Approximation of Protein Structure for Fast Similarity Measures

Fabian Schwarzer and Itay Lotan
Computer Science, Stanford University, Stanford, CA 94305
schwarzf@stanford.edu   itayl@stanford.edu

Abstract

It is shown that structural similarity between proteins can be decided well with much less information than what is used in common similarity measures. The full C_α representation contains redundant information because of the inherent chain nature of proteins and a limit on the compactness due to excluded volume. A wavelet analysis on random chains and proteins suggests approximating subchains by their centers of mass. For not too compact chain-like structures in general, and proteins in particular, similarity measures that use this approximation are highly correlated to the exact similarity measures and are therefore useful, e.g. as fast filters. Experimental results with such simplified similarity measures in two applications, nearest neighbor search and automatic structural classification show a significant speed up.

1 Introduction

Automatic protein structure comparison is an important problem in computational structural biology, e.g., in structural databases and classification [10, 24], structure prediction [6, 7, 19, 26, 30, 31], analysis of trajectories through conformational space generated by molecular dynamics and Monte Carlo simulations [22, 29, 36], graph-based methods for evaluating ensemble properties [2, 3, 33], etc.

As opposed to sequence matching, structural matching requires a similarity measure that is based on spatial atom coordinates. Nevertheless, it is important whether the structures that are compared have the same amino acid sequence or not. For conformational samples from the same sequence there are no ambiguities about correspondences. In this case, similarity measures such as $cRMS$ or $dRMS$ are commonly used [18]. These measures are defined as the root mean square (RMS) of either distances between corresponding atoms in the two candidate structures ($cRMS$) or their corresponding intra-molecular distance matrix entries ($dRMS$). Comparing protein structures that derive from different sequences is more difficult because it is generally not obvious which features (atoms) of one structure should be matched with which features from the other structure. Numerous methods for finding a set of correspondences have been proposed [8, 9, 11, 13, 18, 21, 28, 32, 35]. For example, a dynamic programming approach from sequence alignment was used in [9].

Most similarity measures are based on rather fine granular feature selection. Typically, the coordinates of all C_α atom centers are considered, and sometimes even those of additional atoms. For larger proteins, the number of considered features greatly affect the efficiency of the structural comparison. For example, intra-molecular distance matrices grow quadratically with the number of residues. The complexity of the dynamic programming algorithm in [9] is quadratic in the number of features. While this is not a real problem when comparing a single pair of structures (most proteins have less than 1000 residues), it becomes an important issue when querying a database for similar structures, or when clustering a large set of structures.

In this paper, we show that the complexity of similarity measures can be reduced while introducing only a small error. We uniformly sub-divide the backbone into a small number of contiguous subchains and represent each of the subchains by the average coordinates of its atom centers. Similarity measures can then be defined on these "averaged conformations". Although this simplification introduces some error, we provide evidence that enough information about relative similarity is retained to discriminate structures in practical applications. In particular, the derived similarity measures using the averaged protein representation are highly correlated to their original full-atomic counterparts. If high accuracy is a concern, approximate similarity measures are still useful as a fast filter to considerably reduce the number of pairs that need to be passed to the exact similarity measure. While we cannot give bounds on the error that is introduced, we show through wavelet analysis of protein structures and random chains that averaging is a reasonable method for reducing the dimensionality of structure descriptors.

Reducing the computational complexity of similarity measures significantly accelerates many tasks that involve structural matching. In our experiments we observed decreases in running times by large factors, typically from days to hours or even minutes. For very large sets of proteins, both the efficiency of structure comparison of a single pair and the number of such pairs that are actually evaluated are important. Many approaches require evaluation of all pairs ("brute-force approach") even if the task is to identify only a small constant number of nearest neighbors for each conformation in the set. Using our averaged representation we show that we can avoid this quadratic cost of examining all pairs in a k nearest neighbor application. In another application to automatic structural classification, the complexity of a previous algorithm that matches pairs of structures grows quadratically with the number of residues. In this case as well, a small reduction in the number of features results in substantial savings.

The rest of the paper is organized as follows. In Section 2, we describe in detail the proposed averaging that takes advantage of the chain and excluded volume properties of proteins. In Section 3, we demonstrate our approach in a k nearest neighbor application on a large set of different conformations of the same sequence. In Section 4, we show that our approach can serve as a fast prefilter to significantly speed up the STRUCTAL algorithm for classification of structures with different sequences. Finally, Section 5 concludes the paper.

2 Shape similarity and approximation of chains

Given two sequences of points in 3-space $P = (\mathbf{p}_1, \ldots, \mathbf{p}_n)$ and $Q = (\mathbf{q}_1, \ldots, \mathbf{q}_n)$, their coordinate root mean square deviation ($cRMS$) is a common measure of similarity. It is defined as

$$cRMS(P,Q) = min_T \sqrt{\frac{1}{n}\sum_{i=1}^{n} \| \mathbf{p}_i - T\mathbf{q}_i \|^2} \quad (1)$$

where $\| \cdot \|$ is the Euclidean L_2-norm and T is a rigid body transform (rotation and translation). Note that there is a closed form solution for T that yields the optimal alignment [14].

Another common RMS shape similarity measure, $dRMS$, is based on comparing intra-set point distance matrices, i.e. the matrix of distances between all points within each structure. For a point set P, this matrix is defined as

$$\left(d_{ij}^P\right) = \|\mathbf{p}_i - \mathbf{p}_j\| \quad (2)$$

The distance matrix RMS deviation ($dRMS$) of P and Q is then defined as

$$dRMS(P,Q) = \sqrt{\frac{2}{n(n-1)}\sum_{i=2}^{n}\sum_{j=1}^{i-1}(d_{ij}^P - d_{ij}^Q)^2} \quad (3)$$

When applying these similarity measures to proteins, it is common practice to use the C$_\alpha$ atom centers, ordered along the backbone, as defining points. (Sometimes, additional atoms are included or C$_\beta$ atoms are used instead.) The positions of these atoms are usually considered to determine the shape of the backbone sufficiently well. However, in the case of $dRMS$, the intra-molecular distance matrices grow quadratically with the length of the protein, which significantly slows down the $dRMS$ computation for large proteins.

2.1 Approximate similarity measures

We propose to reduce the number n of sample points in P and Q as follows. In each sequence, we replace contiguous subsequences of points by their centroids. That is, we uniformly partition the sequence P of length n into m contiguous subsequences of length $\lfloor n/m \rfloor$ each. (If n/m is not an integer, some subsequences will be chosen to be longer by one.) For each of the subsequences, we then replace its points by their centroid, which we will denote $\overline{\mathbf{p}}_j$ for subsequence j. For example, if subsequence j spans points $(\mathbf{p}_r, \ldots, \mathbf{p}_s)$ then

$$\overline{\mathbf{p}}_j = \frac{1}{s-r+1}\sum_{i=r}^{s}\mathbf{p}_i \quad (4)$$

Based on these averaged subsequences we define the m-averaged representation $\overline{P}_m$ of P as the sequence of m points $(\overline{\mathbf{p}}_1, \ldots, \overline{\mathbf{p}}_m)$. (For Q, we proceed analogously.) We can now define the simplified RMS measures for $\overline{P}_m$ and $\overline{Q}_m$ analogously to the above RMS measures on the original sequences. That is, in the defining formulas (Equations 1, 2 and 3), we replace $\mathbf{p}_i$ ($\mathbf{q}_i$) by $\overline{\mathbf{p}}_i$ ($\overline{\mathbf{q}}_i$) and n by m. We will call these measures m-averaged measures and denote them by $\overline{c}_m RMS$ and $\overline{d}_m RMS$.

Obviously, the error of these simplified similarity measures continuously approaches zero as the two compared point sets P and Q become more similar, i.e. $\lim_{Q \to P} |\overline{c}_m RMS(P,Q) - cRMS(P,Q)| = 0$ and the same holds for $\overline{d}_m RMS$. For general point sets, the error introduced by this approximation can be quite substantial. However, for proteins the error is small because of their chain nature and because van der Waal forces limit the compactness of possible conformations. In the next section we will try to give an intuition as to why this is the case using random chains and wavelets analysis.

2.2 Random chains and Haar wavelets

In what follows we will use random chains and the Haar wavelet transform to argue that averaging is a reasonable method for reducing the size of the representation of a protein for computing structure similarities. A random chain $C = (\mathbf{c}_0, \ldots, \mathbf{c}_{n-1})$ in 3-D is an ordered set of points in space defined as follows:

$$\begin{aligned} \mathbf{c}_0 &= \mathbf{0}, \\ \mathbf{c}_{i+1} &= \mathbf{c}_i + \mathcal{S}_2 \cdot l \quad i = 0, \ldots, n-2 \end{aligned} \tag{5}$$

where $\mathcal{S}_2$ is a random 3-D vector uniformly distributed on the unit 3-D sphere and l is the fixed Euclidean distance between two consecutive points of the chain. $\mathcal{S}_2$ is sampled as follows:

$$\mathcal{S}_2 = \begin{bmatrix} \sin\phi\cos\theta \\ \sin\phi\sin\theta \\ \cos\phi \end{bmatrix} \tag{6}$$

where $\theta \sim U[0, 2\pi]$ and $\cos\phi \sim U[-1, 1]$. Computing the covariance matrix of $\mathcal{S}_2$ reveals that the off-diagonal elements are identically 0, and as a result the three dimensions of each random step are uncorrelated. Since each step is independent of all other steps the distributions of the three dimensions of any point on the chain are uncorrelated. We have empirically observed that the average chain behavior can be approximated well by replacing $\mathcal{S}_2$ with a 3-vector sampled from the normal distribution $\mathcal{N}(\mathbf{0}, \frac{1}{3} \cdot \mathcal{I})$ (where $\mathcal{I}$ is the 3×3 identity matrix) when $n > 10$.

The Haar wavelet transform of a chain is a recursive averaging and differencing of the coordinates of the points. The transform recursively smoothes the chain while keeping the *detail* coefficients needed to reconstruct the full chain from the smoothed out version. We define the full resolution chain to be of level 0: $C = C^0$. We recursively create smoothed versions of the chain by averaging pairs of consecutive points:

$$\mathbf{c}_i^j = \frac{1}{\sqrt{2}} \left(\mathbf{c}_{2i}^{j-1} + \mathbf{c}_{2i+1}^{j-1} \right) \quad \left\{ \begin{array}{l} j = 1, \ldots, \log n \\ i = 0, \ldots, \frac{n}{2^j} - 1 \end{array} \right. . \tag{7}$$

As each level of resolution is created we also compute the *details* that are smoothed out by the averaging:

$$\mathbf{d}_i^j = \frac{1}{\sqrt{2}} \left(\mathbf{c}_{2i}^{j-1} - \mathbf{c}_{2i+1}^{j-1} \right) \quad \left\{ \begin{array}{l} j = 1, \ldots, \log n \\ i = 0, \ldots, \frac{n}{2^j} - 1 \end{array} \right. . \tag{8}$$

Note that the averages and details are multiplied by a scale factor of $\sqrt{2}$ at each level. Given C^j, the smoothed chain at level j, and D^j, the detail coefficients of level j, it is possible to reconstruct exactly C^{j-1} by inverting the formulas of Equations 7 and 8. The Haar wavelet transform of a chain C is thus defined as:

$$\hat{C} = \left(C^{\log n}, D^{\log n}, D^{\log n - 1}, \ldots, D^1 \right). \tag{9}$$

The length of $\hat{C}$ is the same as C and $C^{\log n}$ is the centroid of the entire chain. Since C can be exactly reconstructed from $\hat{C}$, no information is lost during the transform. This representation can then be compressed by removing (setting to 0) all coefficients in $\hat{C}$ smaller than some threshold. Since the coefficients are scaled, the square of the L_2 error of approximation in this case would be equal to the sum of the squares of the removed coefficients [34].

Given the normal approximation of the random chain construction, we can analytically determine the pdf of each of the coefficients in $\hat{C}$ by adding, subtracting and scaling independent normally distributed variables. The pdf of each detail coefficient in level j can be derived as:

$$\mathbf{d}^j \sim \mathcal{N}\left(\mathbf{0}, \frac{4^j + 2}{36} \cdot \mathcal{I} \cdot l \right), \tag{10}$$

and the pdf of their squared L_2 norm is thus:

$$\|\mathbf{d}^j\|_2^2 \sim \chi^2(3dof) \cdot \frac{4^j + 2}{36} \cdot l \tag{11}$$

with a mean of $\frac{4^j+2}{12} \cdot l$ and variance of $\frac{(4^j+2)^2}{216} \cdot l^2$. Since the pdfs of the detail coefficients are centered at the origin and their variance is decreasing by a factor of roughly 4, they are expected to be ordered (in absolute value) in $\hat{C}$, from largest to smallest. Note that the $\sqrt{2}$ scaling during the construction of the coefficients would account for an average growth of a factor of 2 in their variance from one level to the next. The special structure of random chains accounts for the second factor of 2. Hence as a general policy it is best to remove coefficients starting at the lowest level and climbing up. These coefficients have the lowest variance and thus contain the least information for determining structural similarity. The effect of averaging as described in Section 2.1 for $m = 2^v$ is the same as that of removing the lowest $\log n - v$ levels of coefficients. Since these are expected to be the smallest coefficients, we can conclude that using an m-averaged chain will give the smallest expected error for a representation that uses only m Haar detail coefficients for each dimension. The wavelet analysis allows us to estimate the approximation mean squared (MS) error introduced by removing all coefficients of level j. It can be computed to be $\frac{l}{12}\left(2^i + \frac{1}{2^{i-1}}\right)$. Therefore for an m-averaged approximation the MS error is expected to be on the order of $\frac{n \cdot l}{6m}$.

This behavior of the Haar detail coefficients is a result of the fact that we are dealing with chains that on the average grow further and further away from their starting point. The expected distance of the nth point from the origin is on the order of $\sqrt{n} \cdot l$. In Figure 1 we compare the variance of the coefficients of random chains to those of very compact random chains (chains forced to reside inside small spheres) and to those of point clouds sampled randomly from inside a sphere of radius $\sqrt{n} \cdot l$. All chains are of length $n = 64$.

While it is a well-known fact that the positions of neighboring residues in native protein structures are highly correlated (e.g., see [20]), it was shown in [16] that treating

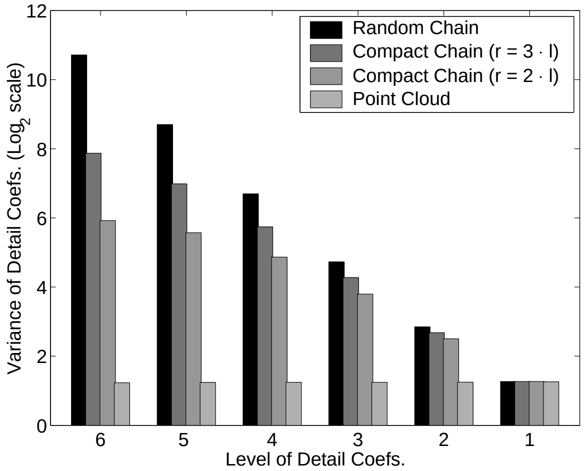

Figure 1: Comparison of the Haar coefficients of random chains, compact random chains and random point clouds.

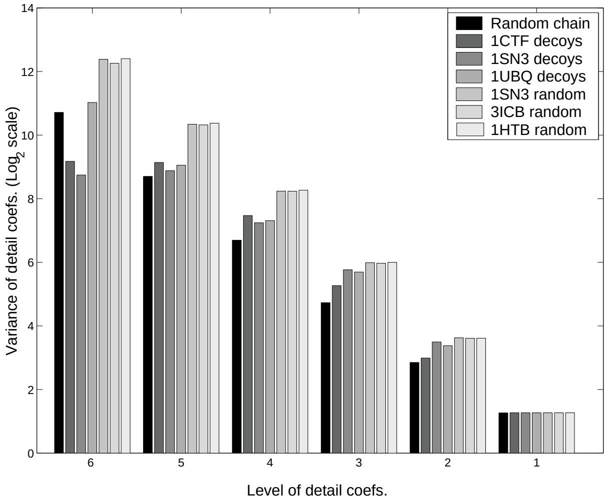

Figure 2: Comparison of the Haar coefficients of decoy sets of protein and randomly generated conformations of proteins to the coefficients of a set of random chains.

the position of each residue as uniformly distributed on a 3-D sphere yields a very good approximation of the average behavior of native protein structures. We performed the same wavelet transform on sets of conformations of actual proteins of length 64 residues (only the C_α atoms) taken from *decoy sets* (conformations which are expected to be similar to the native conformation). See Section 2.3 for more details on these sets. We obtained results similar to those of random chains, namely the detail coefficients are ordered and have a large growth rate. The results for a few sets of proteins are presented in Figure 2. One important difference from random chains is that for decoy sets the growth in the variance of the detail coefficients from one level to the next decreases considerably for the top levels. We explain this as follows. Small pieces of a protein cannot be highly packed because of steric clashes, while intermediate size pieces are often elongated (secondary structure helices and strands) and hence the variance of the coefficients grows considerably from one level to the next at the low and intermediate levels. The tight packing of the secondary structure of the native-like conformations makes the high level coefficients considerably smaller than in the random chain model. We would have liked to give results for decoy sets of proteins longer than 64 residues, however currently no such sets exist and generating them requires a significant effort, and expertise which we do not posses.

Random protein conformations (generated as described in Section 2.3), on the other hand, are considerably less compact than the decoy sets, and hence behave much more like random chains at *all* levels of detail coefficients. As can be seen in Figure 2 the growth in variance (from one level to the next) at the lower levels is even bigger than what is observed for random chains. This is a result of the space taken up by each residue. In our random chain model the points had no volume and the chain may not be self-avoiding. The random protein conformations, however, must respect the ”volume“ of the atoms and are not allowed to have any self-overlaps. Their behavior is actually modelled better by random chains in which the next step is sampled uniformly from a *hemisphere* defined by the direction of the previous step.

We thus can conclude, that while decoy sets cannot be compressed as much as random sets, it is possible to remove the first few levels and still get a very good approximation.

2.3 Correlation of approximate and exact similarity measures

When using the $cRMS$ measure to compute similarity between random chains we find that the approximate m-averaged versions yield very good results for m as small as 8. For $m = 4, 8, 12$ and 20 Pearson’s correlation of the approximate measure to the true one is 0.59, 0.92, 0.97 and 0.99 respectively. When using the $dRMS$ measure to compute similarity between random chains the approximate m-averaged versions is highly correlated for m as small as 12. The correlation values obtained are 0.45, 0.78, 0.88 and 0.94 respectively.

In order to verify that the analogous behavior of the detail coefficients of protein sets and random chains carries over to approximate similarity measures we chose 9 structurally diverse proteins used by Park and Levitt in [26] (1CTF, 1ERP, 1R69, 1SN3, 1UBQ, 2CRO, 3ICB, 4PTI, 4RXN) with between 38 and 76 residues. For these proteins we obtained (1) decoy sets (conformations highly similar to the native conformation) generated by Park and Levitt, containing 10,000 conformations each and (2) randomly generated conformation sets using the program FOLDTRAJ[1] [7] containing 5000 structures each.

For each set, we randomly chose between 1000 and 4000 pairs whose true $dRMS$ distance was less than 5Å and computed their m-averaged distances for different values of m. The correlation of the m-averaged $cRMS$ and $dRMS$ measures to the true $cRMS$ and $dRMS$ measures for the different decoy sets can be found in Table 1(a). As can be seen from the table, for $m = 8$ the approximate $cRMS$ measure is already highly correlated with the true $cRMS$ measure, which means a reduction factor of between 5 and 8 still yields a very good approximation. For $dRMS$, high correlation is achieved for $m = 12$, which means a reduction factor of between 3 and 6 (since the complexity of $dRMS$ is quadratic, the actual gain is by a factor of between 9 and 36). We note that the correlation values obtained are quite similar to those computed for random chains.

The correlation of the m-averaged $cRMS$ and $dRMS$ measures to the true measures for the different random sets can be found in Table 1(b). Here too, a reduction factor of between 5 and 8 yields a highly correlated approximation of the $cRMS$ measure. For $dRMS$ high correlation is achieved for $m = 8$, which means a reduction factor of between 5 and 8 (an actual gain between 25 and 64). Here the correlation values are in fact better than those computed for random chains as would be anticipated from the higher growth ratio of the variance of the detail coefficients of random protein sets in comparison to those of random chains (see Figure 2). We note that when examining all pairs (not only those whose $dRMS$ distance is smaller than 5Å , the Pearson's correlation found is even stronger.

3 Application 1: nearest neighbor search

Simulations and other conformational sampling methods generate large sets of conformations of a particular protein. For example, the project **Folding@Home**[2] runs parallel molecular dynamics simulations on several thousands of computers across the Internet and then centrally evaluates the obtained data. An important step in evaluating such data, e.g. for clustering and related tasks, is the following: given a set of conformations of the same protein, find the k nearest neighbors (NNs) for each sample in the set. Typically, k is a small constant while the size of the set can be very large.

The straight forward ("brute-force") approach is to evaluate the similarity measure ($cRMS$ or $dRMS$) for all pairs and then report the k NNs for each sample. However, the quadratic complexity makes this approach scale badly. Spatial data structures such as the ***kd-tree*** [4] can avoid this complexity under certain circumstances [1, 5, 17, 15, 23, 27]. Note that these data structures allow for exact search, i.e., they return the same NNs as would the brute-force search. However, most of them require a Euclidean metric space of rather limited dimensionality. Unfortunately, $cRMS$ is not a Euclidean metric. Although $dRMS$ is a Euclidean metric, the dimensionality of the space of intra-molecular distance matrices is far too high. (Typically, for dimensions higher than a few tens, none of the nearest neighbor data structures performs better than brute-force search.) Therefore if we hope to use a spatial data structure to speed up NNs search, we must use the $dRMS$ measure but find a way to significantly reduce the dimensionality of the structure descriptors beyond the averaged conformations we presented in Section 2.

3.1 Further reduction of distance matrices

We use the singular value decomposition (SVD) [12] to further compress the intra-molecular distance matrices of averaged proteins, that is to further reduce the number of parameters involved in computing $\overline{d}_m RMS$. SVD is a standard tool for principal components analysis (PCA) and computes directions of greatest variance (and thus distance information) in a given set of high-dimensional points. These directions are called principal components (PCs). The SVD can be used to linearly map a set of high-dimensional input vectors (data points), stored in a matrix A, into a lower-dimensional subspace while preserving most of the variance. Such a transform can be found by decomposing the matrix A of the input vectors into $A = USV^T$, the SVD of A, where U and V are orthogonal matrices and S is diagonal with the singular values of A along the diagonal.

Efficient algorithms exist that compute the SVD in time $O(s^2t)$ where s is the smaller and t the larger dimension of A (rows or columns). Note that while in principle, SVD could be applied to intra-molecular distance matrices without first averaging protein pieces, the quadratic dependency on the smaller dimension of A shows the important advantage of averaging: usually, the larger dimension t will reflect the size of the conformational sample set while the smaller dimension s will correspond to the size of a single intra-molecular distance matrix. Reducing the distance matrix size by using averaged conformations, as described in Section 2, is therefore key to performing the SVD in practice.

[1] `http://bioinfo.mshri.on.ca/trades/`

[2] `http://folding.stanford.edu`

	1CTF		1ERP		1R69		1SN3		1UBQ		2CRO		3ICB		4PTI		4RXN	
m	$cRMS$	$dRMS$	$cRMS$	$dRMS$	$cRMS$	$dRMS$	$cRMS$	$dRMS$	$cRMS$	$dRMS$	$cRMS$	$dRMS$	$cRMS$	$dRMS$	$cRMS$	$dRMS$	$cRMS$	$dRMS$
4	0.38	0.42	0.88	0.86	0.64	0.47	0.54	0.45	0.37	0.47	0.73	0.52	0.57	0.40	0.42	0.46	0.49	0.49
8	0.90	0.77	0.97	0.94	0.97	0.87	0.94	0.78	0.84	0.70	0.97	0.89	0.98	0.86	0.95	0.83	0.93	0.79
12	0.98	0.93	0.98	0.96	0.99	0.92	0.98	0.94	0.98	0.92	0.99	0.95	0.99	0.92	0.98	0.92	0.98	0.94
16	0.99	0.95	0.99	0.98	0.98	0.92	0.99	0.97	0.98	0.95	0.99	0.97	0.98	0.92	0.98	0.94	0.98	0.97
20	0.99	0.96	0.98	0.96	0.99	0.96	0.99	0.97	0.99	0.96	0.99	0.97	0.98	0.93	0.99	0.95	0.98	0.95

(a)

	1CTF		1ERP		1R69		1SN3		1UBQ		2CRO		3ICB		4PTI		4RXN	
m	$cRMS$	$dRMS$	$cRMS$	$dRMS$	$cRMS$	$dRMS$	$cRMS$	$dRMS$	$cRMS$	$dRMS$	$cRMS$	$dRMS$	$cRMS$	$dRMS$	$cRMS$	$dRMS$	$cRMS$	$dRMS$
4	0.69	0.59	0.85	0.92	0.68	0.65	0.68	0.63	0.65	0.53	0.70	0.64	0.65	0.55	0.72	0.71	0.73	0.74
8	0.96	0.90	0.99	0.99	0.97	0.93	0.97	0.92	0.96	0.84	0.97	0.91	0.96	0.86	0.97	0.95	0.97	0.96
12	0.99	0.97	0.99	0.98	0.99	0.97	0.99	0.97	0.99	0.95	0.99	0.97	0.99	0.95	0.99	0.98	0.99	0.98
16	0.99	0.98	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.97	0.99	0.99	0.99	0.97	0.99	0.98	0.99	0.98
20	0.99	0.98	0.97	0.96	0.99	0.98	0.99	0.98	0.99	0.97	0.99	0.98	0.99	0.97	0.99	0.98	0.99	0.97

(b)

Table 1: Pearson's correlation coefficient for different m values evaluated for $cRMS$ and $dRMS$ of (a) decoy sets and (b) randomly sampled conformations of various proteins.

	$k = 10$		$k = 25$		$k = 100$	
	E_1	E_2	E_1	E_2	E_1	E_2
1CTF	1.1	1.04	1.116	1.036	1.104	1.023
1ERP	1.3	1.158	1.292	1.124	1.246	1.083
1R69	1.201	1.093	1.192	1.074	1.165	1.048
1SN3	1.161	1.086	1.155	1.063	1.136	1.04
1UBQ	1.199	1.105	1.184	1.08	1.162	1.045
2CRO	1.118	1.061	1.155	1.056	1.141	1.041
3ICB	1.152	1.054	1.154	1.048	1.129	1.033
4PTI	1.186	1.099	1.188	1.079	1.17	1.055
4RXN	1.179	1.079	1.186	1.072	1.163	1.046

Table 2: Mean errors $E_i = E(err_i)$ for 100 queries of k nearest neighbors. Park-Levitt decoys, 20 PCs.

	$k = 10$		$k = 25$		$k = 100$	
	E_1	E_2	E_1	E_2	E_1	E_2
decoys	1.337	1.248	1.34	1.196	1.347	1.141
decoys2	1.136	1.052	1.171	1.049	1.158	1.036
uniform	1.101	1.035	1.107	1.029	1.113	1.02

Table 3: 1CTF: Mean errors $E_i = E(err_i)$ for 100 queries of k nearest neighbors. Decoys, decoys2: 100,000 decoys from Park-Levitt set ($m = 16$ and 20 PCs). Decoys2: excluding 8 outliers. Uniform: 100,000 uniformly sampled conformations ($m = 16$ and 16 PCs).

To perform a SVD on a set of intra-molecular distance matrices derived from m-averaged conformations, each of these distance matrices is rewritten as an $m(m-1)/2$ dimensional vector and then the SVD is applied to the matrix that contains all these vectors. Taking the resulting U matrix and removing all columns that correspond to small singular values (the directions of little variance), we have the linear map that takes the set of distance matrices into a lower-dimensional Euclidean space while preserving a high percentage of the variance and thus distance information. We found that in practice, a relatively small output dimensionality between 10 and 20 is sufficient to maintain about 90% of the variance of the distance matrices.

In the following, we will denote the $dRMS$ measure obtained from an SVD compressed set of m-averaged distance matrices by $\overline{d}_m^{PC} RMS$. (PC stands for the number of principal components that are used after compression.)

3.2 Evaluation of approximation errors

Of course, by reducing the dimensionality of distance matrix space from $dRMS$ over $\overline{d}_m RMS$ to finally $\overline{d}_m^{PC} RMS$, we lose some information. However, we observed that in general, the introduced errors are small enough to allow for good results in finding most of the true k nearest neighbors. In what follows, we illustrate this with different error measures.

For a set of conformations S and a given query conformation Q from that set, we define two subsets S_1 and S_2 of size k each. S_1 is the set of k nearest neighbors of Q in S using exact $dRMS$. S_2 is the set of k nearest neighbors of Q in S using $\overline{d}_m^{PC} RMS$. Thus, S_2 is the approximation of S_1 using our reduced similarity measure.

In a first experiment, we evaluated how many of the exact k nearest neighbor conformations in S_1 would be contained in S_2 as well. To this end, we looked at the $k = 100$ nearest neighbors for each of 100 query conformations and computed the average number of such matches over the 100 queries. Using $\overline{d}_{16}^{20} RMS$ on sets S of 10,000 decoys of 1R69, 4PTI, 2CRO, 1SN3 and 3ICB, these averages ranged from 74.8 to 83.3 (roughly 80 on average). Using $\overline{d}_{16}^{18} RMS$ on sets S of 5,000 uniform samples of the same proteins, corresponding averages were between 86.1 and 94.2 (about 90 on average). These results show that, using our approximations, not too many extra samples would have to be drawn in order to obtain sufficiently many exact nearest neighbors.

However, in cases in which significantly more than k conformations are clustered around the query conformation, the number of identical matches may not be a good quality measure. In such cases, it may even be possible to obtain a very different candidate set with almost the same distribution of distances from the query conformation. We therefore used the two following error measures to further evaluate the quality of our approximations.

$\mathbf{err_1}$ The ratio of the exact $dRMS$ of the furthest conformation in S_2 to the exact $dRMS$ of the furthest conformation in S_1.

$\mathbf{err_2}$ The ratio of the average exact $dRMS$ of all conformations in S_2 to the average exact $dRMS$ of all conformations in S_1.

We first looked at the decoy sets of size 10,000 for each of the nine Park-Levitt proteins. We used $m = 16$ and 20 PCs. For each set, we randomly chose 100 query conformations and evaluated err_1 and err_2 for each of them. Table 2 shows the means $E_i = E(err_i)$ of both errors over the 100 queries (standard deviations were generally small). The mean error E_1 is usually noticeably smaller than 1.2 which means that the worst-case error by our reduction is smaller than 20%. The mean error E_2 is almost always noticeably smaller than 1.1 indicating an average error of less than 10%. Note that these percentages correspond to small absolute values of about 1.5Å and 0.7Å, respectively.

We also evaluated the error measures for two large sets of 100,000 conformations of 1CTF, a decoy set and a uniformly sampled set. Here, we used $m = 16$ and 20 PCs for the decoys and $m = 16$ and 16 PCs for the uniform samples. The results are presented in Table 3. For the decoys we noticed both errors to be slightly higher than for the smaller data set above. However, when we ignore 8 out of the 100 query conformations, which were obvious outliers, we got comparable results (row labelled $decoys2$). For the uniform samples, both relative errors were significantly smaller than for the decoy set.

3.3 Running time

We now consider the running times in a concrete nearest neighbor task: given a set of 100,000 random conformations of protein 1CTF, find $k = 100$ nearest neighbors for each sample in the set. The reported times in this section refer to a sequential implementation in C running on a single 1GHz Pentium processor on a standard desktop PC.

We first compare the running times for a brute-force (all-pairs) implementation using both $cRMS$ and $dRMS$, and their corresponding averaged similarity measures $\overline{c}_m RMS$ and $\overline{d}_m RMS$. Table 4 shows that the latter measures already result in a notable speed-up. For $\overline{d}_{16}RMS$, a significant speed-up over $dRMS$ is obtained due to the quadratic down-scaling of intra-molecular distance matrices by averaging proteins. For $\overline{c}_{16}RMS$, the improvement over $cRMS$ is smaller. This is because the reduction by averaging affects the number of involved points only linearly and the main effort in computing $cRMS$ comes from finding an optimal rigid body alignment of two point sets.

N	$cRMS$	$\overline{c}_{16}RMS$	$dRMS$	$\overline{d}_{16}RMS$
1,000	18.6s	12.4s	31.0s	2.2s
2,000	74.4s	50.0s	137.5s	8.0s
5,000	464.8s	312.0s	759.8s	43.4s
100,000	∼52h	∼35h	∼84h	∼4.8h

Table 4: Brute-force search using $cRMS$ vs $\overline{c}_{16}RMS$ and $dRMS$ vs $\overline{d}_{16}RMS$ for finding the $k = 100$ nearest neighbors for each of N samples.

Note that the increase in running times agrees quite well with the expected quadratic scaling of the brute-force nearest neighbor approach. The times for $N = 100,000$ samples were therefore extrapolated from the actual running times measured for the smaller values of N. In fact, for $dRMS$ using all C$_\alpha$ atom coordinates, we had problems storing all intra-molecular distance matrices (however, these problems do not occur with averaged proteins and $\overline{d}_m RMS$).

We next address the quadratic scaling problem of the brute-force approach. To be able to apply a kd-tree, we first further reduced the 120-dimensional space of $\overline{d}_{16}RMS$ using SVD and retained 16 principal components. This further compression took about one minute for the complete set of 100,000 samples. Building the kd-tree for the resulting 16-dimensional data took only about 4 seconds. The correlation coefficient of the resulting $\overline{d}^{16}_{16}RMS$ and $dRMS$ was found to be still about 0.94. As reported in the previous section, the approximation errors are also low (cf. Table 3).

We then ran both the brute-force and the kd-tree approach using $\overline{d}^{16}_{16}RMS$ as similarity measure. Table 5 shows the running times for finding $k = 1$ and $k = 100$ nearest neighbors for each sample in the full set of 100,000 samples. The obtained total speed-up of our nearest neighbor search (approximate similarity measures and a kd-tree) over the current best approach (brute-force search using all C_α coordinates) is several orders of magnitude.

In general, the speed-up obtained by using a kd-tree can be expected to increase with increasing sample set size N. Due to its quadratic scaling, brute-force search will become very slow for larger sample sets. On the other hand, the sub-quadratically scaling kd-tree approach should allow to process even much larger sample sets within a few hours without parallelization on a standard desktop PC.

k	Brute-force	kd-tree
1	30min	4min10sec
100	41min	19min

Table 5: Brute-force vs kd-tree search for k nearest neighbors for each of 100,000 samples. Used similarity measure: $\bar{d}_{16}^{16}RMS$.

4 Application 2: structural classification

Given a set of native protein structures each having a different amino-acid sequence, such as the Protein Data Bank (PDB)[3] we would like to automatically classify the structures into groups according to their structural similarity. This task has been performed manually in the SCOP (structural classification of proteins) database[4] [24], where protein structures are hierarchically classified into *classes*, *folds*, *superfamilies* and *families*. The major difficulty in performing this classification automatically lies in the need to decide, given two protein structures, which parts of both structures should be compared, before it can be determined how similar these parts are. This *correspondence problem* does not arise when different conformations of the same protein are compared because in that case the correspondence is trivially determined. For this reason computing the similarity between structures of different proteins requires considerably more computation than the methods described in Section 2.

There are a variety of algorithms that have been created for structural classification. The DALI[5] method [13] starts with the distance matrices of both proteins. It finds all pairs of similar sub-matrices of small fixed size (one from each protein distance matrix) and then uses a Monte Carlo algorithm to assemble the pairs into larger consistent alignments. The PROSUP[6] method [21] initially identifies similar fragments in both proteins and iteratively expands them to create alignments. A dynamic algorithm is then used to iteratively refine each alignment and finally insignificant alignments are removed. The STRUCTAL[7] method [9] directly matches the backbones of the two protein structures by iteratively cycling between a dynamic programming algorithm and least-square fitting to come up with an alignment that minimizes coordinate difference. For other methods see [18].

4.1 The modified STRUCTAL algorithm

All the above methods stand to gain in performance by using our averaging scheme. In order to verify this we tested the speed up and accuracy obtained by using the STRUCTAL method on averaged protein structures. We could not test our approach on PROSUP and DALI because both servers did not accept our averaged structures (for file formatting reasons), and the algorithms were too involved for us to implement reliably.

The STRUCTAL algorithm starts with an initial alignment of the backbone C_α atoms of the two structures according to one of a number of possible heuristics (aligning the beginnings, the ends, random segments, by sequence similarity, etc.) Then a two step process is repeated until convergence. First a dynamic programming algorithm based on the Needleman and Wunsch sequence alignment algorithm[25] finds the correspondence between the two structures that yields the highest score. Scoring is based on assigning a cost to each possible corresponding pair, which is inversely proportional to the distance between C_α positions, and a gap penalty for every gap in the alignment. Computing the best correspondence thus requires $O(n_1 n_2^2 + n_2 n_1^2)$ time (n_1 and n_2 are the number of residues in each structure). Second, an optimal alignment is computed based on the best correspondence using the method in [14]. The $cRMS$ distance of the final alignment and the number of corresponding residues is returned as a measure of the similarity of the two structures. Since the result is sensitive to the initial alignment, the algorithm is usually run a number of times for each pair of structures, each time using a different initial alignment. The best (smallest) result of all the runs is kept.

By using averaging with this algorithm we add another degree of freedom to the computation. If we average each r C_α atom positions to create the averaged structure, we have a choice of $a = mod(n_1, r)$ starting positions to start the averaging process for one structure and $b = mod(n_2, r)$ for the other. As a result there are $a \times b$ possible pairs of averaged structures that could possibly result in different alignment scores. Therefore more runs per pair of structures would be necessary when using our averaging scheme. However, the gain from averaging stands to be very large. Since both n_1 and n_2 are reduced by a factor of r, the complexity of the dynamic programming, which is the main part of the algorithm, is reduced by a factor of r^3.

4.2 Experimental results

We implemented the STRUCTAL algorithm as it is described in [9]. Our implementation initially aligns the two structures by choosing a small random correspondence. Therefore, we ran the algorithm 12 times for every pair

[3] http://www.rcsb.org/pdb/

[4] http://scop.mrc-lmb.cam.ac.uk/scop/

[5] http://www2.ebi.ac.uk/dali

[6] http://lore.came.sbg.ac.at/CAME /CAME_EXTERN/PROSUP

[7] http://bioinfo.mbb.yale.edu/align

of structures and kept the smallest result as the similarity score. In [9] they run the algorithm a few times as well, each time using a different initial alignment heuristic. We selected 103 protein structures of various sizes (between 100 and 600 residues) taken from 5 different fold families in SCOP [24]. We compared all pairs of structures using the full C_α representation and using averages of $r = 5$ (36 runs per pair of structures) and $r = 8$ (60 runs per pair of structures).

Altogether 5253 pairs of structures were evaluated. The correlation of the similarity measure of the averaged representation to the that of the full representation was 0.81 for $r = 5$ and 0.80 for $r = 8$. When only examining pairs where the number of corresponding residues was above 100 the correlation was 0.86 and 0.82 respectively. For correspondence larger than 200 the correlation was 0.93 and 0.85 respectively. We wanted to evaluate how useful the averaging scheme is as a pre-filter for classification. When trying to pick out the 10 most similar structures to each of the structures in the test set, we find on average 6.3 of them when picking the top 10 using the averaged representation with $r = 5$, although 25% of the time we get less than 5. We find 8.3 on average when picking the top 20 using the averaged representation, getting less than 5 only 4% of the time. When trying to pick out the 20 most similar structures to each of the structures in the test set, we find on average 13.8 of them when picking the top 20 using the averaged representation with $r = 5$, although 12% of the time we get less than 10. We find 16.7 on average when picking the top 30 using the averaged representation, getting less than 10 only 3% of the time. The results for averaging with $r = 8$ were only slightly worse. However, when computing similarity between large structures where the correspondence is larger than 200 residues (35 in our set of structures), we find that with the 10 most similar structures using the $r = 5$ averaged representation we find 8.7 of the top 10 structures, only 4 times finding less than 8. Using the top 15 computed with the averaged representation we find 9.9 on average out of the top 10, only once finding as little as 8.

As far as the savings in time gained by averaging, it took 233 hours to compute the similarity measure (using all C_α atoms) between all pairs of structures in our data set. Using $r = 5$ averaging it took less than 4 hours and using $r = 8$ averaging it took one hour and 10 minutes. Clearly the speedup is quite significant. As the results above demonstrate, using the averaged representation as a pre-filter for finding structurally similar proteins is a very fast yet not very precise tool. If one is looking for k similar structures, it is best to first find about $3k$ structures using the averaged representation and then use the exact methods to find the top k. This approach produces the correct set of k most of the time.

5 Conclusion

Two general properties of proteins, their chain nature and limited compactness, can be exploitet to uniformly reduce the number of features for structural similarity computations. Substantial savings in terms of storage and running time are attained with small errors.

In applications in which no approximation error is tolerable, our approach can be used as a first step to filter a small subset of pairs that are within some tolerance band around the desired similarity. More expensive exact similarity measures can then be used on the reduced set of pairs.

Two possible applications were presented: finding k nearest neighbors in large sets of conformations of the same protein and classification of different proteins using the STRUCTAL algorithm. In these examples, the running times were reduced from days to hours or even minutes.

In general, applications that input very large sets of proteins or that employ computationally intensive algorithms on large proteins stand to benefit from approximation of structures as suggested in this paper.

Acknowledgements: The authors would like to thank Patrice Koehl, Pankaj Agarwal and Jean-Claude Latombe for comments and Patrice Koehl for providing the full Park-Levitt decoy set for the nearest neighbor experiments. This work was partially funded by NSF_ITR grant `DUKE UNI/01-SC-NSF-1009-01` and a grant from Stanford's Bio-X program.

References

[1] P. K. Agarwal. Range searching. In Jacob E. Goodman and Joseph O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, pages 575–598. CRC Press, 1997.

[2] M. S. Apaydin, D. L. Brutlag, C. Guestrin, D. Hsu, and J.-C. Latombe. Stochastic roadmap simulation: An efficient representation and algorithm for analyzing molecular motion. In *RECOMB*, 2002. (to appear).

[3] M. S. Apaydin, A. P. Singh, D. L. Brutlag, and J.-C. Latombe. Capturing molecular energy landscapes with probabilistic conformational roadmaps. In *Proc. IEEE Intl. Conf. on Rob. and Autom.*, pages 932–939, 2001.

[4] J. Bentley. multidimensional binary search trees used for associative searching. *commun. ACM*, 18:509 – 517, 1975.

[5] K. L. Clarkson. Nearest neighbor queries in metric spaces. In *Proc. 29th Annnu. ACM Sympos. Theory Comput.*, pages 609–617, 1997.

[6] Borin Fain and Michael Levitt. A novel method for sampling alpha-helical protein backbones. *J. Mol. Biol.*, 305:191–201, 2001.

[7] H. J. Feldman and C. W. Hogue. A fast method to sample real protein conformational space. *Proteins*, 39(2):112–131, 2000.

[8] Z. K. Feng and M. J. Sippl. Optimum superimposition of protein structures: ambiguities and implications. *Fold. Des.*, 1:123–132, 1996.

[9] M. Gerstein and M. Levitt. Comprehensive assessment of automatic structural alignment against a manual standard, the scop classification of proteins. *Protein Science*, 7:445–456, 1998.

[10] J. F. Gibrat, T. Madej, and S. H. Bryant. Surprising similarities in structure comparison. *Curr. Opin. Struct. Biol.*, 6(3):377–385, 1996.

[11] A. Godzik. The structural alignment between two proteins: is there a unique answer? *Protein Sci.*, 5:1325–1338, 1996.

[12] G. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996.

[13] L. Holm and C. Sander. Protein structure comparison by alignment of distance matrices. *J. Mol. Biol*, 233(1):123 – 128, 1993.

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

[15] Poitr Indyk. *High-dimensional Computational Geometry*. PhD thesis, Stanford University, September 2000.

[16] K. Kedem, L.P. Chew, and R. Elber. Unit-vector RMS (URMS) as a tool to analyze molecular dynamics trajectories. *Proteins: Structure, Function and Genetics*, 37:554 – 564, 1999.

[17] J. Kleinberg. Two algorithms for nearest-neighbor search in high dimension. In *Proc. 29th Annu. ACM Sympos. Theory Comput.*, pages 599–608, 1997.

[18] Patrice Koehl. Protein structure similarity. *Current Opinion in Structural Biology*, 11:348–353, 2001.

[19] Patrice Koehl and Marc Delarue. Building protein lattice models using self-consistent mean field theory. *Journal of Chemical Physics*, 108(22):9540–9549, 1998.

[20] R. Kolodny, P. Koehl, L. Guibas, and M. Levitt. Small libraries of protein fragments model native protein structures accurately. *J. of Mol. Biol.*, 323(2):297 – 307, Oct 2002.

[21] P. Lackner, W. Koppensteiner, M. Sippl, and F. Domingues. ProSup: a refined tool for protein structure alignment. *Protein Engineering*, 13(11):745 – 752, 2000.

[22] S. M. Larson, C. D. Snow, M. Shirts, and V. S. Pande. Folding@Home and Genome@Home: Using distributed computing to tackle previously intractable problems in computational biology. Computational Genomics (to appear).

[23] S. Maneewongvatana and D. M. Mount. The analysis of a probabilistic approach to nearest neighbor searching. In *Proc. 7th Workshop on Algorithms and Data Structures (WADS 2001)*, pages 276–286, 2001.

[24] A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.*, 247(536-540), 1995.

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

[26] B. Park and M. Levitt. Energy functions that discriminate X-ray and near-native folds from well-constructed decoys. *J. Mol. Biol.*, 258:367–392, 1996.

[27] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, New York, NY, 1985.

[28] I. Shindyalov and P. Bourne. protein structure alignment by incremental combinatorial extension (CE) of the optimal path. *Protein engineering*, 11(9):739 – 747, 1998.

[29] M. R. Shirts and V. S. Pande. Mathematical analysis of coupled parallel simulations. *Physical Review Letters*, 86(22):4983–4987, 2001.

[30] David Shortle, Kim T. Simons, and David Baker. Clustering of low-energy conformations near the native structures of small proteins. *Biophysics*, 95:11158–11162, 1998.

[31] K. T. Simons, R. Bonneau, I. Ruczinski, and D. Baker. Ab initio protein structure prediction of CASP III targets using ROSETTA. *Proteins*, 37(3):171–176, 1999.

[32] A. P. Singh and D. L. Brutlag. Protein structure alignment: A comparison of methods, 2000. (http://cmgm.stanford.edu/˜brutlag/Abstracts/... singh00.html).

[33] Guang Song and Nancy M. Amato. Using motion planning to study protein folding pathways. In *RECOMB*, pages 287–296, 2001.

[34] Eric J. Stollnitz, Tony D. DeRose, and David H. Salesin. Wavelets for computer graphics: A primer (part 1). *IEEE Computer Graphics and Applications*, 15(3):76–84, 1995.

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

[36] B. Zagrovic, E. J. Sorin, and V. Pande. β-hairpin folding simulations in atomistic detail using an implicit solvent model. *J. Mol. Biol.*, 313:151–169, 2001.