JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 12, Number 1, 2005
© Mary Ann Liebert, Inc.
Pp. 12–32

The URMS-RMS Hybrid Algorithm for Fast and Sensitive Local Protein Structure Alignment

GOLAN YONA[1] and KLARA KEDEM[1,2]

ABSTRACT

We present an efficient and sensitive hybrid algorithm for local structure alignment of a pair of 3D protein structures. The hybrid algorithm employs both the URMS (unit-vector root mean squared) metric and the RMS metric. Our algorithm searches efficiently the transformation space using a fast screening protocol; initial transformations (rotations) are identified using the URMS algorithm. These rotations are then clustered and an RMS-based dynamic programming algorithm is invoked to find the maximal local similarities for representative rotations of the clusters. Statistical significance of the alignments is estimated using a model that accounts for both the score of the match and the RMS. We tested our algorithm over the SCOP classification of protein domains. Our algorithm performs very well; its main advantages are that (1) it combines the advantages of the RMS and the URMS metrics, (2) it searches extensively the transformation space, (3) it detects complex similarities and structural repeats, and (4) its results are symmetric. The software is available for download at *biozon.org/ftp/software/urms/*.

Key words: AU1

1. INTRODUCTION

Functional annotation of proteins as done today is based mostly on inference rather than on experiments. The pace at which new proteins are revealed rules out expensive in vivo experiments to define their functionality, and therefore much effort is directed into improving methods for detecting similarity between proteins as a basis for functional inference. Especially, there is a strong interest in fast and sensitive structure comparison algorithms. Since protein structure is better conserved in evolution than sequence (Holm and Sander, 1996), detecting structural similarity can help identify subtle similarities that are not detected by means of sequence comparison.

The importance of protein structure comparison pertains to other problems beyond homology detection. Advances in functional genomics, from protein structure prediction to studies of structure–function relationships, require a fast, automatic, and well assessed method to compute the similarity of 3D protein structures. For example, it can be used to search for important structural and functional sites, such as active sites or interaction sites. Fold prediction systems require hundreds of thousands of structure comparisons

[1]Department of Computer Science, Cornell University, Ithaca, NY 14853.
[2]Department of Computer Science, Ben Gurion University, Beer-Sheva, Israel.

per prediction, and both the speed and accuracy of the algorithm are important factors in the success of these systems. An effective structure comparison algorithm can be also instrumental in studies of the self-organization of the protein space (Yona *et al.*, 1999; Yona and Levitt, 2000b) as it can help in automatic domain detection and classification, and comparison of protein families.

1.1. Related work

The importance of protein structure comparison has been acknowledged and investigated by many over the years, resulting in several different approaches for structure comparison. In general, these approaches can be divided into algorithms that are based on matching distance matrices and those that are based on minimizing atomic distances.

A number of groups extended classical sequence alignment ideas to handle structural alignment. Some of these methods employ double dynamic programming (DP) techniques for computing protein structure alignment (Taylor and Orengo, 1989; Orengo *et al.*, 1992; Taylor, 1999), where, in Taylor (1999) the first level of DP is applied to create a residue–residue scoring matrix using chemical information on residues, and the secondary structure information is used in the second level DP. In Orengo *et al.* (1992) linear representations of secondary structures are derived and their features are compared to identify equivalent elements in two proteins. The secondary structure alignment then constrains the residue alignment, which compares only residues within aligned secondary structures. Another dynamic programming algorithm is Structal (Gerstein and Levitt, 1996, 1998) which uses iterative DP to align the structures and recompute the scoring matrix based on the distances between atoms in the current alignment. The significance of the alignment is estimated the same way sequence matches are estimated in BLAST, based on the extreme-value distribution.

Algorithms that are based on matching distance matrices were introduced by Vriend and Sander (1991), Griendley *et al.* (1993), and Yee and Dill (1993). These algorithms (such as DALI [Holm and Sander, 1993]) use distance matrices on α-carbons to identify pairs of secondary structure elements that have the same distance profiles in the two structures compared. A Monte Carlo algorithm is then applied to search for the best combination of fragment pairs.

Other methods that do not fall under these categories were reported over the years. Rackovsky and Scheraga (1980) describe a method that is based on local representation of protein chains by discrete analogues of curvature and torsion (a similar approach can also be found in Rackovsky and Goldstein [1988]). The geometric hashing method is introduced by Fischer *et al.* (1992) and Pennec and Ayach (1998). Fischer *et al.* (1992) apply geometric hashing to find a maximal number of matching pairs of alpha carbons between two proteins. A base molecule is preprocessed to find all possible matching coordinate frames. Given a 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 (Pennec and Ayach, 1998). The Combinatorial Extension (CE) algorithm (Shindyalov and Bourne, 1998) starts with aligning short fragment pairs (AFPs) that are then extended based on a set of rules that determine whether the extension results in a better alignment. The MAMMOTH algorithm (Ortiz *et al.*, 2002) uses the URMS measure to compare all heptapeptide pairs of the two structures and transform these values to similarity scores; using this similarity matrix, they search for the best local alignment by applying a heuristic that extends short alignments, similarly to the MaxSub heuristic (Siew *et al.*, 2000).

1.2. Our approach

Although there exist many algorithms for protein structure comparison, the problem is by no means solved. First, there is no natural definition of structural similarity, as is the case for sequence similarity. This problem is most pronounced when one seeks local similarities. The standard measure that is used to evaluate the resemblance of two protein structures is the root-mean-square (RMS) distance of aligned α-carbon 3D positions.[1] However, a major problem with the RMS distance is that it can be overly sensitive **FN1**

[1] In some cases, the RMSD (root-mean-square deviation) is used.

to outliers and is highly length dependent. Moreover, this measure promotes shorter alignments, as they will naturally tend to minimize the RMS distance. Existing algorithms usually address this problem by employing some sort of a transformation to similarity scores, or by avoiding local alignments altogether and performing only global alignments.

Chew *et al.* (1999) presented a new measure, *the unit vector RMS* (URMS). This method is based on a variant of the RMS distance, measuring differences between the global orientation vectors (unit vectors) at corresponding α-carbons of two proteins, rather than the differences between the positions of the α-carbons themselves (the URMS metric is described in more detail in the next section). It is argued in that paper that this measure, when used to compare protein shapes globally, overcomes some of the disadvantages of the RMS, being length independent and robust to outliers. These advantages have been acknowledged by Kedem *et al.* (1999) where the measure has been used to locate a folding site, by Ortiz *et al.* (2002) and Siew *et al.* (2000) where it is used to evaluate ab initio predictions, and by Chew and Kedem (2002) where it is applied to find the structural consensus and alignment of a protein family. A disadvantage of the URMS is that since it ignores translations it can produce incorrect alignments that are composed of several local similarities under different translations.

In this paper, we present a new and fast method to compute *pairwise local structural alignments*. Our method addresses the limitations of both the RMS metric and the URMS metric, while utilizing the merits of each. We search efficiently the space of all possible alignments under the set of allowed transformations. Using the URMS metric, we detect all viable transformations. A dynamic programming algorithm is then invoked to optimize the local match corresponding to each one of the top ranking transformations, using the RMS metric.

The paper is organized as follows: In Section 2, we review the URMS metric and present the main elements of our algorithm. The scoring function is described in Section 3. In Section 4, we discuss the statistical properties of structural matches. Performance evaluation is reported in Section 5, and a few examples are presented in Section 6. We conclude with a discussion in Section 7.

2. METHODS—COMPARING PROTEIN STRUCTURES

2.1. Notations and basic definitions

We denote complete protein structures by uppercase letters $\mathbf{A}, \mathbf{B}$. Each protein structure is given with its complete set of x, y, z coordinates for all its atoms. We reduce this representation to the α-carbon backbone atoms $\mathbf{A} = \{\vec{A_i}\}_{i=1}^{n} = \{(Ax_i, Ay_i, Az_i)\}_{i=1}^{n}$ where n is the number of amino acids. We denote by $\mathbf{a}$ and $\mathbf{b}$ fragments of $\mathbf{A}$ and $\mathbf{B}$, respectively, where a fragment is a short consecutive substructure of the protein.

2.1.1. The URMS distance. The URMS distance (Chew *et al.*, 1999) is a variant of the RMS distance: instead of comparing residue positions, we compare unit vectors. The compared vectors are the vectors between adjacent α-carbons along the protein backbone. We refer to these direction vectors as *unit* vectors since, for proteins, these vectors are all approximately of the same length (about 3.8 Å). By chaining the unit vectors head-to-tail, we obtain the standard model of a protein as a sequence of α-carbons in space. Alternatively, we can place all of the unit vectors at the origin; the protein backbone is thus mapped into unit vectors in the unit sphere (see Fig. 1). Formally, given a protein structure $\mathbf{A}$, its *unit vector model* is the set F1

$$\mathbf{U_A} = \{\vec{u_i}^A\}_{i=1}^{n-1}$$

where $\vec{u_i}^A = (\vec{A_{i+1}} - \vec{A_i})/||\vec{A_{i+1}} - \vec{A_i}||$.

The URMS distance between two protein structures $\mathbf{A}$ and $\mathbf{B}$ is defined as the minimal RMS distance between their unit vector models, under rotation. Formally,

$$DIST_{urms}(\mathbf{A}, \mathbf{B}) = \min_R \{DIST_{rms}(\mathbf{U_A}, \mathbf{U_B^R})\}$$

where $\mathbf{U_B^R}$ is the rotated unit vector model of $\mathbf{B}$. The rotation R that minimizes the distance is referred to as the *URMS rotation*. Since the unit vector models of both proteins are centered at the origin, there is no need to introduce translation.

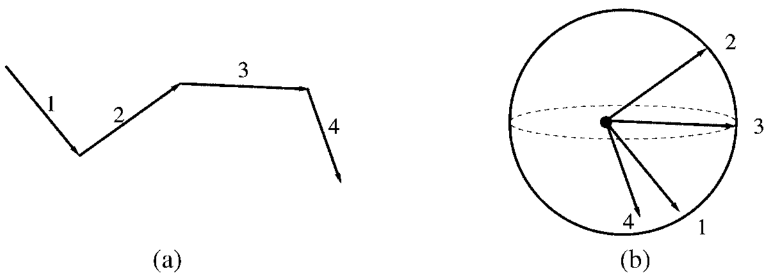

FIG. 1. (**a**) A sketch of the protein backbone; the unit vector i connects amino acid i to amino acid $i+1$, ($i = 1, 2, 3, 4$). (**b**) The unit vectors translated to the origin (into the unit sphere).

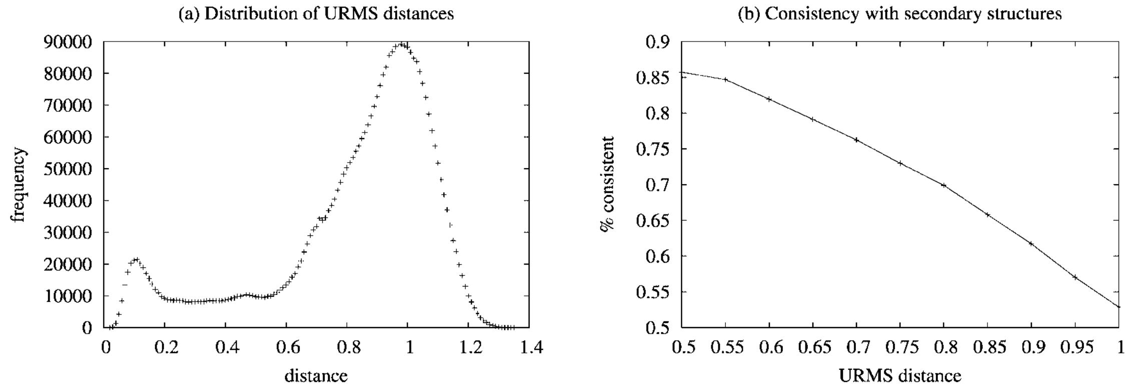

FIG. 2. (**a**) Distribution of URMS distances. (**b**) Correlation of URMS distances with secondary structures. The numbers are averaged over both alpha and beta proteins.

2.1.2. Structural agreement. Given two protein fragments **a** and **b** of length ℓ, we say that the fragments are *in structural agreement under rotation R* if

$$DIST_{urms}(\mathbf{a}, \mathbf{b}) < T_{urms}$$

where T_{urms} is a preset threshold and R is the URMS rotation.

The fragment length ℓ is determined through optimization to be 8. This is consistent with the average length of a typical secondary structure element. To determine T_{urms}, we study the distribution of distances between fragment pairs. As is shown in Fig. 2a the distribution indicates the existence of two populations, one of similar fragments and the other of nonsimilar fragments, suggesting a natural threshold of about 0.6. F2

It should be noted that fragment pairs that are in structural agreement are strongly correlated with secondary structures (Fig. 2b). At a threshold of 0.6, more than 93% of the fragment pairs are secondary-structure consistent for alpha proteins, and more than 70% are secondary-structure consistent for beta proteins.

2.2. Algorithm outline

Given a pair of 3D protein structures, our goal is to find the most similar substructures, under a *local scoring function*, as defined in Section 3. Our algorithm works as follows:

1. **Searching the rotation space:** Search for the complete set of similar fragment pairs using the URMS metric. For each pair of fragments determine the optimal rotation.
2. **Vector quantization:** Cluster rotations into clusters of similar rotations.
3. **Searching the reduced translation space:** For each cluster of similar rotations, identify the most consistent translation amongst its members. The cluster centroid's rotation and the most consistent translation define a candidate transformation.

4. **Alignment:** Given a candidate transformation (rotation and translation), find the best structural alignment using dynamic programming with an RMS-based scoring matrix.
5. **Iterative refinement:** Iteratively redefine the scoring function based on the current alignment, and realign the structures based on the scoring function, repeating steps 4 and 5, until convergence.
6. **Output:** Report the highest scoring transformations and alignments.

A detailed description of each step if given next.

2.2.1. Searching the rotation space. Any pair of similar substructures is composed of smaller fragment pairs that are structurally similar. If there is a single rigid transformation under which substructure **A** is very similar to substructure **B**, then one would expect to find multiple fragment pairs of **A** and **B** that are in structural agreement under this transformation. We explore the rotation space spanned by transformations that bring fragments of **A** and **B** to structural agreement (we refer to this set as the set of *viable rotations*). Specifically, every pair of ℓ-length fragments $(\mathbf{a}, \mathbf{b})$ of **A** and **B**, is compared using the URMS metric. If a fragment pair is in structural agreement under rotation R, then R is considered a *viable rotation* and is added to the set (for implementation details, see Appendix A).

2.2.2. Vector quantization. The space of all viable rotations can be quite large. For two proteins of length 100, the number of fragment pairs that are in structural agreement can be on the order of hundreds or even thousands, each pair with its own optimal rotation. Clearly, if two protein structures are similar, then many of the viable rotations will aggregate into one or a few clusters of similar rotations. In practice, the situation is more complex: slight differences between the viable rotations are very common. Moreover, when the exact transformation between two structures cannot be formulated by means of translation and rotation alone but requires deformation as well (which is quite often the case), the locally optimal (viable) rotations might give rise to elongated and relatively wide clusters. Together with outliers, these lead to a sparsely populated yet fairly well connected rotation space (see Fig. 18 in Appendix B). Clustering such spaces is a challenging task, especially when computational speed is a major issue.

We tested four different clustering algorithms for vector quantization, focusing on sensitivity and speed. The four algorithms that were tested are a greedy algorithm, a grid-like algorithm, connected components, and agglomerative pairwise clustering (see Appendix B). Our tests suggest that the differences between the clustering algorithms do not substantially affect the performance. Because of its speed and its symmetry (see below) we chose to proceed with the connected components algorithm.

To cluster the rotations, one needs a distance function between rotations. We use the *Frobenius distance* (denoted by $DIST_{frob}$) between rotation matrices, defined as the Euclidean distance between their representations as nine-dimensional vectors. An appealing alternative, based on the angle representation of rotation matrices, is described in Appendix B.

Symmetry and determinism. One of the major issues with structure comparison algorithms is their nondeterministic nature. Since the search space is too large to search exhaustively, all algorithms employ some kind of a heuristic to find good alignments, through sampling or directed search that can be highly influenced by the initial configuration (e.g., the initial orientation that is used to align the two structures). One of the consequences of this is that results are not symmetric; namely, when comparing structures A and B the results might differ from those one would get by comparing B with A. Obviously, one can simply run the algorithm twice, once for A,B and the other for B,A, and pick the better of the two results, but at the cost of doubling the computation time. Even then, if there is an element of randomness in the algorithm, then the symmetry is not guaranteed.

Our algorithm addresses this problem with no increase in computation time. By definition, the connected components clustering algorithm finds exactly the same rotations, inverted, when comparing B with A. This simply follows from the fact that

$$DIST_{frob}(R_1, R_2) = DIST_{frob}(R_1^T, R_2^T)$$

and that $R^{-1} = R^T$ for rotation matrices.[2] FN2

[2] To guarantee symmetry with all other clustering methods, our solution is to store both the viable rotation and the inverse rotation for each pair of consistent fragments and treat the inverse rotations the same way we treat the original rotations. This results in only little increase in computation time.

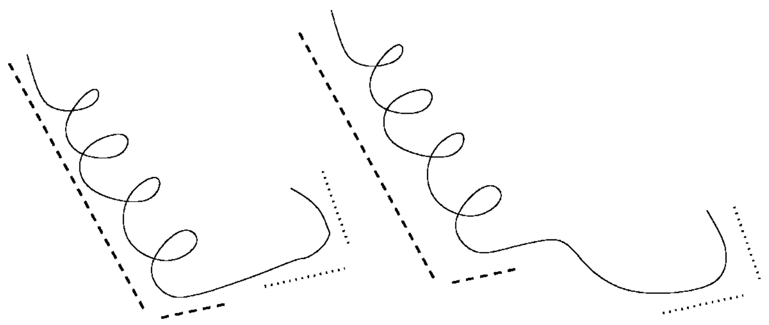

FIG. 3. Both the dashed and the dotted lines show resemblance of fragments under rotation only, but for the dotted match the translation is different than that for the dashed lines.

Picking representative rotations. A representative rotation is selected for each cluster that has at least n_{min} rotations (n_{min} is set to 3 in our tests). Note that the centroid cannot be used since averaging rotation matrices will most unlikely generate a valid rotation matrix. Instead, we select as representative rotation the one that minimizes the total distance from all other rotation matrices in the cluster.

2.2.3. Searching the reduced translation space. While the URMS metric is very effective in finding the optimal rotation, it is not as effective for fine-tuned alignments of protein structures. In particular, different fragment pairs can have similar rotations but different translations (see Fig. 3). $\boxed{\text{F3}}$

Since the RMS metric depends on the absolute coordinates, finding the right translation for each cluster of rotations is crucial for proper orientation of one structure with respect to the other and for accurate alignment. Each cluster of rotations contains similar rotation matrices, derived from different pairs of fragments that are in structural agreement. Our assumption is that for the most part the cluster corresponds to fragments that can form a locally consistent alignment; therefore, the proper translation should be one of the translations that are associated with the fragment pairs. We refer to this set of translations as the *reduced translation space*.

To find the best translation, we run a search in the reduced translation space. The protein structures are first rotated according to the cluster rotation. For each fragment pair, the structures are then translated such that the fragments are perfectly aligned to each other. Given this new orientation, we evaluate the quality of the match by summing the total distance between the atoms in all fragment pairs in the cluster. The final cluster translation is selected as the one that minimizes that total distance.[3] $\boxed{\text{FN3}}$

2.2.4. Alignment. Given the pair of rotation and translation, the structures are oriented accordingly. Our goal now is to find the most similar substructures. This is essentially a search for local similarity, and as such it requires a scoring function that would penalize far-apart residues and gratify close residues. In Section 3, we study the statistical properties of structural alignments to derive an effective scoring function based on the RMS metric. Given the scoring function, we employ the dynamic programming algorithm to maximize the structural similarity. This is repeated for each cluster, given its representative rotation and translation, yielding the best alignment for this cluster.

2.2.5. Iterative refinement. Since the rotation space is not well clustered, the output might be sensitive to the selection of the clustering algorithm and to the representative transformation in each cluster. The selected rotation and translation might be suboptimal, resulting sometimes in alignments with relatively high RMS distance. To improve the quality of the alignment, we apply an iterative refinement procedure that resembles the one described by Levitt and Gerstein (1998). Given the alignment from the previous iteration, we compute the RMS distance for the aligned residues and use the rotation matrix and translation vector to redefine the scoring function (see Section 3). The structures are realigned using the new scoring function and the process is repeated. The procedure continues until a maximal number of iterations has

[3]An alternative quality measure is the total number of fragment pairs whose average distance is below a certain threshold (as defined in Section 3).

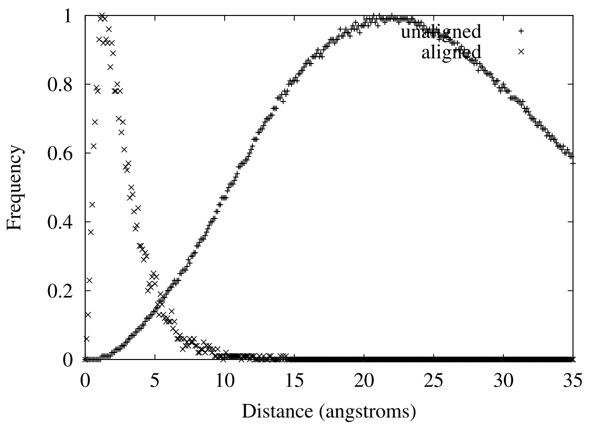

FIG. 4. Distribution of atomic distances between residues in structural alignments.

been reached or until convergence (the alignment remains the same). Since the procedure is not guaranteed to improve the alignment, we save all intermediate results, and at the end of the iterative process we keep the best alignment (the most significant one, as discussed in Section 4). To save computation time, the process is triggered only for alignments that are significant after the first iteration (this option can be turned off to increase sensitivity).

2.2.6. Output. At this final stage of the algorithm, we sift through the set of the alignments computed above. Each significant alignment is reported. Due to repetitions or matches between different domains for multidomain protein, there might be more than a single significant match. We sort the matches based on their significance as described in Section 4. To eliminate overlapping matches, all matches that overlap with previously reported matches are excluded. Each match is reported with its score, the RMS, and the statistical significance (see below).

3. CHOOSING THE OPTIMAL SCORING FUNCTION

Many similarities between protein structures are actually localized to a single domain or structural motif. Detecting these motifs is a hard problem and has been a major hurdle for structure comparison algorithms. The main reason is that unlike sequence comparison, where the notion of the most similar subsequence is well defined (given the scoring matrix), no such notion has been agreed upon for protein structures, partly because there is no natural measure of similarity for individual residues as is the case for protein sequences.

To define a scoring matrix for local structure alignment, we study the statistical properties of a large population of structural alignments (Leung and Yona, unpublished results). Given the alignments that were generated with CE[4] (Shindyalov and Bourne, 1998), we derive the distributions of atomic distances between aligned residues and unaligned residues (Fig. 4). As the graph indicates, the two populations are well separated and the transition occurs around a threshold distance of 5 Å. FN4 F4

We generate the scoring matrix by simply converting the atomic distances to similarity scores. This is repeated for every candidate transformation (rotation and translation). Specifically, if the distance between residues A_i and B_j under the candidate transformation is d_{ij}, then their similarity is defined as $s_{ij} = SHIFT - d_{ij}$ (a similar transformation was suggested by Holm and Sander [1993] and by Levitt and Gerstein [1998].) Based on the graphs in Fig. 4, we conclude that a reasonable range for the *SHIFT* is between 4 Å and 7 Å. Gap penalties are modeled using an affine function, because of its simplicity

[4]We use another algorithm to generate these distributions. However, we believe that these distributions reflect the true properties of structural alignments as they are derived from significantly similar structures.

and speed of computation, and are set to a range similar to that of *SHIFT*, with the gap extension being roughly one order of magnitude less than the gap opening penalty. A range of values is explored for each parameter, and the parameters are optimized over a subset of the data set described in Section 5, using the same methodology.

4. STATISTICAL SIGNIFICANCE

The raw similarity score of the optimal alignment does not necessarily indicate true relationship, as it tends to increase just by chance with the length of compared structures. To assess the biological relevance of a match, one needs to know the expected score for a match between two random proteins of the same length.[5] In our case, however, there are two different measures that can be used to score the match: the $\boxed{\text{FN5}}$ *similarity score* that is based on the atomic distances and *accounts for gaps* and the *RMS distance* between *aligned residues*. It is the combination of both that will determine the relevance of a match. The higher the score and the lower the RMS, the better is the match.

To assess the significance, we ask what is the probability to get a match with the observed score and RMS by chance. Formally, given two proteins of length m and n and a match of length l with similarity score S and RMS R, we compute the probability P

$$\begin{aligned} P(\textbf{Match}/m,n) &= P(S,R/m,n) \\ &= P(S/m,n)P(R/S,m,n). \end{aligned}$$

These probabilities can be derived from the empirical background distributions for random proteins. However, unlike sequences, random structures are not easy to generate. Moreover, a random structure is not likely be stable, as stereo-chemical constraints limit the conformation space to a small fraction of all possible shapes. A more sensible solution is to use existing but unrelated proteins. We generate this population from structure pairs that belong to different SCOP folds and classes (Hubbard *et al.*, 1999). The matches between these pairs are considered random, and a total set of 139,047 alignments is generated and analyzed.[6] $\boxed{\text{FN6}}$

4.1. *Estimating $P(S/m,n)$*

Under our definition of the scoring function, the compared structures can be viewed essentially as sequences with a position-specific scoring function. It is well known that local sequence similarity scores follow the extreme value distribution (Karlin and Altschul, 1990; Dembo *et al.*, 1994a). Many of the principles that underlie the theory for random sequence matches hold here as well, as the properties required from the scoring function are easily met by our structure-based scoring function. Clearly, there are at least a few positive matches (some residues are close to each other); however, when considering all pairs of atoms the average score is negative, as only $O(n)$ positions can be aligned at the most (positive scores), while there are n^2 entries in the matrix. Under these conditions, the score of a random match between a sequence of length m and a sequence of length n has been shown to be centered around $a \cdot \ln(m \cdot n)$ where a is a function of the scoring matrix. Indeed, the distribution of similarity scores as a function of the product of the lengths of the structures compared follows a logarithmic curve, as is shown in Fig. 5a. The transformation of these scores to zscores, by subtracting the expected average and dividing by $\boxed{\text{F5}}$ the standard deviation, results in a distribution that follows nicely the extreme value distribution (Gumbel, 1958) as is shown in Fig. 5b. We denote the pvalue of the similarity score by *Pvalue*1.

[5]Note that this is different from a random match, where the relative position of the fragments and their orientation is selected at random. The dynamic programming algorithm searches for the best match between the input structures, therefore the randomness should be introduced in the input.

[6]Even in a population of supposedly random pairs one might observe structural similarities due to recurrence of small motifs of secondary structures.

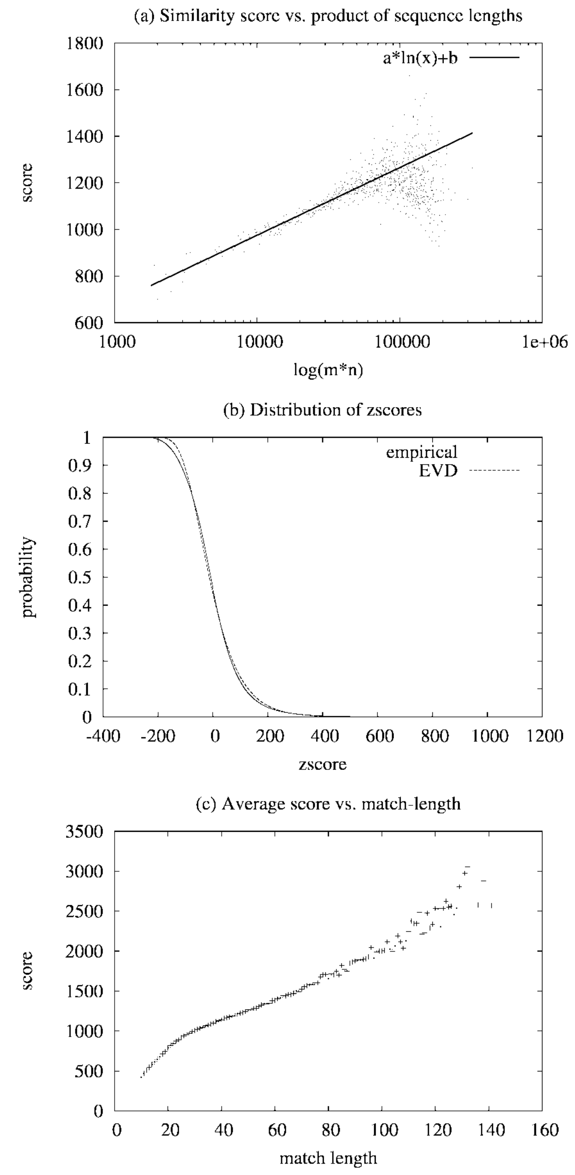

FIG. 5. Statistical properties of the scores of random matches.

4.2. Estimating $P(R/S, m, n)$

Since the score is linearly dependent on the length of the match (see Fig. 5c) we can use the approximation

$$P(R/S, m, n) \approx P(R/l, m, n).$$

Since the RMS depends only on the length of the match,

$$P(R/l, m, n) = P(R/l).$$

The distribution of $P(R/l)$ is characterized in Fig. 6a. Surprisingly, it seems that the RMS reaches saturation for match lengths around 100 residues, and the functional forms of the average as well as of the standard deviation (Fig. 6b) are of tangent hyperbolic functions. The transformation to zscores results in a $\boxed{\text{F6}}$

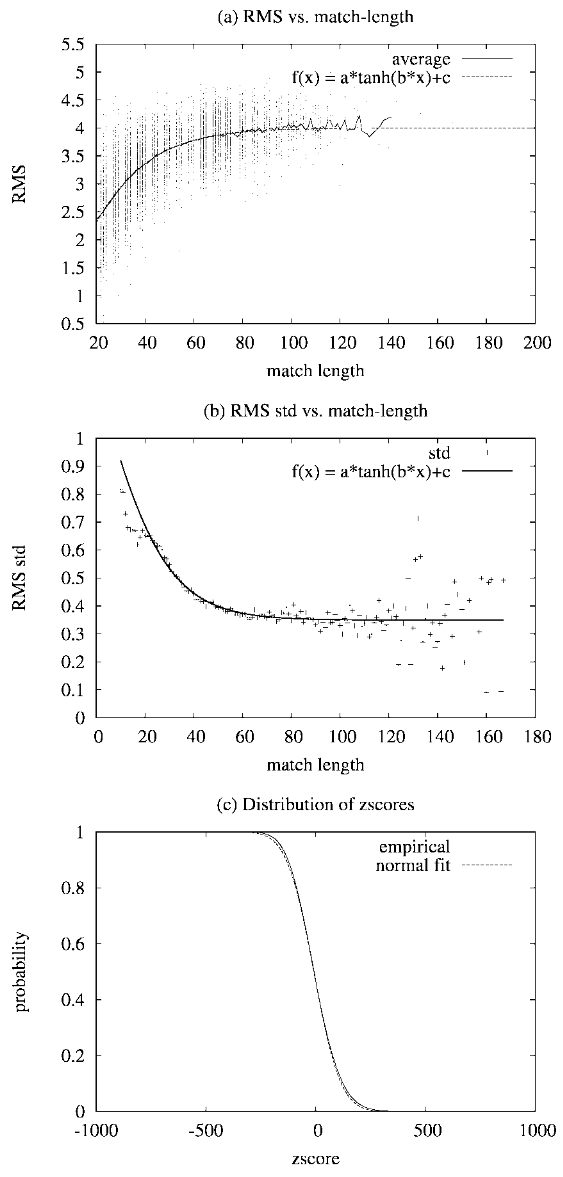

FIG. 6. Statistical properties of the RMS of random matches.

normal distribution, whose cumulative distribution is plotted in Fig. 6c. We denote the pvalue of the RMS by *Pvalue*2.

4.3. The significance of a match

Using the cumulative distributions for the zscores (Fig. 5b, Fig. 6c), we can estimate the probability to obtain an alignment with score $> S$ and RMS $< R$. The significance is estimated by the product of these two probabilities

$$\begin{aligned} Pvalue(\mathbf{Match}/m, n) &= Pvalue(S, R/m, n) \\ &\approx P(S' > S/m, n)P(R' < R/l) \\ &= Pvalue1 \cdot Pvalue2. \end{aligned}$$

In practice, however, the events are not independent, and the estimates might be overly optimistic.

5. PERFORMANCE EVALUATION

To assess our algorithm, we tested it on the SCOP structural classification of proteins (Hubbard *et al.*, 1999). This hierarchy of protein domain families that is generated manually based on expert knowledge is an excellent benchmark for structure comparison algorithms. The SCOP hierarchy consists of class, fold, superfamily, and family. Being able to recover this hierarchy with an automatic structure comparison algorithm is an indication of the algorithm's sensitivity and accuracy. Here we focused on the superfamily level. Our goal was to detect as many as possible family pairs that belong to the same superfamily, while minimizing the number of false positives.

5.1. Datasets

Our dataset consists of 1,287 domain families (SCOP 1.50). These families are organized into 814 superfamilies, 545 folds, and 7 classes. A subset of 456 families (families within superfamilies that contain at least 3 families) is used in our analysis. For each family, a structural representative is selected based on the minimal average sequence distance from all other members of the family and the SPACI index that measures the quality of the structure (Brenner *et al.*, 2000).

5.2. Sensitivity results

To evaluate the accuracy and sensitivity of the method, we use several indices based on the ROC measure. The first measure we used is the ROC1 index. ROC1 is simply the number of true positives detected before the first false positive. To compute this index, we compare the representative structure of each family with the 1,287 other structures in our dataset. The results are sorted based on the pvalue and the number of true positives detected before the first false positive is counted. Finally, the results are summed over all queries.

Our definition of a false positive depends on what we consider to be a positive and what is deemed negative. We use two definitions of negatives. The first, more restrictive one considers all structures outside of the superfamily of the query structure to be negatives. However, structures within the same fold may be significantly similar as well. Our second definition of a negative is a more permissive one and considers all similarities within the same fold to be positives, while similarities outside the fold are considered negatives. The results in both cases are listed in Table 1. T1

We compare our results with those of CE (Shindyalov and Bourne, 1998). This widely used algorithm is generously available from the CE website. As the results indicate, the hybrid algorithm performs very well. The running times of these algorithms are also comparable, with 1.5 seconds per alignment on average with our hybrid algorithm (2.5 seconds with the iterative version), compared to 5.75 seconds with CE.

5.3. Accuracy results

To measure the alignment accuracy, we compute the average RMS over all pairs of families that belong to the same superfamily. Altogether, 678 alignments were processed, and the results are summarized in Table 2. As the results indicate, the hybrid algorithm tends to produce alignments with lower RMS on average. T2

TABLE 1. ROC1 PERFORMANCE EVALUATION RESULTS[a]

	CE	*Hybrid URMS -RMS*	*Hybrid URMS -RMS (iterative)*
Type of first false-positive			
Different superfamily	187	269	322
Different fold	197	282	337

[a]For each method, we report the number of true relationships that are detected before the first false positive occurs. Results are given for the following types of false positive: different superfamily and different fold (see text for details). For the hybrid algorithm, we report results for the single-iteration version and the multiple-iteration version (see "iterative refinement" in Section 2.2.5).

TABLE 2. ACCURACY OF STRUCTURAL ALIGNMENTS[a]

CE	*Hybrid URMS -RMS*	*Hybrid URMS -RMS (iterative)*
3.97 (105 aa)	3.3 (98 aa)	3.16 (107 aa)

[a]For each method, we report the average RMS and the average length of the alignment (in parentheses).

4C

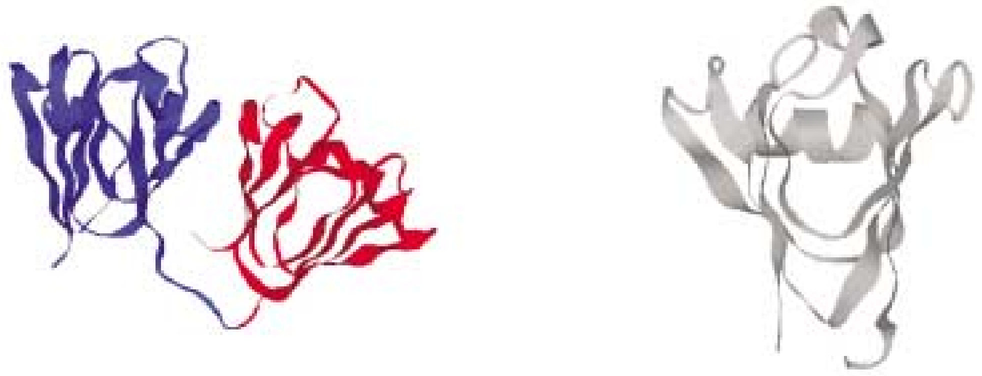

FIG. 7. Left: SCOP domain d1bd7a (b.11.1.1 beta-Crystallin). **Right:** SCOP domain d1prr_1 (b.11.1.1 Protein S). Note that d1bd7a is twice as large as d1prr_1.

6. EXAMPLES

To demonstrate shape similarities that are detected by our hybrid algorithm, we tested it over structures with interesting geometries and compared its performance with other shape comparison systems available on the web, such as CE (Shindyalov and Bourne, 1998) and DALI (Holm and Sander, 1993).

Our first example is of two SCOP domains, d1bd7a and d1prr_1 (Fig. 7). These two domains belong to the same family in SCOP. Our algorithm detects five significant nonoveralpping alignments between the two structures (Fig. 8). F7 F8

Interestingly, the hybrid algorithm detects that domain d1bd7a is comprised of two copies of domain d1prr_1. This is indicated by two structural alignments. The first (corresponding to rotation 1) matches positions 1–86 of d1prr_1 with positions 1–86 of d1bd7a with RMS = 1.913 and 24% sequence identity (see alignments above). The second (corresponding to rotation 2) matches positions 1–85 of d1prr_1 with positions 91–175 of d1bd7a with RMS = 2.265 and 25% identity (the alignment is depicted in Fig. 9a). Furthermore, an internal symmetry is found within each domain. For example, the second half of d1prr_1 (46–86) is aligned with residues 90–131 of d1bd7a with RMS = 2.005 (rotation 11). This alignment and the two other alignments (rotations 3,9) imply that each subdomain of d1bd7a is divided into two (similar) subsubdomains, as is also the case for d1prr_1 (Fig. 9b). These alignments suggest that this domain family can be broken into smaller domains. Note that both CE and DALI report only one of the five alignments that are detected with the RMS-URMS algorithm, as is shown in Fig. 10. Both correspond to rotation 1 in our results. F9 F10

Our second example is the PDB proteins 1dtl (chain A) and 1hqv (chain A). These proteins are composed of two very similar domains (Fig. 11) that are glued to each other in different rotations (see overlay of the domains in Fig. 12). The hybrid method detects and displays the rotations of all of the four possible superpositions of these domains (Fig. 13). DALI reports only one of these matches (Fig. 14), while CE reports three (the second and the third are basically a continuation of the first, broken apart by CE because of physical distance between coordinates.[7]) F11 F12 F13 F14 FN7

[7]The CE algorithm breaks the chain into fragments if the physical distance between consecutive residues exceeds a certain threshold, suggesting discontinuous residue coordinates

d1prr_1 (90 aa) ↔ d1bd7a (176 aa)

Rotation 1

```
RMS:  1.913 along 82 aa overlap    Pvalue1: 3.5e-06   Pvalue2: 6.0e-10  Log(PvalueTotal): -14.67

1     MANITVFYNEDFQGKQVDLPPGNYTRAQLAALGIENNTISSVKVPPGVKAILYQNDGFAGDQIEVV.ANAEELG...PLNNNVSSIRVIS   86
      :::I::::N::F:GK::::::::::  ::A:G:: :::SSV:V:: :::::YQ::G::G:Q:::: :::::::   :::::V:S:R:I:
1     EHKIILYENPNFTGKKMEIVDDDVPS..FHAHGYQ.EKVSSVRVQS.GTWVGYQYPGYRGLQYLLEKGDYKDNSDFGAPHPQVQSVRRIR   86

24% identity in 82 aa overlap
```

Rotation 2

```
RMS:  2.265 along 79 aa overlap    Pvalue1: 4.6e-05   Pvalue2: 6.1e-07  Log(PvalueTotal): -10.55

1     MANITVFYNEDFQGKQVDLPPGNYTRAQLAALGIENNTISSVKVPPGVKAILYQNDGFAGDQIEVVANAE....ELG..PLNNNVSSIRVI   85
      :::I::F::E:FQG::::L :G::::  L:::G:  ::::SV:V:: :::::Y::::::G:Q::::::::    :::  ::::::SS:R:I
91    NPKIIIFEQENFQGHSHEL.SGPCPN..LKETGM..EKAGSVLVQA.GPWVGYEQANCKGEQFVFEKGEYPRWDSWTSSRRTDSLSSLRPI  175

25% identity in 79 aa overlap
```

Rotation 11

```
RMS:  2.005 along 41 aa overlap    Pvalue1: 0.01   Pvalue2: 6.1e-04  Log(PvalueTotal): -5.11

46    PGVKAILYQNDGFAGDQIEVVANAEEL.GPLNNNVSSIRVIS   86
      :::K:I::::::F:G:::E:::::::L ::::::::S::V::
90    GNPKIIIFEQENFQGHSHELSGPCPNLKETGMEKAGSVLVQA  131

19% identity in 41 aa overlap
```

Rotation 9

```
RMS:  2.090 along 41 aa overlap    Pvalue1: 0.02   Pvalue2: 1.2e-03  Log(PvalueTotal): -4.69

47    GVKAILYQNDGFAGDQIEV.VANAEEL..GPLNNNVSSIRVISV   87
      ::K:ILY:N::F:G:::E: :::::::  ::::::VSS:RV:S:
1     EHKIILYENPNFTGKKMEIVDDDVPSFHAHGYQEKVSSVRVQSG   44

34% identity in 41 aa overlap
```

Rotation 3

```
RMS:  2.374 along 43 aa overlap    Pvalue1: 0.01   Pvalue2: 4.7e-03  Log(PvalueTotal): -4.27

2     ANITVFYNEDFQGKQVDLPPGNYTRAQLAALGIENNTISSVKVPP   46
      ::::::::::::G:Q::L::G:Y::  ::::G:::::::SV::::
44    GTWVGYQYPGYRGLQYLLEKGDYKD..NSDFGAPHPQVQSVRRIR   86

18% identity in 43 aa overlap
```

FIG. 8. Alignment of d1prr_1 and d1bd7a using the Hybrid algorithm. Only significant matches are reported (total of five different rotations). For each rotation, we report the match length (excluding gaps), the RMS, and the significance. *Pvalue*1 is based on the score (not shown); *Pvalue*2 is based on the RMS.

7. DISCUSSION

In this paper, we present a new method for protein structure comparison. We study the notion of local structural similarities and investigate the statistical properties of structural matches to derive sensible similarity measures for structural alignments. We then describe an efficient algorithm for local structure comparison. Our algorithm collects information about significant rotations using the URMS metric and generates local pairwise alignments using the RMS-based metric. The algorithm can report more than just a single significant alignment, such as between multidomain proteins.

The hybrid algorithm we present here differs from previous algorithms in several ways: (i) it combines the advantages of the URMS and the RMS metrics; (ii) it searches methodically the transformation space; (iii) it

4C

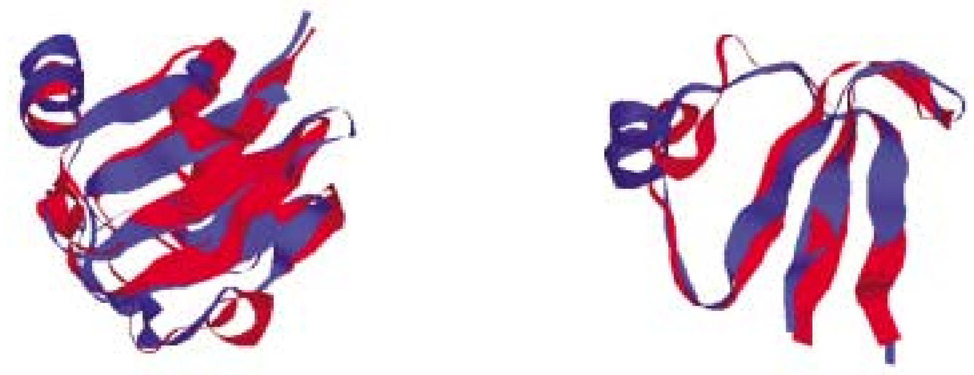

FIG. 9. **Left:** Structural alignment of d1prr_1 with the first half of d1bd7a. **Right:** The first half domain of d1prr_1 (positions 46–86) matches the second half of the subdomain in d1bd7a (positions 90–131).

d1prr_1 (90 aa) ↔ d1bd7a (176 aa)

DALI results

```
RMS: 2.1 along 83 aa residues    Zvalue: 11.2

1     MANITVFYNEDFQGKQVDLPPGNYTRaqLAALGIEnNTISSVKVPPgVKAILYQNDGFAGDQIEVV.ANAEEL....GPLNnNVSSIRVISVP   88
         I .: N .F GK. ::   .       A G :   :SSV:V       : YQ  G: G Q  .   . .:       P    V S:R I
1     EHKIILYENPNFTGKKMEIVDDDVPS..FHAHGYQ.EKVSSVRVQS.GTWVGYQYPGYRGLQYLLEkGDYKDNsdfgAPHP.QVQSVRRIRDM   88

25% identity in 83 aa overlap
```

CE results

```
RMS: 2.06 along 83 aa overlap    Zvalue: 5.00

1     MANITVFYNEDFQGKQVDLPPGNYTRAQLAALGIENNTISSVKVPPGVKAILYQNDGFAGDQIEVVA.NAEELG...PLNNNVSSIRVISV   87
         I .: N .F GK..::   .       A G :   :SSV:V       : YQ  G: G Q  .   . .:       .  V S:R I
1     EHKIILYENPNFTGKKMEIVDDDVPS..FHAHGYQ.EKVSSVRVQS.GTWVGYQYPGYRGLQYLLEKGDYKDNSDFGAPHPQVQSVRRIRD   87

24% identity in 83 aa overlap
```

FIG. 10. Dali and CE alignment of d1prr_1 and d1bd7a.

can detect complex similarities, such as between multidomain proteins; (iv) it can detect structural repetition (multimers); and (v) it yields symmetric outputs, i.e., the same results are obtained when comparing protein **A** with protein **B**, as when the input is reversed.

The algorithm has many computational options, catering for a large variety of applications, and it can be tuned at different resolutions, depending on the desired speed, sensitivity, and accuracy. Using the default set of parameters, the algorithm is very fast, with an average of 1.5 seconds per comparison. As we have shown, the algorithm compares very well with CE on the SCOP benchmark.

In this paper, we adhered to comparing two protein structures using the traditional set of rigid transformations (composed of rotations and translations). However, it is quite common that the optimal transformation involves deformation of the structure that cannot be expressed only in terms of one rigid transformation. Nevertheless, our algorithm generates a number of pairwise alignments that correspond to different significant or popular rotations. We believe that this collection of output transformations sheds more light on the exact relation between the two structures.

The URMS-RMS hybrid algorithm is currently being used to generate structural alignments between all PDB structures. The alignments will be available at the biozon website, at *biozon.org*. The software is available for download at *biozon.org/ftp/software/urms/*.

APPENDIX A—SEARCHING THE ROTATION SPACE EFFICIENTLY

Using secondary structure information

Searching the effective rotation space as described in Section 2.2.1 requires comparing each pair of fragments from structures **A** and **B**. The time complexity of this step is $O(mn)$ where $m = length(\mathbf{A})$ and $n = length(\mathbf{B})$. To speed up this step, one can compare only fragments of similar secondary structures (i.e., a strand with a strand, a helix with a helix, and a coil with a coil). We tested a variant in which we considered only elements of consistent secondary structure elements; however, this reduced the quality of the sample set (sparse statistics) and resulted in inferior results. To obtain better sampling, one can sample secondary structure elements considering all possible shifts; however, this does lead to only moderate saving in computation time.

Speedups

The computation of $DIST_{urms}(\mathbf{a}, \mathbf{b})$ requires finding the optimal rotation for that fragment pair. To speed up the computation, we approximate this distance with the *surface distance* $DIST_{surface}$ based on the length of edges connecting the unit vectors (see Fig. 15). The optimal rotation is computed only for fragment $\boxed{F15}$ pairs that are considered close according to the surface distance. The surface distance between **a** and **b** is defined as

$$DIST_{surface}(\mathbf{a}, \mathbf{b}) = \sum_{i=2}^{l} \left|\ ||\vec{u_i}^{a} - \vec{u_{i-1}}^{a}||^2 - ||\vec{u_i}^{b} - \vec{u_{i-1}}^{b}||^2\ \right|.$$

A fragment pair is said to be a *candidate fragment pair* if $DIST_{surface}(\mathbf{a}, \mathbf{b}) < T_{surface}$ where $T_{surface}$ is set to 2 (based on the distribution of surface distances, see Fig. 16). The intuition is that if the surface $\boxed{F16}$ distance is small, then it is quite likely that the two sequences of unit vectors match well under the optimal rotation, and the URMS-based distance satisfies $DIST_{urms}(\mathbf{a}, \mathbf{b}) < T_{urms}$. Indeed, the correlation between these two measures is strong. On average, for 98% of all fragment pairs of URMS distance $< T_{urms}$, the surface distance is less than 4. This threshold, however, is too permissive as most fragments within surface distance of 4 are not in structural agreement as defined in Section 2.1.2. To improve efficiency, we set the surface distance threshold to $T_{surface} = 2$. With this threshold, about 57% of the candidate fragment pairs are in structural agreement, at a moderate cost in sensitivity (eliminating 21% of the fragment pairs that are in structural agreement).

The main advantage of the surface distance is that it is much faster to compute than $DIST_{urms}$, as it does not involve computing the optimal rotation, thus allowing fast rejection of many fragment pairs at this initial stage.

APPENDIX B—CLUSTERING THE ROTATION SPACE

Clustering methods

We tested four simple and fast clustering algorithms for vector quantization. Other, more sophisticated clustering algorithms can clearly be employed as well, but at the cost of increased computation time.

Greedy clustering: The greedy clustering algorithm is a fuzzy online clustering algorithm. Given a new rotation, we compute its distance from the existing clusters. The distance from a cluster is defined as the minimal distance from a cluster member. The rotation is classified to all the clusters that are within a distance D_0, where D_0 was set through optimization to 0.1 (when using the matrix representation and the Frobenius distance) or $30'$ (when using the angle representation, as described below).

Grid clustering: This algorithm attempts to search the rotation space while eliminating possible biases due to the metric used and the geometry of clusters, by forming a grid and binning the rotation space into fixed-size bins. Each bin is considered a cluster (the size of the bin is determined through optimization).

Pairwise clustering (average linkage): This is the common agglomerative clustering algorithm. The algorithm starts with singletons and successively merges the closest clusters, as long as their distance does

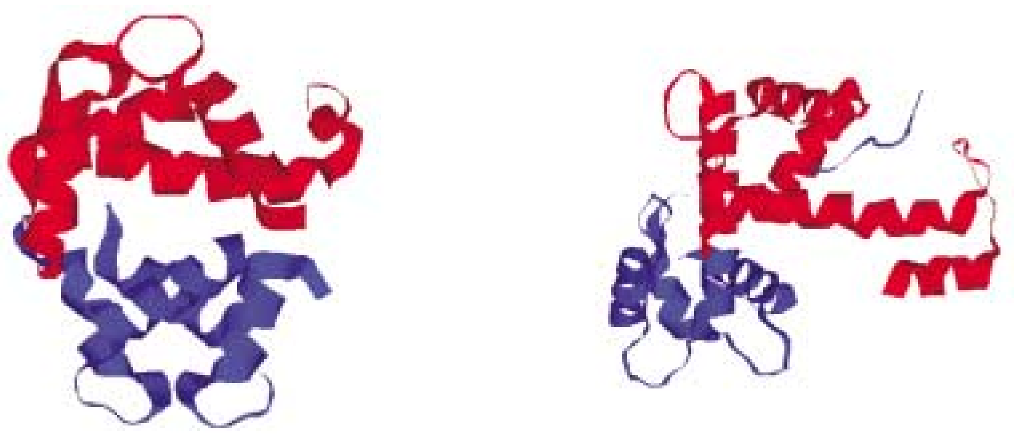

FIG. 11. PDB structures 1dtl:A (**left**) and 1hqv:A (**right**).

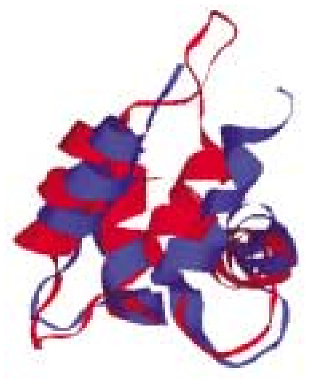

FIG. 12. One of the four matches found between 1dtl:A.pdb and 1hqv:A.pdb.

not exceed a predefined threshold D_0. The distance between a pair of clusters is defined as the average distance over all pairwise distances between the members of these clusters.

Connected components (single linkage): This algorithm is a variation over the general pairwise clustering algorithm described above, where the distance between clusters is defined as the minimal distance of all pairwise distances between the clusters' members. When the merging process stops, the result is a set of connected components. The algorithm can be implemented more efficiently than the general pairwise clustering algorithm. Beacuse of its speed and the symmetry (Section 2.2.2), this algorithm is our default choice.

Sampling

For very large protein structure, the set of viable rotations might include thousands of rotations. To reduce computation time involved with clustering such large sets of rotations, we sample the rotation space randomly. We limit the size of the sample set to 1,000 viable rotations. Note that the symmetry is no longer guaranteed when sampling is used (see Section 2.2.2). This option can be turned off to guarantee symmetry.

Clustering rotation matrices versus clustering angles

To cluster the rotations, one needs a distance function between rotations. The Frobenius distance between rotation matrices (defined as the Euclidean distance between their representations as nine-dimensional vectors) is less than optimal as the high dimensional space of rotation matrices seems to be sparsely

1dtl_A:A (149 aa) ↔ 1hqv_A:A (178 aa)

Rotation 6

```
RMS:  2.289 along 65 aa overlap    Pvalue1: 7.2e-04   Pvalue2: 6.7e-06  Log(PvalueTotal): -8.32

81   MKKSEEELSDLFRMFDKNADGYIDLEELKIMLQA......TTITEDDIEELMKDGDKNNDGRIDYDEFLEFMKG  148
     :::::::L:::F:::DK:::G:I:::EL:::         T::::::::::::::D::N::::::EF::::K:
11   ALPDQSFLWNVFQRVDKDRSGVISDNELQQA...LSNGTWTPFNPVTVRSIISMFDRENKAGVNFSEFTGVWKY   81

21% identity in 65 aa overlap
```

Rotation 3

```
RMS:  2.147 along 61 aa overlap    Pvalue1: 7.3e-03   Pvalue2: 2.3e-06  Log(PvalueTotal): -7.77

83   KSEEELSDLFRMFDKNADGYIDLEELKI......MLQATTITEDDIEELMKDGDKNNDGRIDYDEFLEFMK  147
     K::::::::FR::D::::G:ID::ELK:      ::    ::::::::L::::D::::G:I::D:F:::::
80   KYITDWQNVFRTYDRDNSGMIDKNELKQALSGFGYR....LSDQFHDILIRKFDRQGRGQIAFDDFIQGCI  146

26% identity in 61 aa overlap
```

Rotation 2

```
RMS:  2.954 along 75 aa overlap    Pvalue1: 2.4e-04   Pvalue2: 3.1e-03  Log(PvalueTotal): -6.13

12   QKNEFKAAFDIFVLGAEDGSISTKELGKVMRML.GQNPTPEELQEMIDEVDEDGSGTVDFDEFLVMMVRSMKKSEEELSD   90
     ::::::::F:::: ::::G:IS::EL::::: : :::::P::::::I:::D::::::V:F:EF::::.      ::::::
14   DQSFLWNVFQRVD.KDRSGVISDNELQQALSNGTWTPFNPVTVRSIISMFDRENKAGVNFSEFTGVWKYI...TDWQNVF   89

17% identity in 75 aa overlap
```

Rotation 1

```
RMS:  2.839 along 75 aa overlap    Pvalue1: 8.6e-04   Pvalue2: 1.2e-03  Log(PvalueTotal): -6.00

9    TEEQKNEFKAAFDIFVLGA..EDGSISTKELGKVMRMLGQNPTPEELQEMIDEVDEDGSGTVDFDEFLVMMVRSMKKS..   84
     ::::::::::::::F::::  ::G:I:::EL::::::G:::::::::::I:::D::G:G:::FD:F
75   FTGVWKYITDWQNVFRTYDRDNSGMIDKNELKQALSGFGYRLSDQFHDILIRKFDRQGRGQIAFDDFIQG.......CI  146

85   ...EEELSD   90
        ::::::
147  VLQRLTDIF  155

17% identity in 74 aa overlap
```

FIG. 13. Alignment of 1dtl_A 1hqv_A using the Hybrid algorithm.

populated even for closely related structures (Fig. 17). Therefore, it is hard to determine typical within-cluster distances. F17

Motivated by these issues, as well as by computational considerations, we explored the compact *angles representation* for the rotation space. In the angles representation, a rotation is given by a three-dimensional vector of its x, y, and z rotation angles. Under that representation, the distance between two rotations is defined as the angle-adjusted $L1$ distance between their three-dimensional vectors. Specifically, given two angle-vectors $\vec{\alpha} = (\alpha_1, \alpha_2, \alpha_3)$ and $\vec{\beta} = (\beta_1, \beta_2, \beta_3)$, their distance is defined as

$$DIST_{angles}(\vec{\alpha}, \vec{\beta}) = \sum_{i=1}^{3} f(|\alpha_i - \beta_i|)$$

where $f(x) = \min(x, 360 - x)$.

When clustering rotations using that representation, we save time both when computing distances between rotations and when selecting a representative rotation for a cluster, since angles can be averaged (and the cluster centroid is selected as the representative). It should be noted that in mapping rotation matrices to angles, we detect a relatively small number of rotation matrices with a negative determinant

1dtl_A:A (149 aa) ↔ 1hqv_A:A (178 aa)

DALI results

```
RMS: 7.0 along 95 aa residues    Zvalue: 6.6

1    YKAAVEQl.................TEEQKNEFKAAFDIFVLGAeDGSISTKELGKVMRMLGQNPTPEELQEMIDEVDE   62
                                .   :..  F  :      G I  EL . :   G   . :   :I . D
53   VTVRSII.smfdrenkagvnfseftgVWKYITDWQNVFRTYDRDN.SGMIDKNELKQALSGFGYRLSDQFHDILIRKFDR   130

63   DGSGTVDFDEFLVMMVRSmkk.SEEELSDlfrmfdkn.........ADGYidleelkimlqattiteddieelmkdgdkn   132
      G G : FD:F:   :         ::
131  QGRGQIAFDDFIQGCIVL...qRLTDIFR........rydtdqdgwIQVS..............................   169

133  ndgr.IDYDEF   142
            Y
170  ....yEQYLSM   176

15% identity in 95 aa overlap
```

CE results

```
1dtl_A:A   82 aa         <->     1hqv_A:A:1   168 aa

RMS: 3.13 along 79 aa overlap    Zvalue: 4.60

1    YKAAVEQ...............LTEEQKNEFKAAFDIFVLGAEDGSISTKELGKVMRMLGQNPTPEELQEMIDEVDEDGSGTVDFDEFLVMMVRS    80
        .:                     .   :..  F  :      G I   EL . :   G   . :    :I . D  G G : FD:F:   :
54   TVRSIISMFDRENKAGVNFSEFTGVWKYITDWQNVFRTYD.RDNSGMIDKNELKQALSGFGYRLSDQFHDILIRKFDRQGRGQIAFDDFIQGCIVL   148

16% identity in 79 aa overlap


1dtl_A:A:1   33 aa       <->     1hqv_A:A:1   168 aa

RMS: 2.04 along 32 aa overlap    Zvalue: 3.90

84   SEEELSDLFRMFDKNADGYIDLEELKIMLQAT   115
      :  L ..F.  DK.  G I   EL.  L
14   DQSFLWNVFQRVDKDRSGVISDNELQQALSNG    45

28% identity in 32 aa overlap

1dtl_A:A:2   34 aa       <->     1hqv_A:A:1   168 aa

RMS: 1.11 along 32 aa overlap    Zvalue: 4.10

116  TITEDDIEELMKDGDKNNDGRIDYDEFLEFMK   147
        :  :  . .  DK.  G I  .E  : :
12   LPDQSFLWNVFQRVDKDRSGVISDNELQQALS    43

15% identity in 32 aa overlap
```

FIG. 14. Dali and CE alignment of 1dtl_A 1hqv_A.

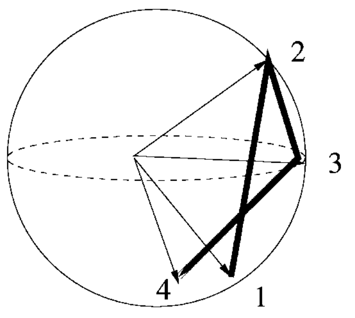

FIG. 15. The unit sphere with four unit vectors; their surface distance is measured along the bold lines.

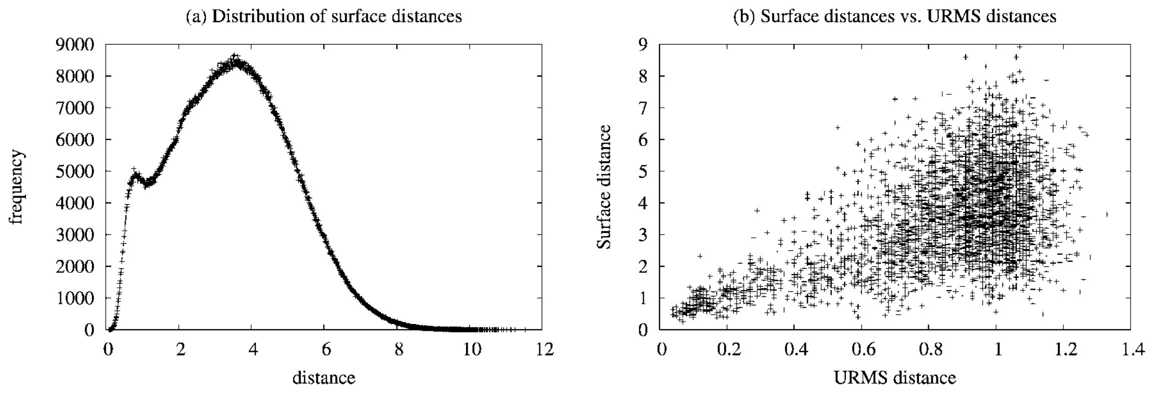

FIG. 16. Surface distances as approximations for URMS distances. (**a**) Distribution of surface distances. (**b**) Surface distances vs. URMS distances.

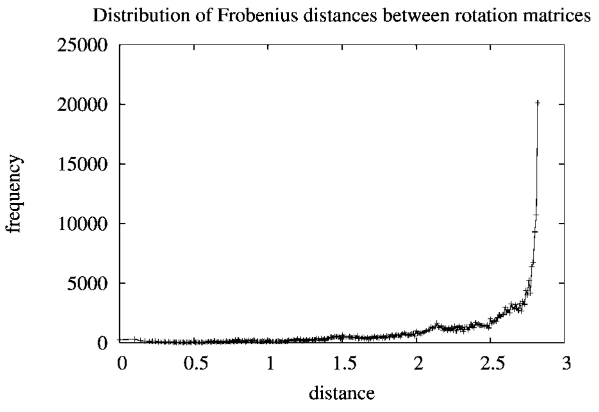

FIG. 17. Distribution of Frobenius distances between rotation matrices.

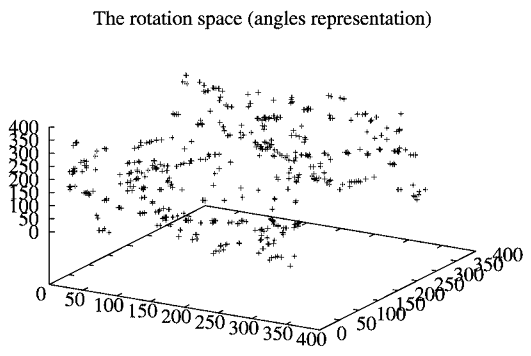

FIG. 18. The rotation space (angles representation). All viable rotations for domain families a.3.1.1 (SCOP:d2ycc Mitochondrial cytochrome c) and a.3.1.2 (SCOP:d1qksa1 N-terminal domain of cytochrome cd1-nitrite reductase) are plotted.

(count for about 10% of the rotation matrices). These matrices represent isometry (rotation, translation, and reflection) and are eliminated from the analysis for computational simplicity.

The angles representation is not only more compact. It is also insightful when characterizing the geometrical properties of the rotation space, and it is interesting to view the rotation space through its angles representation. Clearly, even for closely related structures, the rotation space is noisy and sparsely populated as can be seen in Fig. 18. F18

The only disadvantage of the angles representation is that the symmetry that is guaranteed when clustering rotation matrices (see Section 2.2.2) is not guaranteed with the angles representation, since for some rotations $DIST_{angles}(R_1, R_2) \neq DIST_{angles}(R_1^{-1}, R_2^{-1})$.

ACKNOWLEDGMENTS

This work is supported by the National Science Foundation under Grant No. 0133311 to Golan Yona.

REFERENCES

Altschul, S.F. 1991. Amino acid substitution matrices from an information theoretic perspective. *J. Mol. Biol.* 219, 555–565. AU2

Brenner, S.E., Koehl, P., and Levitt, M. 2000. The astral compendium for protein structure and sequence analysis. *Nucl. Acids Res.* 28, 255–256.

Chew, L.P., Kedem, K., Huttenlocher, D.P., and Kleinberg, J. 1999. Fast detection of geometric substructure in proteins. *J. Comp. Biol*. 6(3–4), 313–325.

Chew, L.P., and Kedem, K. 2002. Finding the consensus shape for a protein family. *Proc. ACM 18th Symp. on Computational Geometry*, 64–73.

Dayhoff, M.O., Schwartz, R.M., and Orcutt, B.C. 1978. A model of evolutionary change in proteins, *in* M. Dayhoff, ed., *Atlas of Protein Sequence and Structure*, vol. 5, suppl. 3, 345–352, National Biomedical Research Foundation, Silver Spring, MD.

Dembo, A., and Karlin, S. 1991. Strong limit theorems of empirical functionals for large exceedances of partial sums of i.i.d variables. *Ann. Prob.* 19, 1737–1755. AU2

Dembo, A., Karlin, S., and Zeitouni, O. 1994. Critical phenomena for sequence matching with scoring. *Ann. Prob.* 22, 1993–2021.

Fischer, D., Nussinov, R., and Wolfson, H. 1992. 3D substructure matching in protein molecules. *Proc. 3rd Intl. Symp. Combinatorial Pattern Matching*, 136–150.

Gerstein, M., and Levitt, M. 1996. Using iterative dynamic programming to obtain accurate pairwise and multiple alignments of protein structures. *Proc. of ISMB'96 Intelligent Systems for Molecular Biology*, 59–66.

Gerstein, M., and Levitt, M. 1998. Comprehensive assessment of automatic structural alignment against a manual standard, the SCOP classification of proteins. *Protein Sci.* 7, 445–456.

Gonnet, G.H., Cohen, M.A., and Benner, S.A. 1992. Exhaustive matching of the entire protein sequence database. *Science* 256, 1443–1445. AU2

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

Gumbel, E.J. 1958. *Statistics of Extremes*, Columbia University Press, New York.

Henikoff, S., and Henikoff, J.G. 1992. Amino acid substitution matrices from protein blocks. *Proc. Natl Acad. Sci. USA* 89, 10915–10919. AU2

Holm, L., and Sander, C. 1993. Protein structure comparison by alignment of distance matrices. *J. Mol. Biol.* 233, 123–138.

Holm, L., and Sander, C. 1996. Mapping the protein universe. *Science* 273, 595–602.

Hubbard, T.J., Ailey, B., Brenner, S.E., Murzin, A.G., and Chothia, C. 1999. SCOP: A structural classification of proteins database. *Nucl. Acids Res.* 27, 254–256.

Karlin, S., and Altschul, S.F. 1990. Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proc. Natl Acad. Sci. USA* 87, 2264–2268.

Kedem, K., Chew, L.P., and Elber, R. 1999. Unit-vector RMS (URMS) as a tool to analyze molecular dynamics trajectories. *Proteins Struct. Funct. Genet*. 37, 554–564.

Levitt, M., and Gerstein, M. 1998. A unified statistical framework for sequence comparison and structure comparison. *Proc. Natl. Acad. Sci. USA* 95, 5913–5920.

Orengo, C.A., Brown, N.P., and Taylor, W.R. 1992. Fast structure alignment for protein databank searching. *Proteins Struct. Funct. Genet.* 14, 139–167.

Ortiz, A.R., Strauss, C.E.M., and Olmeda, O. 2002. MAMMOTH (Matching molecular models obtained from theory): An automated method for model comparison. *Protein Sci.* 11, 2606–2621.

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

Rackovsky, S., and Scheraga, H.A. 1978. Differential geometry and polymer conformations 2. Development of a conformational distance function. *Macromolecules* 13, 1440–1453.

Rackovsky, S., and Goldstein, D.A. 1998. Protein comparison and classification: A differential geometry approach. *Proc. Nat. Acad. Sci.* 85, 777–781.

Shindyalov, I.N., and Bourne, P.E. 1998. Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. *Protein Eng.* 11, 739–747.

Siew, N., Elofsson, A., Rychlewski, L., and Fischer, D. 2000. MaxSub: An automated measure for the assessment of protein structure prediction quality. *Bioinformatics* 16, 776–785.

Taylor, W.R. 1999. Protein stucture alignment using iterated double dynamic programming. *Protein Sci.* 8, 654–665.

Taylor, W.R., and Orengo, C.A. 1989. Protein structure alignment. *J. Mol. Biol.* 208, 1–22.

Vriend, G., and Sander, C. 1991. Detection of common three-dimensional substructures in proteins. *Proteins Struct. Funct. Genet.* 11, 52–58.

Yee, D., and Dill, K. 1993. Families and the structural relatedness among globular proteins. *Protein Sci.* 2, 884–899.

Yona, G., and Levitt, M. 2000. Towards a complete map of the protein space based on a unified sequence and structure analysis of all known proteins. *Proc. of ISMB 2000*, 395–406.

Yona, G., Linial, N., and Linial, M. 1999. ProtoMap: Automatic classification of protein sequences, a hierarchy of protein families, and local maps of the protein space. *Proteins* 37, 360–378.

Address correspondence to:
Golan Yona
Street Address? **AU3**
Department of Computer Science
Cornell University
Ithaca, NY 14853

E-mail: golan@cs.cornell.edu

Author
Right running head okay as shown (short title)?

Author
Revised art okay as is (per emailed files), especially Figure 4?

Pub
Layout of four-color art okay (Figs. 7, 9, 11, and 12 on recto pages)?

AU1
Provide "Key words" here.

AU2
Please cite reference in text or delete entry.

AU3
Please provide street address.