<td>-5</td>
<td>-4</td>
<td>-4</td>
</tr>
<tr>
<td>A</td>
<td>-3</td>
<td>-2</td>
<td>-5</td>
<td>-2</td>
<td>-2</td>
<td>-3</td>
<td>1</td>
<td>-2</td>
<td>7</td>
<td>2</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>B</td>
<td>-7</td>
<td>-6</td>
<td>-10</td>
<td>-5</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>4</td>
</tr>
<tr>
<td>G</td>
<td>-3</td>
<td>-2</td>
<td>-5</td>
<td>-1</td>
<td>-1</td>
<td>-2</td>
<td>0</td>
<td>-3</td>
<td>-1</td>
<td>1</td>
<td>7</td>
<td>3</td>
<td>2</td>
<td>-1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>L</td>
<td>-4</td>
<td>-3</td>
<td>-6</td>
<td>0</td>
<td>-2</td>
<td>-2</td>
<td>0</td>
<td>-2</td>
<td>-1</td>
<td>-1</td>
<td>3</td>
<td>7</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>M</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>-3</td>
<td>-5</td>
<td>-1</td>
<td>-5</td>
<td>0</td>
<td>-1</td>
<td>2</td>
<td>1</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>N</td>
<td>-1</td>
<td>0</td>
<td>-3</td>
<td>3</td>
<td>-3</td>
<td>-4</td>
<td>-1</td>
<td>-4</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
<td>9</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>O</td>
<td>-3</td>
<td>-2</td>
<td>-6</td>
<td>1</td>
<td>-3</td>
<td>-5</td>
<td>0</td>
<td>-4</td>
<td>3</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>P</td>
<td>-5</td>
<td>-3</td>
<td>-6</td>
<td>-4</td>
<td>0</td>
<td>-1</td>
<td>0</td>
<td>-1</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>-1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>Q</td>
<td>-6</td>
<td>-4</td>
<td>-9</td>
<td>-2</td>
<td>0</td>
<td>-1</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>-1</td>
<td>3</td>
<td>-1</td>
<td>1</td>
</tr>
</tbody>
</table>
structure similarity is most often detected and quantified by a protein pairwise structure alignment program. The comparison of protein structures, as an extremely important problem in structural and evolutional biology, has a large spectrum of applications. The detection of local or global structural similarity between a new protein and a protein with known function allows the prediction of the new protein’s function (Roy, Yang, & Zhang, 2012). Structural comparison methods are useful for organizing and classifying known structures (Orengo et al., 1997), and for discovering structure patterns and their correlation with sequences (Bradley et al., 2002). Recently, protein pairwise structure alignment tools have been applied to generate the training set for a template-based protein structure prediction (i.e., threading) tool that is based on Machine Learning (Ma, Peng, Wang, & Xu, 2012; Ma, Wang, Zhao, & Xu, 2013), as well as being extended to some other applications such as binding-site recognition (Wass, Kelley, & Sternberg, 2010). Since structure alignment is an NP-hard problem, which is only computationally tractable by using heuristics (Koehl, 2006). In the following sections, we discuss three levels of heuristics.

3.1.2 First level of heuristics: What are the representations for protein structures

There are overall four types of representations for protein structure, which are C-alpha, C-map, SSE, and Profile. C-alpha stands for the representation of protein structure by backbone atoms (C-alpha). C-map is contact or distance map-based method. SSE stands for using secondary structure element to represent proteins. Profile is the usage of AA type and/or discrete structural alphabets to represent the protein. For example, DALI (Holm & Sander, 1993) represents the protein as a C-alpha-based distance map (C-map); VAST (Gibrat et al., 1997) represents the protein as the SSEs, PB-Align (Tyagi et al., 2008) transforms the protein structure into 1D PB string (Profile), and most other methods use C-alpha to represent the protein. For a more detailed description, see Table 5.4. Many programs mix several representations, such as Daniluk and Lesyngg (2011) and Kolbeck et al. (2006).

3.1.3 Second level of heuristics: What are the perspectives for protein structures

Although no universally acknowledged definition of what constitutes structural similarity, there are two major perspectives for the spectrum of
<table>
<thead>
<tr>
<th>Name</th>
<th>Comments</th>
<th>Represent</th>
<th>Perspective</th>
<th>Score</th>
<th>Search algorithm</th>
<th>Order</th>
<th>Available link</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>DALI</td>
<td>Distance matrix alignment</td>
<td>C-map</td>
<td>Implicit-flexible</td>
<td>2D</td>
<td>Monte Carlo</td>
<td>N</td>
<td><a href="http://ekhidna.biocenter.helsinki.fi/dali_server/start">http://ekhidna.biocenter.helsinki.fi/dali_server/start</a></td>
<td>Holm and Sander (1993)</td>
</tr>
<tr>
<td>CE</td>
<td>Combinatorial extension on aligned fragment pairs (AFPs)</td>
<td>C-map and C-alpha</td>
<td>Rigid-body</td>
<td>2D and 3D</td>
<td>AFP-chaining</td>
<td>Y</td>
<td><a href="http://source.rcsb.org/jfatcatserver/ceHome.jsp">http://source.rcsb.org/jfatcatserver/ceHome.jsp</a></td>
<td>Shindyalov and Bourne (1998)</td>
</tr>
<tr>
<td>ProSup</td>
<td>A refined tool for protein structure alignment</td>
<td>C-alpha</td>
<td>Rigid-body</td>
<td>3D</td>
<td>Maximize Ne with constraint RMSD</td>
<td>Y</td>
<td><a href="https://topmatch.services.came.sbg.ac.at/">https://topmatch.services.came.sbg.ac.at/</a></td>
<td>Lackner, Koppensteiner, Sippl, and Domingues (2000)</td>
</tr>
<tr>
<td>Name</td>
<td>Comments</td>
<td>Represent</td>
<td>Perspective</td>
<td>Search algorithm</td>
<td>Order</td>
<td>Available link</td>
<td>References</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------------------------</td>
<td>-----------</td>
<td>-------------</td>
<td>------------------------</td>
<td>-------</td>
<td>---------------------------------------------------</td>
<td>---------------------------------</td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
<td>Spatial Constraints</td>
<td>Dimension</td>
<td>Search Method</td>
<td>Flexibility</td>
<td>Rigidity</td>
<td>Algorithm Details</td>
<td>Website</td>
</tr>
<tr>
<td>------------</td>
<td>----------------</td>
<td>---------------------</td>
<td>-----------</td>
<td>---------------</td>
<td>-------------</td>
<td>----------</td>
<td>----------------------------------------------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>3D-Blast</td>
<td>Structural alphabets and BLAST type algorithm</td>
<td>Profile</td>
<td>Implicit</td>
<td>Blast-like</td>
<td></td>
<td>Y</td>
<td>Tung et al. (2007)</td>
<td><a href="http://3d-blast.life.nctu.edu.tw/">http://3d-blast.life.nctu.edu.tw/</a></td>
</tr>
<tr>
<td>PB-Align</td>
<td>Protein blocks (PB)-based structure alignment</td>
<td>Profile</td>
<td>Implicit</td>
<td>String</td>
<td></td>
<td>Y</td>
<td>Tyagi et al. (2008)</td>
<td><a href="http://www.bo-protsience.fr/pbe/?page_id=12">http://www.bo-protsience.fr/pbe/?page_id=12</a></td>
</tr>
<tr>
<td>Name</td>
<td>Comments</td>
<td>Represent</td>
<td>Perspective</td>
<td>Score</td>
<td>Search algorithm</td>
<td>Order</td>
<td>Available link</td>
<td>References</td>
</tr>
<tr>
<td>-------</td>
<td>---------------------------------------------------------------------------</td>
<td>-----------</td>
<td>--------------</td>
<td>----------</td>
<td>--------------------------------</td>
<td>-------</td>
<td>---------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>ALADYN</td>
<td>Dynamics-based alignment by collective movements</td>
<td>Profile</td>
<td>Implicit-flexible</td>
<td>3D</td>
<td>Maximize Ne with constraint RMSD</td>
<td>N</td>
<td><a href="http://aladyn.escience-lab.org/">http://aladyn.escience-lab.org/</a></td>
<td>Potestio, Aleksiev, Pontiggia, Cozzini, and Micheletti (2010)</td>
</tr>
<tr>
<td>SPalign</td>
<td>A size-independent score-based protein structure alignment</td>
<td>C-alpha</td>
<td>Rigid-body</td>
<td>3D</td>
<td>Dynamic programming</td>
<td>Y</td>
<td><a href="http://sparks.infomatics.iupui.edu/yueyang/server/SPalign/">http://sparks.infomatics.iupui.edu/yueyang/server/SPalign/</a></td>
<td>Yang, Zhan, Zhao, and Zhou (2012)</td>
</tr>
<tr>
<td>DeepAlign</td>
<td>A score that considers evolutionary information and hydrogen-bonding similarity</td>
<td>Profile and C-alpha</td>
<td>Beyond spatial proximity</td>
<td>1D + 3D</td>
<td>Dynamic programming</td>
<td>Y</td>
<td><a href="http://ttic.uchicago.edu/~jinbo/DeepAlign/DeepAlign_exe_V1.00.tar.gz">http://ttic.uchicago.edu/~jinbo/DeepAlign/DeepAlign_exe_V1.00.tar.gz</a></td>
<td>Wang et al. (2013)</td>
</tr>
</tbody>
</table>

1D and 3D indicates that the algorithm applies different scoring functions during different stages. 1D + 3D indicates that the algorithm applies one scoring function that contains 1D and 3D information.

Y indicates that the alignment results are in sequential order. N indicates that the alignment could result in nonsequential order.
structural alignment methods, say rigid and flexible (Hasegawa & Holm, 2009). These perspectives differ in their treatment of structural variations. In particular, the rigid perspective treats the protein structures as rigid three-dimensional objects since there is a strong tradition to visualize structural alignments by least-squares superimposition. On the other hand, the flexible representations are classified into two groups. Implicit-flexible is measured by the difference between distance matrices, examples are DALI and Vorolign (Birzele et al., 2007). Explicit-flexible are the subtle changes in the angle between concatenated fragments, examples are MATT (Menke et al., 2008) and FATCAT (Ye & Godzik, 2003). All flexible representations carry detailed information about internal motions of the protein structures.

3.1.4 Third level of heuristics: How to construct a protein pairwise structure alignment

A protein pairwise structure alignment method consists of two major components: a scoring function to measure the protein similarity and a search algorithm to optimize the scoring function (Hasegawa & Holm, 2009; Wang et al., 2013). It is proposed that different scoring functions can be classified into three types depending on whether the structural representation is three-dimensional, two-dimensional, or one-dimensional. Once a similarity score has been defined, a search algorithm will be applied to find the optimal set of correspondences. An exhaustive search for such correspondence set between two structures is intractable, and various heuristics have been developed. The most commonly used approaches are Monte Carlo search (Daniluk & Lesyng, 2011; Holm & Sander, 1993), fragment assembly including graph extension algorithms (Krissinel & Henrick, 2004; Shatsky et al., 2002; Shindyalov & Bourne, 1998), an initial guess of the rigid-body transformation followed by DP (Levitt & Gerstein, 1998; Yang et al., 2012; Zhang & Skolnick, 2005), maximizing the number of structurally equivalent residues subject to a fixed Euclidean distance cutoff (Lackner et al., 2000; Potestio et al., 2010), and double dynamic programming (Birzele et al., 2007; Taylor & Orengo, 1989). Many programs mix several approaches, such as Ortiz, Strauss, and Olmea (2002), Shindyalov and Bourne (1998), and Wang and Zhengg (2008).

3.1.5 Why we should go beyond the traditional point of view

It should be noted that under the traditional point of view, whatever representations are used, whether rigid or flexible perspective, the scoring functions are based on spatial proximity, which neglects of considering other
information such as AA mutation score and local structure substitution potential. It is observed that despite proteins in a family share a similar overall shape, their structures exhibit very high local flexibility due to evolutionary events (i.e., mutation, insertion, and deletion) at the sequence or local sub-structure level \((\text{Csaba et al., 2008})\). This kind of local conformation change due to evolutionary events cannot be accurately quantified by spatial proximity of aligned residues (e.g., after rigid-body superposition). Instead, evolutionary distance shall be a better measure. Inspired by this observation, we develop a method named DeepAlign that uses hydrogen-bonding similarity, plus the AA and local substructure substitution matrices, which are derived from evolutionarily related protein pairs, to align protein local structures \((\text{Wang et al., 2013})\). Experimental results show that DeepAlign can generate structure alignments much more consistent with manually curated alignments than other automatic tools especially when proteins under consideration are remote homologs. These results imply that in addition to spatial proximity, evolutionary information and hydrogen-bonding similarity are essential to aligning two protein structures.

3.2. The scoring functions to measure the structure similarity

The scoring functions to measure the protein structure similarity could be categorized into three groups that depend on whether the structural representation is three-dimensional, two-dimensional, or one-dimensional or one number characterizing the whole structure. For more details on scoring function, see Hasegawa and Holmm \((2009)\).

3.2.1 Three-dimensional

Upon rigid-body superimposition, the similarity of 3D objects can be measured by the positional deviations of equivalent atoms. Numerous scoring functions have been proposed based on the rigid-body perspective that depends on the balance between the size of the common core \((\text{Ne})\) and residue positional deviations (root mean squared deviation, RMSD) \((\text{Hasegawa & Holm, 2009})\). For protein superposition, the most common metric is the coordinate root mean square deviation (cRMS) defined as,

\[
cRMS(A_i, B_j) = \| A - B \| = \sqrt{\frac{1}{N} \sum_{k=0}^{N-1} (a_{i+k} - b_{j+k})^2}
\]  

(5.3)
where $A$ and $B$ are two-aligned ungapped fragment pairs with length $n$, begin at $a_i$, $b_j$ position from two input proteins (or in brief, $\langle i, j \rangle$ for later use). Other metrics that are derived from cRMS consist of STRUCTAL-score (Levitt & Gerstein, 1998), TMscore (Zhang & Skolnick, 2004), GDT-score (Zemla, 2003), just name a few. Instead of a single rigid-body transformation, explicit-flexible perspective-based methods concatenate together a series of fragments, which have tight local superimpositions. Two measurements have been proposed, one seeks to globally minimize the number of bends (Ye & Godzik, 2003), while the other allows flexibilities everywhere between short fragments (Menke et al., 2008).

### 3.2.2 Two-dimensional
The 2D measurement is based on the similarity of residue–residue interaction patterns between two proteins without superimposition. Such patterns can be described, for example, by representing the protein structures as distance matrices, contact maps, or Voronoi tessellations. Distance matrices can be calculated by intramolecular Cα–Cα distances (Holm & Sander, 1993). The distance root mean squared deviation (dRMS) that compares corresponding distance matrices in the two sets of points is calculated as,

\[
dRMS(A_i, B_j) = \sqrt{\frac{1}{n^2} \sum_{k=0}^{n-1} \sum_{l=0}^{n-1} \left( a_{i+k} - a_{i+l} \right) \left| \left| b_{j+k} - b_{j+l} \right| \right|^2}
\]

where $A$ and $B$ are two-aligned ungapped fragment pairs with length $n$, begin at $\langle i, j \rangle$. One popular metric derived from dRMS is DALI-score (Holm & Sander, 1993). Moreover, contact maps can be generated by applying a certain distance threshold on distance matrices. Voronoi tessellations (Birzele et al., 2007), which get rid of the distance threshold to define contacting residues, create a mesh grid inside the protein structure such that every point of space is assigned to the nearest residue. The alignment methods that consider the 2D scoring functions could allow flexibility (i.e., implicit-flexible) as long as the interaction networks are conserved.

### 3.2.3 One-dimensional
Structural profiles, which classify each residue according to its AA type and/or discrete structural alphabets, could be applied to homology search by fast string comparison (Carpentier et al., 2005; Tung et al., 2007). Although these profiles have limited power to detect structural similarity
between proteins that deviate largely, they can provide additional evolutionarily related information that goes beyond the spatial proximity that considers proteins as 3D and/or 2D representation. One example that considers the AA and local substructure substitution matrices is as follows:

\[
\text{Local}(A_i, B_j) = \sum_{k=0}^{n-1} \text{CLESUM}(i+k,j+k) + \omega \cdot \text{BLOSUM}(i+k,j+k) \quad (5.5)
\]

where \(\omega\) is a tunable parameter to judge the contribution of AA evolutionary information.

3.3. The search algorithms for pairwise structure alignment

The current search algorithms could be categorized into five groups: Monte Carlo, AFP-chaining, DP, maximize Ne with constraint RMSD, and double dynamic programming. In this section, we introduce six widely used and/or cited protein structure alignment methods that encompass a variety of representations, perspectives, scoring function types, and search algorithms. They are DALI, CE, TMalign/STRUCTAL, MATT/FATCAT, CLEPAPS/ProSup, and Vorolign. For more available pairwise structure alignment tools, see Table 5.4.

3.3.1 DALI (C-map, implicit-flexible, 2D, Monte Carlo)

The scoring function of DALI (Holm & Sander, 1993), as defined below, is derived from dRMS:

\[
\text{DALI}(A_i, B_j) = \sum_{k=0}^{n-1} \sum_{l=0}^{n-1} \left(0.2 - \frac{A(i, k, j, l)}{d_{i,k,j,l}}\right) e^{-\left(d_{i,k,j,l}/20\right)^2} \quad (5.6)
\]

where \(A(i,k,j,l) = ||a_{i+k} - a_{i+l}|| - ||b_{j+k} - b_{j+l}||\) and \(d_{i,k,j,l} = 0.5 \times ||a_{i+k} - a_{i+l}|| + ||b_{j+k} - b_{j+l}||\). Then DALI uses the Monte Carlo sampling method to search for the best consistent set of SFPs to join into an alignment. The basic step in the Monte Carlo search is addition or deletion of residue equivalence assignments. The native score of DALI is a summation, overall aligned residue pairs in both structures, of a bonus score that is maximal when the inter-residue distances in both structures are equal. DALI uses many initial alignments and runs the searching algorithm in parallel. The final output alignment is the one with the best total score. A later version of DALI introduced an initial fast lookup of common SSEs between the
two proteins (Holm & Rosenström, 2010). To determine the statistical significance of output alignments, DALI computes the Z-score of an alignment by using the background distribution of scores from an all-against-all comparison of 225 representative structures with less than 30% sequence identity (Holm & Sander, 1994). As an elastic aligner, DALI could align protein structures in cases where structural flexibilities exist. Due to Monte Carlo methodology, DALI could generate alignments in nonsequential order.

### 3.3.2 CE (C-map and C-alpha, rigid-body, 2D and 3D, AFP-chaining)

CE (Shindyalov & Bourne, 1998) has two stages during alignment, the initial stage and the refinement stage. For the initial stage, the scoring function of CE is based on dRMS (as shown in Eq. 5.4), and the search algorithm is combinatorial extension to successively join new aligned fragment pairs (AFPs) that is consistent with the previous joined AFPs on the alignment path. Such methodology is called, in brief, AFP-chaining. In particular, the AFPs are pairs of eight-residue fragment, which are considered similar if their corresponding internal distances are similar (of low dRMS). Gaps are allowed between neighboring AFPs, but their length is limited (less than 30 residues) to speedup. CE constructs the alignment by choosing an initial AFP and extending it. Even though using greedy algorithm during search, CE also considers AFPs that are not global optimal with respect to their scoring function to widen and improve the search. For the refinement stage, the scoring function of CE is based on cRMS (as shown in Eq. 5.3), and the search algorithm is to lengthen the alignment without compromising its cRMS. The native score of CE is a Z-score that evaluates the statistical significance of the alignment by considering the probability of finding an alignment of the same number of equivalent residues (Ne), number of gaps (Ngaps), and geometrical distance (measured by dRMS). Due to AFP-chaining methodology, CE could generate alignments in sequential order.

### 3.3.3 TMalign and STRUCTAL (C-alpha, rigid-body, 3D, DP)

TMalign (Zhang & Skolnick, 2005) is evolved from STRUCTAL (Levitt & Gerstein, 1998). The scoring function for both methods is a reciprocal-like cRMS score (as shown in Eq. 5.7), and the search algorithm is DP to optimize this score. In detail, both methods start from several initial alignments and the output alignment is the one with the best score. To align two protein structures, the methods iteratively get the optimal rigid-body transformation based on the current alignment by Kabsch method (Kabsch, 1976) and then find an optimal alignment based on the current transformation by DP on the
reciprocal-like cRMS score matrix. The procedure is repeated till it converges to a local optimum. The other methods that apply DP on protein structure alignment include Yang et al. (2012). Due to DP methodology, TMalign and STRUCTAL could generate alignments in sequential order.

The reciprocal-like cRMS score that applied in TMalign and STRUCTAL is defined as follows:

\[
r_{cRMS}(A_i, B_j) = \frac{1}{C_0/C_1} \sum_{k=0}^{n-1} d(i+k, j+k), \text{ where } d(i, j) = \frac{1}{1 + (a_i - b_j)^2/d_0^2}
\] (5.7)

However, the difference between them is that the TMalign score (called TMscore) applies a length-dependent \(d_0 = 1.24 \times \sqrt{L_\text{s} - 15} - 1.8\), where \(L_\text{s}\) is the smaller length of the input proteins, while STRUCTAL-score has a fixed value for \(d_0 = 5.0\) Å. Since the cRMS weights the distances between all residue pairs equally, a small number of local structural deviations could result in a high cRMS, even when the global topologies of the compared structures are similar. TMscore and STRUCTAL-score overcome the problem of cRMS by reweighting the residue pairs with smaller distances relatively stronger than those with larger distances. Therefore, these scores are more sensitive to the global topology than to the local structural variations. Moreover, TMscore is normalized in a way that the score magnitude relative to random structures does not depend on the proteins’ size. TMscore around 0.17 means an average pair of randomly related structures (Zhang & Skolnick, 2004), and 0.5 means highly likely that they have similar folds (Xu & Zhang, 2010).

### 3.3.4 CLEPAPS and ProSup (C-alpha, rigid-body, 3D, maximize Ne with constraint RMSD)

CLEPAPS (Wang, 2009; Wang & Zheng, 2008) has two stages during alignment, the initial stage and the refinement stage. Here, we only focus on the refinement stage of CLEPAPS, which is evolved from ProSup (Lackner et al., 2000). The goal in this stage is to maximize the number of structurally equivalent residues (i.e., Ne) subject to a fixed Euclidean distance cutoff (i.e., constraint RMSD). The search algorithm basically has four steps: (i) identify a seed SFP in both proteins, (ii) iteratively recruit more SFPs to form initial alignments based on the superposed seed SFP, (iii) iteratively apply a DP to select as many equivalent residues as possible
while subject to a fixed distance cutoff, and (iv) evaluate the final alignment based on a certain scoring function. Note that, in the third step, other approaches instead of DP could be applied to maximize the Ne, such as the one used in CLEPAPS. In particular, guided by CLESUM score, it iteratively recruits those SFPs with a smaller length compared to those in the second step, to form the current alignments based on the superposition that is generated from previous alignment. During each iteration, the distance cutoff is lowered down gradually. Such approach, that might be called “Zoom-In,” could generate alignments in nonsequential order, while the approach based on DP could result in sequential order alignments. The other methods that apply the similar refinement approach include Ortiz et al. (2002), Shindyalov and Bourne (1998), and Zemla (2003).

3.3.5 MATT and FATCAT (C-alpha, explicit-flexible, 3D, AFP-chaining)
MATT (Menke et al., 2008) is evolved from FATCAT (Ye & Godzik, 2003). Both methods belong to the AFP-chaining method. Like other structural aligners in this class, it first finds AFPs of between five and nine residues in each chain that share very close spatial proximity. Then, they greedily add these AFPs into a final structural alignment and extend these AFPs by adding adjacent residues into the interblock regions. What MATT and FATCAT differs from other AFP-chaining methods is that these two allow flexible (or bend) operations between adjacent AFPs. In particular, MATT allowing these impossible bends, translations, and twists everywhere between AFPs prior to the final extension phase. FATCAT globally minimizes the number of these flexible points. Therefore, MATT and FATCAT are able to detect regions in close spatial contact and incorporate them into an alignment, while other aligners would erroneously disallow such blocks from entering an alignment. Recently, another software ForMATT (Nadimpalli, Daniels, & Cowen, 2012) has been developed as an extension of MATT by taking into consideration primary sequence similarity in aligning two protein structures.

3.3.6 Vorolign (C-map, implicit-flexible, 2D, double dynamic programming)
Vorolign uses Voronoi tessellations to represent each structure. The method aligns protein structures using double dynamic programming (Birzele et al., 2007) and measures the similarity of two residues based on the evolutionary conservation of their corresponding Voronoi contacts in the protein structure. After the neighbor sets of each residue $a_i$ and $b_j$ in the two structures $A$
and $B$ are defined by Voronoi Tessellation, the first level DP is conducted to calculate the similarity of two neighbor sets. Then the calculated score for $(i, j)$ is used to fill the second-level DP that is applied to calculate the similarity between the two proteins. This similarity measurement allows Vorolign to align protein structures in cases where structural flexibilities exist. The other method that applies double dynamic programming on protein structure alignment includes Taylor and Orengoo (1989).

3.4. DeepAlign: The approach beyond spatial proximity

3.4.1 CLEPAPS

DeepAlign (Wang et al., 2013) is evolved from CLEPAPS (Wang & Zheng, 2008) which is the first trial to go beyond spatial proximity. CLEPAPS distinguishes itself from other existing algorithms by the use of a certain type of structural alphabets, say CLE. Guided by the substitution matrix CLESUM that is available to measure the similarity between any two such letter string by simple string comparison, CLEPAPS regards a SFP as an ungapped string pair with a high sum of pairwise CLESUM scores. A highly scored SFP which is spatially consistent, under cRMS measurement, with several other SFPs determines an initial alignment. CLEPAPS then joins consistent SFPs guided by their similarity scores to extend the alignment by several iteration steps (i.e., the “Zoom-In” approach, as described previously).

3.4.2 Scoring function

Instead of using structural alphabets alone, DeepAlign extends the scoring function considering AA mutation score, local substructure substitution potential, hydrogen-bonding similarity, as well as considering a reciprocal-like cRMS to measure the spatial proximity. In particular, the equivalence of two residues $a_i$ and $b_j$ from two input proteins is estimated by the following scoring function:

$$DeepScore(i,j) = (\max(0, BLOSUM(i,j)) + CLESUM(i,j))$$

$$\times d(i,j) \times v(i,j)$$

(5.8)

Meanwhile, BLOSUM and CLESUM measure the evolutionary distance of two proteins at the sequence and local substructure levels, respectively. BLOSUM is the widely used AA substitution matrix BLOSUM62 (Henikoff & Henikoff, 1992), CLESUM is the local structure substitution matrix, $d(i,j)$ measures the spatial proximity of two aligned residues after
rigid-body superposition with a reciprocal-like cRMS form (see Eq. 5.7), and \( \nu(i,j) \) measures the hydrogen-bonding similarity,

\[
\nu(i,j) = \frac{1}{3} \left( \sum_{x=-1,1} \frac{(a_i - a_{i-x})(b_j - b_{j-x})}{|a_i - a_{i-x}| |b_j - b_{j-x}|} + \frac{(a_i - a_{i_{cb}})(b_j - b_{j_{cb}})}{|a_i - a_{i_{cb}}| |b_j - b_{j_{cb}}|} \right)
\] (5.9)

where \( a_{i_{cb}} \) denotes the corresponding C\( \beta \) atom of \( a_i \), as shown in Fig. 5.2A. This score helps to align hydrogen bonds more accurately. As shown in Fig. 5.2B, the method that optimizes only spatial proximity (e.g., TMscore) leads to a wrong alignment, which can be corrected by incorporating \( \nu(i,j) \) to the scoring function.

In addition to the BLOSUM62 substitution matrix, other matrices (e.g., PAM250) can also be used to measure the evolutionary distance of two proteins at the sequence level (Dayhoff & Schwartz, 1978). The max() function in Eq. (5.8) is used to handle the situation where two proteins to be aligned are distantly related. In this case, we will only rely on CLESUM to measure the evolutionary distance. It is shown that replacing the max() function to a tunable parameter \( \omega \) like Eq. (5.5) also works well. In the future, we may use sequence profile similarity to measure evolutionary distance, which usually is more sensitive than BLOSUM matrices. Moreover, CLESUM disfavors

![Figure 5.2](image-url) The definition and application of the hydrogen-bonding score. (A) This picture shows the three vectors of two proteins used in the hydrogen-bonding score. One protein is represented by solid lines and the other by dashed lines. (B) An illustration of one wrong alignment between two beta-strands. Residues 1, 2, 3, and 4 in dark belong to protein A and residues 1', 2', 3', and 4' in gray belong to protein B. The aligned residue pairs are in dotted circle. On left is the wrong alignment generated by optimizing only TMscore; while on right is the correct alignment optimizing the product of hydrogen-bonding score and TMscore.
the match of two unrelated helices but favors the alignment of two evolutionarily related loop regions. Loop regions are usually harder to align than alpha-helices and beta-strands if only spatial proximity is used in the scoring function.

### 3.4.3 Search algorithm

**Overview.** The DeepAlign algorithm flowchart is illustrated in Fig. 5.3. It consists of the following steps: (i) identifying SFPs using AA and local substructure mutation matrices, (ii) generating an initial alignment from one SFP, (iii) refining alignments by DP, and (iv) gap elimination.

**Similar fragment pairs.** DeepAlign measures the equivalence of two residues \(i, j\) using AA and local substructure substitution matrices as follows:

\[
\text{Similarity}(i, j) = \max(0, \text{BLOSUM}(i, j)) + \text{CLESUM}(i) \tag{5.10}
\]

where CLESUM is the local conformation substitution matrix and BLOSUM is the AA substitution matrix as described before. Again, using Eq. (5.5) to calculate the similarity also works well. Using this score, we can identify two evolutionary-related instead of only geometric similar fragments and thus generate better initial alignments. We use two types of SFPs: short SFP with 6–8 residues and long SFP with 9–18 residues. A short SFP, denoted as SFP_s, shall have a similarity score at least 0 while a long SFP, denoted as SFP_L, shall have a similarity score at least 10. It is obvious that each SFP_L must contain at least one SFP_s. We use SFP_L and SFP_s to build coarse-grained and fine-grained initial alignments, respectively. SFP_L is slightly longer than the average length of a helix, while SFP_s has a similar length as a typical beta strand. By combining long and short SFPs, we can speed up our algorithm without losing accuracy. The higher score one SFP has, the more likely it is contained in the best alignment. Therefore,
we sort all SFPs and only keep those top-ranked SFPs. The idea of using two types of SFPs is derived from CLEPAPS (Wang & Zheng, 2008).

Generating initial alignments using SFPs. We select TopK long SFPs (i.e., SFP_L) and from each of them generate one coarse-grained initial alignment. In particular, we first calculate the rotation matrix using the Kabsch method (Kabsch, 1976) to minimize the RMSD of the two fragments in a SFP. Then we use this rotation matrix to transform one protein and generate an initial alignment using DP to maximize the scoring function, subject to the restriction that the distance deviation of two-aligned residues shall be less than $3 \times d_0$ ($d_0$ is defined in Eq. 5.7, which is length dependent). All these TopK coarse-grained initial alignments are sorted by the alignment score and only top TopJ ($<\text{TopK}$) are kept for further refinement. Starting from a coarse-grained initial alignment, we recalculate the rotation matrix using the SFP_s contained in the SFP_L and then realign the two proteins using DP to maximize the scoring function, which results in a better initial alignment.

Iterative refinement of alignment. Starting from an initial alignment, an iterative DP refinement procedure is applied to improving the alignment, with the goal to maximizing the scoring function. This procedure is very similar to that in many structure alignment methods such as Lackner et al. (2000), Shindyalov and Bournee (1998), and Zhang and Skolnick (2005).

Gap elimination. As shown in Holm and Sanderr (1993), an AFP shall not be too short (say less than four residues). However, since our scoring function does not explicitly penalize gap openings, the resultant alignment may have more gap openings than desirable. To deal with this, we use some heuristics to merge one very short AFP (less than four residues) to its neighboring AFPs to reduce the number of gap openings.

3.5. Result analysis
3.5.1 Overview
Common evaluation tests for structure alignment methods measure first, the accuracy of the alignments; second, the ability of the alignment score to discriminate homologous from unrelated proteins in database-wide comparisons (Hasegawa & Holm, 2009). Here, we first evaluate our program DeepAlign using three manually curated alignment databases on a few metrics: LALI (length of alignment), RefAcc (reference-dependent alignment accuracy), RMSD, TMscore, and mutation scores (i.e., BLOSUM and CLESUM). RefAcc is calculated as the percentage of correctly aligned positions as judged by the gold standard (i.e., manual alignments), measuring
consistency between automatic alignments and human-curated alignments. We then use a database-wide benchmark to test the performance of DeepAlign in identifying distant homologs and structural analogs. We compare DeepAlign with several popular structure alignment tools such as DALI (Holm & Sander, 1993), TMAalign (Zhang & Skolnick, 2005), MATT (Menke et al., 2008), and ForMATT (Nadimpalli et al., 2012), which represent four very different methods. Finally, we also test the performance of multiple solutions of DeepAlign on the three manually curated alignment databases.

3.5.2 The evaluation benchmarks
We use three manually curated benchmarks: (i) A subset of CDD (conserved domain database; Marchler-Bauer et al., 2005) used in Kim and Lee (2007), (ii) MALIDUP (Cheng, Kim, & Grishin, 2008a), and (iii) MALISAM (Cheng, Kim, & Grishin, 2008b) to evaluate the reference-dependent alignment accuracy. In addition, we use (iv) SABmark (Van Walle, Lasters, & Wyns, 2005) to test the performance for identifying distant homologs and structural analogs. The CDD set contains 3591 manually curated pairwise structure alignments. The human-curated alignments for CDD contain only the alignments of core residues. The CDD set has already been used to evaluate a bunch of pairwise structure alignment algorithms, including CE (Shindyalov & Bourne, 1998), FAST (Zhu & Weng, 2005), LOCK2 (Shapiro & Brutlag, 2004), MATRAS (Kawabata, 2003), VAST (Gibrat et al., 1997), and SHEBA (Jung & Lee, 2000). MALIDUP has 241 manually curated pairwise structure alignments for homologous domains originated from internal duplication within the same polypeptide chain. About half of the pairs in MALIDUP are remote homologs. The alignments in these two databases are manually curated, taking into consideration not only geometric similarity but also evolutionary and functional relationship. Therefore, the manually curated alignments make more biological sense, and it is reasonable to use them as reference to judge automatically generated alignments. SABmark (version 1.65) benchmark contains SABmark-sup and SABmark-twi. SABmark-sup is the superfamily set in SABmark, containing 425 protein groups with low to intermediate sequence identity. SABmark-twi is the twilight set in SABmark, containing 209 groups with low sequence identity. Each SABmark-sup (-twi) group contains at most 25 structures sharing a SCOP (Murzin, Brenner, Hubbard, & Chothia, 1995) superfamily (fold). It is believed that if two proteins are in the same SCOP superfamily, it is likely these two proteins are remote homologs. If two proteins share only
the same SCOP fold, it is very likely that they are structural analogs instead of remote homologs.

### 3.5.3 Performance on CDD

DeepAlign obtains the highest reference-dependent alignment accuracy of 93.8% among the five automatic structure alignment methods (Table 5.5). DeepAlign also outperforms the methods evaluated in Kim, Tai, and Lee (2009) in terms of ref-dependent alignment accuracy. That is, DeepAlign is more consistent with human experts than the other programs. In terms of TMscore and RMSD, the TMalign alignments are slightly better than the DeepAlign alignments, but the former are less consistent with manual alignments than the latter. This implies that the geometric similarity score used by TMalign (i.e., TMscore) does not accurately reflect the alignment criteria used by human experts. The DeepAlign alignments also have much better evolutionary scores than the other three programs no MATTer how the mutation scores are calculated. As a control, we also calculate the evolutionary scores of the manual alignments. The manual alignments have much lower mutation score per alignment because only core residues are aligned. However, the manual alignments have the best average mutation scores per aligned position. Note that the manual alignments are not explicitly driven by a specific mutation score. This confirms that human experts indeed take into consideration evolutionary relationship in aligning two protein structures and that TMalign may align many more evolutionarily unrelated residues together than DeepAlign. ForMATT has a similar mutation (i.e., BLOSUM/CLESUM) score per alignment as DeepAlign, but ForMATT has a better average mutation score per aligned position than DeepAlign because ForMATT has a smaller LALI.

### 3.5.4 Performance on MALIDUP

DeepAlign obtains a reference-dependent alignment accuracy of 92%, greatly exceeding the other three tools (Table 5.5). DeepAlign is 6% better than the second best algorithm DALI. Although the TMalign alignments have a longer alignment length and the MATT alignments have a smaller RMSD, both TMalign and MATT have much lower reference-dependent alignment accuracy. This again implies that the TMalign and MATT scoring functions greatly deviate from what are implicitly used by human experts. In terms of TMscore, DeepAlign is only slightly second to TMalign, but better than the others. However, the DeepAlign alignments have much better evolutionary scores, only second to the manual alignments in terms of the
<table>
<thead>
<tr>
<th>Method</th>
<th>LALI</th>
<th>RMSD</th>
<th>TMscore</th>
<th>RefAcc</th>
<th>Blosum1</th>
<th>Clesum1</th>
<th>Blosum2</th>
<th>Clesum2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CCD (3591)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DeepAlign</td>
<td>134.8</td>
<td>2.86</td>
<td>0.667</td>
<td>93.8</td>
<td>0.261</td>
<td>1.782</td>
<td>43.45</td>
<td>243.71</td>
</tr>
<tr>
<td>DALI</td>
<td>130.8</td>
<td>2.75</td>
<td>0.663</td>
<td>92.8</td>
<td>0.165</td>
<td>1.684</td>
<td>28.78</td>
<td>225.15</td>
</tr>
<tr>
<td>MATT</td>
<td>128.6</td>
<td>2.53</td>
<td>0.655</td>
<td>91.4</td>
<td>0.152</td>
<td>1.728</td>
<td>30.19</td>
<td>229.59</td>
</tr>
<tr>
<td>ForMATT</td>
<td>112.3</td>
<td>2.32</td>
<td>0.566</td>
<td>86.4</td>
<td>0.343</td>
<td>1.983</td>
<td>44.11</td>
<td>235.64</td>
</tr>
<tr>
<td>TMalign</td>
<td>138.4</td>
<td>2.84</td>
<td>0.686</td>
<td>85.6</td>
<td>0.047</td>
<td>1.531</td>
<td>15.25</td>
<td>211.88</td>
</tr>
<tr>
<td>Manual</td>
<td>62.6</td>
<td>1.66</td>
<td>0.345</td>
<td>100.0</td>
<td>0.677</td>
<td>2.499</td>
<td>43.89</td>
<td>157.67</td>
</tr>
<tr>
<td><strong>MALIDUP (241)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DeepAlign</td>
<td>85.5</td>
<td>2.61</td>
<td>0.622</td>
<td>92.0</td>
<td>0.314</td>
<td>1.872</td>
<td>29.31</td>
<td>158.28</td>
</tr>
<tr>
<td>DALI</td>
<td>83.5</td>
<td>2.65</td>
<td>0.600</td>
<td>86.4</td>
<td>0.172</td>
<td>1.700</td>
<td>18.63</td>
<td>147.53</td>
</tr>
<tr>
<td>MATT</td>
<td>82.3</td>
<td>2.47</td>
<td>0.608</td>
<td>79.8</td>
<td>0.178</td>
<td>1.824</td>
<td>18.84</td>
<td>150.00</td>
</tr>
<tr>
<td>ForMATT</td>
<td>70.6</td>
<td>2.19</td>
<td>0.542</td>
<td>86.2</td>
<td>0.344</td>
<td>2.196</td>
<td>28.62</td>
<td>154.66</td>
</tr>
<tr>
<td>TMalign</td>
<td>87.0</td>
<td>2.62</td>
<td>0.631</td>
<td>81.0</td>
<td>0.110</td>
<td>1.600</td>
<td>12.50</td>
<td>137.64</td>
</tr>
<tr>
<td>Manual</td>
<td>77.9</td>
<td>2.49</td>
<td>0.587</td>
<td>100.0</td>
<td>0.294</td>
<td>1.853</td>
<td>27.67</td>
<td>154.81</td>
</tr>
<tr>
<td>Method</td>
<td>LALI</td>
<td>RMSD</td>
<td>TMscore</td>
<td>TMscore</td>
<td>LALI</td>
<td>RMSD</td>
<td>TMscore</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>------</td>
<td>------</td>
<td>----------</td>
<td>----------</td>
<td>------</td>
<td>------</td>
<td>----------</td>
<td></td>
</tr>
<tr>
<td>DeepAlign</td>
<td>61.3</td>
<td>2.96</td>
<td>0.521</td>
<td>77.5</td>
<td>-0.601</td>
<td>1.108</td>
<td>-36.48</td>
<td>67.66</td>
</tr>
<tr>
<td>DALI</td>
<td>61.0</td>
<td>3.11</td>
<td>0.515</td>
<td>67.7</td>
<td>-0.595</td>
<td>0.925</td>
<td>-35.52</td>
<td>56.28</td>
</tr>
<tr>
<td>MATT</td>
<td>56.2</td>
<td>2.74</td>
<td>0.486</td>
<td>51.7</td>
<td>-0.625</td>
<td>1.013</td>
<td>-34.05</td>
<td>56.98</td>
</tr>
<tr>
<td>ForMATT</td>
<td>44.9</td>
<td>2.42</td>
<td>0.411</td>
<td>56.3</td>
<td>-0.486</td>
<td>1.489</td>
<td>-21.1</td>
<td>65.69</td>
</tr>
<tr>
<td>TMalign</td>
<td>61.1</td>
<td>3.06</td>
<td>0.517</td>
<td>53.7</td>
<td>-0.684</td>
<td>0.739</td>
<td>-40.04</td>
<td>45.65</td>
</tr>
<tr>
<td>Manual</td>
<td>56.7</td>
<td>2.92</td>
<td>0.488</td>
<td>100.0</td>
<td>-0.556</td>
<td>1.240</td>
<td>-31.58</td>
<td>70.75</td>
</tr>
</tbody>
</table>

See text for the explanation of LALI, RMSD, TMscore, and RefAcc. “Blosum1 (Clesum1)” is the average mutation score per aligned position, while “Blosum2 (Clesum2)” is the average mutation score per alignment. As a control, the performance of manually curated alignments is also shown in the table.
average mutation score per aligned position. Since the TMalign alignments on average are longer, this again confirms that TMalign may align many more evolutionarily unfavorable residues than DeepAlign. ForMATT performs similarly on this dataset as on CDD.

### 3.5.5 Performance on MALISAM

DeepAlign obtains the highest ref-dependent alignment accuracy of 77.5% among all the five computer programs (Table 5.5). DeepAlign is 10% better than the second best algorithm DALI. MALISAM is much more challenging than CDD and MALIDUP. In MALISAM, 80 pairs (i.e., 61.5% of the total) contain proteins with different SCOP folds. The DALI and TMalign alignments have similar average alignment lengths as the DeepAlign alignments, but slightly higher RMSD. Furthermore, the DALI, MATT, and TMalign alignments deviate significantly from the manual alignments. In terms of the BLOSUM scores, the difference between the DeepAlign alignments and others is not very significant. This is not unexpected because the proteins in this dataset are only weakly similar at sequence level and BLOSUM is not sensitive enough. However, the DeepAlign alignments have much better CLESUM score than the others, only slightly second to the manual alignments. That is, the DeepAlign alignments are more evolutionarily favorable than others at the local substructure level.

### 3.5.6 Performance on SABmark

Given a protein structure, we align it to all the proteins in the benchmarks and then rank all the alignments by certain criteria. We examine if the top-ranked protein structures are in the same group as the query protein or not. DeepAlign uses its scoring function, say DeepScore in Eq. (5.8), to rank the proteins. Similarly, TMalign, MATT, and DALI use TMscore, P-value, and Z-score to rank the alignments, respectively. The ranking results are evaluated by ROC (receiver operator curve) and AUC (area under curve). ForMATT has a very similar result as MATT. As shown in Fig. 5.4A, tested on SABmark-sup, DeepAlign has the best ROC curve, especially at the high specificity area. For example, at the specificity level 0.99, DeepAlign has sensitivity around 0.4, while the other three have sensitivity only around 0.2. We also observe the same trend on the SABmark-twi set. SABmark-twi is more challenging because each group in this set consists of proteins similar at only SCOP fold level. However, DeepAlign outperforms others by an even larger margin. As shown in Fig. 5.4B, DeepAlign has sensitivity 0.6 at specificity 0.96, while the second best algorithm DALI has sensitivity only 0.4 at
the same specificity level. MATT and TMalign have only sensitivity around 0.2 at this specificity level. These results imply that DeepAlign scoring function is better than DALI’s Z-score, MATT’s $P$-value, and TMscore in detecting the superfamily relationship of proteins.

### 3.5.7 Multiple solutions

Due to symmetry and repeats, some protein pairs may have multiple correct alignments. To deal with this, DeepAlign generates a few alternative...
alignments for a given protein pair instead of just a single alignment with the highest DeepScore (see Eq. 5.8). DeepAlign fulfills this by searching for the best alignments from many initial alignments and ranking the alignments by DeepScore. For a given protein pair, we generate the topM ($M \leq 10$) alternative alignments using DeepAlign and then pick up the one with the highest ref-dependent alignment accuracy. Table 5.6 lists the average ref-dependent accuracy of DeepAlign on the three benchmarks with respect to $M$. As shown in this table, we can significantly improve alignment accuracy by generating two alternative alignments. Nevertheless, using $\geq 3$ alternative alignments can only slightly improve the accuracy over using two alternatives. When only two alternative alignments are used, DeepAlign obtains alignment accuracy 95.1%, 92.7%, and 87.6% on CDD, MALIDUP, and MALISAM, respectively. As a control, for each protein pair, we also calculate alignment accuracy using the best alignment generated by DeepAlign (only the first-ranked alignment is used), DALI, TMalign, and MATT. It turns out that when only top two alternative alignments are used we can obtain alignment accuracy comparable to the combination of the four tools. Finally, we provide one example to illustrate the application of multiple solutions for the analysis of the symmetry proteins alignment (shown in Fig. 5.5).

<table>
<thead>
<tr>
<th>DeepAlign topM</th>
<th>CDD (3591)</th>
<th>MALIDUP (241)</th>
<th>MALISAM (130)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top1</td>
<td>93.8</td>
<td>92.0</td>
<td>77.5</td>
</tr>
<tr>
<td>Top2</td>
<td>95.1</td>
<td>92.7</td>
<td>87.6</td>
</tr>
<tr>
<td>Top3</td>
<td>95.2</td>
<td>93.1</td>
<td>88.0</td>
</tr>
<tr>
<td>Top4</td>
<td>95.3</td>
<td>93.1</td>
<td>88.9</td>
</tr>
<tr>
<td>Top5</td>
<td>95.3</td>
<td>93.1</td>
<td>89.7</td>
</tr>
<tr>
<td>Top10</td>
<td>95.3</td>
<td>93.1</td>
<td>90.0</td>
</tr>
<tr>
<td>Maximal of four$^a$</td>
<td>96.9</td>
<td>94.2</td>
<td>85.6</td>
</tr>
</tbody>
</table>

$^a$For each protein pair, the best alignment generated by four pairwise structure alignment tools DALI, TMalign, Matt, and DeepAlign is used to calculate the ref-dependent alignment accuracy.
4. PROTEIN MSA

4.1. Introduction of protein MSA

4.1.1 Why we need the protein MSA

MSA carries significantly more information than pairwise alignment and hence has been extensively used for classification, analysis of evolutionary relationship, motif detection, and structure/function prediction (Wang et al., 2011). When proteins are distantly related, sequence methods usually fail to yield accurate alignment. In contrast, structure methods, which exploit geometrical information, may still work well. As more protein structures are experimentally solved, MSA is becoming more useful and important. The applications of MSA programs include understanding evolutionary conservation and divergence (Andersen et al., 2001), functional prediction through the identification of structurally conserved active sites in

Figure 5.5 Three solutions reported by DeepAlign for the proteins 8i1b (in blue) and 4fgf (in red), that belong to SCOP fold beta-trefoil. (A) This fold exhibits a threefold rotational symmetry, which is confirmed by the fragment shown with bold lines in color orange, purple, and yellow, respectively. The corresponding conformational letter is also shown in left. (B) Superposition of the three solutions with 4fgf fixed.
homologous proteins (Irving et al., 2001), construction of benchmark datasets on which to test multiple sequence alignment programs (Edgar & Batzoglou, 2006), and automatic construction of profiles and threading templates for protein structure prediction (Dunbrack, 2006; Panchenko et al., 1999). However, developing computational methods for accurate MSA, especially of a large set of distantly related protein structures, is still regarded as an open challenge (Wang & Zheng, 2009).

4.1.2 The goal of the protein MSA
The common goal of all MSA methods is to identify a set of residue “columns” from each “row” protein that are structurally similar (Wang & Zheng, 2009). For a given MSA, the aligned blocks, which correspond to contiguous columns, is composed of locally similar fragments. This local similarity within a block may be phrased as “vertical equivalency.” The local similarity is necessary to the alignment but is insufficient. For any two structures in the MSA, the transformation to superimpose an AFP in an aligned block should also bring the fragment pairs in other blocks spatially close. This is the “horizontal consistency.” In some MSA methods, these aligned blocks are called “cores.” The full core (partial core) is defined as the aligned blocks that contain all (some) of the input proteins.

4.1.3 Limitations of the current approaches
A protein MSA method consists of two major components: how the pairwise structure alignment is conducted and how to merge all input structures to form a MSA. The current MSA methods to deal with the merging method could be categorized into horizontal-first and vertical-first approaches. The horizontal-first approaches (Lupyan, Leo-Macias, & Ortiz, 2005; Micheletti & Orland, 2009; Ye & Godzik, 2005) progressively merge pairwise alignments into an MSA, which not only might be slow with the increase of the input protein number but also might be suboptimal since pairwise alignment errors carry over to the final result (Wang et al., 2011; Wang & Zheng, 2009). The vertical-first approaches (Dror, Benyamini, Nussinov, & Wolfson, 2003; Shatsky et al., 2004) identify some SFBs among proteins and then extend the SFB alignments to MSAs. The number of SFBs could grow exponentially with respect to the number of proteins, so these methods may have to examine a large number of SFBs in order to not miss the best MSA, which is usually computationally expensive (Wang et al., 2011). As such, the challenge facing a vertical-first method is to identify only those SFBs, which are very likely contained in the best MSA. Recently,
Ilinkin et al. (2010) have developed a consensus-first approach. Starting from an initial consensus structure, it iteratively pairwise aligns each input structure upon the consensus and updates the consensus based on the current MSA. However, the choice of the initial consensus structure would be biased to suboptimal result. Moreover, the pairwise alignment errors would accumulate to deteriorate the consensus structure.

4.1.4 Solutions for the current limitations

In contrary to the sequence alignment, any insignificant trial alignment for structures can be detected by structure superposition and then excluded. Thus, it is practicable to select from locally SFBs to build up a scaffold, and then upon the scaffold to construct the final multiple alignment. In particular, a scaffold, which is defined as one pivot structure plus several SFBs, contains more information than the initial consensus structure as applied in Ilinkin et al. (2010). By checking the spatial consistency for those SFPs between the pivot and the other structure, we could directly know whether this structure is anchored or not (see Fig. 5.6). For unanchored structures, we could realign them upon the pivot by the guidance of these SFBs, which could reduce the pairwise alignment errors. In conclusion, this approach could be regarded as scaffold-first, and the solutions for the current limitations are (i) using HSFB (highly similar fragment block) to solve the exponential nature of SFB with respect to the number of proteins, (ii) constructing scaffold from ranked HSFBs by spatial consistency to get a better initial start, (iii) aligning each proteins upon this scaffold by checking both horizontal consistency and vertical equivalency to reduce the pairwise alignment errors, and (iv) finally, refining the whole MSA by realigning each input structure to the consensus structure.

Figure 5.6 This picture shows the basic structure of a scaffold that contains one pivot structure plus several similar fragment blocks (SFBs). Given a superposition, we could check the spatial consistency (dark colored) for those similar fragment pairs (SFPs) between the pivot and the other structure to directly know whether this structure is anchored or not.
4.2. The current approaches for MSA

In this section, we introduce four protein MSA methods. They are MUSTANG (Konagurthu et al., 2006), CE-MC (Guda, Lu, Scheeff, Bourne, & Shindyalov, 2004), MASS (Dror et al., 2003), and MAPSCI (Ilkinin et al., 2010). For more tools, see Table 5.7.

4.2.1 MUSTANG (C-map, implicit-flexible, 2D, horizontal-first)

MUSTANG (Konagurthu et al., 2006) uses a progressive pairwise framework that derived from DALI (Holm & Sander, 1993) to build the final MSA. At the core of the method is a robust scoring scheme for pairwise alignments. This makes the use of a simple DP algorithm possible, instead of Monte Carlo, for all pairs of structures in the set, in order to gather accurate pairwise residue–residue equivalences. Before the final multiple alignment is constructed along a guide tree, a special extension phase is undertaken in which the pairwise scores of correspondences are recalculated in the context of all the remaining structures. This procedure significantly reduces the effect caused by making incorrect greedy choices while building the final alignment. The DALI-based scoring scheme lends it the ability to handle conformational flexibilities, such as hinge rotations.

4.2.2 CE-MC (C-alpha, rigid-body, 3D, horizontal-first)

CE-MC (Guda et al., 2004) uses CE (Shindyalov & Bourne, 1998) to perform all pairwise alignments. The algorithm refines a set of pairwise structure alignments using a Monte Carlo optimization technique. In particular, the algorithm iteratively modifies the MSA. The initializations are pairwise alignments of a pivot structure against all other structures, by a random set of moves. The random moves are then accepted with a probability proportional to the gain in the alignment score, which is calculated from the column distances in aligned blocks, using the form of reciprocal-like cRMS score (shown in Eq. 5.7). The iterative process is stopped when the optimal alignment cannot be improved by random moves.

4.2.3 MASS (SSE, rigid-body, 3D, vertical-first)

MASS (Dror et al., 2003) decomposes structures into SSEs. For each element, MASS treats them as vectors in 3D space and applies geometric hashing to calculate sequence order independent alignments. The SSE alignment defines an initial superimposition. Then, MASS extends this SSE alignment to a global optimal alignment by searching C-alpha atom positions close in space. The main advantage of MASS is the running speed since
Table 5.7  A selected list of multiple structure alignment tools

<table>
<thead>
<tr>
<th>Name</th>
<th>Comments</th>
<th>Represent</th>
<th>Perspective</th>
<th>Score</th>
<th>Merge method</th>
<th>Order&lt;sup&gt;a&lt;/sup&gt;</th>
<th>Available link</th>
<th>References</th>
</tr>
</thead>
</table>

*Continued*
Table 5.7 A selected list of multiple structure alignment tools—cont’d

<table>
<thead>
<tr>
<th>Name</th>
<th>Comments</th>
<th>Represent</th>
<th>Perspective</th>
<th>Score</th>
<th>Merge method</th>
<th>Order</th>
<th>Available link</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>3DCOMB</td>
<td>Local and contact pattern with machine learning trained</td>
<td>C-map and C-alpha</td>
<td>Rigid-body</td>
<td>2D and 3D</td>
<td>Scaffold-first</td>
<td>Y</td>
<td><a href="http://ttic.uchicago.edu/~jinbo/3DCOMB/3DCOMB_exe_V1.06.7z">http://ttic.uchicago.edu/~jinbo/3DCOMB/3DCOMB_exe_V1.06.7z</a></td>
<td>Wang et al. (2011)</td>
</tr>
</tbody>
</table>

*Y/N indicates that the algorithm has an option to let the alignment result be in sequential or in nonsequential order.*
it reduces the structural complexity to SSEs and applies geometric hashing. The second advantage of MASS is the capability of detecting the partial cores.

4.2.4 MAPSCI (C-alpha, rigid-body, 3D, consensus-first)

MAPSCI (Ilinkin et al., 2010) is a consensus-first approach that contains three major steps. To select the initial consensus structure by four different choices, to compute an initial MSA for all other structures against the consensus structure by a certain pairwise alignment tool, to iteratively get the optimal consensus structure as well as the rigid-body transformations based on the current MSA, and then to update each pairwise alignment between the consensus structure based on the current transformation by DP. MAPSCI considers four choices for initial consensus structure, they are the protein of median length, that minimizes the sum of the pairwise distances to all the other proteins, with the smallest maximum pairwise distance, and that generates the largest initial core.

4.3. 3DCOMB: The scaffold-first approach for MSA

4.3.1 BLOMAPS

3DCOMB (Wang et al., 2011) is evolved from BLOMAPS (Wang & Zheng, 2009), which is the first trial to use the scaffold-first approach for MSA. By means of the CLEs, BLOMAPS turns a structural fragment into a string, and two strings with their CLESUM score being higher than a preset threshold form a SFP. A string from one protein as a seed and its highly similar fragments from other proteins form a SFB. BLOMAPS consists of several steps including finding HSFBs guided by CLESUM, removing block redundancy, constructing scaffold by checking consistency in spatial arrangement among fragments from different blocks, dealing with unanchored structures, and the final step of refinement where the average template for alignment is obtained.

4.3.2 Overview of 3DCOMB

As shown in Fig. 5.7, 3DCOMB first generates a list of pivot structures. By default, this list contains all $M$ input proteins, so TopK is equal to $M$. For each pivot structure, 3DCOMB uses the CRF model to generate HSFBs, which are ranked by their spatial consistency scores and only the TopJ with the highest scores are extended to initial MSAs (i.e., the scaffold). 3DCOMB identifies those “unanchored” proteins which are not well aligned to the pivot. To improve an initial MSA, 3DCOMB conducts TopF trials to
realign each of the “unanchored” proteins to the pivot. Finally, 3DCOMB refines the whole MSA based on the consensus structure derived from the MSA.

### 4.3.3 Detecting HSFBs

Given two protein structures, we can calculate the marginal probability of two fragments being aligned using the forward–backward algorithm (Lafferty, McCallum, & Pereira, 2001). In brief, the marginal probability for each residue pair \((i, j)\) is calculated by a CRF model that uses four local features and two global features. The local features measure the ungapped segment pairs with length 9, 13, 17, 21 centering at \((i, j)\), while the global
features measure the environment fitness sphere with radius 8.0 and 14.0 Å that centers at \( \langle i, j \rangle \). This marginal probability is defined as the similarity score of two structure fragments. However, if we directly use CLESUM or (BLOSUM + CLESUM) (see Eq. 5.5 or Eq. 5.10) as the similarity score, we will not lose much accuracy but significantly increases the running speed. Given a short fragment \( F_1 \) in protein \( p_1 \), let \( F_i \) denotes the fragment of the same size in another protein \( p_i \) which has the highest similarity score with \( F_1 \). All the fragments \( F_1, F_2, \ldots, F_M \) form an HSFB with \( F_1 \) being the pivot fragment. A protein of size \( N \) in total have \( N-L+1 \) HSFBs. Note that the “highest similarity” relationship is asymmetric. That is, among all fragments in protein \( p_i \), \( F_i \) is the most similar one to \( F_1 \) may not imply that among all fragments of \( p_i \), \( F_1 \) also is the most similar one to \( F_i \). Therefore, given \( M \) proteins with lengths \( N_1, N_2, \ldots, N_M \), there are in total \( (\sum_{i=1}^{M} N_i) - ML + M \) HSFBs.

### 4.3.4 Ranking HSFBs by spatial consistency

Two HSFBs may be geometrically inconsistent with each other. That is, we cannot superimpose well the fragments in both HSFBs using a single set of rigid-body transformations. We can calculate the degree to which two HSFBs are geometrically consistent and then rank all the HSFBs according to their consistency with others. The HSFB with the highest consistency score is very likely contained in the best MSA. We use a simple method to estimate the consistency score of one HSFB as follows. For each fragment in the HSFB, we generate a rigid-body transformation (Kabsch, 1976) between this fragment and the pivot fragment. Let \( B_1 = \{ F_{11}, F_{12}, \ldots, F_{1M} \} \) denotes the HSFB for which we want to calculate its consistency score and \( F_{11} \) is the pivot fragment. Let \( T_i (i = 2, 3, \ldots, M) \) denotes the rigid-body transformations derived from superimposing \( F_{1i} \) to \( F_{11} \). Let \( B_2 = \{ F_{21}, F_{22}, \ldots, F_{2M} \} \) denotes another HSFB. For any fragment \( F_{2i} (i = 2, 3, \ldots, M) \) in \( B_2 \), we superimpose \( F_{2i} \) to \( F_{21} \) using the transformation \( T_i \) and then calculate the RMSD between \( F_{2i} \) and \( F_{21} \). If the distance is within 3 Å, we increase the consistency score of \( B_1 \) by 1, otherwise by 0.

### 4.3.5 Scoring function for MSA

A good MSA should have a large number of cores (i.e., CORE-LEN) and also a small core RMSD. A core is a fully aligned column, consisting of one residue from each input protein. In addition, pairwise alignments in an MSA should also be of high quality. It is challenging to develop an algorithm that can optimize these criteria simultaneously since sometimes they contradict
with one another. For example, a large CORE-LEN usually leads to a large RMSD value. A simple solution is to fix one criterion and then optimize the others, for example, maximizing CORE-LEN while restricting RMSD. This solution is not very flexible in that we have to determine RMSD in advance, not to mention that neither CORE-LEN nor RMSD is the best measure. We use \( \text{CORE-LEN} \times \text{TMscore} \) as the scoring function where TMscore of an MSA is defined as the average TMscore (see Eq. 5.7) of all the pairwise alignments implied in the MSA.

4.3.6 Building an initial MSA from an HSFB for scaffold

Given an HSFB of \( M \) structures, we first generate a set of \( M - 1 \) rigid-body transformations by superimposing each fragment in the HSFB to the pivot fragment and minimizing the RMSD of these two fragments. Then we superimpose each structure to the pivot structure using the transformation generated from fragment superimposition, and run DP to generate an alignment of the two structures by maximizing the TMscore. Finally, we assemble the \( M - 1 \) pairwise alignments into an initial MSA using the pivot structure as the anchor.

4.3.7 Adjustment of pairwise alignment

Given the initial MSA, we may refine it by adjusting the pairwise alignment between each input structure and the pivot structure. First, we calculate the TMscore of the pairwise alignment between each input structure and the pivot. In addition, we also calculate the percentage of spatial consistency SFPs between each input structure and the pivot. If TMscore < 0.5 (Xu & Zhang, 2010) or the percentage < 50%, this input structure is called “unanchored.” We adjust the alignment between each unanchored structure and the pivot using rigid-body transformations derived from other TopF HSFBs. In particular, for each top HSFB, let \( F_1 \) and \( F_2 \) denote the fragments in the HSFB belonging to the pivot and the unanchored structure, respectively. We realign the unanchored structure to the pivot structure using the rigid-body transformation generated from minimizing the RMSD between \( F_1 \) and \( F_2 \). The pairwise alignment with the maximum TMscore is kept in the MSA.

4.3.8 Consensus-based MSA refinement

3DCOMB refines an MSA by realigning each input structure to the consensus structure, which is constructed as follows. At each column of this MSA, we calculate the center of all the aligned residues (only C\( \alpha \) is considered).
Second, we merge two neighbor columns into a single one if the following two conditions are satisfied: (A) the total number of aligned residues in these two columns is not more than the total number of input structures and (B) the distance between their two centers is less than 3.0 Å. We use 3.0 Å as the cutoff because in native protein structures more than 99% of Cα–Cα virtual bonds are longer than 3.0 Å. This merge procedure is repeated until no columns can be merged. The consensus structure consists of all the centers. This refinement procedure is repeated until a given number of iterations or the scoring function cannot be improved further.

4.4. Result analysis

4.4.1 Overview

We evaluate our program 3DCOMB using three large-scale benchmarks on three metrics: CORE-LEN (number of fully aligned columns in the MSA), RMSD (root mean squared deviation of the fully aligned columns in the MSA), and TMscore (the average TMscore of all the pairwise alignments in the MSA). We compare 3DCOMB with several MSA tools such as BLOMAPS (Wang & Zheng, 2009), MAPSCI (Ilinkin et al., 2010), MAMMOTH (Lupyan et al., 2005), MUSTANG (Konagurthu et al., 2006), MATT (Menke et al., 2008), and MultiProt (Shatsky et al., 2004), which represent six different methods that include the three major merging methods.

4.4.2 The evaluation benchmarks

We use three benchmarks: HOMSTRAD (Mizuguchi, Deane, Blundell, & Overington, 1998), SABmark-sup, and SABmark-twi (Van Walle et al., 2005). HOMSTRAD contains 398 homologous protein families, each with at least three structures. SABmark-sup is the superfamily set in SABmark (version 1.65), containing 425 families with low to intermediate sequence identity. SABmark-twi represents the twilight set in SABmark, containing 209 families with low sequence identity.

4.4.3 Performance on HOMSTRAD

3DCOMB obtains the largest average CORE-LEN and the third best average core RMSD (slightly larger than MultiProt and MAPSCI). Note that because MultiProt uses a very strict cutoff to determine if an aligned column is a core or not, it always obtains the smallest core RMSD and also very small CORE-LEN on all the benchmarks. As shown in Table 5.8, MAMMOTH and MUSTANG can generate alignments with CORE-LEN comparable to
Table 5.8 Alignment accuracy of seven MSA tools on three benchmarks HOMSTRAD, SABmark-sup, and SABmark-twi

<table>
<thead>
<tr>
<th>Method</th>
<th>CORE-LEN</th>
<th>RMSD</th>
<th>TMscore</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HOMSTRAD</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3DCOMB</td>
<td>170.58</td>
<td>2.00</td>
<td>0.800</td>
</tr>
<tr>
<td>MAPSCI</td>
<td>162.55</td>
<td>1.87</td>
<td>0.792</td>
</tr>
<tr>
<td>MAMMOTH</td>
<td>169.84</td>
<td>3.03</td>
<td>0.786</td>
</tr>
<tr>
<td>MATT</td>
<td>169.53</td>
<td>2.00</td>
<td>0.781</td>
</tr>
<tr>
<td>BLOMAPS</td>
<td>169.27</td>
<td>2.18</td>
<td>0.779</td>
</tr>
<tr>
<td>MUSTANG</td>
<td>169.49</td>
<td>2.66</td>
<td>0.765</td>
</tr>
<tr>
<td>MultiProt</td>
<td>140.82</td>
<td>1.33</td>
<td>0.649</td>
</tr>
<tr>
<td><strong>SABmark-sup</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3DCOMB</td>
<td>106.66</td>
<td>2.59</td>
<td>0.655</td>
</tr>
<tr>
<td>MAPSCI</td>
<td>89.51</td>
<td>2.95</td>
<td>0.627</td>
</tr>
<tr>
<td>MAMMOTH</td>
<td>105.50</td>
<td>5.78</td>
<td>0.614</td>
</tr>
<tr>
<td>MATT</td>
<td>104.12</td>
<td>2.59</td>
<td>0.613</td>
</tr>
<tr>
<td>BLOMAPS</td>
<td>101.82</td>
<td>3.11</td>
<td>0.613</td>
</tr>
<tr>
<td>MUSTANG</td>
<td>103.86</td>
<td>4.20</td>
<td>0.583</td>
</tr>
<tr>
<td>MultiProt</td>
<td>68.70</td>
<td>1.61</td>
<td>0.404</td>
</tr>
<tr>
<td><strong>SABmark-twi</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3DCOMB</td>
<td>71.63</td>
<td>3.02</td>
<td>0.526</td>
</tr>
<tr>
<td>MAPSCI</td>
<td>50.11</td>
<td>4.38</td>
<td>0.466</td>
</tr>
<tr>
<td>MAMMOTH</td>
<td>64.97</td>
<td>8.31</td>
<td>0.436</td>
</tr>
<tr>
<td>MATT</td>
<td>67.08</td>
<td>2.89</td>
<td>0.453</td>
</tr>
<tr>
<td>BLOMAPS</td>
<td>67.20</td>
<td>4.22</td>
<td>0.457</td>
</tr>
<tr>
<td>MUSTANG</td>
<td>66.89</td>
<td>5.10</td>
<td>0.422</td>
</tr>
<tr>
<td>MultiProt</td>
<td>36.38</td>
<td>1.75</td>
<td>0.259</td>
</tr>
</tbody>
</table>

The accuracy is measured by CORE-LEN, RMSD, and TMscore. The values in the table are averaged over an individual benchmark.
3DCOMB, MATT, and BLOMAPS, but much larger RMSD. 3DCOMB not only achieves the best average TMscore but also excels others on almost each individual structure group.

### 4.4.4 Performance on SABmark

As shown in Table 5.8, 3DCOMB obtains the best average CORE-LEN and TMscore for SABmark-sup. By RMSD, 3DCOMB is second to only MultiProt, but MultiProt obtains a much smaller CORE-LEN. By TMscore, 3DCOMB is about 4.5% better than the second best method MAPSCI and also outperforms others on almost each individual structure group. For SABmark-twi, it is more challenging because it consists of mostly distantly related proteins. However, 3DCOMB excels others at an even larger margin. By CORE-LEN, 3DCOMB is 6.6% better than the second best method MATT and excels others on a majority of structure groups. By core RMSD, 3DCOMB is second to MultiProt and slightly to MATT. By TMscore 3DCOMB outperforms the second best algorithm MAPSCI by 13% and also excels others on almost each individual structure group.

### 5. CONCLUSIONS AND PERSPECTIVES

Using secondary structures to describe protein structures has its own limitations. Therefore, protein structural alphabets, which are discretized states of backbone fragment conformations, have been proposed to bridge the gap between the secondary and tertiary structures. Among all the work structural alphabets, our CLEs aptly balance precision with simplicity by a probabilistic perspective (Zheng, 2008). In order to measure the similarity between these discrete letters, a substitution matrix CLESUM is developed by using a similar way as BLOSUM. Based on the newly developed structural alphabets, we invented a protein pairwise structure alignment program (DeepAlign). The scoring function (DeepScore) of our program contains the following key factors: (i) AA mutation score, (ii) local substructure substitution potential, (iii) hydrogen-bonding similarity, and (iv) traditional geometric similarity measure. DeepAlign could generate alignments highly consistent with manually curated alignments. We use manually curated alignments as the reference because they usually make much more biological sense since they are built by human experts taking into consideration evolutionary and functional relationship besides spatial proximity alone. Note that DeepScore (see Eq. 5.8) is the natural combination of four different
items and there are no parameters to be fine tuned. Therefore, we do not bias DeepAlign toward a specific performance metric.

It is obvious that MSA will bring more information than pairwise alignment alone. However, the limitation for MSA lies in how to merge the input structures into a MSA. Here, we introduce our method 3DCOMB that applies a novel approach, namely, scaffold-first, to go beyond the current limitations. By using a probabilistic model to combine both local and global structure information, we can accurately identify the most conserved short fragment blocks among proteins to be aligned. These conserved fragment blocks plus a pivot protein form the scaffold, which are very likely contained in the best MSA. Then by pairwise adjusting other proteins with the pivot, 3DCOMB can quickly get the best MSA through a consensus-first approach. 3DCOMB also introduces a novel scoring function to generate an MSA with a large number of cores and high-quality pairwise alignments.

Structure alignment tools have a wide spectrum of applications, ranging from constructing classification database for protein structures (Holm & Sander, 1994; Murzin et al., 1995; Orengo et al., 1997) to protein function annotation (Brylinski & Skolnick, 2008; Roy et al., 2012). Recently, DeepAlign and 3DCOMB have been utilized in numerous different applications, such as (i) constructing the training data for template-based protein structure prediction tools that based on Machine Learning method (Källberg et al., 2012; Ma et al., 2012; Ma, Peng, Wang, & Xu, 2013; Ma, Wang, et al., 2013; Peng, Bo, & Xu, 2009; Shao et al., 2011), for position-specific distance-dependent protein statistical potential (Zhao & Xu, 2012), and for protein residue contact prediction (Wang & Xu, 2013); (ii) searching the ligand database for homology-based binding-site prediction (Källberg et al., 2012); and (iii) building MSA for multiple-template-based threading (Peng & Xu, 2011). These new areas of DeepAlign and 3DCOMB have applied broaden the spectrum of the application list of structure alignment tools.

Despite success, the following open problems for structure alignment need to be solved in the future.

How to deal with the inconsistency problem. Sadowski and Taylorr (2012) have found that even for relatively similar proteins the degree of inconsistency level is high among the structure alignments generated by different tools. One source of inconsistency is found to be around the region near gaps and for proteins with low structural complexity, as well as for helices. Another source of inconsistency is found in those proteins containing periodic or repetitive structures. One possible solution to deal with the periodic or repetitive case is to allow multiple solutions (see Fig. 5.5).
How to process gaps in both pairwise and multiple alignment. As shown in Holm and Sander (1993), an AFP shall not be too short (say less than four residues). Also, as mentioned in Sadowski and Taylor (2012), since indels are responsible for much of the structural change which occurs through evolution, it is necessary to develop a more accurate gap score for structural alignment. Moreover, how to deal with the exponentially growth number of gaps with respect to the number of proteins in MSA remains unsolved.

How to conduct fast and reliable database search. Although a variety of tools have been developed, such as 3D-BALST (Tung et al., 2007) and YAKUSA (Carpentier et al., 2005), a database search approach that could deal with large scale of distant structures is deficiency because of the combinatorial explosion for search space (Hasegawa & Holm, 2009). The ultimate goal is to project an entire protein structure to a fingerprint, that is, one number or histogram. Recently, Harder, Borg, Boomsma, Røgen, and Hamelryck (2012) have applied Gauss Integrals calculated from protein backbones as the protein global descriptor for fast large-scale clustering of protein structures. This methodology might be able to be applied for fast and reliable database search in the future.

How to improve flexible alignment. Although structure comparison methods that allow flexibility and plasticity could generate the most biologically meaningful alignments (Hasegawa & Holm, 2009), it is well known that introducing flexibility into alignments greatly enlarges the searching space, exacerbating problems in generating high-quality alignments for more distantly related proteins (Sadowski & Taylor, 2012). It is also suggested that, instead of allowing flexible “everywhere,” we should focus on those functional and/or hinge regions (Sadowski & Taylor, 2012). As a choice, RAPIDO (Mosca et al., 2008), that only lets the region between two domains to be flexible, could serve as a candidate for the purpose. Moreover, Elastic Network Model might be applied to define those highly flexible hinge regions, such as used in Emekli, Schneidman-Duhovny, Wolfson, Nussinov, and Haliloglu (2008) and Yang, Song, and Jernigan (2009).

How to conduct complex–complex alignment. Sippl and Wiederstein (2012) pointed out that the comparison of oligomers and molecular complexes differs largely. They also provide several challenging examples that cannot be handled by conventional structure alignment techniques since multiple chains are involved. To deal with this problem, TopMatch (Slater, Castellanos, Sippl, & Melo, 2013) has been developed. The basic idea of TopMatch is that, in a pairwise comparison of two structures, it first concatenates all chains together and then generally yields several basic alignments.
For large proteins and complexes, the number of basic alignments may rise to several hundred. Finally, TopMatch merges these basic alignments, according to a spatial consistency measurement, to form a composite alignment. It should be noted that, in the alignment of multiple chains, the relative order of monomers is arbitrary. This property made TopMatch possible to deal with the complex alignment without considering the chain ordering. However, due to the increase number of large complex protein structures in PDB, the alignment tools with more efficiency and more reliability are still needed in the future.

*How to raise the speed and accuracy for large-scale MSA.* As mentioned early in Koehl (2006), some of the most accurate MSA methods are still computationally prohibitive to be applied in large scale and continuous experiments. This statement is also true for 3DCOMB. Although has improved accuracy, the running speed of 3DCOMB for large-scale input protein is still far from satisfactory.

**REFERENCES**


