Using Neighborhood Preserving Projections for Comparing Protein Structures

Sourangshu Bhattacharya Chiranjib Bhattacharyya

IISc-CSA-TR-2004-14
http://archive.csa.iisc.ernet.in/TR/2004/14/

Computer Science and Automation
Indian Institute of Science, India

October 2004

Using Neighborhood Preserving Projections for Comparing Protein Structures

Sourangshu Bhattacharya and Chiranjib Bhattacharyya
Dept. of Computer Science & Automation,
Indian Institute of Science,
Bangalore -560012, India.
{ sourangshu, chiru } @ csa.iisc.ernet.in

Abstract

This paper reports a new protein structure comparison algorithm, Matchprot, which is based on comparison of optimal neighborhood preserving 1-dimensional projection of the structure. A novel similarity measure based on the above idea is suggested and is used to obtain a set of equivalences between the residues of the two proteins by a local alignment based formulation. The algorithm makes connection with spectral graph theory, which in turn suggests a interesting link between weighted graph matching and protein structure comparison. Matchprot runs in $O(n^3)$ time when two proteins of n residues are compared. Empirical testing shows that the method identifies the correct alignment when the protein pairs are structurally similar. Moreover, it discovers the complete structural alignment even when the protein sequences are related by circular permutations, which are not detected by many standard protein structure comparison programs.

Keywords: Protein structure comparison, Circular Permutations, Distance matrices

1 Introduction

Structures of proteins are more descriptive of their functions and evolutionary history than sequences. Structural similarity can often shed light on a common function or the same origin of different proteins. Comparing proteins is thus an important task for structural biologists. The rapid growth of in the no. of entries in the protein structures databases e.g. PDB[1], calls for development of fast and accurate structure comparison methods.

The problem of comparing two protein structures is often posed as that of finding an optimal structural alignment. The structural alignment problem can be understood as that of identifying maximal substructures from each structure, which have a high structural similarity. A protein structure is often described by the position of one atom of a residue, typically but not neccessarily the C^{α} atom. There are two approaches to determine structural similarity. The first approach determines a rigid transformation (i.e. rotation and translation) that optimally superimposes the two structures one against the other. After superposition, the Euclidean distance between the matched atoms of the two proteins is taken as a measure of similarity. The alternative approach is based on comparing *distance matrices*. The distance matrix of a protein is consisting of pairwise euclidean distances of all pairs of residues in a protein. Havel et. al.[2] showed that one can reconstruct the structure from the distance matrix, but with an ambiguity of the chirality. This implies that there is enough information about protein structure in the distance matrices and it can serve as a basis for protein structure comparison. This approach sidesteps the computation of the expensive transformation step in the former approach. However both the approaches gives rise to extremely hard computational problems. The work reported in this paper uses distance matrices

A wide variety of programs have been reported in literature for comparing protein structures following both the approaches. SSAP[3, 4], DALI [5, 6], and CE [7] use distance matrices, while LOCK [8, 9] uses the transformation based approach. These algorithms solve the structural alignment problem using very clever heuristics. DALI and LOCK use iterative algorithms for solving the optimization problem. ever, due to the complexity of the function to be optimized, no convergence proofs or bounds on conrevergence rates can be given. This is unsatisfactory as protein structure comparison programs are basic and are to be used repetitively in many experiments.

Most of the approaches mentioned here use sequence order information heavily and hence will not be able to detect non-topological similarity. A simple kind of non-topological similarity occuring in nature is the case of *circularly permuted* proteins. A pair of proteins is said to be circularly permuted if the N-terminal of one protein has significant sequence similarity with C-terminal of the other and vice versa[10]. Many naturally occurring pairs of circularly permuted proteins have been reported [11, 10]. However, SSAP and CE can align 2 residues only if they appear on the same side of the already aligned residues. They will detect a portion of the match when there is a circular permutation between the two proteins. A robust protein structure comparison algorithm should be able to detect such permutations.

In this paper we propose a heuristic algorithm, Matchprot, for structural alignment. We propose to compare optimal neighborhood preserving 1- dimensional projections of the protein structures instead of the actual structures. We develop a novel similarity measure based on the above view, that gives a similarity value to 2 residues, one from each structure. It also makes an important connection between the protein structure comparison problem and weighted graph matching techniques. Finally, we have proposed a new method for retrieving optimal equivalences between residues of 2 structures based on the above similarity measure and the information given by the protein sequences. The proposed algorithm is fast and has a worst case running time cubic in the size of the proteins. Experimental results show that our algorithm performs well on diverse types of protein structures as compared to other standard programs. Though the method for retrieving the equivalences between residues of 2 proteins uses sequence information, it can retrieve them correctly even in case of a permutation in the sequences. This is further validated by experimental results.

The paper is organized as follows In section 3 the matchprot algorithm is described. Experimental results are presented in section 4. We conclude with some directions for further research.

2 Preliminaries

In this paper we view proteins as a chain of amino acid residues in 3-dim space. Denote protein A, having n residues, by $A = \{x_1, x_2, \ldots, x_n\}$, where each $x_i \in \mathbb{R}^3$ is the coordinate of the ith C_α atom. The distance matrix of protein A is defined as the $n \times n$ matrix with elements $d^A_{ij} = \|x_i - x_j\|$ where $\|x\|$ denotes the Euclidean norm. Consider two proteins $A = \{x_1, x_2, \ldots, x_m\}$ and $B = \{y_1, y_2, \ldots, y_n\}$ having m and n residues respectively. The set $\phi = \{(x_{i_l}, y_{j_l}) | x_{i_l} \in A, y_{j_l} \in B, 1 \leq l \leq K, i_l \neq i_k \& j_l \neq j_k \text{ if } l \neq k\}$ is called a structural alignment of length K. In the alignment the residue i_l of protein A is said to be matched or equivalenced with residue j_l of protein B.

3 The Algorithm: Matchprot

3.1 DALI score function

To derive the intuitions for our algorithm it is instructive to understand what alignments DALI[5, 6] might prefer. DALI defines the score of an alignment ϕ as follows

$$\begin{aligned} S_{A,B}(\phi) = & \quad \sum_{i=1}^{K}\sum_{j=1}^{K}\phi(i,j) \\ \phi(i,j) = & \quad \left(0.2 - \frac{|d_{ij}^{A} - d_{ij}^{B}|}{d_{ij}^{AB}}\right)\exp\left(-\left(\frac{d_{ij}^{AB}}{20}\right)^{2}\right) \end{aligned} \tag{1}$$

where $d_{ij}^{AB} = (d_{ij}^{A} + d_{ij}^{B})/2$. DALI searches the alignment space for that ϕ which maximises S. This problem is extremely difficult and is solved with several heuristics.

There are two properties that make terms in the DALI scoring function (1 contribute high values to the total score. The first point to be noted is that residues which are spatially far do not contribute much to the DALI score. See that whenever residues i and residue j are far apart d_{ij}^{AB} is high, driving the exponential weighting term to 0 and hence spatially far residues have no impact on the score function. DALI thus considers only those residues which are spatially close, hence effectively defining a neighbourhood. The second observation is that DALI will try to match those residues which have a similar neighbourhood. This is because the DALI score (1) gives a higher score to that ϕ, whose residues are such that $|d_{ij}^{A} - d_{ij}^{B}|$ is low. Thus DALI tries to pick up alignments whose residues have similar spatial neighbourhoods. Searching for such alignments in the space of all possible alignments is an extremely difficult problem. The key for efficiently locating such alignments lies in characterizing the neighbourhood information. In the next section we discuss one such characterization.

3.2 Optimal neighbourhood preserving projection

A protein is essentially a collection of points in $\mathbb{R}^3$. We propose to project these points on the real line such that neighbourhood information is preserved. It is hoped that these projected points serve as characterization of the neighbourhood and should be comaparable across structures. In this section we discuss a formulation to optimally compute such projections.

Let two points i and j be spatially close in a given protein structure then their projections should also be close. To parametrize the notion of closeness we define the nearness matrix of a protein A with n residues by the following nonlinear decreasing function

$$\mathcal{A}_{ij} = e^{\frac{-d_{ij}}{\alpha}},\ \alpha > 0 \tag{2}$$

of the distance between ith and jth residues. The parameter α governs the rate of decrease of the nearness value.

Let $f = [f_1, \ldots, f_n]^T$ be the vector of projections of a protein having n residues. We require that whenever $\mathcal{A}_{ij}$ is high, $|f_i - f_j|$ should be low. This can be directly translated as the following objective function

$$\min_{f\in\mathbb{R}^n} \sum_{i=1}^{n}\sum_{j=1}^{n} \mathcal{A}_{ij}(f_i - f_j)^2 \tag{3}$$

An obvious solution to the above minimization problem is $f = ce$, where c is a scalar and e is a vector of all 1s. This is not very useful because it results in all residues being projected on the same point on the real

line. This can be avoided by imposing the constraint that the solution should be orthogonal to ce. This is implemented by requiring that f should satisfy

$$f^T e = \sum_{i=1}^{n} f_i = 0 \tag{4}$$

Suppose f^* solves formulation (3) with the above constraint then cf^* also solve the same problem where c is any scalar. To deal with this extra degree of freedom one more scaling constraint is introduced, leading to the following formulation

$$\begin{array}{l} \min_{f\in\mathbb{R}^n} \sum_{i=1}^{n}\sum_{j=1}^{n} \mathcal{A}_{ij}(f_i - f_j)^2 \\ \text{Subject to} \\ \sum_{i=1}^{n} f_i = 0 \\ \sum_{i=1}^{n} f_i^2 = k \end{array} \tag{5}$$

For a given $k > 0$ this problem can be efficiently solved by quadratic programming. The optimal solution of formulation (5) be f^* and is the optimal neighbourhood preserving projection of a protein with nearness matrix $\mathcal{A}$. Given two proteins, A and B, consisting of m and n residues, it is desired that the respective projections f^A and f^B be comaparable. As argued before if f^* is the optimal projection then one can also consider cf^* as the optimal projection as it will solve (3) with the constraint (4). Thus instead of comparing f^A and f^B one may want to compare f^A and cf^B, where c is some constant. To fix the constant c one can impose various criterion. One could have required that the range of values for both the proteins are same which gives $c = \frac{\max f^A - \min f^A}{\max f^B - \min f^B}$. Alternatively we define $\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(f_i - f_j)^2$ as a measure of average granularity of the structure. We propose to compute c such that the two structures have the same granularity and hence the projections are comparable. After computing the projections of two protein structures, f^A and f^B consisting of m and n residues respectively, we choose c to be

$$\frac{1}{m^2}\sum_{i=1}^{m}\sum_{j=1}^{m}(f_i^A - f_j^A)^2 = c^2\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(f_i^B - f_j^B)^2 \tag{6}$$

The vectors satisfy the constraints of (5) and we obtain $c^2 = \frac{n}{m}$. Equivalently one can fix k as the number of residues in formulation (5) i.e. set $k = n$, and obtain $c = 1$. We propose to compute the projection of a protein structure having n residues with nearness matrix A by solving the following

$$\begin{array}{l} \min_{f\in\mathbb{R}^n} \sum_{i=1}^{n}\sum_{j=1}^{n} \mathcal{A}_{ij}(f_i - f_j)^2 \\ \text{Subject to} \\ \sum_{i=1}^{n} f_i = 0 \\ \sum_{i=1}^{n} f_i^2 = n \end{array} \tag{7}$$

problem which can be solved in $O(n^3)$ time. This formulation has an extremely important relation with algebraic graph theory. We digress to discuss this issue in the next section.

3.3 Protein Structures as Graphs

We start by viewing protein structures as weighted graphs, with residues as the vertices having the nearness matrix $\mathcal{A}$ as the adjacency matrix. For sake of brevity we will call such graphs as nearness graphs.

Following Mohar [12, 13], we define the Laplacian of the nearness graph as $\mathcal{L} = \mathcal{D} - \mathcal{A}$, where $\mathcal{D}$ is a diagonal matrix defined as $\mathcal{D}_{ij} = \sum_{k=1}^{m}\mathcal{A}_{ik}$ if $i = j$, 0 otherwise. By straightforward algebra one obtains

$$f^T\mathcal{L}f = \sum_{i=1}^{m}\sum_{j=1}^{m}\mathcal{A}_{ij}(f_i - f_j)^2$$

Using the variational characterization of eigenvalues one can show that solving (7) is equivalent to finding the eigenvector corresponding to the smallest non-zero eigenvalue of $\mathcal{L}$[13]. This eigenvector is extremely useful in various disciplines, for a review of its applications see [12].

It is clear that, a good match between the nearness graphs of 2 protein structures will give a good structural match between the corresponding structures and vice versa. Also, it follows from the continuity of the Euclidean distance function and the nearness function, that small perturbations in the positions of the C_α atoms of residues, induce small errors in the weights of edges connected in the nodes corresponding to the residue. Thus, even in presence of small errors, matching nearness graphs will also match the corresponding structures. This observation suggests that solution to general weighted graph matching problems e.g. [14, 15] maybe of interest for protein structure comparison. This opens a very interesting area of study which will be taken up elsewhere.

3.4 A measure of similarity

The optimal one dimensional projections by solving (7) can be used to derive a similarity function between various residues of different structures. In a later section it would be extremely useful in retrieving the correspondences.

The similarity between 2 points, one from each structure, can be defined as a decreasing function of the difference between the corresponding projections. If f^A and f^B are the optimal projections corresponding to structures A and B, similarity between residue i belonging to structure A and residue j belonging to structure B is defined as:

$$s(i,j) = T - |f_i^A - f_j^B| \tag{8}$$

This measure essentially tries to characterize the similarity between the neighbourhoods of residue i and residue j. The most important feature of the above similarity measure is that it can score 2 residues one from each structure based on purely structural properties (no information about the protein sequence or secondary structural elements have been used). The DALI [5] and CE [7] scoring functions can give a measure of similarity, when four residues, 2 from each structure are supplied. With such a scoring function, a typical strategy of retrieving correspondences is to generate one and test for high score. DALI generates random equivalences and tries to add them to the existing list, while CE tries to extend a seed alignment using heuristic cutoffs. Generally, such methods tend to be slow due to the exponential number of possibilities that are searched.

SSAP [3] proposes a scoring function of the type developed here by using the global alignment scores with the residues in question constrained to be aligned to each other. However, this can generate different alignments for each residue pair, thereby making the scores incomparable. LOCK [8] uses secondary structure information to construct vectors and derives a scoring measure out of them.

A problem with the scoring function is that the projection values of vertices are evaluated with respect to vertices in a neighborhood of the original vertex in the graph. Thus, inclusion of too many unmatched vertices (which do not have any corresponding ones in the other graphs) may hamper the projection value of the current vertex beyond recognition. Thus, significant difference in size of the structures can have negative impact on the similarity function.

3.5 Retrieval of Correspondences

Various strategies can be used for retrieving equivalences based on the similarity function derived in 8. For example, one can use DALI type heuristics to find an alignment ϕ which maximizes the score $S(\phi) = \sum_{l=1}^{k} s(i_l, j_l)$, where $\phi = \{(i_l, j_l) | 1 \leq l \leq k\}$. However, the similarity function is nondiscriminating, as two different residues can have same one dimensional projection values. It is worthwhile to consider the sequence information along with the similarity function as a basis for determining equivalences.

One can compute a structural alignment between two protein structures by aligning the sequences of the two proteins using $s(i,j)$ as the score function 8. This can be posed as a global alignment problem [16] which can be efficiently solved. The problem with the above approach is the resultant alignment will be heavily dependent on the sequence order and hence will not be able to find equivalences if the sequence of one protein is permuted as compared to the other.

We propose to find subsequences, with high structural similarity. The problem can be posed as that of finding two subsequences which has the maximum similarity when the score is given by 8, this is also known as the Local Alignment problem. This can be efficiently solved using dymamic programming [17]. We will call such sequences as *High Scoring Fragment Pairs(HSFPs)*. It can be noted that DALI and CE also build the alignment from protein fragments rather than residues (hexapeptides in DALI and fragments of length 8 in CE). In our case, the fragment are of variable size giving more flexibility to the algorithm.

The algorithm operates in 2 stages:
(1) Calculation of the local alignment matrix with similarity score given by equation 4.
(2) Iterative determination of HSFPs and their elimination from the local alignment matrix.
The local alignment matrix L, is computed in the same way as by Smith and Waterman, except that we use the structure based similarity measure (Eqn. 8. Thus, $L_{i,j}$ is calculated, for $0 \leq i \leq m,\ 0 \leq j \leq n$, as:

$$L_{i,j} = \begin{cases} 0 & \text{, if } i = 0 \text{ or } j = 0 \\ \max \begin{cases} L_{i-1,j-1} + s(i,j) \\ L_{i-1,j} - g \\ L_{i,j-1} - g \\ 0 \end{cases} & \text{, otherwise} \end{cases} \tag{9}$$

Next the highest scoring entry (corresponding to highest scoring fragment) is detected and traced back to get the highest scoring fragment. The indices corresponding to the residues participating in the current alignment are eliminated from the matrix and the above step is repeated to get the next highest scoring fragment. This is stopped when there are no more positive scoring fragments. The steps are given in Algorithm 1.

Finally, the fragment pairs found above are concatenated to get the whole alignment. To get the equivalences, all the residues aligned to gaps are discarded, and corresponding residue pairs are taken as equivalenced residues. The structural alignment is the set of equivalenced residue pairs obtained above.

Algorithm 1 Retrieving correspondences

```
Alignment ← ϕ.
Compute highest = max_{i,j} L_{i,j}.
Compute (p1, p2) = arg max_{i,j} L_{i,j}.
while highest > 0 do
  Alignment ← Alignment ∪ traceback(p1, p2) {traceback returns the alignment obtained by
  tracing back from it's argument}
  Mark the rows and columns of L corresponding to the residues returned in the current
  alignment done.
  Compute highest = max_{i,j} L_{i,j} such that i or j is not marked done.
  Compute (p1, p2) = arg max_{i,j} L_{i,j} such that i or j is not marked done.
end while
```

Computation of the $(m+1) \times (n+1)$ entries of the local alignment matrix takes $O(mn)$ time. Detection of alignment fragments is done by searching through the $(m+1) \times (n+1)$ matrix for at most $min(m,n)$ times, which takes $O(min(m^2n, mn^2)$ time. Thus the overall time complexity of the algorithm is $O(max(m^3, n^3))$. At any point of time, the program stores a constant number of $m \times m$, $n \times n$ and $m \times n$ matrices. Thus, it consumes $O(max(m^2, n^2))$ memory space.

3.6 Superposition, RMSD and Statistical Significance

The root mean square deviation (RMSD) between 2 aligned and superimposed structures is a measure of closeness between the corresponding residues in 3D space. Let $X_1, X_2, \ldots, X_l$ and $Y_1, Y_2, \ldots, Y_l$ be the position vectors of the transformed and aligned residues. RMSD is calculated as $RMSD = \sqrt{\frac{1}{l}\sum_{i=1}^{l} \|X_i - Y_i\|^2}$ From the equivalences, we can compute the rigid transformation of one structure into the other that minimizes the RMSD between the corresponding residues, using the method by Horn [18]. The optimal transformation can then be applied to the appropriate structure to compute the superposition.

Though, the RMSD is an indicator of the closeness of a given structural match between two structures, it is not a good measure for the significance of a match. For this reason, we compute the statistical significance of the match generated. We use the DALI Z-score [19] as a measure of the same. The DALI score is computed using Eqn. 1. The mean score $m(l)$ as a function of size l is calculated as:
$m(l) = 7.95 + 0.71l - 2.59 \times 10^{-4} l^2 - 1.92 \times 10^{-6} l^3$
where $l \leq 400$ and $l = \sqrt{l_A l_B}$. In case of $l > 400$, a linear extrapolation above $m(400)$ is used.
The Z-score is calculated as $Z(A,B) = \frac{2(S(A,B) - m(l))}{m(l)}$

4 Results and Discussion

In order to verify the accuracy and effectiveness of the algorithm developed above, we implemented the algorithm in Java on JDK 1.4.0 running on a Linux machine. JMAT [20], a free linear algebra library was used for the matrix computations. The protein structures were obtained from the PDB [1]. All references to protein classes, folds, and families are taken from SCOP 1.65 [21]. The images of protein structures were generated using Rasmol.

The objectives of the experiments were to: (a) benchmark Matchprot with other softwares, (b) study the effect of parameter variation and (c) to detect circularly permuted proteins. To measure the quality of the alignments root mean square deviation (RMSD) was used and statistical significance was evaluated with the DALI Z-score.

4.1 Study and Comparison of Alignments Generated by Matchprot

In order to check the correctness of the alignments generated by Matchprot, we experimented with more than 30 protein structure pairs. The protein pairs were chosen to ensure variety in terms of the structures and the degree of similarity in the structures. Also, we compared the alignments generated by Matchprot with those generated by standard protein structure comparison programs e.g. CE [7], LOCK [8] and Flexprot [22].

Table 1: Comparison of results for pairwise protein structure comparison from different programs

Protein1 (length)	Protein2 (length)	C.E. RMSD (N)	Flexprot RMSD (N)	LOCK RMSD (N)	Matchprot RMSD (N)	Z - score Matchprot
1DWT:A (152)	2MM1 (153)	0.7 (152)	0.749 (152)	0.656 (147)	0.749 (152)	23.901
5CNA:A (237)	2PEL:A (232)	1.2 (115)	1.987 (119)	1.068 (160)	1.911 (223)	23.884
2ACT (218)	1PPN (212)	1.0 (211)	2.075 (212)	0.741 (157)	0.877 (210)	29.026
1HTI:A (248)	1TIM:A (247)	0.9 (246)	1.023 (247)	0.844 (235)	1.023 (247)	28.072
3SDH:A (146)	1COL:A (204)	3.9 (124)	2.337 (138)	2.951 (89)	8.868 (116)	-8.043

Table 1 shows RMSD and no. of aligned residues for pairwise comparison of 5 protein pairs as reported by 4 protein structure comparison programs. Also, the DALI Z-score calculated on the alignment generated by Matchprot is reported. The first 4 protein pairs are taken from 4 different SCOP classes and chosen to be

very similar to each other, structurally. Lock generates a very low RMSD due to the core superposition step which tries to detect a core of residues explicitly minimizing RMSD. The other programs give similar values for RMSD and no. of aligned residues for 1DWT:A-2MM1, 2ACT-1PPN and 1HTI:A-1TIM:A. Also, note that LOCK reports a lower no. of aligned residues than the other programs except for 5CNA:A-2PEL:A. The difference in the alignments in case of 5CNA:A-2PEL:A is due to presence of a circular permutation in the two sequences. While CE and Flexprot detect only about half of the equivalences, LOCK detects some more equivalences due to atomic superposition carried out by it. Matchprot detects all the equivalences which involves nearly the whole structure. The low RMSD and high Z-score further confirms a good alignment.

Figures 1, 2, 3 and 4 show the alignment graphs of structural alignments generated by LOCK, CE, Flexprot and Matchprot for the first 4 protein pairs. As a structure alignment is viewed as a set of equivalences the order of which do not matter, we chose the index of an equivalence after sorting the equivalences on the basis residue numbers of one of the proteins. In an alignment graph, the equivalence index is plotted on the X-axis while the residue numbers of the participating residues are plotted on the Y axis. I can be seen that the equivalences retrieved are almost same in all the cases except 5CNA:A-2PEL:A. The alignment graph for 5CNA:A-2PEL:A is explained in section 4.3. Figure 5 shows the superpositions generated by Matchprot for the first four protein pairs.

3SDH:A and 1COL:A have low structural similarity (as shown by RMSDs given by other protein structure comparison programs). Moreover 1COL:A has about 25% more residues than 3SDH:A. Thus, Matchprot is unable to detect the optimal alignment in this case as detected by the other programs. However, as figure 10(b) suggests, it still detects a reasonably close alignment.

4.1.1 Randomly Selected Protein Pairs from 4 SCOP Families

We also performed all-pair comparison on randomly chosen protein pairs from 4 different SCOP families, in order to ensure a more systematic experimentation. Five proteins were chosen from the SCOP family A.1.1.1 (sunid 46459), seven were chosen from SCOP family B.1.1.1 (sunid 48727), six were chosen from SCOP family C.1.1.1 (sunid 51352) and six were chosen from SCOP family D.1.1.2 (sunid 81307). So, a total 61 pairwise structure comparisons were made, each comparison being repeated for 1600 different parameter values making a total of 97600 runs of the program. The results were compared with those obtained from CE [7]. Percentage aligned is calculated with respect to maximum possible alignment length l_m (which is the minimum of lengths of the two sequences), as $(l_a/l_m) \times 100$, l_a being the actual length of alignment.

Tables 2, 3, 4, and 5 show representative results from the experiments. The average difference in RMSD values (in Å) given by Matchprot and CE are 1.04 for A.1.1.1, 0.29 for B.1.1.1, 0.07 for C.1.1.1 and 0.009 for D.1.1.2. The average difference in percentage of residues aligned in alignments given by Matchprot and CE are 9.6 for A.1.1.1, 2.19 for B.1.1.1, 0.67 for C.1.1.1 and 0.25 for D.1.1.2.

The families were chosen from 4 different SCOP classes covering the breadth of the protein fold space, and structures were chosen randomly from each SCOP family. Thus experimentally, it can be said that Matchprot generates alignments very close to those generated by CE. The relatively high deviation in the results in family A.1.1.1 are due to `d1mbwa_`, which is has high structural divergence from the rest of the structures (RMSD 3.3, 3.5, 3.5 and 3.7 at % aligned 83, 86, 87 and 89 rpoduced by CE). So, Matchprot can detect the structural similarity, in case a high (family-level) structural similarity exists between the structures.

Table 2: Minimum RMSD obtained for proteins from SCOP family a.1.1.1

Protein 1	Protein 2	RMSD Matchprot	RMSD CE	% Aligned Matchprot	% Aligned CE
d1dlya_	d1dlwa_	1.451	1.06	98.27	99.13
d1idra_	d1dlwa_	1.174	0.99	99.13	99.13
d1idra_	d1dlya_	2.024	1.20	94.21	95.04
d1mwba_	d1dlwa_	5.467	3.33	94.82	86.20
d1mwba_	d1dlya_	4.400	3.47	96.69	89.25
d1mwba_	d1idra_	5.248	3.71	13.82	87.80
d1ngka_	d1dlwa_	2.632	2.28	100.0	100.0
d1ngka_	d1dlya_	3.409	2.30	96.69	95.86
d1ngka_	d1idra_	3.245	2.21	91.33	92.06
d1ngka_	d1mwba_	5.408	3.46	84.55	86.99

Table 3: Minimum RMSD obtained for proteins from SCOP family b.1.1.1

Protein 1	Protein 2	RMSD Matchprot	RMSD CE	% Aligned Matchprot	% Aligned CE
d12e8l1	d12e8h1	2.552	2.05	92.52	97.19
d15c8h1	d12e8h1	1.594	0.92	99.11	94.95
d15c8h1	d12e8l1	1.901	1.73	96.26	97.19
d1a0qh1	d12e8h1	1.153	0.94	100.0	99.00
d1a0qh1	d12e8l1	1.954	1.93	93.45	90.00
d1a0qh1	d15c8h1	1.004	0.78	98.18	99.00
d1a14h_	d12e8h1	1.005	1.13	98.19	98.33
d1a14h_	d12e8l1	2.632	1.96	95.32	94.39
d1a14h_	d15c8h1	1.499	1.30	98.19	94.95
d1a14h_	d1a0qh1	1.659	1.00	97.27	100.0
d1a2ya_	d12e8h1	2.437	2.02	95.32	98.13
d1a2ya_	d12e8l1	0.759	0.76	100.0	100.0
d1a2ya_	d15c8h1	1.686	1.60	96.26	98.13
d1a2ya_	d1a0qh1	1.864	1.80	93.4	90.00
d1a2ya_	d1a14h_	2.544	2.17	95.32	98.13

Table 4: Minimum RMSD obtained for proteins from SCOP family c.1.1.1

Protein 1	Protein 2	RMSD Matchprot	RMSD CE	% Aligned Matchprot	% Aligned CE
d8timb_	d8tima_	0.579	0.61	99.59	100.0
d1amk__	d8tima_	1.251	1.23	98.38	98.78
d1amk__	d8timb_	1.335	1.41	97.97	98.78
d1aw1a_	d8tima_	1.390	1.43	96.76	98.78
d1aw1a_	d8timb_	1.435	1.54	96.76	98.38
d1aw1a_	d1amk__	1.289	1.19	98.0	98.00
d1aw2a_	d8tima_	1.322	1.32	97.97	98.78
d1aw2a_	d8timb_	1.273	1.22	97.97	98.38
d1aw2a_	d1amk__	1.613	1.56	97.6	98.00
d1aw2a_	d1aw1a_	0.810	0.95	99.21	100.0
d1tpe__	d8tima_	1.088	1.03	98.78	98.78
d1tpe__	d8timb_	1.012	0.96	98.78	98.78
d1tpe__	d1amk__	0.922	1.05	99.19	100.0
d1tpe__	d1aw1a_	1.639	1.54	97.18	98.39
d1tpe__	d1aw2a_	1.406	1.29	97.99	98.39

Table 5: Minimum RMSD obtained for proteins from SCOP family d.1.1.2

Protein 1	Protein 2	RMSD Matchprot	RMSD CE	% Aligned Matchprot	% Aligned CE
d1b20a_	d1a2pa_	0.299	0.30	100.0	100.0
d1baoa_	d1a2pa_	0.198	0.20	100.0	100.0
d1baoa_	d1b20a_	0.371	0.37	100.0	100.0
d1bnr__	d1a2pa_	1.554	1.57	99.07	100.0
d1bnr__	d1b20a_	1.582	1.59	99.08	100.0
d1bnr__	d1baoa_	1.552	1.57	99.07	100.0
d1bsaa_	d1a2pa_	0.156	0.16	100.0	100.0
d1bsaa_	d1b20a_	0.296	0.30	100.0	100.0
d1bsaa_	d1baoa_	0.194	0.19	100.0	100.0
d1bsaa_	d1bnr__	1.557	1.57	99.06	100.0
d1goya_	d1a2pa_	0.471	0.47	100.0	100.0
d1goya_	d1b20a_	0.562	0.56	100.0	100.0
d1goya_	d1baoa_	0.449	0.51	100.0	100.0
d1goya_	d1bnr__	1.452	1.45	100.0	100.0
d1goya_	d1bsaa_	0.497	0.50	100.0	100.0

4.2 Variation of RMSD with Parameters

In order to further ascertain the predictability of the program, we studied the variations in results with the parameters of the program e.g. α, T and g. Data regarding the variation was extracted from the experiments described in section 4.1.1.

α (equation 2) governs the decay of nearness function, thus governing the range of distances which influence the structure graph heavily. An increase in α results in higher distances contributing significantly to the nearness graph. This results in improvement of accuracy with increase in α, for low value of α. Figure 10(a) shows this trend in the average value rmsd.

T (equation 8) is the cutoff for difference in Fiedler vector values, above which, the respective vertices are considered dissimilar. Thus, a higher value of T gives more tolerance to the match, thereby increasing

the RMSD. Also, it is observed that for very low values of T, the RMSD becomes high. This is due to spurious matches which dominate the correct ones due to inherent noise in the data.

The gap penalty g (equation 9) doesn't have a profound effect on the RMSD, except when it is to very high values compared to T, the RMSD tends to increase. This is expected, as the program detects multiple local alignments, which act as replacement for gapped ones.

Figures 6, 7, 8 and 9 show graphs depicting the variation of average RMSD with the T and g for various values of α for protein from 4 different classes. The increase of RMSD with T is clearly seen in all the graphs. However, for low values of α, high values of RMSD are observed for very low values of T due to picking up of more noise. Thus, it is observed that the program gives results in accordance with our expectations.

4.3 Detection of Circular Permutations

To check the capability of Matchprot in detecting circular permutations, we performed an all-pair comparison on a non-redundant database of 84 proteins taken from the SCOP fold "Concanavalin like lectins/glucanases" (sunid 49898), which is believed to have many instances of circular permutations. The fold contains 592 domains. A set 84 proteins which do not share more than 90% sequence similarity over at least 90% of the total length with any other protein in the set was prepared. Matchprot was used structurally align all the protein pairs (3486 of them) over 256 different values of parameters making a total of 892416 pairwise structure comparison. The experiment took average time of 0.176 s per pairwise comparison on an Intel Pentium4 2.4 GHz workstation.

Table 6: 10 Significant circular permutations found in SCOP fold "Concanavalin A-like lectins/glucanases"

Protein 1	Protein 2	Naligned	RMSD	Z-score	No. of fragments
1glh__	1cpn__	207	0.584	30.472	2
1glh__	1ajoa_	212	1.223	30.456	2
1gbg__	1ajoa_	212	1.272	30.006	2
1cpn__	1ajoa_	206	0.694	29.897	2
1gbg__	1cpn__	208	1.044	29.777	2
1qmo.1	1scs__	229	1.299	28.782	2
1qmo.1	1dgla_	229	1.262	28.672	2
1qmo.1	1h9wa_	228	1.379	28.646	2
1ajoa_	1ajka_	210	2.411	27.363	2
1glh__	1ajka_	211	2.544	27.259	2

85 pairs of proteins showing significant (Z-score > 5) circular permutations were detected. The top 10 sorted on the Z-score are shown in Table 2. The Conserved Domains Database [24] shows that all of 1GLH, 1CPN, 1AJO, 1GBG and 1AJK, have the "Glycosyl hydrolases family 16" domain. The regions in sequence that have the domain are: from 25 to 209 in 1GLH, 0 to 151 in 1CPN, 113 to 213 and 0 to 83 in 1AJO, 25 to 209 in 1GBG, and 0 to 126 and 156 to 213 in 1AJK. Thus, it is clear that all the 7 pairs of circular permutation detected are correct. Also, all of 1QMO (chain A and E), 1SCS, 1DGL, and 1H9W have the "Legume lectins beta" and "Legume lectins alpha" domain. The regions in the sequence containing the "Legume lectins beta" domain are 1 to 112 in chain A and 1 to 67 in chain E in 1QMO (which constitutes the d1qmo.1 SCOP domain), 124 to 237 and 1 to 68 in 1SCS, 124 to 237 and 1 to 68 in 1DGL, and 124 to 237 and 1 to 68 in 1H9W. The regions in sequence which contain the "Legume lectins alpha" domain are 73 to 119 in 1QMO chain E, 74 to 120 in 1SCS, 74 to 120 in 1DGL, and 74 to 120 in 1H9W. So, the 3 circular permutations detected involving these residues are also correct.

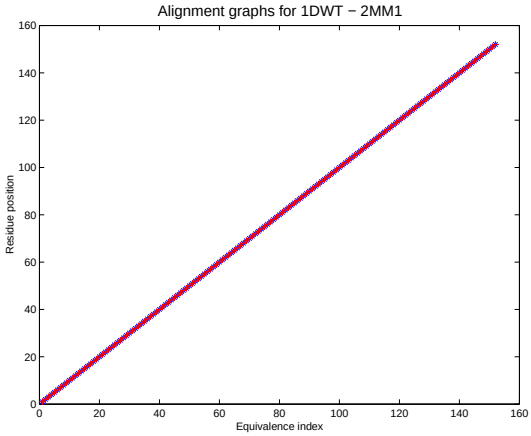
Figure 10(c) shows the alignment graph and structural superposition generated by Matchprot for 1GLH-1AJO:A. The circular permutation is characterized by a negative jump in the alignment graph. The 4 circularly permuted fragment from two structures are shown in different colors in the superposition.

Also, the alignment graph of figure 5 between 2PEL:A-5CNA:A shows that Matchprot is able to detect circular permutation while CE is not. It was seen that Flexprot and LOCK were also not able to detect the full circular permutation. Also, the web version of DALI could not detect the circular permutation. Thus experimentally we see, that circular permutations are detected correctly by Matchprot whereas the other programs fail to do so.

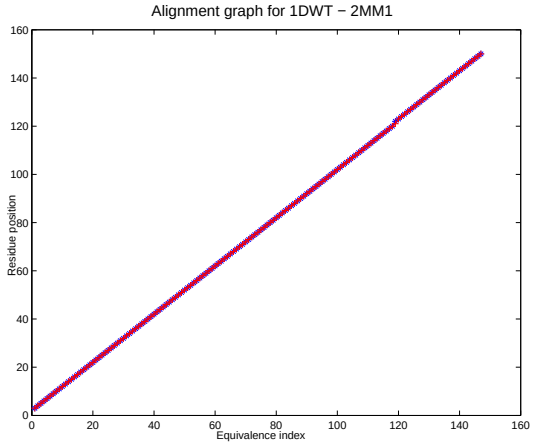
5 Conclusion and Future Work

In this article, we have described the development and experimental validation of a novel algorithm, Matchprot, for comparing protein structures. The proposed algorithm takes $O(n^3)$ time, and is extremely fast. It is competitive with other state of the art methods and it outperforms them in detecting circularly permuted proteins.

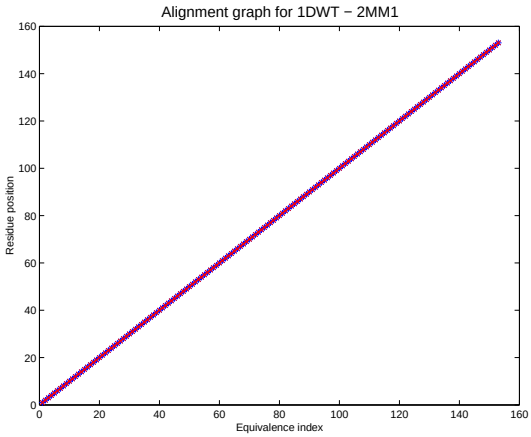
This work opens up many interesting directions. The idea of computing 1-dim neighbourhood preserving projections can be accomplished by various methods like Prinicipal Components Analysis, Topographic mappings etc. It would be interesting to study their applicability for protein structure comparison. It also touches upon the possibility of using graph matching techniques for protein structure comparison. Another interesting direction is to use the pair wise similarity function developed here for multiple structure alignment along with existing multiple alignment techniques.



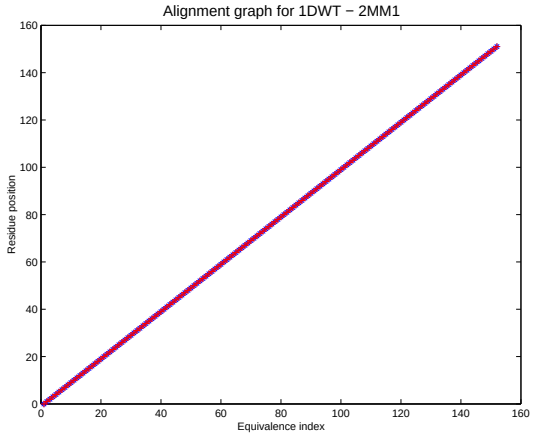
Flexprot



LOCK

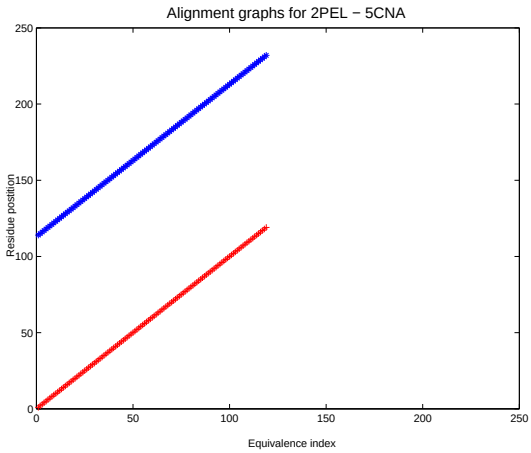


CE

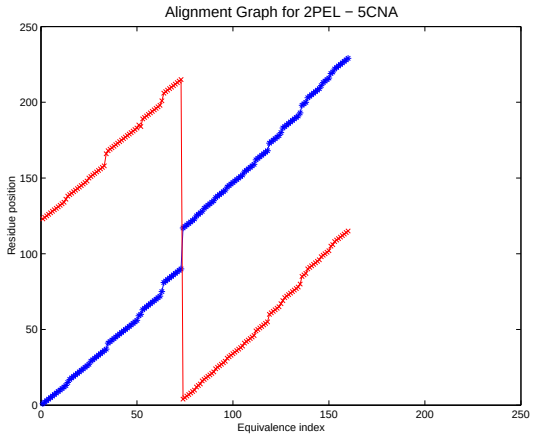


Matchprot

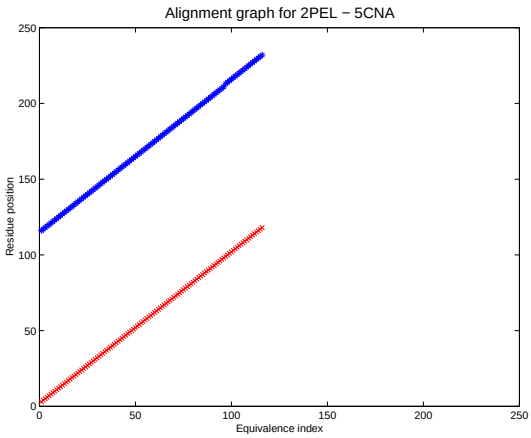
Figure 1: Alignment graph for proteins 1DWT and 2MM1 as calculated by the 3 standard structure alignment programs and Matchprot. In the x-axis the index of the equivalenced residue pairs sorted on the position of the first protein is plotted. In the y-axis the actual position of the residues is plotted.



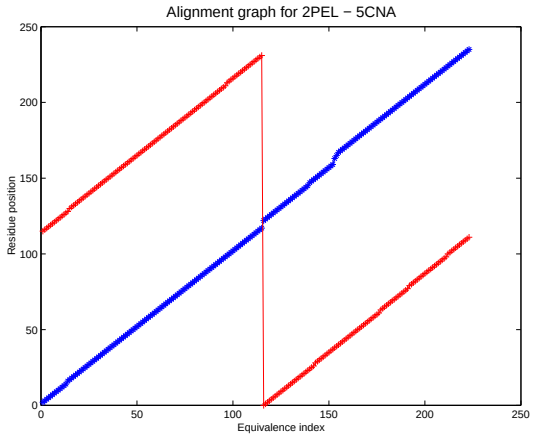
Flexprot



LOCK

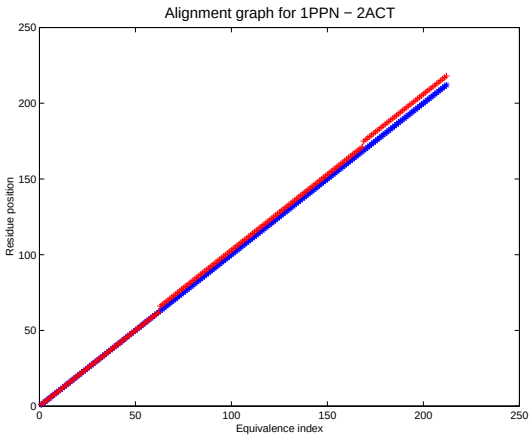


CE

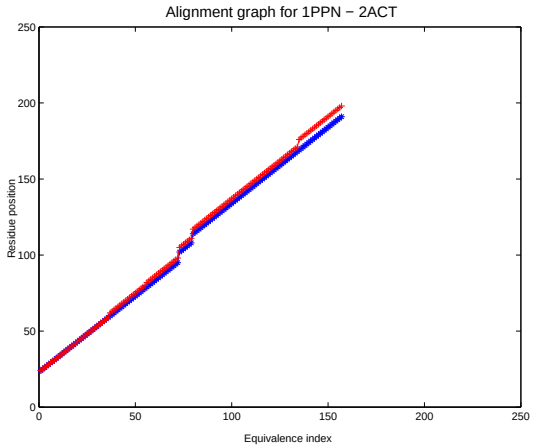


Matchprot

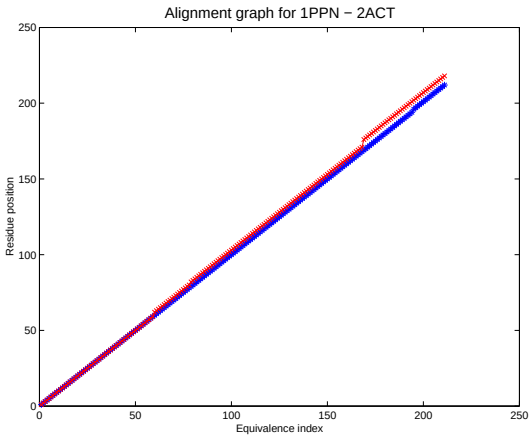
Figure 2: Alignment graph for proteins 2PEL and 5CNA as calculated by the 3 standard structure alignment programs and Matchprot. In the x-axis the index of the equivalenced residue pairs sorted on the position of the first protein is plotted. In the y-axis the actual position of the residues is plotted.



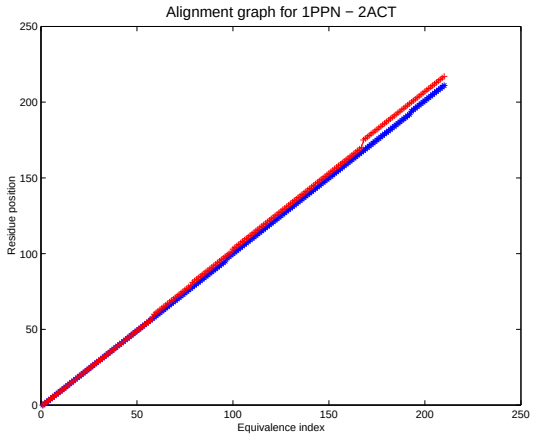
Flexprot



LOCK

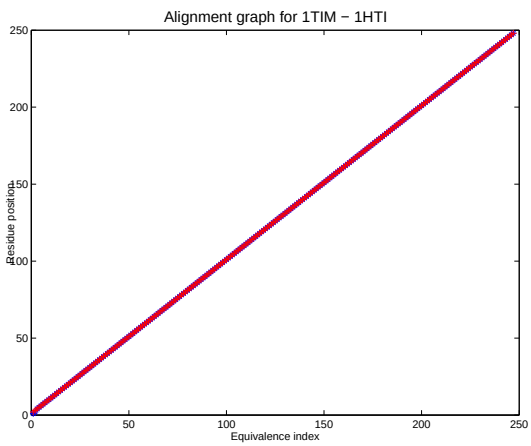


CE

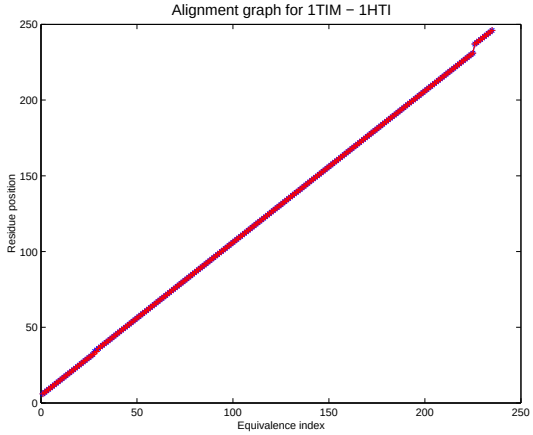


Matchprot

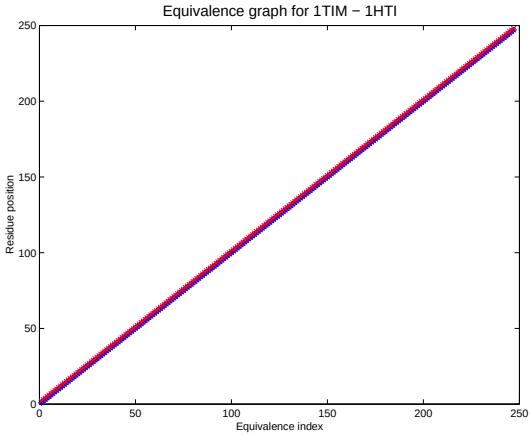
Figure 3: Alignment graph for proteins 1PPN and 2ACT as calculated by the 3 standard structure alignment programs and Matchprot. In the x-axis the index of the equivalenced residue pairs sorted on the position of the first protein is plotted. In the y-axis the actual position of the residues is plotted.



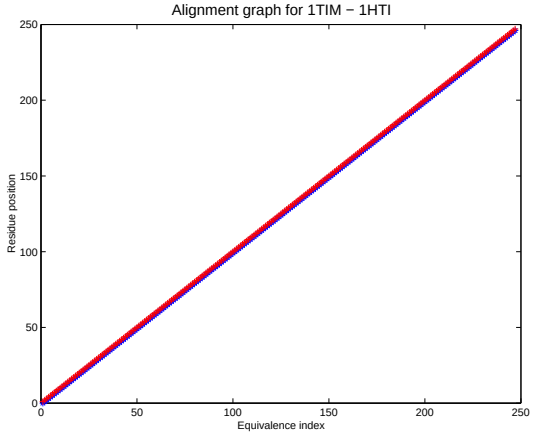
Flexprot



LOCK

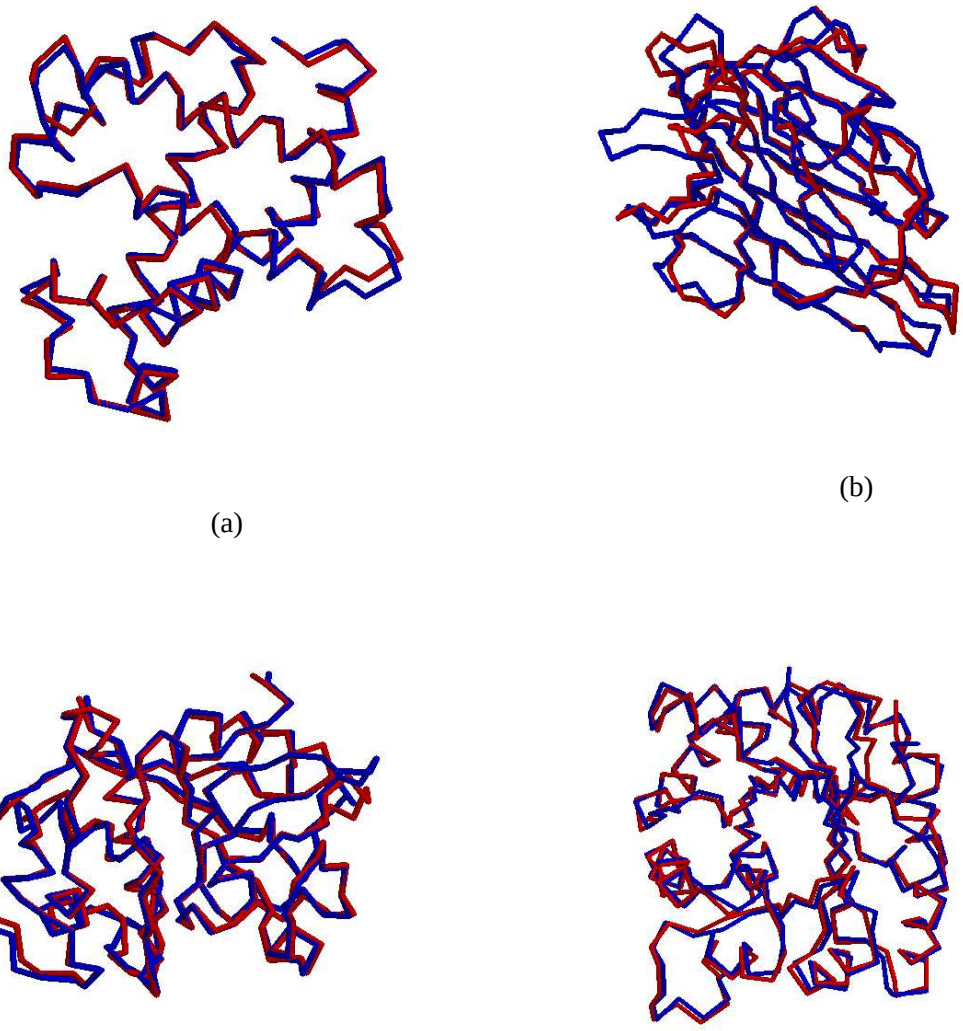


CE



Matchprot

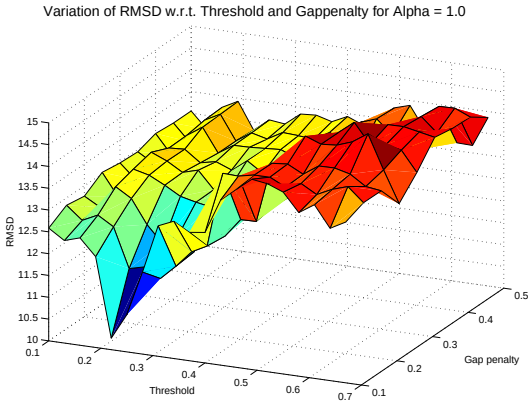
Figure 4: Alignment graph for proteins 1TIM and 1HTI as calculated by the 3 standard structure alignment programs and Matchprot. In the x-axis the index of the equivalenced residue pairs sorted on the position of the first protein is plotted. In the y-axis the actual position of the residues is plotted.



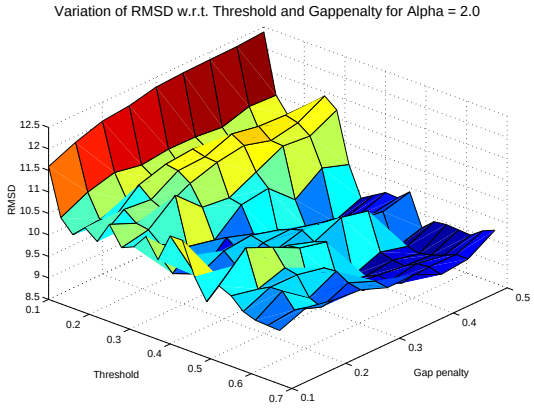
(a) (b)

(c) (d)

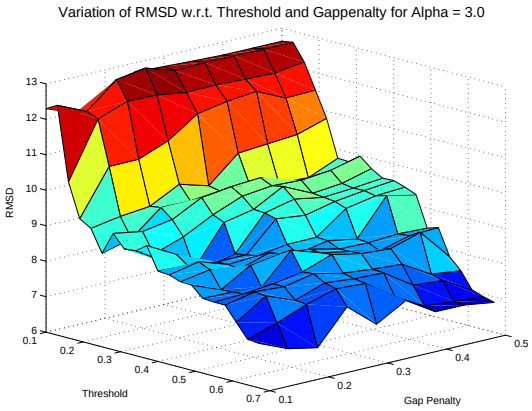
Figure 5: Structural superposition for the protein backbones of (a)1DWT and 2MM1 (b) 2PEL and 5CNA (c) 2ACT and 1PPN and (d) 1TIM and 1HTI as calculated by Matchprot



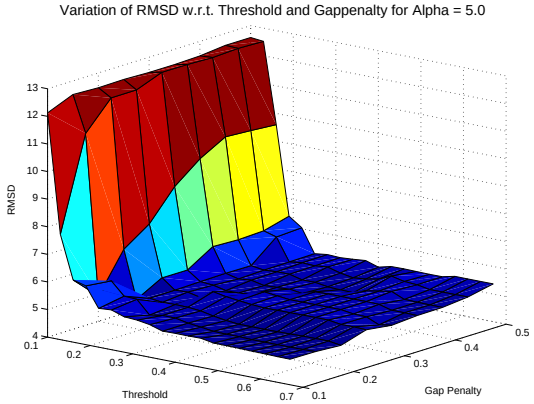
a



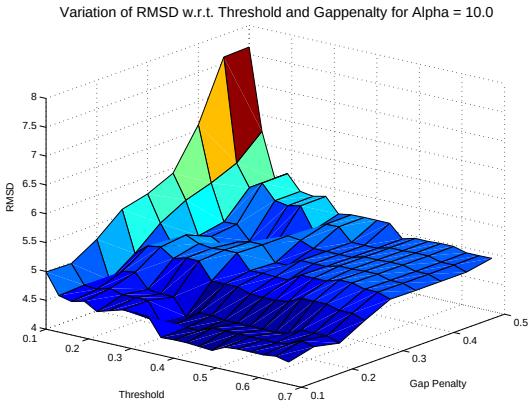
b



c

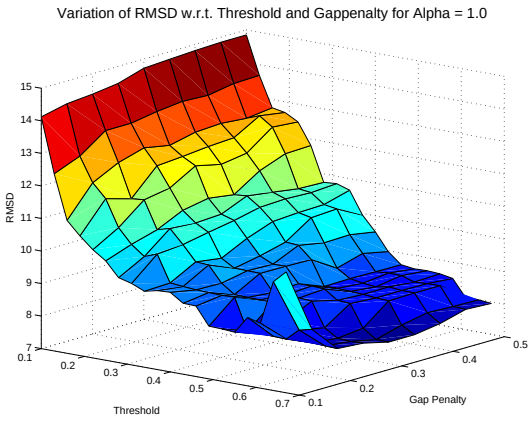


d

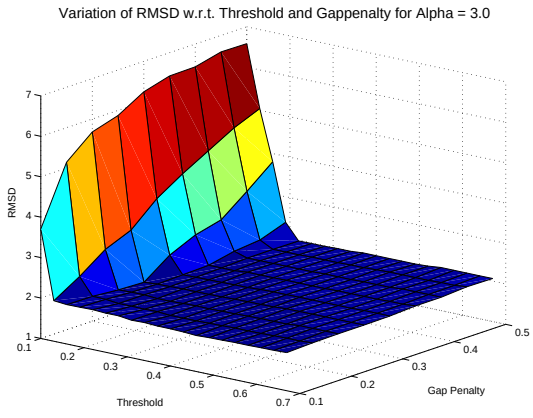


e

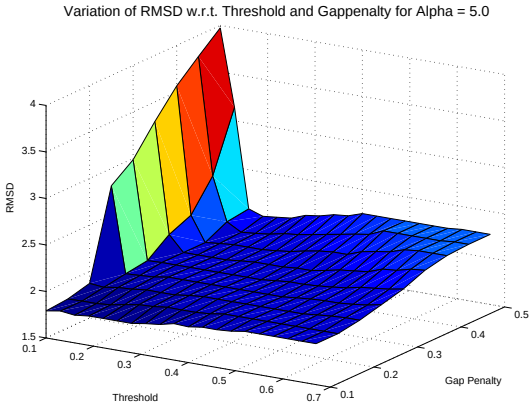
Figure 6: Variation of average RMSD for a set of 10 randomly chosen pair of polypeptides chosen from the SCOP family A.1.1.1. with respect to Threshold(x-axis) and Gap Penalty(y-axis) for (a) $\alpha = 1$, (b) $\alpha = 2$, (c) $\alpha = 3$, (d) $\alpha = 5$ and (e) $\alpha = 10$.



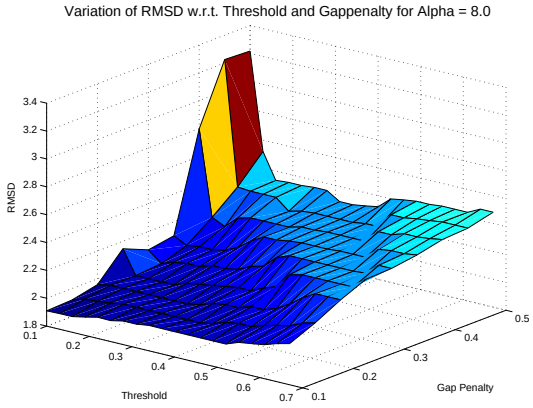
a



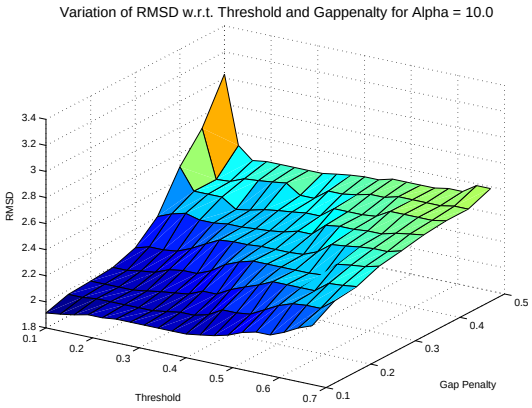
b



c

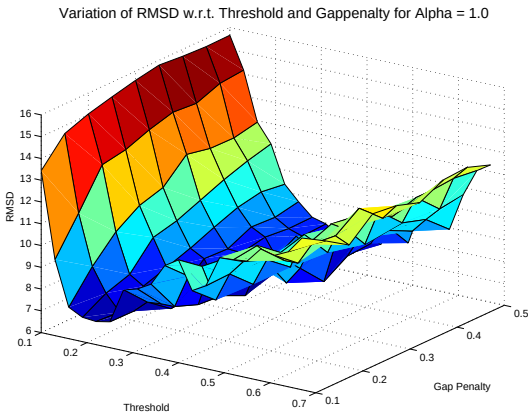


d

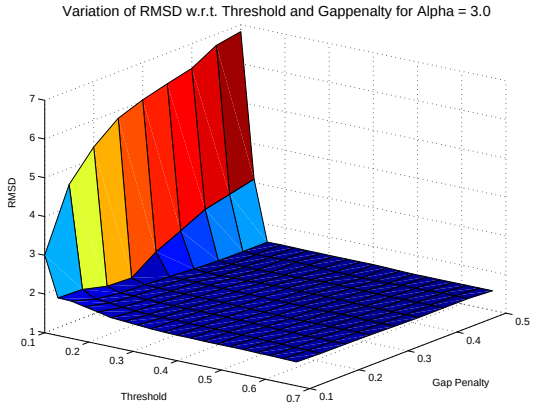


e

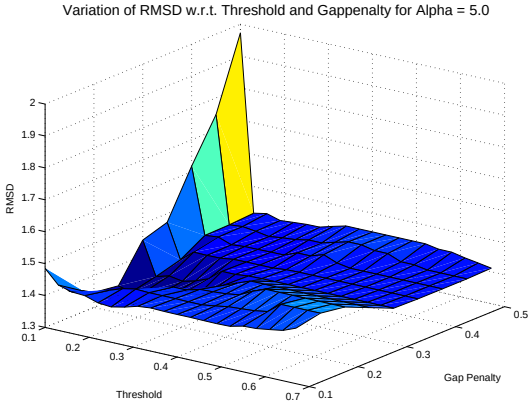
Figure 7: Variation of average RMSD for a set of 21 randomly chosen pair of polypeptides chosen from the SCOP family B.1.1.1. with respect to Threshold(x-axis) and Gap Penalty(y-axis) for (a) $\alpha = 1$, (b) $\alpha = 3$, (c) $\alpha = 5$, (d) $\alpha = 8$ and (e) $\alpha = 10$.



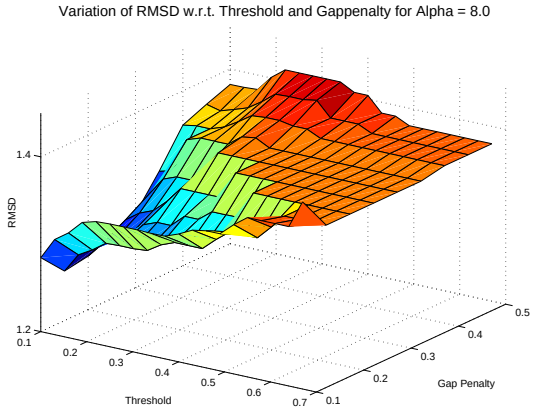
a



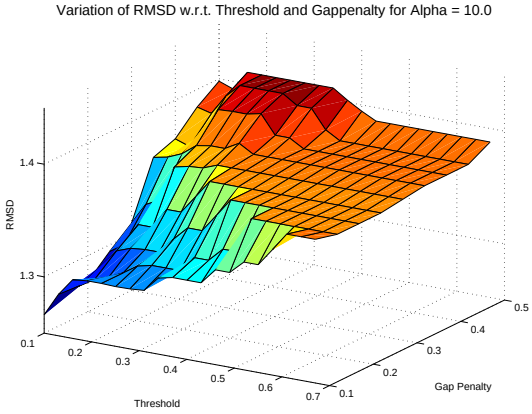
b



c

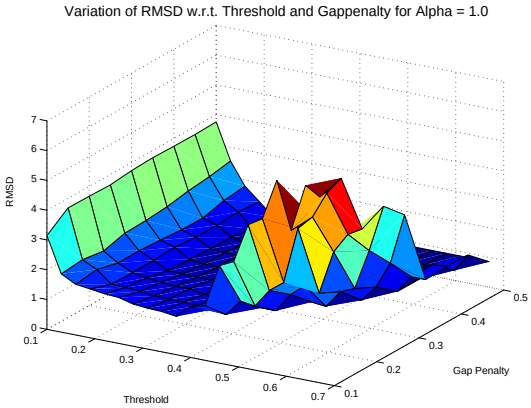


d

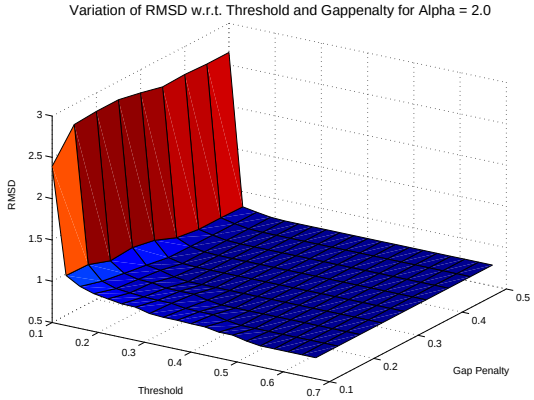


e

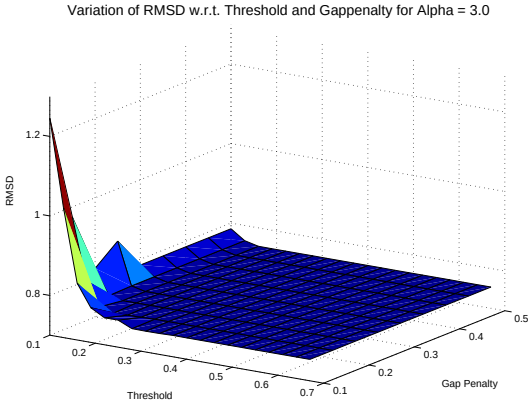
Figure 8: Variation of average RMSD for a set of 15 randomly chosen pair of polypeptides chosen from the SCOP family C.1.1.1. with respect to Threshold(x-axis) and Gap Penalty(y-axis) for (a) $\alpha = 1$, (b) $\alpha = 3$, (c) $\alpha = 5$, (d) $\alpha = 8$ and (e) $\alpha = 10$.



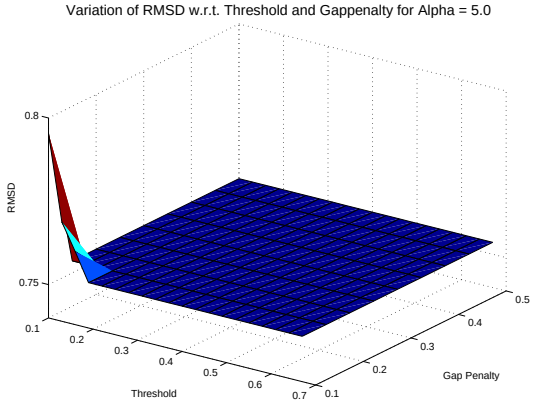
a



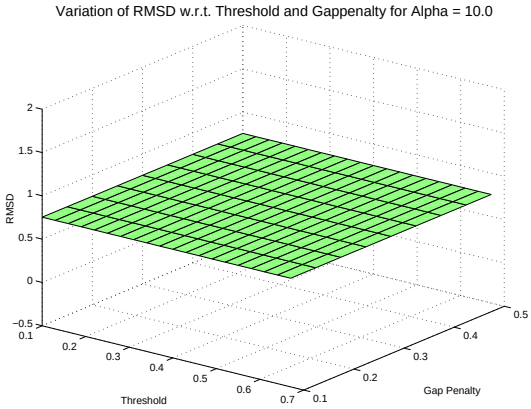
b



c

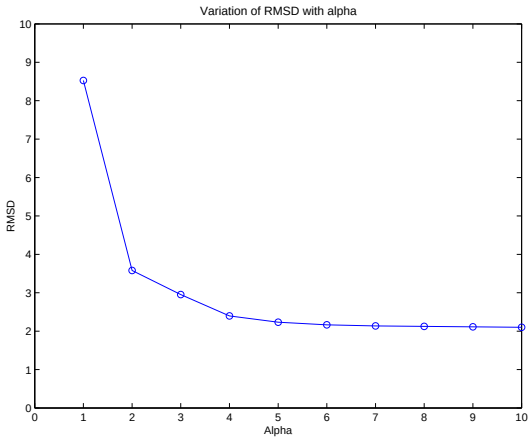


d



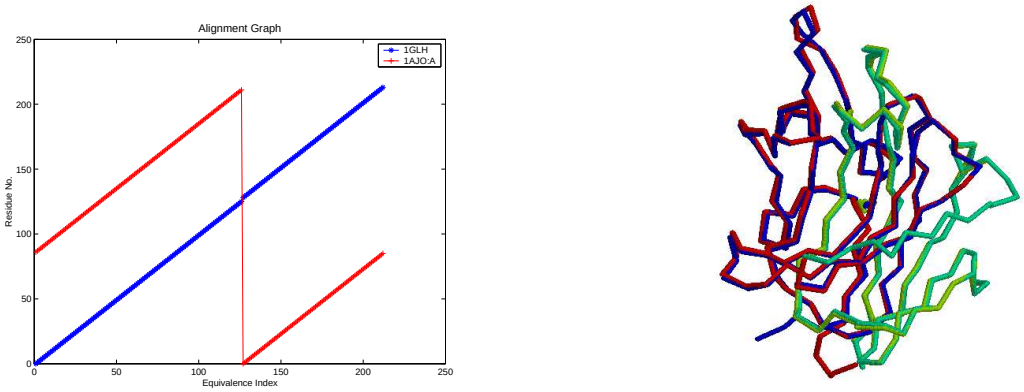
e

Figure 9: Variation of average RMSD for a set of 15 randomly chosen pair of polypeptides chosen from the SCOP family D.1.1.2. with respect to Threshold(x-axis) and Gap Penalty(y-axis) for (a) $\alpha = 1$, (b) $\alpha = 2$, (c) $\alpha = 3$, (d) $\alpha = 5$ and (e) $\alpha = 10$.



(a)

(b)



(c)

Figure 10: (a) α vs. Average RMSD plot for Matchprot. (b) Similar structural superposition generated by Matchprot and CE for 3SDH - 1COL (c) Alignment graph and Structural superposition generated by Matchprot for 1GLH - 1AJO:A.

Acknowledgements

The authors are highly grateful to Dr. Nagasuma R. Chandra, Bioinformatics Center, Indian Institute of Science, for all the valuable suggestions and insightful comments.

References

[1] Z Feng G. Gilliland T N Bhat H Weissig I N Shindyalov H M Berman, J Westbrook and P E Bourne. The protein data bank. *Nucleic Acids Research*, 28:235–242, 2000.

[2] I D Kuntz T F Havel and G M Crippen. The theory and practice of distance geometry. *Bulletin of Mathematical Biology*, 45:665–720, 1983.

[3] William R Taylor and Christine A. Orengo. Protein structure alignment. *Journal of Molecular Biology*, 208:1–22, 1989.

[4] C A Orengo and W R Taylor. Ssap: Sequential structure alignment program for protein structure comparison. *Methods in Enzymology*, 266:617–635, 1996.

[5] Liisa Holm and Chris Sander. Protein structure comparison by alignment of distance matrices. *Journal of Molecular Biology*, 233:123–138, 1993.

[6] Liisa Holm and Chris Sander. Mapping the protein universe. *Science*, 273(5275):595–602, 1996.

[7] P E Bourne and I N Shindyalov. Protein structure alignment by incremental combinatorial extension of optimal path. *Protein Engineering*, 11(9):739–747, 1998.

[8] Amit P. Singh and Douglas L. Brutlag. Hierarchical protein structure superposition using both secondary structure and atomic representations. In *Proceedings of International Conference on Intelligent Systems in Molecular Biology*, volume 5, pages 284–293, 1997.

[9] Jessica Shapiro and Douglas Brutlag. Foldminer: Structural motif discovery using an improved superposition algorithm. *Protein Science*, 13(278-294), 2004.

[10] Amit Fliess Shai Uliel and Ron Unger. Naturally occuring circular permutations in proteins. *Protein Engineering*, 14(8):533–542, 2001.

[11] Ylva Lindqvist and Gunter Schneider. Circular permutations of natural protein sequences: structural evidence. *Current Opinion in Structural Biology*, 7:422–427, 1997.

[12] B. Mohar. Some applications of laplace eigenvalues of graphs. In G. Hahn and G. Sabidussi, editors, *Graph Symmetry: Algebraic Methods and Applications*, pages 225–275. Kluwer, 1997.

[13] F.R.K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

[14] Shinji Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE transactions on pattern analysis and machine intelligence*, 10(5):695–703, 1988.

[15] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(4):377–388, 1996.

[16] S B Needleman and C D Wunsch. A general method applicable to the search of similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48:443–453, 1970.

[17] T.F. Smith and M.S. Waterman. The identification of common molecular subsequences. *Journal of Molecular Biology*, 1981.

[18] B K P Horn. Closed form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America*, 4(4):629–642, 1987.

[19] Liisa Holm and Chris Sander. Dictionary of recurrent domains in protein structures. *Proteins: Structure, Function and Genetics*, 33:88–96, 1998.

[20] Java matrix tools package. http://jmat.sourceforge.org.

[21] A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. SCOP: a structural classification of proteins database for the investigation of sequences and structures. *Journal of Molecular Biology*, 247:536–540, 1995.

[22] H J Wolfson M Shatsky and R Nussinov. Flexible protein alignment and hinge detection. *Proteins: Structure, Function, and Genetics*, 48:242–256, 2002.

[23] Sourangshu Bhattacharya and Chiranjib Bhattacharyya. Comparison of protein structures using spectral graph theory. Technical report, Dept. of Computer Science & Automation, Indian Institute of Science, 2004.

[24] Carol DeWeese-Scott Natalie D. Fedorova Lewis Y. Geer Siqian He David I. Hurwitz John D. Jackson Aviva R. Jacobs Christopher J. Lanczycki Cynthia A. Liebert Chunlei Liu Thomas Madej Gabriele H. Marchler Raja Mazumder Anastasia N. Nikolskaya Anna R. Panchenko Bachoti S. Rao Benjamin A. Shoemaker Vahan Simonyan James S. Song Paul A. Thiessen Sona Vasudevan Yanli Wang Roxanne A. Yamashita Jodie J. Yin Aron Marchler-Bauer, John B. Anderson and Stephen H. Bryant. Cdd: a curated entrez database of conserved domain alignments. *Nucleic Acids Research*, 31(1):383–387, 2003.

A The Algorithm

ALGORITHM : MATCHPROT

Input:
Two sets of coordinate lists X^1, Y^1, Z^1 and X^2, Y^2, Z^2, of lengths m_1 and m_2 respectively specified in the order they appear in the peptide chain.
α: a parameter governing the decay of the adjacency function.
Threshold: a parameter governing the allowed tolerance in fiedler vector values between the residues of the two structures.
GapPenalty: a parameter governing the penalty assigned for aligning a residue to nothing i.e. eliminating it from the structural alignment.

Output:
The equivalence lists E^1 and E^2 of length n.
The optimally transformed coordinates of the second structure in the coordinate system of the first structure, X', Y', Z' of length m_2.
RMSD: Root mean square deviation of the resultant superposition.
zscore: DALI Z-score of the resultant superposition.

Algorithm:

for all l such that $l \in \{1,2\}$ **do** {For both the structures}

Compute the distance matrix $\mathbf{D}$ with entries given by

$$D^l_{i,j} = \sqrt{(X^l_i - X^l_j)^2 + (Y^l_i - Y^l_j)^2 + (Z^l_i - Z^l_j)^2} \ , 1 \leq i,j \leq m_l$$

Compute the adjacency matrix $\mathbf{A}$ with entries given by

$$A^l_{i,j} = \begin{cases} e^{\frac{-D^l_{i,j}}{\alpha}} & \text{, if } i \neq j \\ 0 & \text{, otherwise} \end{cases}$$

Compute the diagonal degree matrix $\mathcal{D}$ as:

$$\mathcal{D}^l_{i,j} = \begin{cases} \sum_{k=1}^{n} A^l_{i,k} & \text{, if } i = j \\ 0 & \text{, otherwise} \end{cases}$$

Compute the Laplacian matrix as:

$$\mathbf{L}^l = \mathcal{D}^l - \mathbf{A}^l$$

Compute the eigenvalues $\lambda^l_i,\ 1 \leq i \leq m_l$ and eigenvectors $\Phi^l_i,\ 1 \leq i \leq m_l$ of the Laplacian as:

$$\mathbf{L}^l \Phi^l_i = \lambda^l_i \Phi^l_i,\ 1 \leq i \leq m_l$$

Sort the eigenvectors according to their respective eigenvalues, thus getting, $\Phi^l_1, \Phi^l_2, \ldots, \Phi^l_{m_l}$ such that $\lambda^l_1 \leq \lambda^l_2 \leq \cdots \leq \lambda^l_{m_l}$.

Calculate the Fiedler vector V^l as:

$$V^l = \min_i \Phi^l_i \ \text{, such that } \lambda^l_i \neq 0.$$

Normalize the Fiedler vector V^l to get:

$$\sum_{i=1}^{m_l} \left(V^l_i\right)^2 = m_l$$

end for

Compute similarity matrix as:

$$Sim_{i,j} = Threshold - |V^1_i - V^2_j| \ , \ 1 \leq i \leq m_1,\ 1 \leq j \leq m_2$$

for $i = 0$ to m_1 **do** {Local alignment matrix}

  for $j = 0$ to m_2 **do**

    Compute

$$LA_{i,j} = \begin{cases} 0 & \text{, if } i = 0 \text{ or } j = 0 \\ \max \begin{cases} LA_{i-1,j-1} + Sim_{i,j} \\ LA_{i-1,j} - GapPenalty \\ LA_{i,j-1} - GapPenalty \\ 0 \end{cases} & \text{, otherwise} \end{cases}$$

  end for

end for

Alignment $\leftarrow \phi$ {alignment set is null}

Compute $highest = max_{i,j}\ LA_{i,j}$.

Compute $(p1, p2) = arg\max_{i,j}\ LA_{i,j}$.
while $highest > 0$ **do**
  $Alignment \leftarrow Alignment \cup traceback(p1, p2)$ {*traceback* returns the alignment obtained by tracing back from it's argument}
  Mark the rows and columns of LA corresponding to the residues returned in the current alignment *done*.
  Compute $highest = max_{i,j}\ LA_{i,j}$ such that i or j is not marked *done*.
  Compute $(p1, p2) = arg\max_{i,j}\ LA_{i,j}$ such that i or j is not marked *done*.
end while
Generate the lists of equivalent residues E^1 and E^2 by eliminating residues that are aligned to gaps.
Calculate the optimal rigid body transformation (rotation and translation) T of the second structure superposing it with the first one, so that the equivalenced residue pairs have minimum RMSD (This is done using the method described in [18]).
for $i = 1$ to m_2 **do** {Calculating transformed coordinates}
  $(X_i', Y_i', Z_i') \leftarrow T(X_i^2, Y_i^2, Z_i^2)$
end for
Compute the RMSD and Z-score of the superposition generated above, as descirbed in section 3.6.
