5-5-2012

Fundamentals of Protein Structure Alignment

Allen Holder
Rose-Hulman Institute of Technology, holder@rose-hulman.edu

Mark Brandt
Rose-Hulman Institute of Technology, brandt@rose-hulman.edu

Yosi Shibberu
Rose-Hulman Institute of Technology, shibberu@rose-hulman.edu

1 Fundamentals of Protein Structure Alignment

MARK BRANDT[a], ALLEN HOLDER[b], and YOSI SHIBBERU[b]
Rose-Hulman Institute of Technology
[a] Department of Chemistry and Biochemistry
[b] Department of Mathematics

1.1 INTRODUCTION

The central dogma of molecular biology asserts a one way transfer of information from a cell's genetic code to the expression of proteins. Proteins are the functional workhorses of a cell, and studying these molecules is at the foundation of much of computational biology. Our goal here is to present a succinct introduction to the biological, mathematical, and computational aspects of making pairwise comparisons between protein structures. The presentation is intended to be useful for those who are entering this research area. The chapter begins with a brief introduction to the biology of protein comparison, which is followed by a brief taxonomy of the different mathematical frameworks for protein structure alignment. We conclude with a couple of recent pairwise comparison techniques that are at the forefront of efficiency and accuracy. Such methods are becoming important as structural databases grow.

1.2 BIOLOGICAL MOTIVATION OF PROTEIN STRUCTURE ALIGNMENT

Proteins are crucially important molecules that are responsible for a large variety of biological functions required for life to exist. The DNA sequence of the genome provides a one-dimensional descriptive code; proteins are self-organizing systems that allow expansion of this one-dimensional code into complex three-dimensional structures possessing a great diversity of functions. Understanding the types of protein structures that are possible is an important part of understanding the existing biological systems, of understanding the aberrant processes that result in genetic disorders, and in the engineering of proteins with novel functions.

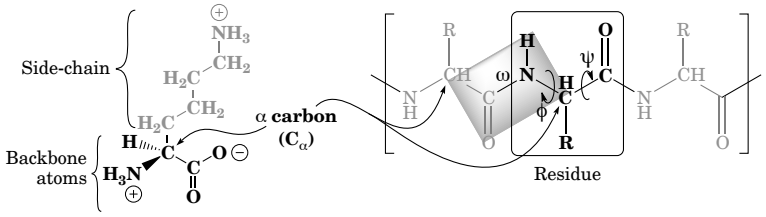


Fig. 1.1 The structure of one type of amino acid (lysine) is shown on the left, with the side-chain and backbone atoms indicated. Proteins are comprised of amino acid residues, where the backbone atoms are linked together to form the chain, and the side-chains determine the folded structure and much of the function of the protein. In the partial protein shown on the right, the side-chains are abbreviated as "R", and the three types of dihedral angles (the specific angle formed by the atoms bonded at each position along the backbone) are shown. The ϕ and ψ angles may vary, within geometric limits imposed by the surrounding atoms; the ω angle is fixed, with the six atoms forming the planar structure shown.

Proteins are synthesized as linear polymers of amino acids; the vast majority of proteins are comprised of a set of twenty different types of amino acids, in defined sequences that, depending on the protein, vary from about 50 to more than 28,000 amino acid residues. Although there are exceptions, in general, the specific sequence of amino acids is specified by the genome; this linear sequence determines the three dimensional structure of the protein.

The three-dimensional structure of a protein is an emergent property of the linear sequence. Predicting the three dimensional structure based entirely on the linear sequence has proven to be challenging because the defined structure exhibited by most proteins is a consequence of a large number of relatively weak interactions. Existence of a defined structure is possible because of geometric constraints imposed by the backbone atoms, and because of geometry dependent hydrogen bonding, electrostatic, and non-polar interactions between the atoms of the backbone and side-chains.

Prior to the experimental determination of the first protein structures, Linus Pauling predicted the formation of regular repeating structures (especially α-helices and β-sheets), based on a theoretical understanding of the geometric constraints inherent in the backbone structure. The increasing number of experimentally determined protein structures has confirmed that most proteins are comprised of arrangements of α-helices and β-sheets, along with regions of less well-defined structural elements. Because proteins are such large molecules, and because the secondary structural elements are important parts of the overall structure, most proteins are represented in ways that emphasize the arrangement of secondary structural elements.

Analysis of protein structures has revealed that many proteins are comprised of one or more separate domains, which are regions within the protein that fold independently of the remainder of the protein. Protein domains are currently considered

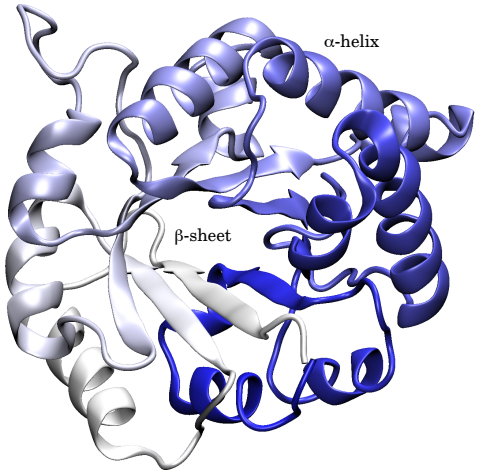


Fig. 1.2 An α-carbon trace emphasizing the secondary structure of one monomer of the enzyme triose phosphate isomerase (from PDB ID 2YPI). This protein folds into an α/β barrel structure, with a multistrand β-sheet (the thick arrows near the center of the structure) surrounded by α-helical elements. A number of other proteins exhibit this α/β barrel structure, in spite of considerable difference in both their sequence and their function.

to be units of evolution: one major constraint on tolerated mutations is the result of the requirement to maintain the folded structure of the domain.

Comparison of different proteins is crucial to an understanding of the relationships between protein amino acid sequences, protein structure, and protein function. Protein sequence information can be obtained from the genome sequencing projects, and protein sequences can be readily compared. However, many proteins are known to have limited sequence similarity and yet have structures that visually appear similar. This raises the question of how to compare complex three-dimensional structures both quantitatively and in ways that will allow a better understanding of the relationships between their structure and their function.

Many proteins exhibit generally similar structures and similar functions. For example, a considerable number of serine protease enzymes have been discovered from species as widely divergent as mammals and bacteria. Although the two proteins shown in Figure 1.3 are comprised of very different amino acid sequences, portions of the structure match rather closely. However, in analyzing the structures, it is less clear how important the structural differences are in the subtle differences in function between these proteins. In addition, it is less clear how best to represent the structural differences in a manner that is both consistent and informative.

Another example of the importance of structure is provided by the prion proteins. Prions are monomeric proteins normally found on the surface of a variety of cells; however, these proteins are capable of undergoing an incompletely understood con-

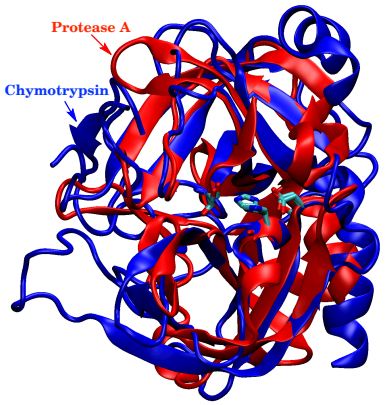


Fig. 1.3 An overlay of two proteins with similar function, Protease A from the bacterium *Streptomyces griseus* (PDB ID 3SGA) and *Bos taurus* chymotrypsin (PDB ID 1YPH). The two proteins share limited sequence identity (∼20%), but similar catalytic mechanisms, and considerable structural similarity especially in the core of the protein. In contrast, another protein, subtilisin from *Bacillus amyloliquefaciens*, also exhibits a similar catalytic mechanism but has a very different structure.

formational change that results in oligomerization, with lethal effects to the affected individual. The spongiform encephalopathy diseases are one of a significant number of diseases caused by protein misfolding. An improved understanding of protein structure and protein folding processes might allow intervention in disease processes that are currently untreatable. In addition, many genetic disorders result from altered protein structure and function. While it is apparent that the changes in one of a small number of residues within a large protein cause disease, only a better understanding of the elaboration of sequence information into an overall structure will allow insight into possible approaches for treatment.

A final purpose of studying protein structure is to allow the design of novel proteins. Enzymes are phenomenal catalysts, which generally exhibit both high reaction rates and high levels of specificity. While biological enzymes catalyze a large range of reactions, no enzymes exist to catalyze many industrially useful reactions. The ability to design new enzyme mechanisms is extremely attractive as a method for carrying out reactions at higher rates, with less expense from heating costs and waste product formation. Current methods for protein design are inefficient and essentially entirely empirical and are largely limited to minor alterations to existing proteins.

We have an increasing database of protein structures; however, we still lack a full understanding of how protein sequence, structure, and function are related. Comparing the structures of existing proteins of different sequences provides important data that will lead to an improved understanding of the mechanisms by which existing protein sequences give rise to their corresponding functional protein structures.

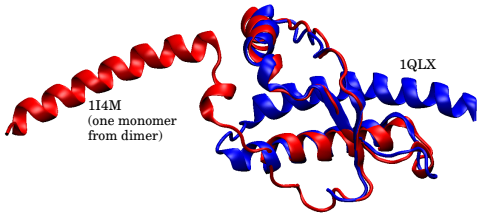


Fig. 1.4 The same protein in two different conformations: a fragment from the human prion protein. The 1I4M structure is part of a dimer, which may represent a stage in the structural transformation from the largely helical protein shown here to the toxic β-sheet conformation thought to cause the lethal spongiform encephalopathy diseases such as Creutzfeldt-Jakob disease and kuru.

1.3 MATHEMATICAL FRAMEWORKS

The two main mathematical frameworks studied in the protein alignment literature are the contact map overlap (CMO) problem and the largest common point set (LCP) problem under bottleneck distance constraint [17]. Figure 1.5 illustrates a two-dimensional version of each framework.

Let $A = \{a_1, a_2, \ldots, a_m\}$ and $B = \{b_1, b_2, \ldots, b_n\}$ be the sets of C_α atom coordinates of two proteins, protein A and protein B, that we wish to align. (These are the carbon atoms in Figure 1.1 that bind to the side chains.) For a given distance cutoff value $\kappa > 0$, define $E_A = \{(a_i, a_j) : \|a_i - a_j\| \leq \kappa \text{ for } i < j\}$ and $E_B = \{(b_i, b_j) : \|b_i - b_j\| \leq \kappa \text{ for } i < j\}$ to be the sets of edges in the contact graphs of proteins A and B respectively. (Edges are represented by arcs in Figure 1.5.) Define $\Pi : A' \to B'$ to be a bijection (one-to-one, onto map) from a subset $A' \subset A$ to a subset $B' \subset B$. Define $T : B \to B'$ to be a rigid body transformation of the fold of protein B. (In Figure 1.5, T is simply a translation of fold B onto fold A.)

CMO Problem Determine the bijection $\Pi : A' \to B'$ that maximizes the size of the matched subsets of edges, $E'_A \subset E_A$ and $E'_B \subset E_B$ where an edge $(a_i, a_j) \in E_A$ is considered a match if $(\Pi(a_i), \Pi(a_j))$ is an edge in E_B.

LCP Problem For all rigid body transformations $T : B \to B'$, determine the largest subset $A^* \subset A$ for which there exists a bijection $\Pi : A^* \to B$ such that $\|a_i - T(\Pi(a_i))\| \leq \kappa$ for all $a_i \in A^*$.

Goldman et al. [9] proved that the CMO problem is NP-hard. Caprara et al. [6] apply linear integer programming methods to obtain exact solutions to the CMO problem for proteins that are similar to one another. Their methods have been improved significantly by Andonov et al. [1], see Section 1.4. By imposing a proximity requirement for aligned residues and employing packing constraints satisfied by actual protein folds, Xu et al. [25] and Li et al. [17] have developed polynomial-time

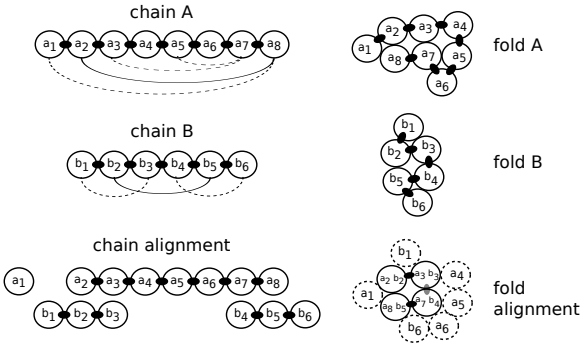


Fig. 1.5 A two-dimensional depiction of protein alignment. Chain A collapses into fold A creating long range contact depicted by arcs on chain A. Likewise, chain B collapses into fold B. Then a rigid body transformation, in this case a vertical translation, superimposes folds A and B to create a fold alignment. In the CMO problem, the arcs on chain A and B are aligned directly (solid arcs) without reference to the superimposed folds. (The consecutive chain contacts are not considered.) In the LCP problem, the superimposed folds determine the chain alignment.

approximation schemes for the CMO problem. The CMO problem is discussed further in Section 1.4.

The LCP problem appears to be easier to solve than the CMO problem. The LCP problem is more geometric in character where as the CMO problem is graph-theoretic in nature. A possible disadvantage of the LCP problem, however, is that the problem treats proteins as rigid objects. In reality proteins are quite flexible. Hasegawa and Holm [10] claim that alignment methods that allow for flexibility give the most biologically meaningful results.

The main ideas used to develop a polynomial-time algorithm for the LCP problem are due to Kolodny and Linial [13] and Poleksic [19, 20]. We describe next a polynomial-time algorithm for solving the LCP problem. The basic ideas were developed by Kolodny and Linial [13] and extended by Poleksic [20]. Although the analysis in [13] and [20] is for the three-dimensional alignment problem, here, for simplicity of exposition, we describe the algorithm for the two-dimensional alignment problem.

In two-dimensions, we can parametrize rigid body transformations by $r = (\theta, x, y)$ as follows:

$$T_r(b_i) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} b_i + \begin{pmatrix} x \\ y \end{pmatrix}$$

where b_i is the coordinates of a C_α atom in protein fold B. For a prescribed distance cutoff value, $\kappa > 0$, and for a given rigid body transformation $T_r(b)$, define A_r to be

the largest subset of A for which there exists a bijection $\Pi_r : A_r \rightarrow B$ such that

$$\|a_i - T_r(\Pi(a_i))\| \leq \kappa \text{ for all } a_i \in A_r.$$

The set A_r and the bijection Π_r can easily be computed in $O(m\,n)$ time by applying dynamic programming, see Section 1.4 to the score matrix $[C]_{ij}$ where

$$C_{ij} = \begin{cases} 1 & \text{if } \|a_i - T_r(b_j)\| \leq \kappa \\ 0 & \text{otherwise.} \end{cases}$$

The solution to the LCP problem is then given by $A_r^* = \max_r\{A_r\}$.

A key observation of Kolodny and Linial [13] is that only a finite set of rigid body transformations need to be considered in order to optimize commonly used alignment scoring functions. To demonstrate this, first observe that only the compact subset, $R = \{r : |\theta| \leq \pi, |x| \leq \gamma, |y| \leq \gamma\}$ of rigid body transformations needs to be considered because if protein B is translated by $|x| > \gamma$ or $|y| > \gamma$, where γ is sufficiently large, A_r will be the empty set since proteins A and B will have no points in common. Since $T_r(b)$ is continuous and R is compact, $T_r(b)$ is uniformly continuous on R. Uniform continuity implies there exists a $\delta_\epsilon > 0$ such that for any $r_1 \in R$ and $r_2 \in R$ satisfying $\|r_1 - r_2\| < \delta_\epsilon$ we have that $\|T_{r_1}(b_i) - T_{r_2}(b_i)\| < \epsilon$ for all $b_i \in B$. Now, consider the open balls $B(r_o, \delta_\epsilon) = \{r : \|r - r_o\| < \delta_\epsilon\}$ where $r_o \in R$. The open balls $B(r_o, \delta_\epsilon)$ cover R, i.e. $R \subset \cup_{r_0 \in R} B(r_o, \delta_\epsilon)$. Since R is compact, a finite subset of these open balls also covers R, i.e. there exit $r_i \in R$ for $i = 1, 2, \ldots, N$ such that $R \subset \cup_{i=1}^N B(r_i, \delta_\epsilon)$. Thus, all the rigid body transformations in the compact set R can be approximated to within a distance of ϵ by the finite set of rigid body transformations $\{r_1, r_2, \ldots, r_N\}$.

The alignment scoring functions considered by Kolodny and Linial [13] must satisfy a Lipschitz condition. Their algorithm only computes an ϵ approximation of the optimal solution. Poleksic [20] extended Kolodny and Linial's approach to the LCP problem. Moreover, Polesic's extension computes the exact solution to the LCP problem in expected polynomial time $O(n^{11})$ for globular proteins of size n [19, 20]. We describe Poleksic's algorithm next.

Let $r_* \in \text{argmax}_{r \in R}\{A_r\}$. Also, for $\kappa > 0$, define the function $S_r(\kappa) = |A_r|$, where $|A_r|$ is the number of elements in A_r. Since $r_* \in R \subset \cup_{i=1}^N B(r_i, \delta_\epsilon)$, there exists an i_* such that $r_* \in B(r_{i_*}, \delta_\epsilon)$, which implies

$$\|T_{r_*}(b_j) - T_{r_{i_*}}(b_j)\| < \epsilon \text{ for all } b_j \in B.$$

From the above inequality, we can conclude that

$$S_{r_*}(\kappa) \leq S_{r_{i_*}}(\kappa + \epsilon)$$

because if a_i and $T_{r_*}(b_j)$ are in contact for cutoff κ, then a_i and $T_{r_{i_*}}(b_j)$ remain in contact for cutoff $\kappa + \epsilon$. We also have that

$$S_{r_{i_*}}(\kappa - \epsilon) \leq S_{r_{i_*}}(\kappa) \leq S_{r_*}(\kappa).$$

Combining the above inequalities, we have that

$$S_{r_{i_*}}(\kappa - \epsilon) \leq S_{r_*}(\kappa) \leq S_{r_{i_*}}(\kappa + \epsilon).$$

The LCP problem can be solved exactly in a finite number of steps if we can determine an $\epsilon > 0$ for which

$$S_{r_{i_*}}(\kappa - \epsilon) = S_{r_{i_*}}(\kappa + \epsilon)$$

since this would imply that $S_{r_*}(\kappa) = S_{r_{i_*}}(\kappa - \epsilon) = S_{r_{i_*}}(\kappa + \epsilon)$. Poleskic [20] showed that such an ϵ exists for all but a finite set of cutoff values $\kappa_1, \kappa_2, \ldots, \kappa_{|A|}$ as follows. The function $S_r(\kappa) = |A_r|$ is a nondecreasing function of κ having integer values in the range 0 to $|A|$. The function $S_r(\kappa)$ is therefore piecewise constant except for possible jump discontinuities at cutoff values $\kappa_1, \kappa_2, \ldots, \kappa_{|A|}$. If we avoid this finite set of cutoff values, then we can choose an $\epsilon > 0$ such that $S_{r_{i_*}}(\kappa - \epsilon) = S_{r_{i_*}}(\kappa + \epsilon)$. The size of the finite set of rigid body transformations $T_{r_i}(b)$ that we need to search to determine i_* is determined by how small ϵ is. The size of ϵ is determined by how close κ is to one of the jump discontinuity points κ_i, $i = 1, 2, \ldots, |A|$. These jump discontinuity points are not known in advance. However, Poleskic [20] provides a proof that the overall expected complexity of the algorithm is polynomial in time.

Spectral Methods

Spectral methods have emerged as a new mathematical framework for the protein structure alignment problem. An advantage of spectral methods over CMO methods pointed out by Bhattacharya et al. [4] is that comparisons with spectral methods are based on two residues (one from each protein) rather than the four residues (two from each protein to define an edge) required by CMO methods. Another advantage over CMO pointed out by Lena et al. [15] is that unlike CMO, spectral methods scale well with the size of the distance cutoff parameter κ.

In 1988, Umeyama [23] published a spectral, polynomial-time algorithm for computing a bijection between two weighted graphs that are isomorphic. The adjacency matrix of each graph is assume to have distinct eigenvalues. Umeyama's algorithm also works well if the graphs are nearly isomorphic. We describe the details of Umeyama's algorithm next.

Let C_A and C_B be the adjacency matrices of two undirected, weighted graphs with the same number of nodes and distinct eigenvalues. The goal is to determine a permutation matrix Ω which minimizes $\|\Omega C_A \Omega^\top - C_B\|$. Let $C_A = U_A D_A U_A^\top$ and $C_B = U_B D_B U_B^\top$ be the eigensystem decomposition of C_A and C_B. It is possible to prove that

$$\|C_A - C_B\|^2 \geq \sum_{i=1}^{n} (\lambda_i^A - \lambda_i^B),$$

where λ_i^A and λ_i^B, $i = 1, 2, \ldots, n$, are the ordered eigenvalues of C_A and C_B. Umeyama showed that there exists an orthogonal matrix $U^* = U_B D U_A^\top$ for some

diagonal matrix D, where D has diagonal elements -1 or 1, for which

$$\|U^* C_A (U^*)^\top - C_B\| = \min_U \|U C_A U^\top - C_B\| = \sum_{i=1}^{n} (\lambda_i^A - \lambda_i^B).$$

For isomorphic graphs, Umeyama proved that U^* is a permutation matrix Ω^*. Moreover, Ω^* can be computed in polynomial-time by solving the assignment problem

$$\Omega^* = \operatorname*{argmax}_\Omega \text{ trace } \left(\Omega |U_A| |U_B^\top|\right)$$

using the Hungarian algorithm. (The entries of the matrices $|U_A|$ and $|U_B|$ are the absolute values of the entries in U_A and U_B.) Umeyama also observed that the optimal solution can often be obtained for non-isomorphic graphs by using the solution to the above assignment problem as an initial guess and then applying a local optimization algorithm.

Umeyama's algorithm for the graph isomorphism problem does not apply directly to the protein structure alignment problem, which is a subgraph isomorphism problem. In other words, Umeyama's algorithm can not handle insertions and deletions. In addition, the weighted graphs of protein structures may not be similar to one another, a requirement for Umeyama's algorithm to work reliably.

Bhattacharya et al. [4] attempt to overcome the limitations of Umeyama's algorithm by considering local alignments to identify similar neighborhoods of the same size and then piecing these neighborhoods together. Rather than apply Umeyama's algorithm directly, Bhattacharya et al. normalize each eigenvector by the size of the protein and then compare the eigenvectors without taking absolute values. (Like Umeyama, they do not use the eigenvalues in their algorithm.) A complication arising in the approach used by Bhattacharya et al. is the fact that the eigenvector decomposition of an adjacency matrix does not specify an orientation of the eigenvectors. In other words, if v_i is an eigenvector, so is $-v_i$. This requires 2^k eigenvector orientations to be checked, where k equals the number of eigenvectors compared. The time complexity of the resulting algorithm, called Matchprot, is $O(2^k \max\{m^3, n^3\})$. Bhattacharya et al. point out that empirical observations suggest that $k = 3$ eigenvectors is sufficient for good results. However, Matchprot has difficulty with alignments involving large insertions and deletions.

Lena et al. [15] introduced a spectral algorithm called Al-Eigen. Unlike Matchprot, Al-Eigen uses a global alignment. Al-Eigen scales eigenvectors by the square root of the corresponding eigenvalues. Like Matchprot, Al-Eigen searches through the 2^k orientations of the k eigenvectors it compares, starting with comparing just one eigenvector from each protein, up to t eigenvectors. The complexity of Al-Eigen is $O(2^{t+1} m\, n)$.

In this section we have tersely reviewed the primary mathematical models associated with optimally aligning protein structures. The intent of the CMO, the LCP, and the spectral frameworks is to discern biologically relevant alignments between two proteins. Each paradigm has advantages and disadvantages, and continued research is important. The algorithmic complexity and resulting solution times were substantial

enough that, until recently, undertaking the task of completing numerous pairwise comparisons was a weighty computational burden. However, a host of modern algorithms is emerging that hasten the comparison procedure. These are discussed in the next section.

1.4 RECENT ADVANCES WITH DATABASE QUERIES

Protein databases contain tens of thousands of structures and continue to grow. One of the research fronts in computational biology is the design of algorithms that can efficiently search and organize these databases. For example, a researcher may want to find those proteins whose structure is similar to one under investigation, or he or she may want to navigate a database by functionally similar proteins. Such queries require comparisons against an entire database. As protein databases grow, undertaking these numerous comparisons requires efficient and accurate comparison algorithms.

We review three methods that are designed for making efficient structural comparisons: 1) a geometrical approach that encodes each residue with angle and distance information called GlObal Structure SuperposItion of Proteins (GOSSIP) [12], 2) a spectral approach called EIGAs that assigns each residue an eigenvalue associated with a high-dimensional feature [22], and 3) a solution procedure to the CMO problem called A_purva [1]. While these methods represent protein chains differently, they share the common algorithmic solution procedure of dynamic programming (DP). The emerging literature on protein structure alignment points to the important role that DP is fulfilling as an efficient algorithmic framework. Our specific goal here is to highlight this observation and to develop the use of DP in each of these alignment procedures.

Dynamic programming was invented by Bellman in the 1940s [3], and its use in biological applications began in 1970 [18]. The discrete version we consider calculates an optimal match between two sequences, say $r_1, r_2, \ldots, r_n$ and $r'_1, r'_2, \ldots, r'_m$. The algorithm requires that each possible match be scored, and we let $S_{ij} = S(r_i, r'_j)$ be the score associated with matching element i of the first sequence with element j of the second. Dynamic programming follows a recursion to calculate an optimal match between the two sequences:

$$V(i,j) = \text{opt} \begin{cases} V(i-1,k) + \rho \\ V(i,j-1) + \rho \\ V(i-1,j-1) + S(r_i, r'_j), \end{cases} \tag{1.1}$$

where ρ is a gap penalty. The gap penalty can depend on if a gap is being initiated or continued; in the former the penalty is called a gap opening penalty and in the latter it's called a gap extension penalty. Completing this recursion over i and j calculates the optimal value of the matching, and backtracing the optimal iterations lists the optimal matching. The optimal match depends not only on the scoring matrix S but also on the penalties to open and continue a gap, and hence, the use

of DP to optimally align sequences requires parameter tuning. The computational complexity of calculating an optimal matching is $O(mn)$, showing that DP is an efficient polynomial algorithm.

The coordinates of the C_α atoms of each residue are used to describe the protein in each of the techniques presented below. Due to the lack of an absolute coordinate system, a protein is commonly abstracted into pairwise comparisons between C_α atoms, which provides a coordinate and rotation free description of the protein. Each of A_purva, GOSSIP, and EIGAs assigns different information to each C_α atom, and hence, each imposes a different pairwise relationship between residues. In what follows, we briefly describe each technique so as to highlight the similarities and dissimilarities between the algorithms. In particular, we show how to construct the scoring matrices used for DP so that each method is seen as an application of DP applied to sequence similarity.

1.4.1 GlObal Structure SuperposItion of Proteins

The algorithm GlObal Structure SuperposItion of Proteins (GOSSIP) uses DP with a scoring matrix created by local three-dimensional geometry. A residue is encoded with 8 characteristics that depend on a parameter q that defines the local geometry. The characteristics for residue r_i depend on the polygon created by residues r_i, r_{i+2}, r_{q-2}, and r_q. Five of the characteristics are distances, and we use $d(r_i, r_j)$ to denoted the Euclidean distance between the C_α atoms of residues i and j. The characteristics are

Char.	1	2	3	4
	$d(r_i, r_{i+2})$	$d(r_{i+2}, d_{i+q-2})$	$d(r_{i+q-2}, r_{i+q})$	$d(r_{i+q}, r_i)$
Char.	5	6	7	8
	$d(r_i, r_{i+q-2})$	θ_i	i	$n-i+1$

The angle θ_i is created by the line segments (r_i, r_{i+q}) and (r_i, C), where C is the centroid of the protein structure and n is the number of residues in the protein chain. See Figure 1.6 for an example. All but the last $n-q$ residues are encoded.

The score assigned to matching residue i of the first protein to residue j of the second is

$$S_{ij} = 3 - 2|d(r_i, r_{i+q}) - d(r_j, r_{i+q})| - 0.1|\theta_i - \theta_j|,$$

where the coefficients 2 and 0.1 are experimentally decided. DP is applied to the scoring matrix $[S]_{ij}$, but with a few adaptations. First, an indicator function $\delta(i, j)$ is used to decide if the characteristics of residues i and j are similar enough to consider a match. Each characteristic is compared, and agreement is decided based on a threshold value that is characteristic dependent. If enough characteristics agree, then

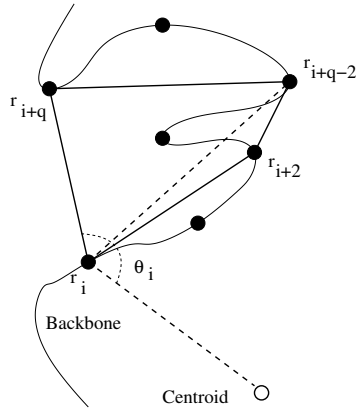


Fig. 1.6 Reside i is characterized by lengths of a polygon, the length of a diagonal, the angle θ_i, and two indices.

$\delta(i,j) = 1$, but if not $\delta(i,j) = 0$. This adaptation alters the recursion in (1.1) to

$$V(i,j) = \max \begin{cases} V(i-1,k) + \rho \\ V(i,j-1) + \rho \\ \delta(i,j)(V(i-1,j-1) + S(r_i, r'_j)). \end{cases}$$

A second adaptation is that $V(i,j)$ is not calculated if the difference between i and j exceeds a threshold indicating the maximum number of gaps allowed in a match. GOSSIP further limits the number of the protein chains that are compared by associating with each chain a collection of chains of similar length.

Numerical work for GOSSIP is promising. Two data sets described in [12] were used for testing, one containing $2,930$ protein structures from CATH and one containing $3,613$ structures from Astral. An all-against-all comparison requires $4,290,985$ alignments in the CATH dataset and $6,525,078$ comparisons in the Astral dataset. To gauge the effectiveness of GOSSIP, 267 proteins from the CATH dataset and 348 from the Astral dataset were selected, and all-against-all comparisons were made on these smaller datasets. Run times were then scaled to the larger datasets. The GOSSIP adaptations to DP were estimated to complete all pairwise comparisons in the large datasets in the range from 9 (17.3) hours to 5.1 (9.1) minutes on CATH (Astral), depending on how similar the lengths of the protein chains needed to be to consider an alignment. The longest run times were for comparisons in which the chain lengths only need to be within 60% of each other, and the shortest times required the chain lengths to be within 95% of each other. The accuracy of the method lies between the results of MultiProt [21] and YAKUSA [7], and while MultiProt better identifies structural similarity, it requires several hundred hours of computational time. GOSSIP bests YAKUSA's results while comparing reasonably with regard to run time.

1.4.2 EIGensystem Alignment with the Spectrum

As with the other methods described in this section, the central theme of aligning protein chains by Eigensystem Alignment with a Spectrum (EIGAs) is to align the residues so that the matching is as similar as possible with regard to a measure of intra-similarity between the residues of the individual proteins. EIGAs uses a scaled version of the Euclidean distance between the C_α atoms of any two residues, a measure called smooth-contact. A smooth-contact matrix, C, has as its ij-th element the smooth-contact between residues i and j,

$$[C]_{ij} = \begin{cases} 1 - d(r_i, r_j)/\kappa, & d(r_i, r_j) \leq \kappa \\ 0, & \text{otherwise,} \end{cases}$$

where $d(r_i, r_j)$ is the Euclidean distance between the C_α atoms of residues i and j. Euclidean distances less then the cutoff parameter κ are scaled linearly between the maximum smooth-contact value of 1, which occurs if $i = j$, and the minimum value of 0, which occurs if the distance between the residues exceeds κ. The smooth-contact measure is an assessment of the proximity of the residues. So the smooth-contact between a residue and itself is 1. If κ is small, then the smooth-contact between most residues is zero, but if κ is large, then more values are nonzero.

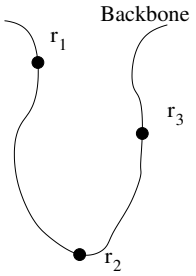
Smooth-contact matrices have the favorable mathematical property that they are positive definite, i.e. that all eigenvalues are positive, for suitably small κ values. This follows from the fact that the diagonal elements are 1 independent of the value of κ, whereas the off diagonal elements can be made arbitrarily small as κ decreases. Hence the sum of the off diagonal elements of any row is guaranteed to be less than the diagonal element itself for sufficiently small κ, a property called diagonal dominance. A well known result in linear algebra is that a diagonally dominant matrix is positive definite. The positive definite property guarantees that a smooth-contact matrix can be factored as,

$$C = UDU^T = (U\sqrt{D})(U\sqrt{D})^T = R^T R, \tag{1.2}$$

where the columns of U form an ortho-normal set of eigenvectors of C and D is a diagonal matrix of the positive eigenvalues. Moreover, a smooth-contact matrix imposes an inner product, and subsequently a norm and a metric, on $\mathbb{R}^n$, where n is the number of residues in the protein chain. So, each protein chain coincides with a geometric rendering of $\mathbb{R}^n$.

To illustrate the mathematical and geometrical properties induced by a smooth-contact matrix, we consider a 3 residue protein chain whose smooth-contact matrix is located in Figure (1.7). The inner product of the vectors v and w induced by the smooth-contact matrix is $< v, w >_C = v^T C w$, which has the following properties:

- $\sqrt{v^T C v} = \|v\|_C$ is the norm of v relative to C,
- $v^T C w / \|v\|_C \|w\|_C$ is the cosine of the angle between v and w, and
- $\sqrt{(v-w)^T C (v-w)} = \|v - w\|_C$ is the distance between v and w relative to C.



$$C = \begin{bmatrix} 1.00000 & 0.11471 & 0.36274 \\ 0.11471 & 1.00000 & 0.19764 \\ 0.36274 & 0.19764 & 1.00000 \end{bmatrix}$$

Fig. 1.7 The path of the backbone indicates that the Euclidean distances between residues 1 and 3 and between 2 and 3 are shorter than the distance between residues 1 and 2. This is seen in the smooth-contact matrix since both $C_{13} = 0.36274$ and $C_{23} = 0.19764$, both of which are greater than $C_{12} = 0.11471$.

The inner product supports an embedding of the residues in an n-dimensional space so that the smooth-contact between any two residues is the result of an inner product calculation. Let the first residue be associated with $e_1 = (1, 0, 0, \ldots, 0)^T$, the second with $e_2 = (0, 1, 0, \ldots, 0)^T$), and so on. Then $e_i^T C e_j = C_{ij}$, which is the smooth-contact between residues i and j, e.g. $e_1^T C e_3 = C_{13} = 0.36274$ for the smooth-contact matrix in Figure 1.7.

The vector space $\mathbb{R}^n$ with the inner produce $< v, w >_C$ is called a contact space, and the geometry induced by C is, in some sense, a 'skewed' Euclidean geometry. As an illustration we note that a sphere is the collection of unit length vectors, which in contact space means v is on the sphere only if $\|v\|_C = 1$. Such a collection is an ellipsoid due to the fact that the metric is scaled by C. A picture of the unit ellipsoid with respect to the contact matrix in Figure 1.7 is shown in Figure 1.8. The residues are represented by the thick blue vectors e_1, e_2, and e_3, which lie along the standard axes. The angle between e_i and e_j is not as it appears in the figure. For example, in Euclidean geometry the angle between e_1 and e_3 is $90°$, but in contact space the angle is $\cos^{-1}(0.36274) = 68.731°$. Also, in Euclidean geometry the distance between between e_1 and e_3 is $\sqrt{2} \approx 1.4142$, but in contact space the distance is $\sqrt{(e_1 - e_3)^T C (e_1 - e_3)} = 1.1289$.

The motivation behind EIGAs is that each protein is associated with an n-dimensional geometry, and this perspective allows the problem of aligning protein structures to be re-stated as a question of optimally aligning the geometries of contact spaces. However, contact spaces vary in dimension just as protein chains vary in length, and the algorithmic design question is to create a method of matching the residues so that the lower dimensional geometries of the two contact spaces are as similar as possible.

Contact geometries are defined by the eigenvectors and eigenvalues, which give the direction and length of the unit ellipsoid's principal axes. Moreover, the eigenvectors are linear combinations of the e_i vectors. So in terms of the protein, each eigenvector represents a weighted sum of the residues in contact space. In some research communities these weighted sums would be called 'features' of the protein. Each

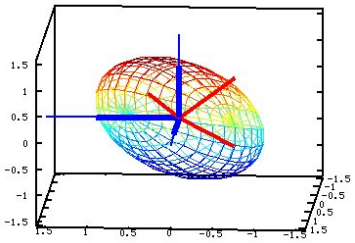


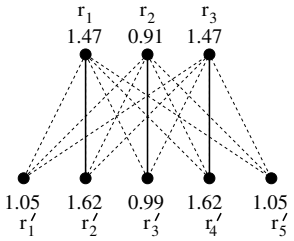
Fig. 1.8 The ellipsoid is the collection of all unit length vectors with respect to the contact matrix in Figure 1.7. The thick blue vectors represent the residues and are e_1 (pointing to the left), e_2 (pointing out from the page), and e_3 (pointing upward). The red vectors are the ortho-normal eigenvectors that form the columns of U in (1.2). The eigenvectors lie along the principal axes of the ellipsoid, and the length of each axis is $2/\sqrt{\lambda}$.

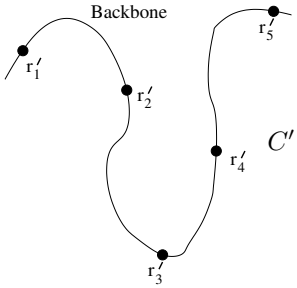
residue is associated with its nearest eigenvector and is assigned its corresponding eigenvalue. Formally, residue r_i is assigned the eigenvalue λ_k, where the associated eigenvector u_k of the smooth-contact matrix solves

$$\max_t \left| \frac{u_t^T C e_i}{\|u_t\|_C \|e_i\|_C} \right| = \max_t |R_{ti}|,$$

where u_t indexes the eigenvectors that comprise the columns of U and $R = \sqrt{D}U^T$ from (1.2). This eigenvalue assignment associates each residue with one of the principal axes of the ellipsoid defined by the smooth-contact map. A matching of the residues between the protein chains is scored by comparing the eigenvalues associated with the corresponding principal axes.

Consider the problem of matching the protein chain in Figure 1.7 with the chain in Figure 1.9. The middle three residues of the chain in Figure 1.9 are in the same configuration as the residues in Figure 1.7, and one would expect an alignment technique to match r_1 with r_2', r_2 with r_3', and r_3 with r_4'. The eigenvalues of the smooth-contact matrix in Figure 1.7 are 1.47, 0.91 and 0.63, and the eigenvalues of the smooth-contact matrix in Figure 1.9 are 1.62, 1.05, 0.99, 0.76, and 0.57. Assigning the residues the eigenvalues of their nearest eigenspaces leads to the matching problem depicted below.





$$C' = \begin{bmatrix} 1.00000 & 0.21301 & 0.01150 & 0.08055 & 0.00013 \\ 0.21301 & 1.00000 & 0.11471 & 0.36274 & 0.10040 \\ 0.01150 & 0.11471 & 1.00000 & 0.19764 & 0.00233 \\ 0.08055 & 0.36274 & 0.19764 & 1.00000 & 0.26100 \\ 0.00013 & 0.10040 & 0.00233 & 0.2610 & 1.00000 \end{bmatrix}$$

Fig. 1.9 Residues r'_2, r'_3 and r'_4 are in the same geometric configuration with r_1, r_2 and r_3 in Figure 1.7

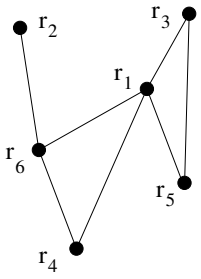
The optimal 3 residue matching is shown by the solid edges. The optimal value is the combined eigenvalue difference, $|1.47-1.62|+|0.91-0.99|+|1.47-1.62| = 0.38$, which is the lowest possible among all matchings of 3 residues. Note that this is the anticipated alignment.

The value of matching residue i with residue j is estimated by $S_{ij} = |\lambda_i - \lambda_j|$. If the eigenvalues are similar, then the principal axes of the unit ellipsoids associated with the residues are approximately the same length. Hence the contact geometries are similar along these particular axes. EIGAs uses the scoring matrix $[S]_{ij}$ and DP to find an optimal matching between the residues. EIGAs was reported in [22] to correctly identify the SCOP classification of the Skolnick40 dataset by completing all 780 comparisons in about 58 seconds. On Proteus300 EIGAs completed all $44,850$ comparisons in about 1.2 hours. The best results in terms of quality were achieved with $\kappa = 14$ Å and a gap penalty of 1. More recent results in [11] show that EIGAs can complete all $44,850$ comparison in 137 seconds if the DP is coded in C++ and the computational effort is distributed over 8 cores. This more recent numerical work shows that if $\kappa = 17$ Å, the gap opening penalty is 0.9, and the gap extension penalty is 0.4, then EIGAs correctly identifies the SCOP classifications of Proteus300 (perfect classifications were reported for many parameter combinations).

1.4.3 A_purva

As is often the case in applied mathematics, the concept of structure can be represented in terms of graph theory, and this is the manner in which contact map overlap (CMO) is derived. The backbone of each protein chain is represented as a graph, say (V, E). The vertices in V represent the C_α atoms, and two C_α atoms are adjacent, meaning that they form an edge in E, if their distance satisfies a contact criteria. For example, if $0 < d(r_i, r_j) < \kappa$, then residues i and j are in *contact* and (r_i, r_j) induces an edge in E. We let V be the index set $\{1, 2, \ldots, |V|\}$ and E be the collection of edges (i, j) such that $d(r_i, r_j)$ satisfies the contact criteria. A_purva [1] uses an upper bound of $\kappa = 7.5$ Å, and edges for consecutive residues along the protein backbone are not permitted. If we let A_{ij} form a binary matrix with the value being 1 if residues i and

j satisfy the contact criteria, then we have represented the protein chain as a graph associated with this adjacency matrix, see Figure 1.10.



$$A = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Fig. 1.10 A graph representation of a protein chain of six residues along with its adjacency matrix.

The goal of CMO is to optimize a pairing between the graphical representations of the two proteins, meaning that we want to pair residues so that their contact structures match as well as possible. This question is similar to the classic NP-hard problem of finding a maximum induced subgraph, which is to find the largest possible subgraph of one graph that is a subgraph of the other. However, CMO adds the constraint that the sequential ordering of the nodes must be preserved in the vertex matching. For example, if r_1 of the first protein is matched with r'_3 of the second, then r_2 of the first protein cannot be matched to r'_2 of the second since this would violate the linear ordering inherited from a protein's backbone. Residue r_2 could be matched to any r'_k of the second protein as long as $k > 3$.

To accomplish the residue matching, the graphs of two proteins are joined to create a bi-partite graph. Let (V^1, E^1) and (V^2, E^2) be the graphs of two proteins and A and A' be their respective adjacency matrices. Form the complete bi-partite graph between the two vertex sets, meaning that every possible edge between V^1 and V^2 is included. The edge sets E^1 and E^2 are used to weight every possible pair of matches, indexed by (i, j, k, l) to indicate that r_i of the first protein is matched with r'_k of the second and that r_j of the first is matched with r'_l of the second. Each (i, j, k, l) is assigned the product of the corresponding elements of the adjacency matrices,

$$A_{ij}A'_{kl} = \begin{cases} 1, & (i,j) \in E^1 \text{ and } (k,l) \in E^2 \\ 0, & \text{otherwise.} \end{cases}$$

We say that the paired residue matches (r_i, r'_k) and (r_j, r'_l) share a common contact provided that $A_{ij}A'_{kl}$ is 1. As an example, if we are trying to align the protein depicted in Figure 1.10 with a 4 residue protein in which the only contacts are between r'_1 and r'_3 and between r'_2 and r'_4, then $A_{13}A'_{13} = 1$ but $A_{26}A'_{12} = 0$. A schematic is depicted in Figure 1.11.

The bi-partite weights are used to score matchings between the residue sets by adding all possible weights in the matching. If r_i is matched with r'_i, for $i = 1, 2, 3, 4$,

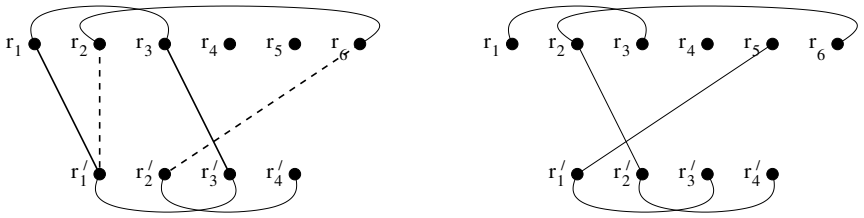


Fig. 1.11 The graph on the left depicts two possible pairs of matchings between two proteins. The pair depicted with solid lines is weighted with a one since r_1 is in contact with r_3 and r'_1 is in contact with r'_3. The pair depicted with dashed lines is weighted with a zero since r'_1 is not in contact with r'_2. Note, not all edges from Figure 1.10 are shown for the top protein. The graph on the right shows a pairing that would not be allowed since it violates the residue ordering imposed by the proteins' backbones.

in the example above, then this matching has a contact score of

$$A_{12}A'_{12}+A_{13}A'_{13}+A_{14}A'_{14}+A_{23}A'_{23}+A_{24}A'_{24}+A_{34}A'_{34} = 0+1+0+0+0+0 = 1.$$

This value indicates the only overlap of this residue matching is due to the fact that matching r_1 to r'_1 and r_3 to r'_3 yields a common contact. In matrix terms, the score is half of $\|(A_{I,R} \circ A'_{I,R})\|_1$, where I indexes the residues from the first protein, R indexes the residues from the second protein, the set subscripts indicate the submatrices whose rows and columns are listed in the sets, and $\circ$ is the Hadamard (elementwise) product. The 1-norm is the elementwise norm and not the operator norm. To see the above calculation in this matrix form, let $I = R = \{1, 2, 3, 4\}$, which gives

$$A_{I,R} \circ A'_{I,R} = A_{I,R} \circ A' = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \circ \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

As above, the only nonzero elements result from residues r_1 and r_3 being in contact in the first protein and residues r'_1 and r'_3 being in contact in the second. The symmetry of the adjacency matrices doubles the score. The matrix description shows that we are looking for ordered index sets of the residues from both proteins so that the elementwise product of the resulting adjacency submatrices has as many ones as possible.

The combinatorial problem of maximizing the contact overlap can be stated as a binary optimization problem. For the i-th residue in V^m, with m indexing either the first or second protein, let $\delta_m^+(i) = \{j : j > i,\ (i,j) \in E^m\}$ and $\delta_m^-(i) = \{j : j < i,\ (i,j) \in E^m\}$. We let y_{ijkl} be a binary variable indicating if we match r_i of the

first protein with r'_k of the second and r_j of the first with r'_l of the second. We also let x_{ik} be a binary variable indicating if r_i of the first protein is matched with r'_k of the second. With this notation, the standard formulation of the CMO problem is,

$$\left.\begin{array}{rcll} \max \sum\limits_{\substack{(i,j)\in E^1\\(k,l)\in E^2}} y_{ijkl} & = & \max \sum\limits_{(ijkl)} A_{ij}A'_{kl}y_{ijkl} & \\ & & & \\ \text{s.t.} & & & \\ & & & \\ \sum\limits_{j\in\delta_1^+(i)} y_{ijkl} & \leq & x_{ik}, & \forall\, i \in V^1,\ (k,l) \in E^2 \\ \sum\limits_{i\in\delta_1^-(j)} y_{ijkl} & \leq & x_{jl}, & \forall\, j \in V^1,\ (k,l) \in E^2 \\ \sum\limits_{l\in\delta_2^+(k)} y_{ijkl} & \leq & x_{ik}, & \forall\, k \in V^2,\ (i,j) \in E^1 \\ \sum\limits_{k\in\delta_2^-(l)} y_{ijkl} & \leq & x_{jl}, & \forall\, l \in V^2,\ (i,j) \in E^1 \\ x_{ik} + x_{jl} & \leq & 1, & \forall\, 1 \leq i \leq j \leq |V^1|,\ i \neq j \\ & & & \forall\, 1 \leq k \leq l \leq |V^2|,\ \ k \neq l \\ x_{ik} & \in & \{0,1\}, & \forall\, i \in V^1,\, k \in V^2 \\ y_{ijkl} & \in & \{0,1\}, & \forall\, (i,j) \in E^1,\, (k,l) \in E^2. \end{array}\right\} \quad (1.3)$$

Solving the CMO problem has drawn much attention, and many have argued favorably for this alignment method. However, the combinatorial complexity of the problem has challenged it with becoming an efficient model for working with entire databases. The recent work in [1] shows otherwise. The insight is to reformulate the problem so that the new formulation lends itself to DP.

The reformulation is interpreted on a new graph $(V^1 \times V^2, \mathcal{E})$, where $\mathcal{E} = \{(i,k,j,l) : A_{ij}A'_{kl} = 1\}$. In this new graph each vertex is an ordered pair of vertices, and each edge corresponds to a matched pair of residues that share a contact. In terms of the protein chains, each vertex of this graph is a possible match between the residues of the different proteins, and an edge only exists between a pair of possible residue matches if they share a common contact. A depiction of this graph is found in Figure 1.12.

The reformulation hints at how DP can be used to solve the CMO problem. The central concept is to move from the lower left corner toward the upper right corner in an optimal fashion. Unlike the other alignment algorithms discussed in this section, which require a single application of DP, the method of A_purva requires a coupling of two DP algorithms. The two decisions are 1) where to link any possible match, called the local problem, and 2) how to construct on optimal sequence from the local decisions, called the global problem. The example in Figure 1.12 illustrates the decisions. From the match (r_1, r'_1) we could have selected any of (r_3, r'_3), (r_4, r'_3), (r_5, r'_3) or (r_6, r'_3). In all but (r_6, r'_3) we could have found a second non-intersecting edge that would have given a CMO score of 2. For example, if we had instead used the edge from (r_1, r'_1) to (r_5, r'_3), then we could have added the edge from (r_4, r'_2)

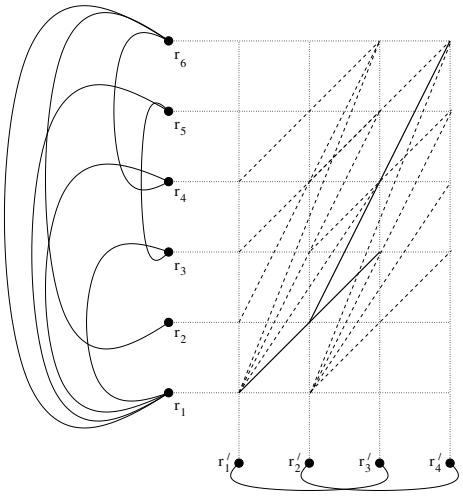


Fig. 1.12 The contacts of the protein from Figure 1.11 are shown on the left and the contacts for the second protein are shown at the bottom. The new graph links pairs of residue matches if they share a contact. All edges of the new graph are shows as either dashed or solid lines, with the solid lines indicating an optimal solution of (r_1, r'_1), (r_2, r'_2), (r_3, r'_3) and (r_6, r'_4). The optimal value of the CMO problem is 2 because this is the maximum number of non-crossing edges.

to (r_6, r'_4). The local decision has to assess which edges should be considered. Once we know the optimal local solutions for each (r_i, r'_k), we use this information to select the global collection of edges so as to maximize the number of edges while maintaining that edges don't cross.

The forward looking perspective means that we are always making decisions for indices in δ_m^+, and hence we need to account for the constraints whose summands are over δ_m^-. Andonov et al. [1] use a classic Lagrangian relaxation that moves these constraints to the objective and penalizes them with multipliers. The relaxed problem

maximizes

$$\sum_{\substack{(i,j)\in E^1\\(k,l)\in E^2}} y_{ijkl}$$

$$+ \sum_{\substack{j\in V^1\\(k,l)\in E^2}} \lambda_{jkl} \left(x_{jl} - \sum_{i\in\delta_1^-(j)} y_{ijkl} - \sum_{\substack{s=k+1\ldots l-1\\t=1\\(t,s)\in E^1}} y_{tskl} - \sum_{\substack{s=1\ldots k-1\\t=j-1\\(t,s)\in E^1}} y_{tskl} \right)$$

$$+ \sum_{\substack{l\in V^2\\(i,j)\in E^1}} \sigma_{ijl} \left(x_{jl} - \sum_{k\in\delta_2^-(l)} y_{ijkl} - \sum_{\substack{s=k+1\ldots j-1\\t=1\\(s,t)\in E^2}} y_{stkl} - \sum_{\substack{s=1\ldots k-1\\t=l-1\\(s,t)\in E^2}} y_{stkl} \right)$$

subject to

$$\begin{aligned}
\sum_{j\in\delta_1^+(i)} y_{ijkl} &\leq x_{ik}, && \forall\, i\in V^1,\ (k,l)\in E^2\\
\sum_{l\in\delta_2^+(k)} y_{ijkl} &\leq x_{ik}, && \forall\, k\in V^2,\ (i,j)\in E^1\\
x_{ik}+x_{jl} &\leq 1, && \forall\, 1\leq i\leq j\leq |V^1|,\ i\neq j\\
& && \forall\, 1\leq k\leq l\leq |V^2|,\ \ k\neq l\\
x_{ik} &\in \{0,1\}, && \forall\, i\in V^1,\, k\in V^2\\
y_{ijkl} &\in \{0,1\}, && \forall\,(i,j)\in E^1,\,(k,l)\in E^2.
\end{aligned}$$

We note that this is not directly a Lagrangian relaxation of (1.3) due to a tighter description of the feasible region. Specifically, the authors make the following replacements in (1.3),

$$\sum_{i\in\delta_1^-(j)} y_{ijkl}\leq x_{jl} \quad\Rightarrow\quad \sum_{i\in\delta_1^-(j)} y_{ijkl} + \sum_{\substack{s=k+1\ldots l-1\\t=1\\(t,s)\in E^1}} y_{tskl} + \sum_{\substack{s=1\ldots k-1\\t=j-1\\(t,s)\in E^1}} y_{tskl}\leq x_{jl}$$

$$\sum_{k\in\delta_2^-(l)} y_{ijkl}\leq x_{jl} \quad\Rightarrow\quad \sum_{k\in\delta_2^-(l)} y_{ijkl} + \sum_{\substack{s=k+1\ldots j-1\\t=1\\(s,t)\in E^2}} y_{stkl} + \sum_{\substack{s=1\ldots k-1\\t=l-1\\(s,t)\in E^2}} y_{stkl}\leq x_{jl}.$$

The additional terms continue to satisfy the non-crossing property, but they give a more accurate description of the search space.

The Lagrange multipliers λ_{jkl} and σ_{ijl} are restricted to be nonnegative, and through the application of DP, the relaxed problem can be solved in polynomial time for any collection of Lagrange multipliers, the complexity being no worse than $O(|V^1||V^2|+|\mathcal{E}|)$. The local use of DP creates a scoring matrix for each (i,k) in $V^1\times V^2$. Specifically, let S_{ik}^L be the matrix whose rows are indexed by

$\delta_1^+(i)$ and whose columns are indexed by $\delta_2^+(k)$. The score at each position is $[S_{ik}^L]_{jl} = 1 - \lambda_{jkl} - \sigma_{ijl}$, which corresponds with the coefficient of the y_{ijkl} variable in the objective. Let c_{ik}^* be the optimal value of applying DP to S_{ik}^L.

The global problem uses the local scores to create an optimal solution to the relaxed problem. Specifically, let S^G be the scoring matrix whose components for each $i \in V^1$ and $k \in V^2$ are

$$\left[S^G\right]_{ik} = c_{ik}^* + \sum_{j \in \delta_1^-(i)} \lambda_{jkl} + \sum_{k \in \delta_2^-(l)} \sigma_{ijl}.$$

Notice that the last two summations are the coefficients for the x_{jl} variables, and that the remaining portion of the objective collapses into c_{ik}^*. Hence, an optimal solution to the Lagrangian relaxation for any nonnegative collections of λ_{jkl} and σ_{ijl} can be calculated with a double application of DP.

While the relaxation can be solved efficiently, the relaxed solution is not a solution to the original CMO problem unless the Lagrange multipliers satisfy an optimality condition of their own. The optimization problem that defines a collection of multipliers for which the relaxed problem actually solves the CMO problem is called the dual optimization problem, which is a minimization problem over the Lagrange multipliers for a fixed collection of x and y variables. A discussion of this topic is outside the scope of this article, but interested readers can find descriptions in most nonlinear programming texts [2]. The Lagrangian process is iterative in that starting with initial Lagrange multipliers, the relaxation is solved with DP, and then the Lagrange multipliers are updated by (nearly) solving the dual problem. The process repeats and is stopped once the value of the CMO problem is sufficiently close to the optimal value of the dual problem or due to a time limitation. For details about how A_purva updates the Lagrange multipliers, see [1].

Iterative Lagrangian algorithms are a mainstay in the nonlinear programming community, and a common sentiment among experts in optimization is that problems either lend themselves to this tactic or not. The numerical tests for A_purva show that the relaxation of CMO lends itself nicely to this solution procedure. A_purva was tested as a database comparison algorithm on both Skolnick40 [6] and Proteus300 [1]. Reported solution times were 357 seconds for all 780 pairwise comparisons for Skolnick40 (approximately 0.46 seconds per comparison) and 13 hours and 38 minutes for all $44,850$ comparisons for Proteus300 (approximately 1.09 seconds per comparison). The comparisons were perfect in their agreement with the SCOP classification via clustering with `Chavl` [16]. While not as fast as either of the other techniques discussed in this section, the results are significant advances for CMO, which before A_purva would have been impractical for a dataset like Proteus300.

1.4.4 Comments on Dynamic Programming and Future Research Directions

The application of DP to align protein structures is not restricted to the recent literature on database applications, and in particular, DP was one of the early solution methods for the CMO problem, see [14, 6]. However, the recent literature indicates that DP

is becoming a central algorithmic method to efficiently align protein structures well enough for classification across a sizable database. Other research groups are arriving at the same conclusion about DP [24]. Indeed, as the authors were preparing this document they found additional examples in the recent literature [5].

With the emerging success that DP is having, we feel compelled to note a few of its limitations. First, DP is a sequential decision process that is optimal only under the Markov property, i.e. that the current decision does not depend on past decisions. All the applications of DP discussed here use the natural sequence of the backbone to order the decisions process, but using DP in this fashion does not allow for the possibility of nonsequential alignments. The ability to identify nonsequential alignments is argued by some as a crucial element to identifying protein families [8]. Adapting the DP framework to consider nonsequential alignments is a promising and important avenue for future research.

One of the concerns about three-dimensional alignment methods is that they don't easily adapt to a protein's natural flexibility. Many proteins have several confirmations, and an alignment method that can correctly classify the different confirmations would be beneficial. Moreover, the crystallography and NMR experiments from which we gain three-dimensional coordinates are not without error, and the alignment methods should be robust enough to correctly classify proteins under coordinate perturbations. Whether or not methods based on DP provide the robustness to handle a protein's flexibility and experimental error is not yet well established. Hasegawa and Holm [10] suggest more attention be focused on the robustness of the optimization procedures used to align protein structures.

Finally, DP requires tuning that is experimental in nature. The gap opening and gap extension penalties need to be experimentally set to achieve agreement with biological classifications. Parameter tuning has not been shown to be database independent, and without such experimental validation, we lack the confidence that DP based methods can be used to organize a database without an apriori biological classification, which somewhat defeats the purpose of automating database classification.

References

1. R Andonov, N. Malod-Dognin, and N. Yanev. Maximum contact map overlap revisited. *J Comput Biol*, 18(1):27–41, Jan 2011.

2. M. Bazaraa, H. Sherali, and C. Shetty. *Nonlinear Programming: Theory and Algorithms*. Wiley-Interscience, 3rd edition, 2006.

3. R. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, 1957.

4. S. Bhattacharya, C. Bhattacharyya, and N. Chandra. Projections for fast protein structure retrieval. *BMC Bioinformatics*, 7 Suppl 5:S5, 2006.

5. N. Bonnel and P. Marteau. Lna: Fast protein classification using a laplacian characterization of tertiary structure. Technical report, Université de Bretagne Sud, France, 2011.

6. A. Caprara, R. Carr, S. Istrail, G. Lancia, and B. Walenz. 1001 optimal pdb structure alignments: Integer programming methods for finding the maximum contact map overlap. *J Comput Biol*, 11(1):27–52, 2004.

7. M. Carpentier, S. Brouillet, and J. Pothier. Yakusa: A fast structural database scanning method. *Proteins*, 61(1):137–151, Oct 2005.

8. L. Chen, L. Wu, Y. Wang, S. Zhang, and X. Zhang. Revealing divergent evolution, identifying circular permutations and detecting active-sites by protein structure comparison. *BMC Struct Biol*, 6:18, 2006.

9. D. Goldman, S. Istrail, and C. Papadimitriou. Algorithmic aspects of protein structure similarity. In *Foundations of Computer Science, 1999. 40th Annual Symposium on*, pages 512–521, 1999.

10. H. Hasegawa and L. Holm. Advances and pitfalls of protein structural alignment. *Curr Opin Struct Biol*, 19(3):341–348, Jun 2009.

11. A. Holder, J. Simon, J. Strauser, and Y. Shibberu. An investigation into the robustness of algorithms designed for efficient protein structure alignment across databases. Technical report, Rose-Hulman Institute of Technology, USA, 2012.

12. I. Kifer, R. Nussinov, and H. Wolfson. Gossip: A method for fast and accurate global alignment of protein structures. *Bioinformatics*, 27(7):925–932, 2011.

13. R. Kolodny and N. Linial. Approximate protein structural alignment in polynomial time. *Proc Natl Acad Sci U S A*, 101(33):12201–12206, Aug 2004.

14. G. Lancia, R. Carr, B. Walenz, and S. Istrail. 101 optimal pdb structure alignments: A branch-and-cut algorithm for the maximum contact map overlap problem. In *Proceedings of the Fifth Annual International Conference on Computational Biology*, pages 143–202, New York, NY, 2001. ACM Press.

15. P. Di Lena, P. Fariselli, L. Margara, M. Vassura, and R. Casadio. Fast overlapping of protein contact maps by alignment of eigenvectors. *Bioinformatics*, 26(18):2250–2258, Sep 2010.

16. I. Lerman. Likelihood linkage analysis (lla) classification method. *Biochimie*, 75:379–397, 1993.

17. S. Cheng Li and Y. Kaow Ng. On protein structure alignment under distance constraint. *Theoretical Computer Science*, 412(32):4187 – 4199, 2011. Algorithms and Computation.

18. S. Needleman and C. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol*, 48(3):443–453, Mar 1970.

19. A. Poleksic. Algorithms for optimal protein structure alignment. *Bioinformatics*, 25(21):2751–2756, Nov 2009.

20. A. Poleksic. Optimizing a widely used protein structure alignment measure in expected polynomial time. *IEEE/ACM Trans Comput Biol Bioinform*, 8(6):1716–1720, 2011.

21. M. Shatsky, R. Nussinov, and H. Wolfson. A method for simultaneous alignment of multiple protein structures. *Proteins*, 56(1):143–156, Jul 2004.

22. Y. Shibberu and A. Holder. A spectral approach to protein structure alignment. *IEEE/ACM Trans Comput Biol Bioinform*, Feb 2011.

23. S. Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE Trans on Pattern Analysis and Machine Intelligence*, 10(5):695–703, 1988.

24. I. Wohlers, R. Andonov, and G. Klau. Algorithm engineering for optimal alignment of protein structure distance matrices. Technical report, CWI, Life Sciences Group, Netherlands, 2011.

25. J. Xu, F. Jiao, and B. Berger. A parameterized algorithm for protein structure alignment. *J Comput Biol*, 14(5):564–577, Jun 2007.