Fast Detection of Common Geometric Substructure in Proteins [*]

L. Paul Chew [†] Dan Huttenlocher [‡] Klara Kedem [§]
Jon Kleinberg [¶]

Abstract

We consider the problem of identifying common three-dimensional substructures between proteins. Our method is based on comparing the shape of the α-carbon backbone structures of the proteins in order to find 3D rigid motions that bring portions of the geometric structures into correspondence. We propose a geometric representation of protein backbone chains that is compact yet allows for similarity measures that are robust against noise and outliers. We represent the structure of the backbone as a sequence of unit vectors, defined by each adjacent pair of α-carbons; we then define a measure of the similarity of two protein structures based on the RMS (root mean squared) distance between corresponding orientation vectors of the two proteins.

Our measure has several advantages over measures that are commonly used for comparing protein shapes, such as the minimum RMS distance between the 3D positions of corresponding atoms in two proteins. This new measure behaves well for identifying common substructures, in contrast with position-based measures where the nonmatching portions of the structure dominate the measure. At the same time, it avoids the quadratic space and computational difficulties associated with methods based on distance matrices and contact maps. We show applications of our approach to detecting common contiguous substructures in pairs of proteins, as well as the more difficult problem of identifying common protein domains (i.e., larger substructures that are not necessarily contiguous along the protein chain).

[*]A preliminary version of this work appears in the Proceedings of the 3rd ACM RECOMB Intl. Conference on Computational Molecular Biology

[†]Department of Computer Science, Cornell University, Ithaca NY 14853. Supported in part by the National Science Foundation (EIA-9726388), by DARPA through ONR (N00014-96-1-0699), and by a grant from the US-Israeli Binational Science Foundation.

[‡]Department of Computer Science, Cornell University, Ithaca NY 14853. Supported in part by gifts from Xerox, Microsoft and Intel and by a grant from the US-Israeli Binational Science Foundation.

[§]Department of Computer Science, Cornell University, Ithaca NY 14853, and Department of Mathematics and Computer Science, Ben-Gurion University of the Negev, Beer-Sheva 84105, Israel. Supported in part by a grant from the Israel Science Foundation, by the Mary Upson Award from the College of Engineering at Cornell University, and by a grant from the US-Israeli Binational Science Foundation.

[¶]Department of Computer Science, Cornell University, Ithaca NY 14853. Supported in part by an Alfred P. Sloan Research Fellowship and by NSF Faculty Early Career Development Award CCR-9701399.

1 Introduction

As an increasing number of protein structures become known, the need for algorithms to analyze three-dimensional conformational structure increases as well. The search for common substructure is of value in uncovering relationships among different proteins — for inferring similarities in function and discovering common evolutionary origins. There is now widespread agreement that similarities among distantly related proteins are often preserved at the level of three-dimensional structure, even after very little similarity remains at the sequence level [7]. We are interested in developing efficient comparison methods for three-dimensional protein structures.

The goal of our work is to automatically identify common substructures shared by two proteins, based on similarity in their three-dimensional structure. A key element of our approach is the development of a geometric notion of similarity that is appropriate to the task of protein structure comparison. This is a problem that has received considerable attention in the biological research literature (cf. [22]), and a variety of different methods have been proposed.

For protein chains that are very similar (e.g., candidate structures produced in crystallography experiments), a common measure of similarity is the RMS (root mean square) distance between the positions of corresponding atoms of the proteins, after the proteins have been superimposed by a three-dimensional rigid motion (e.g. [10, 19]). While this approach is reasonable for comparing two structures that are highly similar, it does not work well for finding similar substructures in otherwise dissimilar proteins. The difficulty with dissimilar proteins is that the portions of the structure that do not match dominate the RMS value, making it difficult to identify matching substructures. This is an instance of the classical issue of outliers, or bad data, dominating the measure in fitting or estimation problems.

Other methods for structural matching address the limitations of techiques that use the RMS distance. One widespread set of approaches is based on comparing the distance matrices or contact maps of two proteins (see e.g. [3, 9, 13, 24, 25]). The distance matrix records all pairwise distances between selected atoms within a given protein — often for all the α-carbons of the residues. For a protein with n residues, such a distance matrix therefore contains $O(n^2)$ entries. The distance matrix representation is invariant with respect to the position and orientation of the protein; thus the distance matrices for two proteins can be compared directly, without the need to find a rigid motion superimposing them as RMS distance approaches require. The use of distance matrices has a number of drawbacks, however. First of all, a full distance matrix is quadratic in the size of the protein it represents. Moreover, algorithms for aligning distance matrices have typically been computationally expensive; identifying corresponding substructures between two proteins requires finding corresponding submatrices that are nearly the same, and this appears to be a difficult optimization problem (e.g. [7, 9]).

Finally, a different set of approaches to structure comparison is based on using a purely *local* representation of a protein chain. Some common local representations are the set of (ϕ, ψ) torsion angles along the backbone [3, 22]; the set of virtual-bond angles and virtual-bond dihedral angles [11, 12, 21] between consecutive α-carbons; and variations on these (e.g. [17, 18]). Such representations are compact and independent of the position and orientation of the protein; however, there are settings in which two proteins with similar local representations can be geometrically very different. Consider, for example, two protein structures that are the same except for a bend in the middle. Although the sets of three-dimensional coordinates and pairwise distances of the proteins will be quite different, the data in a local representation of the two proteins will be the same except for those residues directly at the bend.

In this paper we present a new algorithm for efficiently detecting common substructures in proteins, based on a comparison methodology for protein chains that has both global and local aspects. For a given protein, we begin with the sequence of α-carbons that form its backbone chain. Rather than encoding the three-dimensional coordinates of these atoms, we record the unit vector v_i in the direction from the i^{th} α-carbon to the $(i+1)^{\text{st}}$ α-carbon along the chain. The distance between successive α-carbons is essentially a fixed constant (approximately four angstroms), and thus this set of unit vectors captures the structure of the backbone. Indeed, these unit vectors correspond to the *virtual-bond vectors* of the protein [11, 12, 21].

By chaining the unit vectors v_i end to end, we obtain the standard model of a protein as a sequence of α-carbons in space. By placing all of the unit vectors at the origin of $\mathbf{R}^3$, we can view the protein as a sequence of points on the unit sphere. We will represent the protein in this latter way: by its collection $\{v_i\}$ of unit vectors with respect to a common origin. Thus we see that our representation has both a local and a global nature: although each individual vector v_i records the position of the $(i+1)^{\text{st}}$ α-carbon in a *local* reference frame centered at the i^{th} α-carbon, our representation implicitly situates all the unit vectors $\{v_i\}$ in a common *global* reference frame. This is contrast to traditional approaches that have used virtual bonds among α-carbons to define purely local representations.

Our representation is therefore compact, having size $O(n)$ for a protein with n residues, and it reflects global aspects of a protein's structure — introducing a single bend in the middle of structure affects *all* the vectors v_i that come after the bend. Our unit-vector representation is not independent of the orientation of the protein, since the vectors $\{v_i\}$ are placed in a global frame of reference. However, we will see that unlike representations based on raw three-dimensional coordinates, the unit-vector representation allows for geometric similarity measures that are robust against outliers.

Our algorithm for comparing two protein chains, based on the unit-vector representation, consists of the following three steps.

1. Determine a small set of *shifts*, or *contiguous alignments*, that are likely to include those alignments that bring matching substructures into 3D correspondence.

2. For each shift chosen above, determine the contiguous substructures for which there is a 3D *geometric alignment* (i.e., substructures of the two proteins that can be superimposed using a 3D rigid motion).

3. Combine the contiguous substructures chosen above to identify common protein *domains* (larger substructures that are not necessarily contiguous along the protein chain) that can be *superimposed*.

The first step of our algorithm runs in $O(n \log n)$ time, where n is the number of α-carbons in the larger of the two proteins being compared. The second step takes $O(n)$ time for each shift identified in step one; typically only the best 10 shifts provide any useful information, but our current implementation checks the best 20. The running time of the third step depends on s, the number of substructures found in the second step.

Our approach makes use of two very different kinds of *alignment*. The first of these, employed in Step 1, is the alignment of two sequences by a simple shift relative to one another. We refer to this as *contiguous alignment*, as it is simpler than the standard biological notion of *sequence alignment* (which further allows for insertions and deletions into a sequence). More formally, for integers $k = 0, 1, \ldots, n-1$, we match each residue i of the shorter protein with residue $i + k$ of the longer protein. Here addition is taken modulo n, so that our shift includes a "wrap-around". The wrap-around ensures that each unit vector in protein A is compared against each unit vector in protein B.

The second kind of alignment that we employ is *geometric alignment*, which is the 3D positioning of two proteins such that corresponding atoms in the two structures are approximately superimposed.

Our method will be based on a variant of the RMS distance, measuring differences between the global orientation vectors v_i at corresponding α-carbons of two proteins, rather than measuring differences between the positions of the α-carbons themselves. The key underlying observation is that the difference between two orientation vectors is bounded by a small constant, whereas the distance between two points can grow arbitrarily large. This makes the *unit-vector RMS* (URMS) that we propose naturally robust with respect to differences between nonmatching parts of the structure, whereas the conventional *position-RMS* is not.

The RMS distance. A basic method for determining geometric similarity between two sets of points is the *root mean square* (RMS) distance on point sets. This measure of distance requires a one-to-one correspondence between the point sets. The standard (position-based) RMS distance between point sets $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_n\}$ is defined as the square root of the sum of the squared distances between the points of A and the corresponding points of B. Generally one seeks the best possible RMS distance; that is, the RMS distance between A' and B where A' is the set A after application of the 3D rigid motion that minimizes the RMS distance. We refer to this as the *minimum RMS distance* between A and B.

To compute the minimum RMS distance, first transform A to make it have the same center-of-mass as B. Then compute the *correlation matrix* for A and B. This is a 3×3 matrix in which the (p, q) entry is $\sum_{i=1}^{n} a_{i,p} b_{i,q}$. This matrix is factored using the singular value decomposition; the resulting factors can be used to construct in constant time both (1) the orthogonal matrix that brings A into the closest possible match with B and (2) the resulting RMS distance. For a concise description of the algorithm and an explanation of why it works see Golub and Van Loan [6] (where it is referred to as the orthogonal Procrustes problem) or Lesk [10].

The minimum RMS distance between two sets of atoms in $\mathbf{R}^3$ is useful primarily when aligning two structures that are extremely similar. Thus, it is widely used for aligning structures produced in the course of crystallographic or NMR analysis of molecules.

It is less appropriate for comparing structures that have only a distant similarity. This is due to its sensitivity to outliers. Indeed, a few outliers can have a significant effect on the resulting minimum RMS distance. For instance, consider sets A and B, each with n points, where the minimum RMS distance between A and B is 0. Now suppose we take the point b_1 and gradually move it along a straight line. As we move b_1, the minimum RMS distance between A and B increases without bound, and the rigid motion corresponding to this minimum distance also changes so that the congruent point sets $A - \{a_1\}$ and $B - \{b_1\}$ become arbitrarily far apart. Thus a small number of gross outliers can hide a large shared substructure.

Another issue with the minimum RMS distance is that points far from the center of mass are, in effect, weighted more heavily than points near the center of mass. Consequently, a common "core" shared by two structures is difficult to detect using the minimum RMS distance. This is problematic in the case of proteins, where some crucial similarities occur near the center of mass.

Our Approach. Using an RMS distance on the unit-vector representation $\{v_i\}$ allows us to avoid the drawbacks discussed above while retaining the computational efficiency of the minimum RMS calculations. Recall our view of the representation $\{v_i\}$ as consisting of a sequence of points on the unit sphere in $\mathbf{R}^3$. To compare two proteins, with sets of unit vectors $\{v_i\}$ and $\{w_i\}$, we compute the minimum RMS distance between the two sets $\{v_i\}$ and $\{w_i\}$, viewed as subsets of the unit sphere. Since we wish the origins of the vectors to remain fixed in space, we minimize the RMS distance only over possible rotations of the vectors, not over translations. Henceforth we use the term *position-RMS* to refer to the standard, position-based, minimum RMS distance; we use the term *unit-vector RMS* (URMS) to refer to the minimum RMS distance for a pair of sets in our unit-vector representation.

Because the maximum squared distance between any two unit vectors is 4, no single outlier can have a large effect on the URMS. As a result, the URMS exhibits a natural resilience to the presence of outliers, such as those arising due to portions of two protein structures that do not match.

Other Related Work. Above, we have indicated a number of the approaches that have been proposed for protein structure comparison. Here, we discuss several of these in more detail; see [2, 8, 22] for recent surveys on the topic.

There has been work on discovering geometric alignments that are not required to respect the order of the sequence, using distance matrices on α-carbons and secondary structural elements of proteins (Vriend and Sander [24], Holm and Sander [9], Grindley et al. [5]). In particular, Vreind and Sander [24] use the distance matrix to find pairs of matching fragments of two proteins and then combine them into larger units which are compatible in $\mathbf{R}^3$. The latter step is based on comparisons of the centers of mass of the fragments and of relative rotations between combined fragments. Yee and Dill [25] use distance matrices as part of an algorithm whose high-level structure is similar to ours. They also search over contiguous alignments of two protein sequences to discover similarities, but their notion of *similarity* is different from ours. It is based on shifting the alignments of the distance matrices of the two proteins and computing the "distance" between the matrices (by means of the Frobenius norm). The shifts that give the smallest norms are predicted to yield matching contiguous alignments. Though the running time is not discussed explicitly in their work, it is easy to see that it can be $O(n^3)$. (It is possible that an application of the Fast Fourier Transform could improve this to $O(n^2 \log n)$.) There has also been work by a number of groups on extending classical sequence alignment algorithms to handle pairwise relationships among residue positions (Taylor and Orengo [23], Orengo et al. [14], Sali and Blundell [20]).

Rackovsky and Scheraga [17, 18] develop a purely local representation of protein chains, by defining discrete analogues of curvature and torsion in terms of the virtual bond vectors. They propose a method of searching for structural similarities based on comparing these parameters for different proteins [16, 18].

Fischer et al [4] apply *geometric hashing* to find matching pairs of atoms (alpha carbons) between two molecules. The common subsequences they find are not necessarily consecutive. To use geometric hashing, a *base molecule* is preprocessed to find all possible matching coordinate frames. This may involve up to $O(n^3)$ pre-processing and storage, since a rigid motion in three dimensions is specified by its action on three points. (Distance constraints imposed by the protein can improve this bound somewhat.) Given a second *target molecule*, each triplet of points "votes" for a rigid motion with respect to the base molecule; a large number of common votes indicates a possible large match between the two molecules. Assuming that the rigid motions are indexed in a well-distributed hash table, the running time is $O(n^3)$. Recent work on this technique has suggested that it may be possible to reduce the running time for protein matching applications to a roughly quadratic bound [15].

2 Fast Substructure Detection

The first step of our algorithm exploits the fact that it is possible to quickly detect *contiguous alignments* (or shifts) of two proteins such that significant substructures of the two proteins can be *geometrically aligned*. This operation simply identifies the shifts for which there are matching substructures, without determining which portions of the two proteins correspond. The second step of our algorithm, discussed below, then identifies these matching substructures.

Consider the case in which two proteins contain *contiguous* substructures that are approximately similar. By *contiguous* we mean a portion of structure corresponding to an unbroken substring of the protein's primary sequence. We first compute the URMS, as defined above, for each relative shift of the two protein sequences. Using the FFT and assuming each protein has length bounded by n, this can be accomplished in $O(n \log n)$ time rather than the cruder bound of $O(n^2)$. When we reach the shift at which there is a substructure match, we expect the URMS to drop considerably. By looking for shifts at which there is an unusually low URMS, we can detect approximate common substructures.

Why should we expect a drop of this sort? It is here that the use of URMS, rather than position-RMS, is crucial. Whereas a single "bad" match for URMS matching contributes at most 4 to the sum of squared distances, a single bad match in the position-RMS can have an unbounded effect. Thus, each outlier in the URMS setting contributes at most a constant amount to the error, and cannot prevent a good matching region from standing out.

Intuitively, two unrelated proteins "look random" to each other. We now consider some properties of the URMS for sets of random unit vectors in $\mathbf{R}^3$. Of course, real proteins are not random sequences of unit vectors; indeed, adjacent vectors within a protein are highly correlated. For instance, the angles between adjacent vectors are highly constrained and there are larger substructures that occur with high frequency (e.g. alpha helices and beta sheets). The *Ramachandran plot* of angles between adjacent residues [3] depicts exactly the correlations that occur in typical secondary structural units. Nevertheless, we show by experiment that many of our observations about comparing sets of random unit vectors apply in an approximate sense to proteins; though adjacent vectors within a protein are not random, it appears that the intuition that pairs of unrelated proteins "look random" to each other has some validity.

Lemma 1 *The expected (nonminimized) RMS distance between two sequences of randomly chosen unit vectors in* $\mathbf{R}^3$ *is* $\sqrt{2}$.

Proof: By linearity of expectation, we may focus on a single pair of randomly chosen unit vectors x and y; and by symmetry, we may assume that $y = (0,0,1)$. For a unit vector x, let $f(x)$ denote the result of reflecting x through the xy-plane. If we generate the unit vector x from the uniform distribution on the sphere, and then output x or $f(x)$ with equal probability, the result is still uniformly distributed. But

$$(x-y)^2 + (f(x)-y)^2 = (x-y)^2 + (x+y)^2 = 2x^2 + 2y^2 = 4,$$

and so the expected squared distance between x and y is 2. ■

One intuitively expects that the URMS (the RMS distance between sets of random unit vectors *minimized* over rotations) should achieve a value not much below $\sqrt{2}$. To achieve a more precise bound, we analyze the correlation matrix used during the RMS calculation.

The minimized RMS distance between two sequences of unit vectors, each of length n, is

$$\sqrt{\frac{2n - 2 * \text{trace}(\text{svd}(A))}{n}}$$

where A is the correlation matrix and $\text{svd}(A)$ is the diagonal matrix in the SVD decomposition of A [6]. For unit vectors picked uniformly in the unit sphere (say (x_1, x_2, x_3) from the first protein and (y_1, y_2, y_3) from the second) we need the distribution for the pairwise products $x_i y_i$, as these are the terms that appear in the correlation matrix. These terms all have the same probability distribution, so we analyze the random variable $X_1 Y_1$.

For $x_1 y_1$, we can assume that x_1 and y_1 are independent and uniformly distributed in the interval $[-1, 1]$. This follows from the correspondence between uniform density on a sphere's surface and uniform density on the sphere's diameter.

The random variable $X = X_1 Y_1$ is not uniform, but we're adding a large number of such random values (one for each unit vector in the protein chain), so, once we have the mean and variance, we should be able to use the Central Limit Theorem to obtain an approximation.

$$E[X] = \frac{1}{4} \int_{-1}^{1} \int_{-1}^{1} x_1 y_1 dx_1 dy_1 = 0$$

n	$D_{formula}$	$D_{experiment}$
100	1.310	1.311
200	1.341	1.341
300	1.355	1.354
400	1.363	1.363
500	1.369	1.368

Table 1: The expected URMS for different length sequences of random unit vectors. The formula is derived in the text. The experiments used sequences of random unit vectors generated using MATLAB. For each length, the experimental value shown is the mean for 1000 trials.

$$E[X^2] = \frac{1}{4}\int_{-1}^{1}\int_{-1}^{1} x_1^2 y_1^2 dx_1 dy_1 = \frac{1}{9}$$

Thus, we have a standard deviation of $\sigma = \frac{1}{3}$.

Applying the Central Limit Theorem, we conclude that every individual entry in the correlation matrix can be approximated by a random variable of the form $\frac{\sqrt{n}}{3}\eta(0,1)$ where $\eta(0,1)$ denotes the normal distribution with mean 0 and standard deviation 1. Unfortunately, this isn't enough to determine the final form of the correlation matrix, as the entries are not independent. Still, an analysis under the (false) assumption of independence may be suggestive of how the matrices behave in practice; we confirm this with experimental results.

Let B be a 3×3 matrix with entries chosen independently from $\eta(0,1)$. We need to determine expected values for the trace of the SVD of this matrix. It's a simple matter to run 10,000 experiments of this type using MATLAB. Experimentally, the desired trace has mean 4.26 with standard deviation 1.06.

The analysis above indicates that trace(svd(A)) in the formula for distance can be replaced by its expected value which is $\frac{\sqrt{n}}{3}b$ where b is the expected value of trace(svd(B)). Substituting our experimentally derived value for b, the expected distance D is $\sqrt{2 - \frac{2.84}{\sqrt{n}}}$. This is the approximate distance that we should see between two unrelated proteins after the proteins have been rotated to agree as much as possible.

Table 1 shows values for the expected value of the URMS using (1) the formla derived above and (2) experimental data using sequences of random unit vectors. The formula and experiments agree more closely than we had expected. Apparently, the (false) assumption of independence that we used above still allowed the derivation of a predictive formula.

These results agree quite well with our experiments on real proteins, although the URMS for real proteins is a bit smaller than the values derived above. See Figures 1 and 2. These figures show the URMS plotted for each possible shift. Figure 1 compares a protein of length 65 with one of length 104. The expected URMS derived using the above formula (for a comparison of length 65) is 1.284 while the figure appears to indicate an average value somewhat below this. Figure 2 compares a protein of length 382 with one of length 457. The expected URMS, by formula, is 1.362 while, again, the figure indicates a value below this.

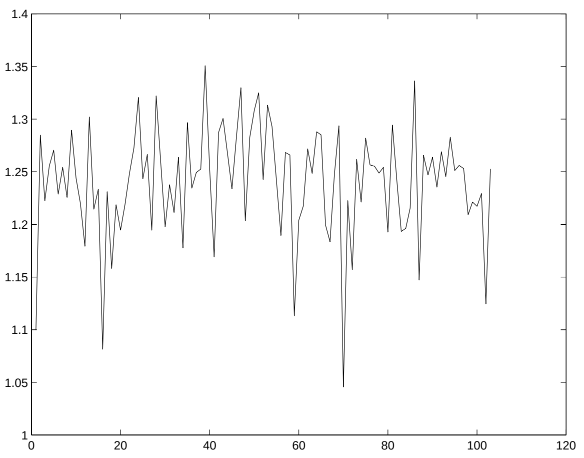

Figure 1: Orientation-RMS for each shift of 2cro and 2wrp.

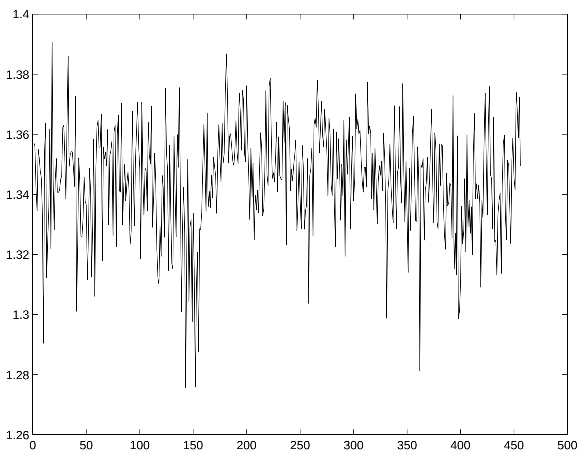

Figure 2: Orientation-RMS for each shift of 1hsc and 2yhx.

The discrepancy can be explained by the fact that our formula models proteins as sequences of random unit vectors, an assumption that is certainly untrue.

In summary, both analysis and experiment indicate that for two unrelated proteins (1) the expected (nonminimized) RMS-distance for the corresponding sequences of unit vectors is $\sqrt{2}$ when the proteins are placed at random orientations and (2) the expected URMS (i.e., the RMS-distance for unit vectors when the proteins are rotated to align as well as possible) is roughly equal to $\sqrt{2 - \frac{2.84}{\sqrt{n}}}$ where n is the length of the smaller of the two proteins.

If two proteins have a substructure in common and if the protein sequences are shifted to bring the two similar substructures into correspondence then the URMS should decrease. The following lemma tries to capture this notion, building on our view of unrelated portions of protein chains as looking "random" with respect to one another.

Lemma 2 *Let $v_1, \ldots, v_k$ be unit vectors in $\mathbf{R}^3$, and let $a_1, \ldots, a_n, b_1, \ldots, b_n$ be unit vectors chosen uniformly and independently at random in $\mathbf{R}^3$. Consider the following two sequences of orientation vectors:*

$$\{a_1, a_2, \ldots, a_p, v_1, v_2, \ldots, v_k, a_{p+1}, \ldots, a_n\},$$

$$\{b_1, b_2, \ldots, b_q, v_1, v_2, \ldots, v_k, b_{q+1}, \ldots, b_n\}.$$

Define $f = \frac{k}{n+k}$, the fraction of the sequence corresponding to the set $\{v_i\}$. Then at the shift that brings the two copies of v_1 into alignment, the value of the expected URMS is at most $\sqrt{2(1-f)}$.

Proof: We can provide an upper bound on the value of the URMS by computing the value of the RMS at a particular orientation: namely, the one which geometrically aligns the two copies of the sets $\{v_1, \ldots, v_k\}$. The crucial observation is that we can consider the vectors in $\{a_i\}$ and $\{b_j\}$ to have been generated after choosing this alignment; thus, the expected squared distance between each corresponding pair is 2, by Lemma 1. Thus, at this orientation, the expected sum of all the squared distances is $2(1-f)n$, and so the expected value of the RMS-distance is $\sqrt{2(1-f)}$. The value of the URMS, which minimizes over all possible orientations, is bounded by this quantity. ■

Consider two proteins of length 300 with a common substructure of length 45. This lemma implies that the expected URMS for these proteins should be $\leq \sqrt{1.7}$ or about 1.30. In fact, the URMS for proteins of length about 300 drops considerably below 1.30 even for smaller common substructures. Again, this is probably due to the nonrandom nature of real proteins.

Our strategy is thus to try all possible shifts of the two proteins looking for a drop in the URMS. Experiment indicates that this strategy is quite effective. For a given shift, we compute the correlation matrix and use this to compute the RMS-distance between the two sets of corresponding unit vectors. The FFT can be used to compute the matrix entries for each of the n shifts. It takes $O(n \log n)$ time to compute n 3-by-3 matrices. For each of these matrices we can compute in constant time the RMS-distance between the corresponding unit vectors. Thus, in $O(n \log n)$ time we can compute the URMS for all possible shifts.

Identifying Matching Substructures

The method so far can be used to detect which shifts might lead to a correspondence between common substructures, but we have not described how that common substructure can be recognized. In other words, we have an efficient method for indicating when two proteins have a common substructure, but it does not actually tell us where the substructure is or how to rotate (in 3D) the structures so that the common substructures have the same orientation. We do know the shift providing the correspondence between the two protein sequences, so we have the correct 1-to-1 map between the two proteins.

Consider the rotation needed to superimpose the substructures. If the substructures are sufficiently similar then this rotation is the same as that needed to superimpose subparts of the substructures. We can use this observation as a way to recognize the substructure. The idea is to compute the rotation matrices needed to bring each small piece of the first protein into rotational alignment with the corresponding piece of the second protein. For pieces that are part of the matching substructure, these rotation matrices will agree. We compute the rotation matrix at each unit vector by computing the rotation that most closely aligns the vector and its successor in the first protein with the corresponding two vectors in the second protein.

To determine when two rotation matrices are similar, we use the Frobenius norm. The Frobenius norm of a matrix is the square root of the sum of the squares of all the matrix entries. This is relatively easy to compute and is invariant under rotation. Let Q and R be two rotation matrices (i.e., orthogonal matrices). Then the Frobenius norm of $Q - R$ is the same as the Frobenius norm of $QR^T - I$ where I is the identity matrix. This fits our intuition for how we want our matrix norm to behave: if Q and R are *close* as rotation matrices then QR^T should be *close* to the identity matrix.

For our problem, we are looking for sequences of these matrices that agree. Thus it is sufficient to compare only adjacent matrices as we move down the protein sequence. This requires just $O(n)$ time.

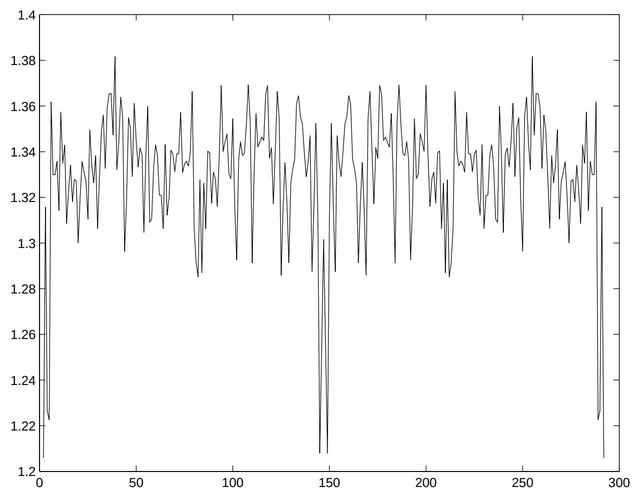

Figure 3: Orientation-RMS for each shift of 1rhd against itself.

3 Implementation and Experimental Results

We have implemented our algorithm in Matlab. The protein structures were taken from the Protein Data Bank (PDB) [1] and our results were compared against substructure similarities reported in the literature. Some of our experiments are summarized in this section.

Assume the shorter protein is A and the longer one is B. The first residue in A is shifted to correspond in turn with each of the residues in B. At each shift, we truncate the unmatched portion of B. When protein A extends past the end of B, we wrap around to the beginning of B. We compute the rotation that minimizes the RMS-distance for the unit-vector representations of A and the truncated B.

Figures 1, 2, and 3 show the URMS for each possible shift of one protein against another. Our experiments have shown that only the first 10-20 lowest values are likely to represent shifts that bring significant substructures into correspondence. Our program takes just the best 20 shifts and proceeds to the next stage of finding the actual matching contiguous substructures for these shifts. To ensure that we have not ignored potential substructure matches, we have also run experiments to check for matching substructures for *all* the shifts, not just the best 20. So far, without exception, the other shifts did not yield any matching substructures of significant size.

Notice that in comparing 1rhd (293 residues) with itself the URMS plot (Figure 3) is symmetric about the median shift. This is caused by our cyclic shifting of residues and happens whenever we compare a protein with itself. Note that when comparing a protein with itself, we do not graph the zero-shift, since the URMS for that shift is 0. Including this shift would distort our scale.

In some structure comparisons (e.g., Figure 1: 2cro (65 residues) with 2wrp (104 residues)), a large consecutive substructure (22 out of 65 residues) is responsible for the drop in the URMS. In other cases (e.g., Figure 2: 1hsc (382 residues) with 2yhx (457 residues)), a number of small matching substructures (about three substructures of average length 12-13) were found for one shift and they cumulatively contributed to the drop in the URMS.

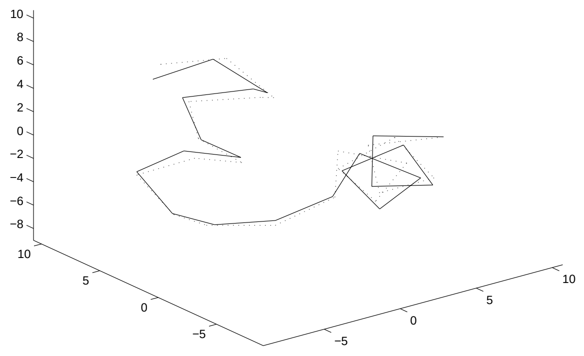
Figure 4: Subsequence match of 2cro(17–40) and 2wrp(62–85).

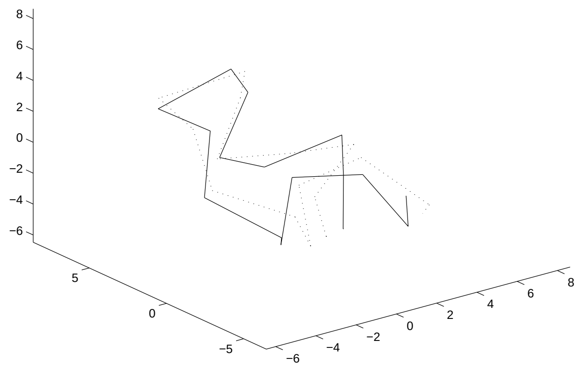
Figure 5: Subsequence match of 2cro(48–65) and 2wrp(65–82).

We compared our results with others in the literature. In Fischer et al. [4] a helix-turn-helix motif is found in tryptophan repressor (PDB code: 2wrp) and in phage 434 cro (PDB code: 2cro). Our graph showing the URMS for each shift is shown in Figure 1. We find a substructure match between the same set of 23 consecutive residues (see Figure 4). Another long match our program discovers for these proteins is shown in Figure 5. Notice that these two matches involve the same section in 2wrp.

An internal duplication of motifs in the molecule of bovine liver rhodanese (PDB code: 1rhd, 293 residues) is well known ([24]). Our algorithm detects large matching substructures (see Figures 6 – 9 in the Appendix.) We do not present matching substructures of length less than 12.

In a third example, we compare the heat shock cognate protein (PDB code: 1hsc, 382 residues) with the protein hexokinase (PDB code: 2yhx, 457 residues). These proteins were compared in [5] though we find it hard to compare our results with theirs. A beta-sheet of 27 residues is found to match residues 193-220 in 1hsc with 63–100 in 2yhx (see Figure 10 in the Appendix).

4 Finding Common Protein Domains

We have preliminary results for composing larger structures (protein domains) by combining the smaller contiguous substructures detected using the above algorithm. Assume the substructure match program outputs s matching pairs of substructures. A large structure is a disjoint union of matching substructures that undergo the same transformation (rotation and translation) for the best RMS. Currently the match is found in a primitive way by comparing the $O(s^2)$ rotation matrices and finding a clique of similar matrices (according to the Frobenius norm) and then comparing the translation. (Notice that just comparing the rotations will yield bogus results on alpha-helix-rich structures, because a substructure a_1 of protein A might be similar to many substructures in B under the same rotation, but with different translations.) The matching substructures detected for 1rhd above were found to undergo similar transformation. We tested two SH2 protein fragments, the SH2 domain 1a1e (103 residues) and SH3-SH2 domain fragment of Human Bcr-Abl tyrosine Kinase Transferase (PDB code: 2abl, 164 residues). We found two large matching substructures (see Figures 11 and 12 in the Appendix) that sum up to 68 residues. Notice that the two matching substructures have a gap of 31 residues in 1a1e and 34 residues in 2abl.

Acknowledgements. The authors would like to thank Ron Elber, David Shalloway, and Steve Vavasis for very helpful discussions.

References

[1] F.C. Bernstein, T.F. Koetzle, G.J.B. Williams, E.F. Meyer, M.D. Brice, J.R. Rodgers, O. Kennard, T. Shimanouchi, M.Tasumi, *J. Mol. Bio.* 112(1977), pp. 535–542.

[2] N.P. Brown, C.A. Orengo, W.R. Taylor, "A protein structure comparison methodology," *Computer Chem.*, 1996.

[3] T.E. Creighton, *Proteins: Structures and Molecular Properties*, Freeman, 1992.

[4] D. Fischer, R. Nussinov and H. Wolfson, "3D substructure matching in protein molecules", *Proc. 3rd Intl. Symp. Combinatorial Pattern Matching*, Lecture Notes in Computer Science, 644, Springer-Verlag, pp. 459–477.

[5] H.M. Griendley, P.J. Artymiuk, D.W. Rice and P. Willett, "Identification of tertiary structure resemblance in proteins using a maximal common subgraph isomorphism algorithm", *J. Mol. Biol.* 229(1993), pp. 707 – 721.

[6] G.H. Golub and C.F. Van Loan, *Matrix Computations, 3rd Edition* (1996), Johns Hopkins University Press.

[7] L. Holm, C. Sander. "Mapping the protein universe." Science 273 (1996), 595.

[8] L. Holm, C. Sander. "Searching protein structure databases has come of age." *Proteins: Structure, Function, and Genetics*, 1994.

[9] L. Holm and C. Sander, "Protein Structure Comparison by Alignment of Distance Matrices." J. Mol. Biol. (1993) 233, 123-138.

[10] A.M. Lesk, "A toolkit for computational molecular biology II. On the optimal superposition of two sets of coordinates," *Acta Crystallographica*, A42(1986), pp. 110–113.

[11] M. Levitt, "A simplified representation of protein conformations for the rapid simulation of protein folding," *J. Mol. Bio.* 104(1976), pp. 59–116.

[12] K. Nishikawa, F. Momany, H.A. Scheraga, "Low-energy structures of two dipeptides and their relationship to bend conformations," *Macromolecules* 7(1974), pp. 797–806.

[13] K. Nishikawa, T. Ooi, "Comparison of homologous tertiary structures of proteins," *J. Theoretical Bio.* 43(1974), pp. 351–374.

[14] C.A. Orengo, N.P. Brown, and W.R. Taylor, "Fast Structure Alignment for Protein Databank Searching." *PROTEINS: Structure, Function, and Genetics* 14:139-167 (1992).

[15] X. Pennec and N. Ayache, "A geometric algorithm to find small but highly similar 3D substructures in proteins", *Bioinformatics*, 14 (1998), pp. 516–522.

[16] S. Rackovsky, D.A. Goldstein, "Protein comparison and classification: A differential geometry approach," *Proc. Nat. Acad. Sci.* 85(1988), pp. 777–781.

[17] S. Rackovsky, H.A. Scheraga, "Differential geometry and polymer conformations 1. Comparison of protein conformations," *Macromolecules* 11(1978), pp. 1168–1174.

[18] S. Rackovsky, H.A. Scheraga, "Differential geometry and polymer conformations 2. Development of a conformational distance function," *Macromolecules* 13(1980), pp. 1440–1453.

[19] S.T. Rao, M.G. Rossman, "Comparison of super-secondary structures in proteins," *J. Mol. Bio.* 76(1973), pp. 211–256.

[20] A. Sali, T. Blundell, *J. Mol. Bio.*, 212(1990), pp. 403–428.

[21] R. Srinivasan, R. Balasubramanian, S.S. Rajan, "Some new methods and general results of analysis of protein crystallographic structure data," *J. Mol. Bio.*, 98(1975), pp. 739–747.

[22] W.R. Taylor (editor), *Patterns in Protein Sequence and Structure*, Springer Series in Biophysics, Vol 7, Springer Verlag 1992.

[23] W.R. Taylor, C.A. Orengo, "Protein structure alignment," *J. Mol. Bio.*, 208(1989), pp. 1–22.

[24] G. Vriend and C. Sander, "Detection of common three-dimensional substructures in proteins", *PROTEINS: Structure, Function and Genetics*, 11(1991), pp. 52–58.

[25] D. Yee, K. Dill, "Families and the structural relatedness among globular proteins," *Protein Science*, 2(1993).

5 Appendix

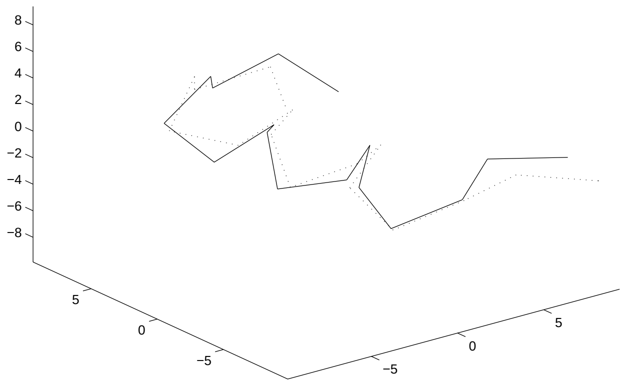

Figure 6: 1rhd internal match of residues 9–26 with 161–178.

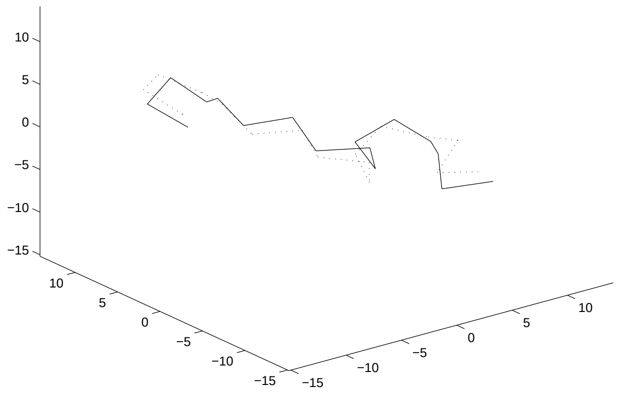

Figure 7: 1rhd internal match of residues 52–69 with 204–221.

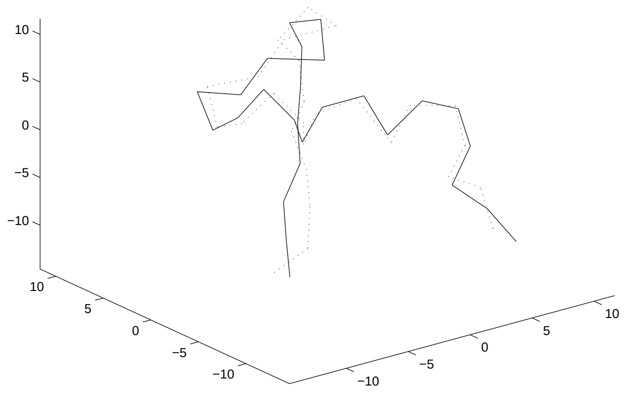

Figure 8: 1rhd internal match of residues 73–102 with 221–250.

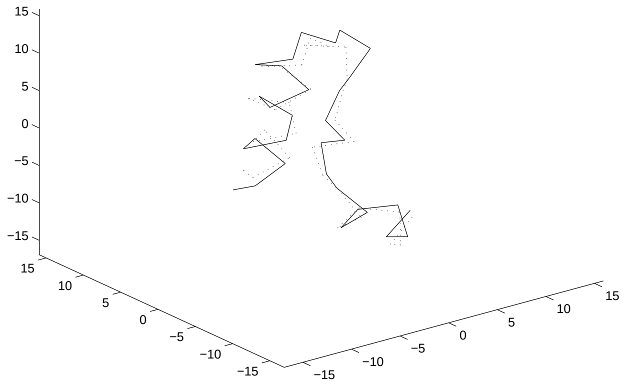

Figure 9: 1rhd internal match of residues 106–138 with 251–283.

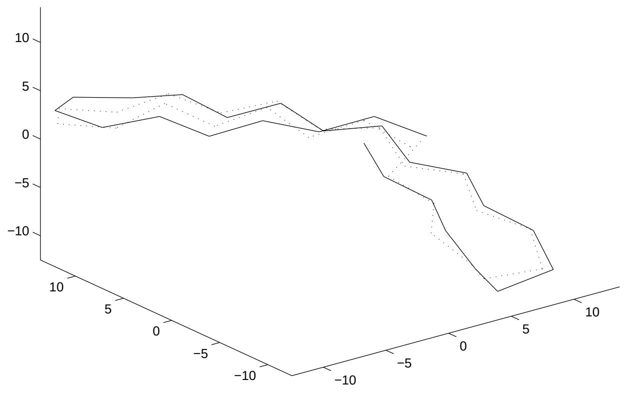

Figure 10: 1hsc(193–220) and 2yhx(63–90) superimposed.

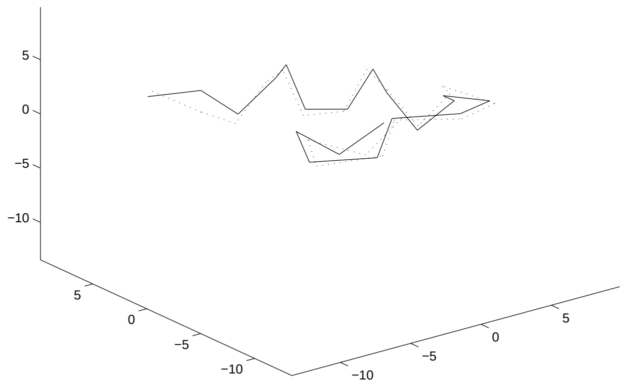

Figure 11: 1a1e(3–24) and 2abl(69–80) superimposed.

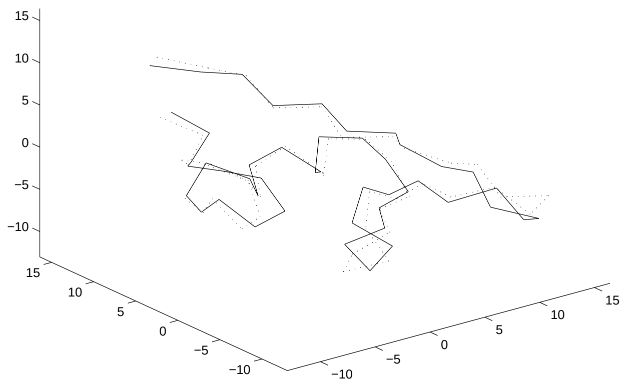

Figure 12: 1a1e(55–102) and 2abl(114–161) superimposed.