BIOINFORMATICS ORIGINAL PAPER

Vol. 26 no. 9 2010, pages 1176–1184
doi:10.1093/bioinformatics/btq127

Structural bioinformatics

Advance Access publication April 5, 2010

Alignment-free local structural search by writhe decomposition

Degui Zhi[1,*,†], Maxim Shatsky[1,2] and Steven E. Brenner[1,2]

[1]Department of Plant and Microbial Biology, UC Berkeley and [2]Physical Biosciences Division, LBNL, Berkeley, CA 94720, USA

Associate Editor: Anna Tramontano

ABSTRACT

Motivation: Rapid methods for protein structure search enable biological discoveries based on flexibly defined structural similarity, unleashing the power of the ever greater number of solved protein structures. Projection methods show promise for the development of fast structural database search solutions. Projection methods map a structure to a point in a high-dimensional space and compare two structures by measuring distance between their projected points. These methods offer a tremendous increase in speed over residue-level structural alignment methods. However, current projection methods are not practical, partly because they are unable to identify local similarities.

Results: We propose a new projection-based approach that can rapidly detect global as well as local structural similarities. Local structural search is enabled by a topology-inspired writhe decomposition protocol that produces a small number of fragments while ensuring that similar structures are cut in a similar manner. In benchmark tests, we show that our method, writher, improves accuracy over existing projection methods in terms of recognizing SCOP domains out of multi-domain proteins, while maintaining accuracy comparable with existing projection methods in a standard single-domain benchmark test.

Availability: The source code is available at the following website: http://compbio.berkeley.edu/proj/writher/

Contact: dzhi@compbio.berkeley.edu

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Received on July 19, 2009; revised on February 25, 2010; accepted on March 19, 2010

1 INTRODUCTION

In the era of structural genomics, comparing a large number of structures can be a dauntingly time-consuming task. Therefore, a number of databases store precomputed structural similarities to accelerate structural comparison queries (Dietmann *et al.*, 2001; Madej *et al.*, 1995; Shindyalov and Bourne, 1998). However, such approaches have several limitations. First, these databases are not always updated in a timely fashion due to the sheer burden of computational requirements. Even if one applies a program that can compare two structures in 1 s, it would take more than 1 year to perform all-versus-all comparisons of 7897 proteins from the 40% non-redundant protein set of Astral 1.71 (Chandonia *et al.*, 2004), which is only a subset of over 50 000 solved protein structures in the PDB (Berman *et al.*, 2000). The second drawback of existing structural similarity databases is that they offer a rigid classification of structural relationships according to some predefined set of parameters. This precludes production of alternate classifications, such as building phylogenies for a particular subset of proteins while applying different scoring functions. In addition, analysis of a large number of artificial structures, such as clustering of models generated by structure prediction efforts, demands ultra-fast structure comparison. Therefore, there is an increasingly compelling need for tools that can rapidly compare a large set of structures.

Similarity of protein structures is typically measured via structural alignment, whose goal is to find a three-dimensional (3D) transformation that brings into correspondence the largest number of atoms between two structures. The quality of a 3D superposition is often measured by the number of matched C_α atoms and the distances between the matched atoms. The exact solution for the pairwise structural alignment is computationally expensive (Ambuhl *et al.*, 2000). Therefore, heuristic approaches have been developed to find good solutions with remarkable efficiency (for a review see, Eidhammer *et al.*, 2000). Even so, typical superposition-based heuristic programs are still too slow for many purposes: they take hours to perform a one-versus-all structural query. For example, running combinatorical extension (CE; Shindyalov and Bourne, 1998) for one protein structure against 2930 non-redundant structures would take about 1 h on a modern desktop computer (Kolodny *et al.*, 2005) and thus several CPU-months to complete an all-versus-all comparison.

A major speed increase for structural comparison was achieved by considering schemes of reduced representation of protein structures, instead of their full 3D coordinates of atoms. The most popular scheme for reduced structure representation is the secondary structure element (SSE), and a number of structure matching methods based on SSEs have been developed since the 1990s (Alexandrov and Fischer, 1996; Dror *et al.*, 2003; Harrison *et al.*, 2003; Holm and Sander, 1995; Koch *et al.*, 1996; Madej *et al.*, 1995; Mizuguchi and Go, 1995; Rufino and Blundell, 1994).

Since the 1980s, ideas from differential geometry and topology have been explored to encode 3D structures into 1D representations (Levitt, 1983; Rackovsky and Scheraga, 1984). Like a comparison of protein amino acid sequences, a 1D representation allows much faster structure comparison than a full 3D alignment. More recently, several groups developed alternative 1D methods using innovative shape descriptors, such as discrete torsion angles (Gong *et al.*, 2005), turning angles between smoothed backbone segments with

*To whom correspondence should be addressed.

†Present address: Department of Biostatistics, Ryals Public Health Bldg 327, University of Alabama at Birmingham, Birmingham, AL 35294, USA.

a certain distance (Zhi *et al.*, 2006), shape symbols (Ison *et al.*, 2005) or residue connectivity profiles (Teichert *et al.*, 2007) [see (Shu *et al.*, 2008) for a recent review of 1D methods]. Indeed, these methods can handle one-versus-all queries in a matter of minutes with a somewhat lower accuracy. However, they are still not fast enough for all-versus-all structure comparisons of datasets of current scale.

The ultimate reduction of running times are achieved by a new approach called projection methods, or '0D' methods (Hasegawa and Holm, 2009). The philosophy behind this approach is that the evaluation of overall structural similarity does not require an explicit specification of a 3D transformation and a residue-to-residue correspondence. Instead this approach maps a complete structure into a fixed length vector, corresponding to a point in a high-dimensional space, where each dimension encodes some transformation-invariant structural feature. Similarity between structures thus can be evaluated by the geometric distance of their projection points. Since the mapping step only needs to be done once and the comparison step demands little computation, projection methods can drastically speed up structural comparison. A typical one-versus-all query with projection methods takes only a fraction of second. A non-exhaustive list of examples of projection methods are: the PRIDE/PRIDE2 method (Gaspari *et al.*, 2005), which uses histogram of C_α distances; the LCM method (Lisewski and Lichtarge, 2006), which uses histogram of backbone distances between contacting C_α atoms; the SGM method (Røgen and Fain, 2003), which uses Gauss integrals (GIs); the LFF method (Choi *et al.*, 2004), which uses descriptors of SSE pairs; and the SSE footprints method (Zotenko *et al.*, 2006), which uses descriptors of SSE triples.

Projection methods can serve as fast filters for comprehensive structure search systems. Current practical structural search programs typically use SSE matching programs as filters before engaging full alignment. For example, DaliLite (Holm and Park, 2000) employs 3D-lookup (Holm and Sander, 1995) and Vorolign (Birzele *et al.*, 2007) employs SSE alignment algorithm (SSEA, McGuffin *et al.*, 2001) as fast filters. Projection methods are typically even faster than these SSE-based methods and thus have potential to be fast structural search filters.

However, there are several challenges preventing current projection methods from being practical. First, since current projection methods use a single vector encoding global properties of the structure—we call them *global projection methods*—they are unable to detect local structural similarities. For example, they cannot detect the similarity between a single-domain protein to one of the domains in a multi-domain protein. Second, certain relatively small structural changes in a protein structure, e.g. loop movement or loop indels, may cause significant changes in some type of global descriptors, e.g. GIs (Røgen and Fain, 2003). Conversely, some structures coincidentally share similar global features and thus introduce false positives.

Here, we propose a new projection-based scheme, writher, for structural comparison that will allow for the detection of local similarities. In this scheme, we view a structure as an ensemble of representative fragments. We employ the writhe decomposition, a novel procedure that decomposes a structure into fragments based on its intrinsic topological characteristics. Instead of using one global descriptor for the entire structure, we consider a set of descriptors for all representative segments. The conceptual novelty of our approach lies in that this representation allows for a comprehensive yet rapid approach to structural comparison: the similarity between pairs of segments is measured in rapid constant time, in the same fashion as in SGM (Røgen and Fain, 2003), and the global or local similarities are evaluated by scoring functions integrating similarities of individual fragments. To control the number of fragments to be matched, we apply a database-indexing scheme to efficiently retrieve only the fragments that are potentially similar to the query.

We test our method in two ways: a standard single-domain benchmark test and also a multi-domain test, where a set of multi-domain proteins are queried against a database of single-domain structures. We compare the performance of writher against existing projection methods. In addition, we also compare writher with leading non-projection-based structure comparison filters. Our test result shows that writher outperforms all these competing methods in the multi-domain test, while it has a comparable accuracy in the standard single-domain test. We believe that writher offers a key advance toward building a practical structure comparison tool for the richness of structures available.

2 METHODS

2.1 Preliminaries

Røgen and Fain (2003) pioneered the application of GIs (also known as Vassiliev knot invariants) as topological descriptors of protein structure. To be self-contained, we include a brief description of the GIs for protein structures below adapted from Røgen and Fain (2003). Readers are referred to Røgen and Fain (2003) and references therein for further details.

We represent a protein structure by $C=<C(i)>$, $i=1,\ldots,n-1$, the polygonal curve over protein's C_α atoms, where $C(i)$ represents the line segment connecting C_α atoms of residues i and $i+1$. The discrete version (first order) GI of the structure, also known as the writhe number (Levitt, 1983), may be described as:

$$G_{(1,2)}(C)=\sum_{1\le i_1<i_2\le n-1} w(i_1,i_2), \tag{1}$$

where $w(i_1,i_2)$ is the probability of observing the signed crossing of line fragments $C(i_1)$ and $C(i_2)$ from an arbitrary angle in the space. Therefore, the GI is the average number of signed self-crossings seen in the curve C from an arbitrary angle in the space. Here, we define the matrix of contribution of residue pairs i_1 and i_2 to the overall writhe number, $W=w(i_1,i_2), 1\le i_1<i_2\le n-1$, as the *writhe-matrix*. Similarly, the average number of unsigned self-crossings of curve C is: $G_{|1,2|}(C)=\sum_{1\le i_1<i_2\le n-1}|w(i_1,i_2)|$. $G_{(1,2)}(C)$ and $G_{|1,2|}(C)$ are called first-order GIs. Higher order GIs can be defined, such as second-order GI $G_{(1,3)|2,4|}(C)=\sum_{1\le i_1<i_2<i_3<i_4\le n-1} w(i_1,i_3)|w(i_2,i_4)|$. The higher order GIs describe the configurations of the crossings.

Røgen and Fain (2003) used 30D vector consisting of all first-, second- and some third-order GIs to represent a protein backbone, thus providing a mapping from an arbitrary protein structure to the 30D Euclidean space. The GI values were scaled to have uniform variances. They showed that a simple nearest neighbor-based algorithm can reproduce the structural similarity among CATH (Orengo *et al.*, 1997) family members with a considerable accuracy. We follow Røgen and Fain (2003) and use the 30D SGM vectors in our experiments.

The GIs can also be defined over a smoothed backbone, rather than the original backbone. GIs over the original backbone are heavily influenced by the local geometry within SSEs. In particular, α-helices contribute large negative values to the total writhe value. Defining GIs over smoothed backbone can effectively reduce the signal due to the local geometrical properties of SSEs and thus emphasize the global folding topology. For smoothing, we assign the center of gravity of every k consecutive C_α atoms as

a new pseudo-C_α atom. We will discuss selection of k in the next subsection. Lindorff-Larsen *et al*. (2005) and Røgen (2005) discussed the effect of defining SGM vector over smoothed backbone versus that over original backbone. Generally, different dimensions of the SGM vector over smoothed backbone are more independent than that over original backbone (Røgen, 2005). However, an SGM vector over original backbone leads to slightly better classification performance (Lindorff-Larsen *et al.*, 2005), probably because that it contains more SSE information, which is the basis for most existing structural classification schemes. In this work, the GI over both original and smoothed backbones is used.

In this work, we note the critical role of the writhe matrix in defining GI. The writhe matrix is the 'gradient' of the first-order GI, as the latter is summation (integral) of the former [Equation (1)]. Moreover, from the definitions of higher order GIs it is clear that the writhe matrix determines GIs of all orders. While Røgen and Fain (2003) only implicitly use the writhe matrix for the computation of GI vectors, we make an explicit representation of the writhe matrix and explore its characteristics, which will lead to our decomposition method.

2.2 Writhe decomposition

Our approach to the local structure comparison problem represents a protein structure by a set of substructures that reflect its local properties. The premise is that if two structures are similar, globally or locally, they should have a high chance of sharing some similar substructures. Therefore, local structural similarity between proteins can be inferred by matching their substructures using global structural comparison methods. The challenge is to obtain a sensible way of decomposing a structure into substructures.

A naive approach is to use all curve fragments, $\{C(i,j):\forall i,j\}$, where $C(i,j)$ denotes the structural fragment between i and j. However, this would produce too many fragments, and most fragments will be redundant. We wish to find a compact representative set of possibly overlapping fragments of the input backbone curve, such that similar structures are cut in a similar way, thus likely to share some fragments. Therefore, the fragments should be defined based on some intrinsic topological characteristics of the structures. Next, we introduce a compact decomposition that exploits the sparseness of the writhe matrix.

We observe that the writhe matrix over the smoothed backbone is often sparse, i.e. only a small number of entries in the writhe matrix deviate significantly from zero. This is because most residue segments are too far apart to generally appear to be crossing, and most nearby segments are parallel. We are interested in matrix values that are above some specified threshold that we termed 'significant entries' (Fig. 1c). Since computing GIs (of all orders) amounts to taking sums of entries in the writhe matrix, only significant entries will have significant contribution to GI.

Significant entries in the writhe matrix correspond to backbone crossings, where backbone segments with non-parallel directions pass one another at a close distance. Significant entries describe the configuration of backbone crossings, which reflects the pattern of backbone packing in the structural core of the protein. Since backbone crossings often involve several residues in a row, significant w-values due to one crossing event organize into a patch. We find local extreme values within a patch as concise representation of the crossing event.

Our writhe decomposition method is based on these local extreme values. For a writhe matrix, we first set to zero all entries with an absolute value smaller than a threshold of 0.03. This threshold is sufficient to retain ~10% of the entries in the writhe matrix. In case there are no entries above the threshold, we keep reducing the threshold by half until some entries are found. As a result, small patches of non-zero entries with significant values in the writhe matrix are identified. Within small patches, we find local extreme values (i,j), where $\mathrm{abs}(w(i,j)) \geq \mathrm{abs}(w(i\pm\{0,1\},j\pm\{0,1\}))$, and then include $C(i,j)$ in the set of fragments for writhe decomposition. Finally, we add the entire protein chain, i.e. $C(1,n-1)$. Notice that in our protocol multiple local extreme values per patch are permitted so long as they are not adjacent.

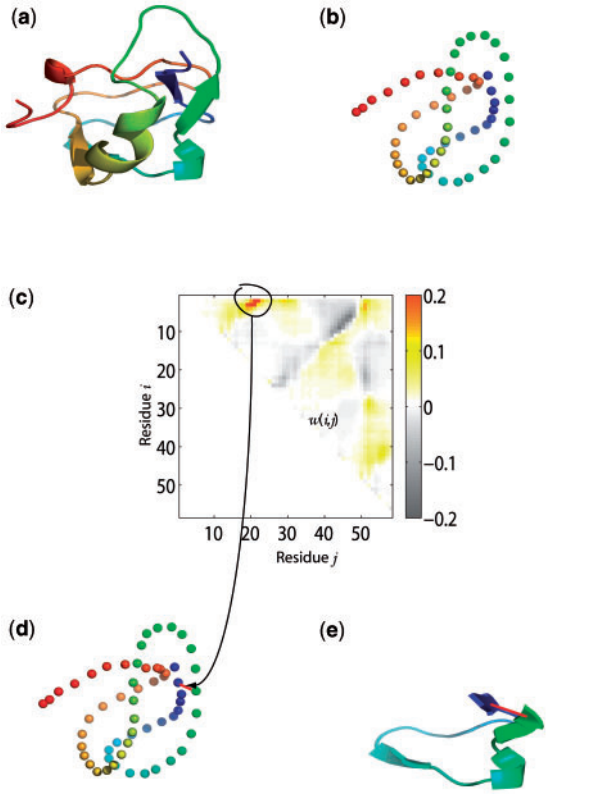

Fig. 1. Writhe decomposition. As an example, we use PDB (Berman *et al.*, 2000) entry 1UCS:A (Ko *et al.*, 2003), structure of a Type III antifreeze protein RD1. (**a**) Cartoon representation of the original chain. (**b**) Smoothed backbone. (**c**) Writhe matrix (only upper triangle entries are shown). (**d**) Local extreme values in writhe matrix are mapped back to the smoothed backbone. One extreme value is used as an illustration. As is typical, it represents a close perpendicular crossing of secondary structure, in this case two adjacent strands of a highly twisted sheet. (**e**) A segment in the original chain corresponding to the red linker in (d).

Table 1. Smoothing dependency of writhe decomposition

k	1	3	5	7	9	11	13
n, # frags	544.8	90.6	62.1	36.0	28.7	19.1	16.2

The average numbers of fragments from writhe decomposition over 10 representative structures (See caption of Supplementary Fig. S2 for PDB codes) with different smoothing parameter k are shown. $k=1$ means no smoothing. Smoothing with even $k=3$ significantly reduces the number of fragments compared with non-smoothing. We observe a generally exponentially decay in number of fragments with the smoothing parameter k, which we fit to $\log(n)=-0.18k+7.3$ by linear regression (Supplementary Fig. S2). The value of $k=7$ yields the largest negative deviation (residue of regression) from the regression line, and thus representing a balance of structural details (small k) and concise writhe decomposition (small n).

This modest redundancy allows flexibility in later fragment matching of similar fragments with slight variations in their ending points.

The number of extreme values strongly depends on the smoothing parameter k, the number of consecutive C_α atoms to be averaged. As shown in Table 1, higher values of k result in smoother backbone and consequently reduce the number of fragments in writhe decomposition. We chose the value

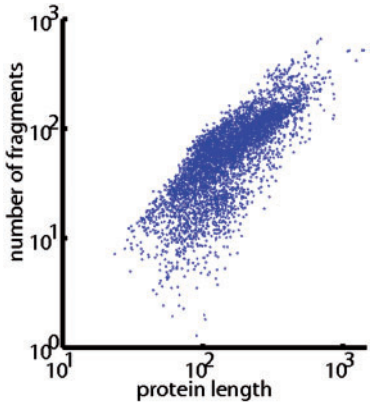

Fig. 2. Scatter plot of length of protein versus number of fragments from writhe decomposition.

Table 2. Accuracy of Astral 1.65 40% non-redundant superfamily benchmark

Method	Coverage (%)	Precision (%)	Time (s/query)
3D-lookup	89.0	90.0	163.8
SSEA	90.3	70.1	0.14
SSEF	90.3	70.2	3.8
SGM	85.6	64.8	0.28
Writher	83.6	65.6	4.3
Writher with RandDecomp	83.6	55.6	3.6

Coverage is the portion of queries for which the program gives an answer among 5345 proteins queried ($\frac{TP+FP}{5345}$). Precision is the portion of answers that are correct, i.e. the top-scoring hit excluding self is from the same SCOP superfamily ($\frac{TP}{TP+FP}$). Time is average time for handling a one-versus-all query. RandDecomp is writher with random (not writhe) decomposition (see text in Section 2.2).

$k=7$ for writhe decomposition as it is large enough to smooth out most of curvatures in α-helices and yet small enough to retain most of structural details.

We observe a linear correlation between the number of fragments and the length of proteins (Fig. 2) in the SCOP 1.65 40% non-redundant set (Chandonia *et al.*, 2004). This verifies that our writhe decomposition procedure produces a compact set of fragments. According to linear regression, there are on average 41 fragments per 100 amino acids.

To verify that writhe decomposition produces informative fragments, we compared writhe decomposition with a random decomposition that generates the same number of fragments with the same length distribution, but with randomized starting positions. The comparative benchmark results are shown in Table 2.

Writhe decomposition, inspired by the definition of first- and higher order of GIs, offers a concise 'sampling' of the major folding content of a structure. Due to the sparse structure of the writhe matrix, the folding content, reflected by GIs, only changes drastically at the significant entries of the writhe matrix. Geometrically, these significant entries correspond to backbone crossings, where non-parallel backbone segments pass each other at a close distance. In other words, fragments resulting from writhe decomposition correspond to 'loop' segments with a pair of twisted termini. These loops are distinctive from obvious and frequently recurring motifs such as, for example, hairpin motifs. Moreover, the size of these loops is not restricted and thus this definition captures structural shapes beyond the motif-level and may include the subdomain and domain levels.

2.3 Database search

In a structural database search task, one structure (query) is compared against a database of structures. The major advantage of projection methods for structural comparison is speed. Our method considers the matching between all fragments of one structure to all fragments of the other. If we assume matching a pair of fragments takes constant time $O(1)$, the same as matching two structures by traditional projection methods, then a naive implementation of fragment comparison between a query protein against the database would take time $O(Nn^2)$, where N is the number of proteins in the database and $O(n)$ is the number of fragments of a query protein and each database protein. To accelerate structural comparison for the database scale, we use the range query on B-tree indexing technique similar to the one used in Camoglu *et al.* (2003) to achieve an average speed of $O(n\log(N))$ per query.

Our method uses a relational database in two steps: indexing and query-handling. In the indexing step, all the database proteins are first decomposed into the fragments. Subsequently, the descriptors of individual fragments are computed as by Røgen and Fain (2003). Finally, these descriptors are organized into a database table and indexed using B-tree. In the query-handling step, the goal is to find proteins in the database similar to a query q. We first decompose q into the fragments, and then query each fragment against the database table to identify all similar fragments in the database. Finally, the proteins in database are ranked by a scoring function that evaluates the overall similarity based on the fragment matches.

A key issue is to quickly identify fragments similar to a query fragment f using database queries. The Euclidean norm (l^2-norm), as used in Røgen and Fain (2003), is difficult to implement using existing database-indexing techniques: commercial spatial index techniques are only practical for low-dimensional data, e.g. 2D or 3D, whereas our fragments are indexed by m-dimensional vectors. Therefore, we instead use the l^∞-norm, in which two fragments f and g are considered a 'match' if $|f^i - g^i| < \epsilon$ for all dimensions $1 \le i \le m$ (here we use f to represent both the fragment and the SGM vector of the fragment). This is equivalent to the intersection of results of range queries $g^i - \epsilon < f^i < g^i + \epsilon$ at all dimensions. MySQL can handle a such range query using B-tree indexing in logarithmic time of the number of fragments in the database, whereas a brute-force scan would take a linear time. Since values at all dimensions are normalized so that the mean is 0 and the SD 1, only a single universal ϵ is used for all dimensions.

The ϵ value determines the sensitivity and specificity of the fragments returned by the range query. Since the similarity of these fragments is going to be more accurately re-evaluated in later stages, the choice of ϵ is done based on achieving a higher sensitivity. We found that the most similar, based on l^∞-norm, 1000 fragments provide a sufficient sensitivity in practice. Therefore, we use a simple dynamic approach where we determine a proper ϵ value by the following protocol: initially ϵ is set to 0.5; if the number of proteins returned is sufficient (default > 1000), exit loop; otherwise set $\epsilon = \epsilon * 2$ and loop.

We note that this change from l^2-norm to l^∞-norm is not without cost. In fact, the overall accuracy of the SGM method over the Astral 1.65 40% non-redundant set drops 7% from nearest neighbor using l^2-norm to nearest neighbor using l^∞-norm. Therefore, we first use l^∞-norm queries as a filter only to quickly collect potential similar fragments, before applying a scoring function.

Unlike the existing projection methods where similarity between proteins is simply established by the distance between their projected vectors, writher measures structural similarity through matching fragment sets. We employ a scoring algorithm considering both the matching quality of the best matching fragment pair and the number of fragment pairs consistent with the best matching pair. Specifically, the score of query q against a protein p in the database is the product of the following two terms.

(1) $maxnorm = \alpha m - \min_{f \in \text{Frags}(p)} ||f - q(f)||$ is defined as the maximum fragment matching score, where Frags(p) is the set of fragments in p, $q(f)$

is the best matching fragment of f in q and $||f-q(f)||$ is the l^1-distance (l^2-distance gives essentially the same result) between the SGM vectors f and $q(f)$ of the non-smoothed chain segments (Røgen, 2005). m is the dimensionality of the GI-vector. α is the maximum average dimension-wise deviation allowed for a fragment pair to be considered similar. Thus, it is expected that $||f-q(f)|| > \alpha m$ for similar fragment pairs and otherwise for dissimilar pairs. Fragment pairs with negative *maxnorm* values are ignored. Based on experiments, we set $m=30$ [as by Røgen and Fain (2003)] for global matching, and $m=15$ (only including first- and second-order GIs) for local matching; α is determined to be 0.3.

(2) *numfrag* is defined as the number of non-redundant matching fragment pairs between p and q that are consistent with the best matching fragment pair. A representative fragment pair $\{(x_p^1, y_p^1), (x_q^1, y_q^1)\}$ is *consistent* with fragment pair $\{(x_p^2, y_p^2), (x_q^2, y_q^2)\}$ if $|(x_p^1-x_q^1)-(x_p^2-x_q^2)| < c$, where c is some cutoff (default 10). This definition ensures that the difference between the gap lengths is limited by 10 residues.

Redundancy still exists after the writhe decomposition, despite of extreme-value selection procedure. This creates over-counting of number of consistent matching pairs. Redundancy among matching fragment pairs is removed by first clustering and then selecting one representatives for each cluster. The *matching distance* between two fragment pairs is defined as the maximum of the offsets among their corresponding coordinates: $d(\{(x_p^1, y_p^1), (x_q^1, y_q^1)\}, \{(x_p^2, y_p^2), (x_q^2, y_q^2)\}) = max(|x_p^1-x_p^2|, |x_q^1-x_q^2|, |y_p^1-y_p^2|, |y_q^1-y_q^2|)$. Based on this distance measure, fragment pairs are clustered using the following protocol. Initially, there is no cluster; and then the fragment pairs are considered one at a time in a random order. If the matching distance between the fragment pair in consideration and the center of any existing cluster is small (default $d \leq 10$), the fragment pair is inserted into the existing cluster and the center of the cluster is updated to the center of gravity of its fragment pairs. If no centers of existing clusters are within a small matching distance with the fragment pair, a new cluster containing just the fragment pair is created.

For global matching tasks, e.g. in the single-domain benchmark test shown below, we wish to measure the consistency of the relative positions of the best matching fragment pair. For this purpose, the score is also multiplied by a global matching adjustment factor, defined as follows. Let L_p and L_q be the lengths of p and q, respectively, and $\{(x_p, y_p), (x_q, y_q)\}$ be the highest scoring matching fragment pair between p and q based on the score $maxnorm * numfrag$, then the relative positions of the matching fragments are $(x_p/L_p, y_p/L_p)$ and $(x_q/L_q, y_q/L_q)$. The global matching adjustment factor is $1-|x_p/L_p - x_q/L_q| - |y_p/L_p - y_q/L_q|$.

Writher uses the writhe matrix twice: the first time to derive a decomposition protocol, and the second time to calculate the GI vectors for individual fragments for structural search. In both uses, we face a choice of using either the writhe matrix over the smoothed or the unsmoothed backbone. We made our choices based on different characteristics of these two kinds of matrices. On one hand, we chose the writhe matrix over the smoothed backbone for fragment decomposition because that smoothing removed unnecessary details that obscure the overall topology of the protein fold, and as a result the writhe matrix over the smoothed backbone is sparse. This is not the case for the unsmoothed version (Supplementary Fig. S1). On the other hand, we chose the writhe matrix over the unsmoothed backbone for structural search after writhe decomposition. This is because as shown by Røgen (2005) that an SGM vector over original backbone leads to slightly better classification performance than the smoothed version.

3 RESULTS

Before describing our assessment protocol, we discuss a common bias in existing benchmark tests. Existing benchmark tests typically use representative SCOP (Murzin *et al.*, 1995) or CATH (Orengo *et al.*, 1997) domains. However, SCOP or CATH domains are not generally typical protein structures a user may encounter: SCOP or CATH domains are normally cut from whole PDB structures by experts of structure classification and automated tools. For a new protein to be compared against existing ones, delineating its domain boundaries is a non-trivial task for its own right (Holland *et al.*, 2006).

Therefore, we use two benchmark tests for evaluating our method. In the first experiment, we follow the common benchmark scenario and use single domains from a representative set of SCOP proteins. We use this test to verify that our method's performance is comparable with existing projection methods and also to provide an assessment consistent with those used previously. The second experiment adopts a more realistic setting, where multi-domain proteins are queried against a SCOP representative set of single-domain protein structures.

Projection and SSE-based methods are aimed to be used as quick filters for more elaborated, but slower, similarity search methods. A good example is 3D-lookup method used as a filter in DaliLite program. A filter method should quickly compute a short list, e.g. about 100, of potentially similar structures. Some protein superfamilies (or families) may contain a large number of structures. Therefore, returning all similar structures from one superfamily may result in too many matches if the number of total matches is capped: some high-scoring and truly similar superfamilies might be excluded from the filter if they are simply less similar than the highest scoring superfamily. This problem becomes especially acute when a query is a multi-domain protein and we search for all domain representatives. Therefore, we suggest to use a scheme where instead of returning all similar structures, only superfamily representatives that received the highest score are returned. In other words, a result list of a filter method does not contain two structures from the same superfamily. In this way we keep results list short and it covers the largest number of structural superfamilies. This scheme was used to evaluate all filter methods below.

Writher is compared against the following projection methods and leading non-projection filter programs.

(1) SGM (Røgen and Fain, 2003): a projection method based on GI comparison, employing the same procedure used in the basic stage of writher where two segments are compared. The GI vectors are generated using the program `GI.c` (Røgen and Fain, 2003) and the vector comparison is carried out by an inhouse Perl implementation.

(2) SSEF (Zotenko *et al.*, 2006): a leading projection method. The SSE footprint vectors are computed by a downloadable program (Zotenko *et al.*, 2006) and the vector comparison is carried out by an inhouse Perl implementation.

(3) SSEA (McGuffin *et al.*, 2001): the filter for Vorolign (Birzele *et al.*, 2007). SSEA treats a structure as a string of SSEs and aligns a pair of structures by the standard dynamic programming algorithm on this SSE string. As the raw alignment score gave unsatisfactory results, we normalized it as follows: $S(p,q) = \frac{S_{\text{raw}}(p,q)}{S_{\text{raw}}(p,p)+S_{\text{raw}}(q,q)-S_{\text{raw}}(p,q)}$, where q and p are query and target proteins, and $S_{\text{raw}}(p,q)$ is the raw score from SSEA method. The algorithm is evaluated using an inhouse C++ implementation. SSEA is not a projection method.

(4) 3D-lookup algorithm (Holm and Sander, 1995): the filter for DaliLite (Holm and Park, 2000). The database search is carried out by the `wolf` program in the DaliLite package.

To ensure a fair comparison, we run 3D-lookup against all chains in the database. This is different from the setting used by the Dali server where the 3D-lookup is used to identify the first entry in the database with a significantly large number of SSE pair matches to the query (and then the full DALI alignments are carried out in the precomputed structure neighborhood of the identified entry). 3D-lookup uses 3D hashing and is not a projection method in the sense we used here.

3.1 SCOP benchmark: single-domain chains

Our first experiment follows the model of Zotenko *et al.* (2006). The dataset is the Astral 1.65 40% non-redundant set (Chandonia *et al.*, 2004), which contains 5345 SCOP chains, with pairwise sequence identities <40%. Zotenko *et al.* compared the performance of different projection methods for the task of classifying a new structure into its proper SCOP categories. For a projection method, one chain is considered being correctly classified if its nearest neighbor in the projected high-dimensional space (its highest scoring hit from database, self-hits excluded) belongs to the same SCOP category.

Writher is not designed to compete with single projection methods in identifying global structural similarities. Since writher needs to handle the added 'noise' from decomposed fragments, it is expected to suffer from more false positives than global, single-chain, projection methods. The additional fragments also make writher run slower. However, the test result shown in Table 2 demonstrate that our fragment-based method produces an accuracy comparable with SGM in terms of classifying SCOP domains according to global structural similarity. It is not surprising that SSE-based methods SSEA, SSEF and 3D-lookup achieve better accuracies, since SCOP classification is largely based on SSE packing patterns. 3D-lookup outperforms other methods with a significant margin, with a trade-off of much longer running time.

In terms of running time, Writher is slower than SGM and SSEF; but it is still orders of magnitudes faster than residue-level structural alignment methods. Writher's running time includes the time for database queries and the time for scoring the structures by summarizing database hits. Database queries only consume 28% of the total wall time. Since writher is currently implemented using Perl as a prototype, the running time could be improved further by using faster programming languages.

3.2 SCOP benchmark: multi-domain chains

In the second experiment, we test the writher's ability to detect local structural similarities between multi-domain proteins (queries) and a database of SCOP domains (mostly single-domain chains). We construct our benchmark query set by selecting whole PDB chains with at least two domains present in the Astral 1.65 40% non-redundant set, yielding 699 multi-domain chains. These queries were searched against the same Astral-40% database as in the single-domain test. As a reference, we also bring the frequently used full 3D alignment method CE into comparison; we certainly do not expect any projection methods would outperform CE, but CE may be thought loosely as a ceiling on performances for the projection methods.

Our goal is to measure methods' ability to recognize the superfamily memberships for parts of a multi-domain query. Some protein superfamilies may be over- or underrepresented in the database. For example, if a domain of a multi-domain protein belongs to a large superfamily then the result list of a query search might contain mostly proteins from that superfamily. However, in this test, we are interested in the ability to identify representative protein chains for each domain of a multi-domain protein. For this purpose in the search results we designate the score of a superfamily by its top-scoring member (the Astral domains that are part of the query multi-domain protein are excluded from the results). Superfamilies containing a domain in the query protein are considered 'trues', and others 'falses'.

We generate the ROC curve as following. At a rank cutoff k, $1 \leq k \leq 300$, and for each query, the following quantities for computing ROC curves are defined: TP is the number of trues with a rank better than k; TN is the number of falses with a rank worse than k; FP is the number of falses with a rank better than k; and FN is the number of trues with a rank worse than k. Consequently, sensitivity = TP/(TP + FN); specificity = TN/(TN + FP). Now, for a single k, we take the average of sensitivity for all queries as the estimated sensitivity for k, the average of specificity for all queries as the estimated specificity for k, and draw a ROC curve for the estimated sensitivity versus the estimated specificity (Fig. 3a). Notice that, due to a non-uniform distribution of multi-domain proteins in Astral superfamilies, the ROC-like curve from random ordering of the chains in Astral 1.65 40% set is different from the diagonal line with a 0.5 area.

In this test, writher achieves the highest accuracy among projection methods. It is not surprising to see CE reach the highest accuracy, at a price of being 1000 times slower than writher. The SCOP classification is closely related to secondary structure composition and SSEF (Zotenko *et al.*, 2006) uses a 1500-dimensional vector that represents extensive information regarding the configurations of SSE triples. Therefore, it is quite significant that writher also outperforms SSEF and SSEA. The top performer in single-domain test, 3D-lookup, is only better than writher at the very specific range (better than 99% specificity). Since SGM is the underlying global projection method of writher, and writher leads SGM by a wide margin, we ascribe the performance of writher to the fragment-decomposition stage we introduced here.

We also computed the test statistics for the query subset of 399 chains belonging to superfamilies with at least four structures in the database (Fig. 3 and Table 3). In this subset, writher outperforms SSEF with an even larger margin. For example, for a query multi-domain protein, selecting writher's top 114 superfamily hits (out of 1287) includes each of the true superfamilies (the ones containing any of the domains in the query protein) with a 80% chance, while the same cutoff by the second best performer, SSEF, has a 68% chance. Moreover, selecting a protein's top 260 superfamily hits by writher has only a 10% chance missing any of its true superfamilies.

Surprisingly, we found that writher even outperforms CE for a number of multi-domain structures. For example, there are 42 cases among queries with two domains for which writher identifies the correct superfamilies as its top two hits. Among those, there are 16 cases where CE fails to correctly identify superfamilies from its top two hits. This result has not occurred by chance, since SGM, SSEA and Random have not identified correctly any of the two-domain proteins. One of the examples is 1l3s chain A, crystal structure of *Bacillus* DNA polymerase I fragment complexed to DNA (Johnson *et al.*, 2003). SCOP classifies it into two domains

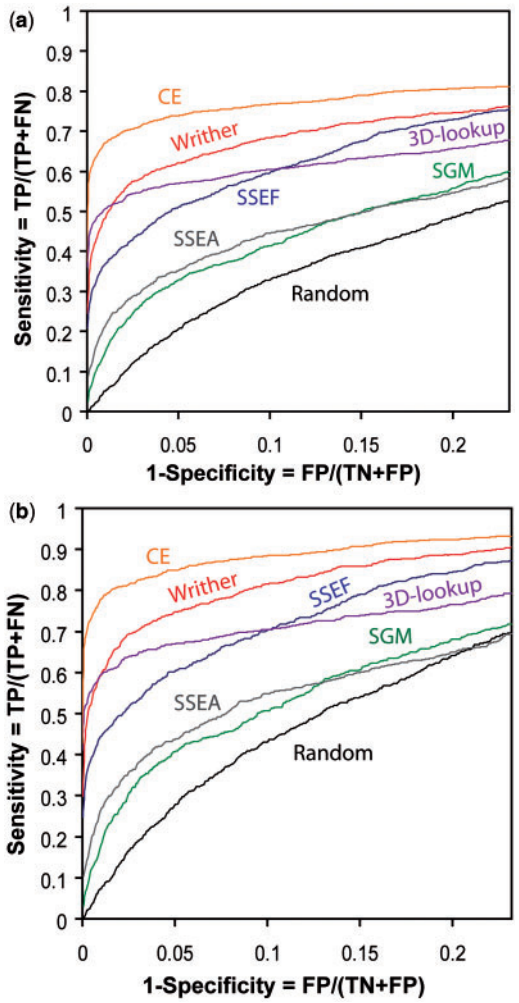

Fig. 3. (**a**) Multi-domain benchmark result. Multi-domain proteins are queried against the database of single-domain proteins. See text for the exact protocol for generating these ROC curves. (**b**) Multi-domain benchmark result using only the multi-domain proteins that belong to superfamilies with at least four database members.

belonging to the Ribonuclease H-like superfamily (e.8.1) and to the DNA/RNA polymerases superfamily (c.55.3). Writher identified the correct superfamilies for both domains as its top two hits—domains d1t7pa2 and d1kfsa1. CE also identified the ribonuclease H-like domain as the top hit, but failed to rank the DNA/RNA polymerase domain as the second hit. Although CE outperforms writher in general, this result shows that writher has potential to improve sensitivity of a full alignment method such as CE in a local alignment task.

In addition we investigated performance of these methods for domains in four major SCOP classes, including all-α, all-β, α/β and $\alpha+\beta$, among these 399 chains. As shown in Supplementary Figure S3, writher displays a superior performance in all classes except the all-α class. In the case of the all-α class all methods perform poorly, comparable with random classification. Surprisingly, results of writher is even comparable with CE for all-β, α/β and $\alpha+\beta$ classes, only losing by a large margin for the all-α class. We speculate that the poor performance of writher for all-α domains may be due to that the backbone smoothing procedure before writhe decomposition has over simplified α-helices. A special smoothing treatment for α-helix-rich proteins may improve writher further.

Table 3. Multi-domain benchmark test result

	Sensitivity at				
	99.7% Specif.	99% Specif.	95% Specif.	90% Specif.	Time (s/query)
CE	71	78	85	88	17748.9
3D-lookup	**54**	**59**	67	70	404.2
Writher	50	**59**	**75**	**81**	17.3
SSEF	37	44	60	68	1.9
SSEA	15	26	44	53	0.34
SGM	9	18	41	48	0.14
Random	2	7	27	41	NA

Multi-domain chains are queried against the Astral 1.65 40% non-redundant superfamily benchmark set. Only those multi-domain chains with at least four members in its SCOP superfamily are used for the query. Sensitivities at several specificity levels are reported. Bold values signify the highest sensitivity achieved at a specificity level.

Compared with the running time of the single-domain test, the running time of multi-domain test for global projection methods SSEF and SGM remain unchanged. The running times for writher, SSEA and 3D-lookup increase as the length of the queries increases. However, writher is still quite rapid and remains an order of magnitude faster than 3D-lookup.

4 CONCLUSIONS AND DISCUSSIONS

In this article, we present a new method, writher, for rapid protein structure similarity search. Writher compares structures by first decomposing them into a compact representative set of fragments, and then matching fragments by a fast projection technique. Our approach allows efficient detection of both global and local structural similarities. To provide computation time practical for real-time queries, we employ a database-indexing scheme. In a benchmark test for recognition of SCOP domains from a set of full-length multi-domain proteins, we show that writher is able to identify local structural similarities substantially more effectively than existing filter methods.

Our writhe decomposition has some resemblance to fragment-based protocols employed in structure prediction (see Tyagi *et al.*, 2008, for a recent review). However, writhe decomposition is fundamentally different from existing fragment decomposition methods. First, writhe decomposition is designed to comprehensively represent all major local structural features of a single structure, while fragmentation for structure prediction are to build empirical library of possible local folding patterns in all existing structures. Moreover, the fragments in writhe decomposition are not limited in length and can be as long as hundreds of residues; while the local structure libraries for structure prediction usually contain short fragments of fixed length (4–15 residues).

In practice, our decomposition scheme is rather general, in that it can transform any global projection method into a local projection method using any meaningful fragmentation procedure. In this work, we employed the projection method SGM (Røgen and Fain, 2003) to describe individual. It is possible to generalize this approach to use other projection methods, such as the SSE-based methods (Choi *et al.*, 2004; Holm and Sander, 1995; Zotenko *et al.*, 2006).

Moreover, while writher has been evaluated for superfamily classification in our benchmark experiment, it is possible to apply writher to detect local similarities that have functional characterization. For example, it may be possible to use ligand binding site knowledge to limit the search only to fragments proximate to the binding site, and to alter the smoothing associated with such focused studies.

Just as BLAST allows for flexible specifications of sequence similarity that arise in various biological enquiries, a structural search engine would empower biologists with means for searching user-defined structural similarity, to complement the existing carefully curated but rigid structural classification systems, allowing one to realize goals such as protein function prediction (Petrey and Honig, 2009). In 1990, when BLAST was published, the number of sequences in the Genbank was only 39 533. The number of structures in the PDB is already over 55 000, yet structural biologists still do not have a correspondingly fast search engine for structural similarity. We believe that this lack of tool is not due to the lack of demand but rather due to the lack of technological development. Like BLAST, a successful structural search system would meet the following specifications: (i) fast, able to search a whole database in interactive time; (ii) accurate in detecting local similarities; and (iii) accurate in aligning amino acids. Achieving all these goals is a grand goal which is unlikely to be feasible with one algorithmic solution. In this article, we proposed a method that can be used as a fast filter to screen a whole structural database. The main advantage of this method is that, unlike previous filter approaches, writher is sensitive not only to global similarities but also to local structural similarities. Therefore, we believe writher is an important step toward a rapid practical search system for structural biology research.

ACKNOWLEDGEMENTS

We wish to thank Fabian Birzele for help with applying the SSEA method. We thank Peter Røgen for discussions.

Funding: National Institute of Health (R01 GM073109 and K99/R00 RR024163); the Department of Energy (DE AC03 76SF00098); an Alfred P. Sloan Foundation research fellowship.

Conflict of Interest: none declared.

REFERENCES

Alexandrov,N. and Fischer,D. (1996) Analysis of topological and nontopological structural similarities in the PDB: new examples with old structures. *Proteins Struct. Funct. Bioinform.*, **25**, 354–365.

Ambuhl,C. *et al.* (2000) Computing largest common point sets under approximate congruence. In *Proceedings of the 8th Annual European Symposium on Algorithms*. Springer, London, UK, pp. 52–63.

Berman,H. *et al.* (2000) The protein data bank. *Nucleic Acids Res.*, **28**, 235–242.

Birzele,F. *et al.* (2007) Vorolign–fast structural alignment using Voronoi contacts. *Bioinformatics*, **23**, e205–e211.

Camoglu,O. *et al.* (2003) Towards index-based similarity search for protein structure databases. In *IEEE Computer Society Bioinformatics Conference*, pp. 148–158.

Chandonia,J. *et al.* (2004) The ASTRAL compendium in 2004. *Nucleic Acids Res.*, **32**, D189–D192.

Choi,I. *et al.* (2004) Local feature frequency profile: a method to measure structural similarity in proteins. *Proc. Natl Acad. Sci. USA*, **101**, 3797–3802.

Dietmann,S. *et al.* (2001) A fully automatic evolutionary classification of protein folds: Dali domain dictionary version 3. *Nucleic Acids Res*., **29**, 55–57.

Dror,O. *et al.* (2003) Multiple structural alignment by secondary structures: – algorithm and applications. *Prot. Sci.*, **12**, 2492–2507.

Eidhammer,I. *et al.* (2000) Structure comparison and structure patterns. *J. Comput. Biol.*, **7**, 685–716.

Gaspari,Z. *et al.* (2005) Efficient recognition of folds in protein 3D structures by the improved PRIDE algorithm. *Bioinformatics*, **21**, 3322–3323.

Gong,H. *et al*. (2005) Building native protein conformation from highly approximate backbone to torsion angles. *Proc. Natl Acad. Sci. USA*, **102**, 16227–16232.

Harrison,A. *et al.* (2003) Recognizing the fold of a protein structure. *Bioinformatics*, **19**, 1748–1759.

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

Holland,T. *et al.* (2006) Partitioning protein structures into domains: Why is it so difficult? *J. Mol. Biol.*, **361**, 562–590.

Holm,L. and Park,J. (2000) DaliLite workbench for protein structure comparison. *Bioinformatics*, **16**, 566–567.

Holm,L. and Sander,C. (1995) 3-D lookup: fast protein structure database searches at 90% reliability. In *Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology*, pp. 179–187.

Ison,R. *et al.* (2005) Proteins and their shape strings. *Eng. Med. Biol. Mag. IEEE*, **24**, 41–49.

Johnson,S.J. *et al.* (2003) Processive DNA synthesis observed in a polymerase crystal suggests a mechanism for the prevention of frameshift mutations. *Proc. Natl Acad. Sci. USA*, **100**, 3895–3900.

Ko,T. *et al.* (2003) The refined crystal structure of an Eel Pout Type III antifreeze protein RD1 at 0.62-A resolution reveals structural microheterogeneity of protein and solvation. *Biophys. J.*, **84**, 1228–1237.

Koch,I. *et al.* (1996) An algorithm for finding maximal common subtopologies in a set of proteins. *J. Comput. Biol.*, **3**, 289–306.

Kolodny,R. *et al.* (2005) Comprehensive evaluation of protein structure alignment methods: scoring by geometric measures. *J. Mol. Biol.*, **346**, 1173–1188.

Levitt,M. (1983) Protein folding by restrained energy minimization and molecular dynamics. *J. Mol. Biol.*, **170**, 723.

Lindorff-Larsen,K. *et al.* (2005) Protein folding and the organization of the protein topology universe. *Trends Biochem. Sci.*, **30**, 13–19.

Lisewski,A. and Lichtarge,O. (2006) Rapid detection of similarity in protein structure and function through contact metric distances. *Nucleic Acids Res.*, **34**, e152.

Madej,T. *et al.* (1995) Threading a database of protein cores. *Proteins*, **23**, 356–369.

McGuffin,L.J. *et al.* (2001) What are the baselines for protein fold recognition? . *Bioinformatics*, **17**, 63–72.

Mizuguchi,K. and Go,N. (1995) Comparison of spatial arrangements of secondary structural elements in proteins. *Protein Eng.*, **8**, 353–362.

Murzin,A. *et al.* (1995) SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.*, **247**, 536–540.

Orengo,C. *et al.* (1997) CATH - a Hierarchic Classification of Protein Domain Structure. *Structure*, **5**, 1093–1108.

Petrey,D. and Honig,B. (2009) Is protein classification necessary? toward alternative approaches to function annotation. *Curr. Opin. Struct. Biol.*, **19**, 363–368.

Rackovsky,S. and Scheraga,H.A. (1984) Differential geometry and protein folding. *Acc. Chem. Res.*, **17**, 209–214.

Røgen,P. (2005) Evaluating protein structure descriptors and tuning Gauss integral based descriptors. *J. Phys. Condens. Matter*, **17**, S1523–S1538.

Røgen,P. and Fain,B. (2003) Automatic classification of protein structure by using Gauss integrals. *Proc. Natl Acad. Sci. USA*, **100**, 119–124.

Rufino,S. and Blundell,T. (1994) Structure-based identification and clustering of protein families and superfamilies. *J. Comput. Aided Mol. Des.*, **8**, 5–27.

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

Shu,N. *et al.* (2008) Describing and comparing protein structures using shape strings. *Curr. Protein Pept. Sci.*, **9**, 310–324.

Teichert,F. *et al.* (2007) Sabertooth: protein structural alignment based on a vectorial structure representation. *BMC Bioinformatics*, **8**, 425.

Tyagi,M. *et al.* (2008) Protein structure mining using a structural alphabet. *Proteins Struct. Funct. Bioinform.*, **71**, 920–937.

Zhi,D. *et al.* (2006) Representing and comparing protein structures as paths in three-dimensional space. *BMC Bioinformatics*, **7**, 460.

Zotenko,E. *et al.* (2006) Secondary structure spatial conformation footprint: a novel method for fast protein structure comparison and classification. *BMC Struct. Biol.*, **6**, 12.