Artif Intell Rev
DOI 10.1007/s10462-013-9416-9

# Going over the three dimensional protein structure similarity problem

**Nantia Iakovidou · Eleftherios Tiakas · Konstantinos Tsichlas · Yannis Manolopoulos**



**Abstract** This article presents in detail our novel proposed methodology for detecting similarity between or among three dimensional protein structures. The innovation of our algorithm relies on the fact that during the similarity process, it has the ability to combine many attributes together and fulfill lots of preconditions, which are extensively discussed throughout the paper. Our concept is also supported by an efficient and effective indexing scheme, that provides convincing results comparing to other known methods.

**Keywords** Protein structure similarity · Indexing scheme · Combined linear measure

## 1 Introduction

A significant task in the area of structural biology and bioinformatics is finding proteins, whose structures or substructures are similar to those of other proteins. Since the knowledge of the 3D structure of a protein can yield useful information about its functional properties, then structural similarity can be a very good predictor of functional similarity and evolutionary-related proteins (protein families) (Needleman and Wunsch 1970). Several opinions support that sequence alignment methods can also be useful for this purpose, but this turns out to be less the case when the sequence identity of the involved proteins is lower than 30 % (Gan 2002; Lesk 2004; Micheletti and Orland 2009). This practically means that below this threshold the involved proteins have low sequence similarity and consequently possible detection of

N. Iakovidou (✉) · E. Tiakas · K. Tsichlas · Y. Manolopoulos
Department of Informatics, Aristotle University of Thessaloniki, 54124 Thessaloníki, Greece
e-mail: niakovid@csd.auth.gr

E. Tiakas
e-mail: tiakas@csd.auth.gr

K. Tsichlas
e-mail: tsichlas@csd.auth.gr

Y. Manolopoulos
e-mail: manolopo@csd.auth.gr

Published online: 12 July 2013

evolutionary and/or functional relatedness is more appropriately complemented by structural alignment techniques (Koehl 2001; Lichtarge and Sowa 2002), which is in fact the focus of the present study.

Various methods for detecting protein structural relationships have been proposed in recent years. Some of them perform alignment in rigid structures at the level of C-alpha atoms of proteins. A typical example of this category is the root mean square deviation measure (RMSD), which computes the minimum average distance between the backbones of superimposed proteins. According to Cohen and Sternberg (1980), Koehl (2001), Rogen and Fain (2003), Zhi et al. (2006) RMSD provides a pragmatic definition of structural similarity but consists an excellent measure of it only for nearly identical structures. Once the shape of two proteins begins to diverge, RMSD looses its effectiveness. In general, the majority of these approaches obtain best alignment by minimizing a suitable measure of geometric compatibility, such as the similarity of distance matrices in DALI over the possible amino acid pairings (Holm and Sander 1993), or the weighted sums of distances of equivalent C-alpha atoms that are used in CE algorithm (Shindyalov and Bourne 1998) and MAMMOTH-mult (Lupyan et al. 2005). Usually, the deficiencies of these methods arise from the fact that in many cases the distance-based measures of similarity do not satisfy the triangle inequality: $d(x, y)+d(y, z) \geq d(x, z)$. When a method violates the triangle inequality, it is fundamentally unable to judge dissimilarity and this problem worsens with increasing distance (Rogen and Fain 2003).

Several other methods align proteins by reducing them to a coarse metric such as secondary structure elements (SSEs). Such kind of approaches require some criterion for assigning secondary structures to proteins, that is to say some method to classify residues in the protein as belonging to helices, strands (namely $\alpha$ -helices and $\beta$ -strands) or loops (of various types) or not being part of an SSE at all. Tableau-based methods generally belong to this category, such as SA Tableau Search (Stivala et al. 2010) and IR Tableau (Zhang et al. 2010). Other representative methods of this group are VAST (Gibrat et al. 1996; Madej et al. 1995), SSM (Krissinel and Henrick 2004), GANGSTA (Kolbeck et al. 2006), LOCK2 (Shapiro and Brutlag 2004) and SARF2 (Alexandrov 1996). The truth about SSE methods is that there is not an exact procedure of assignment and also opinions vary about the precise beginning and end of SSEs (Stivala et al. 2010). Also a major drawback of these approaches is that they need to perform an exhaustive sequential scan of a structure database to find similar structures to a target protein (Park et al. 2006). Furthermore, using only SSEs means that regions of protein structures that are not defined as being part of an SSE are not used at all and this fact can lead to less sensitive results.

Methods that do not perform any kind of alignment have also been proposed (Budowski-Tal et al. 2010; Carugo and Pongor 2002; Rogen and Fain 2003; Zhi et al. 2006). These methods use various ways to represent the protein molecules for example as paths or vectors, in order to use this representation to identify candidate sets of structural neighbors. Some other algorithms also step away from distance-based criteria by focusing on statistical distributions of local distances or by comparing and classifying proteins on the basis of their topological properties. The drawback of these approaches is that they can lead to loss of secondary structure information and less accurate results. This happens because in order to perform faster, they use an approximate representation of the protein molecule, ignoring the fact that utilizing secondary structure information aids in filtering out noisy solutions and achieving efficiency and robustness (Dror et al. 2003).

A parameter that is also taken into account during the study of 3D protein structures is sequentiality, which means that subsequent amino acids in one protein must correspond to subsequent amino acids in the partner protein. The majority of methods follow this restriction while the number of methods that are non-sequential is still limited. Some of

the methods included in the non-sequential category are CA, MULTIPROT, SCALI and GANGSTA+ (Bachar et al. 1993; Guerler and Knapp 2008; Shatsky et al. 2004; Yuan and Bystroff 2005). The drawback of sequential approaches is that they can decrease the possibility to discover evolutionary relationships (Xie and Bourne 2008) as new protein structures can arise from the combination and permutation of substructures of a protein (structural rearrangements) (Bashton and Chothia 2007; Fong et al. 2007). In this way, they fail to capture similar structures with extensive conformation changes such as internal rearrangements, which means that these methods ignore the flexibility of the polypeptide chains (Zhi et al. 2006).

Certain other issues also arise when structural comparison of proteins is studied such as pairwise and multiple comparison. The result of a pairwise alignment or comparison procedure concerns two particular proteins, while the multiple comparison procedure gives similarity results between a certain protein and a list of other known proteins. Another term called 'dynamics-based-alignment' has also been recently introduced (Zen et al. 2008) and is intended to compare the dynamic motions of different proteins, such as the ALADYN approach (Potestio et al. 2010). Furthermore, methods that perform structural alignment between a model protein and the true known structure of the protein have also been proposed (Ortiz et al. 2002) and they are known as model comparison methodologies, but these last two subjects are out of the scope of the current paper.

In this paper we present in detail our prototype algorithm that performs 3D protein structure comparison at the level of C-alpha atoms and aims at detecting similarity between a query protein and a set of one or more other known protein structures. In contrast to many of the aforementioned algorithms, ours does not perform an alignment and in the same time it uses an exact, but simplified representation of the protein molecule (without loss of secondary structure information) and a robust indexing technique to store and classify the data.

In particular, in our framework each protein molecule is represented as a sequence of weights, which come up from the combination of two measures. If we suppose for example that $n$ is the number of C-alpha atoms in the protein molecule, then we compute (a) the $n-1$ distances between consecutive C-alpha atoms and (b) the $n-2$ cosines of their associated turning angles, which according to Zhi et al. (2006) consist a sufficient angle descriptor and no use of torsion angles is needed. Then we combine these two quantities $n-1$ times in order to produce a series of the hybrid measure $w\,(w_1, w_2, \ldots, w_{n-1})$, which, as will be proved on the next section, satisfies the triangular inequality. In this way, local secondary structural information is maintained and also the use of the DIStance and COsine measures, from which our algorithm's name (DISCO) is produced, ensures that the algorithm is rotation and translation independent.

On the next step, we divide this sequence of weights into overlapping subsequences, which correspond to all possible consecutive protein substructures of the molecule. After repeating the same procedure for all proteins, we make use of an indexing technique (M-tree) in order to store these data and search for similarities either between pairs of proteins (Pair-DISCO Algorithm), either between a particular query protein and a set of other proteins (Multi-DISCO algorithm). The use of overlapping subsequences enables us to detect not only similar structures but also similar substructures (motifs) in protein molecules, without thinking about sequentiality, which means about the order of appearance of the substructures into the molecule. This is very important because it means that we can capture both local and global similarity of protein structures and also because it enriches our algorithm with flexibility.

In fact, the use of indexing technique in our approach helps us to construct a database of protein structures in order to efficiently support and handle similarity queries of a specific

protein across the totality of proteins that are stored in the determinate database. As will be shown in the paper, the indexing method calculates accurate similarities without having access to the whole database. In this way, the searching procedure becomes more efficient and rapid providing also dynamic data handling.

Until now, DISCO is the only algorithm that combines and supports so many options and characteristics such as pairwise and multiple comparisons, motif extraction, translation-rotation independence, non-sequentiality, flexibility, detection of global and local similarity, absence of any kind of alignment, maintenance of secondary structure information and in the same time using a robust indexing technique with a distance-based similarity measure that satisfies triangular inequality. For this reason it is difficult to be compared with other methods that do not perform so many procedures simultaneously. Nevertheless, in this paper we are compared with two representative and most close to our algorithm's philosophy methods that perform pairwise and multiple comparisons respectively.

The rest of the paper is organised as follows. In Sects. 2 and 3 our proposed approach is described. Its experimental evaluation is presented in Sect. 4 and discussion follows in Sect. 5.

## 2 The proposed approach

### 2.1 Data representation

Let $P$ be a set of proteins, and let a protein $P_a \in P$, and its *length* $n = |P_a|$, which is defined as the total number of C-alpha atoms that it contains. For the protein $P_a$ let $x[1 \ldots n]$, $y[1 \ldots n]$, $z[1 \ldots n]$ be the corresponding 3D coordinates of the C-alpha atoms that it contains in the Euclidean three dimensional space, following the order of the protein structure.

We compute the $n-1$ Euclidean distances between consecutive C-alpha atoms using the equation:

$$d_i = \sqrt{(x[i]-x[i-1])^2 + (y[i]-y[i-1])^2 + (z[i]-z[i-1])^2} \tag{1}$$

for $i = 2, \ldots, n$. Note that $d_1 = 0$.

At the following, we compute the $n-2$ convex angles $\theta_i \in [0, \pi]$ between the linear vector segments $\mathbf{u}$, $\mathbf{v}$ associated to the secondary structures of proteins, using the cosine formula:

$$\cos\theta_i = \frac{\mathbf{u}\cdot\mathbf{v}}{|\mathbf{u}||\mathbf{v}|} \tag{2}$$

where,

$$\begin{aligned}
\mathbf{u}\cdot\mathbf{v} &= (x[i-1]-x[i])\,(x[i+1]-x[i]) + (y[i-1]-y[i])\,(y[i+1]-y[i]) \\
&\quad + (z[i-1]-z[i])(z[i+1]-z[i]) \\
|\mathbf{u}| &= \sqrt{(x[i-1]-x[i])^2 + (y[i-1]-y[i])^2 + (z[i-1]-z[i])^2} \\
|\mathbf{v}| &= \sqrt{(x[i+1]-x[i])^2 + (y[i+1]-y[i])^2 + (z[i+1]-z[i])^2}
\end{aligned}$$

for $i = 2, \ldots, n-2$. Note that $\theta_1 = \theta_{n-1} = 0$.

We compute the convex angles between all the protein atoms sequentially according to the Protein Data Bank file format (Berman 2007). In this way we can capture the struc-

tural characteristics of all the recorded side-chains of the proteins in their regular backbone structure.

For the protein $P_a$ we normalize the calculated distances $d_i$ and angles $\theta_i$ in the interval [0, 1]:

$$nd_i = \frac{d_i}{max_i\{d_i\}} \tag{3}$$

$$n\theta_i = \frac{\theta_i}{\pi} \tag{4}$$

and we calculate the following combined linear measure for the C-alpha protein atoms:

$$w_i = a \cdot nd_i + (1 - a) \cdot n\theta_i \tag{5}$$

for $i = 1, \ldots, n-1$, and for a selected real number $a \in [0, 1]$, which defines how much the distances or the angles will affect the final calculated weights $w_i$ between the protein C-alpha atoms.

**Proposition 1** *The combined linear measure $w_i = a \cdot nd_i + (1-a) \cdot n\theta_i$ satisfies the metric space properties, which are symmetry, positivity and triangular inequality.*

*Proof* The cosine similarity (Eq. 2) is a measure of similarity between two vectors $\mathbf{u}$, $\mathbf{v}$, but it does not satisfy the metric properties (symmetry, positivity, triangular inequality). From the cosine similarity, the angular similarity is derived, which expresses the normalized angle between the vectors and it is a similarity function within [0, 1]:

$$1 - \frac{\theta_i}{\pi}, \qquad \text{where} \qquad \theta_i = \cos^{-1}\left(\frac{\mathbf{u} \cdot \mathbf{v}}{|\mathbf{u}| \cdot |\mathbf{v}|}\right)$$

The advantage of the angular similarity is that when used as a distance measure (by subtracting it from 1), it is a proper distance metric (it satisfies the metric space properties) (Wikipedia http://en.wikipedia.org). Therefore, the corresponding distance:

$$1 - \left(1 - \frac{\theta_i}{\pi}\right) = \frac{\theta_i}{\pi} = n\theta_i$$

is a metric distance function, returning values into [0, 1].

Moreover, the distances $d_i$ are calculated using the Euclidean distance metric $L_2$, thus the normalized distance:

$$nd_i = \frac{d_i}{\max_i\{d_i\}}$$

satisfies the metric space properties, and is a proper metric distance function, returning values into [0, 1].

Therefore, we have two norms (metric distances) $nd_i$ and $n\theta_i$ which return values into [0, 1]. Also, it is well known (http://en.wikipedia.org), that the linear combination:

$$a \cdot nd_i + (1 - a) \cdot n\theta_i$$

where $a \in [0, 1]$ is also a norm (metric distance function), i.e. satisfies the metric space properties.

Finally, we construct the protein's data as a sequence of the weights $w_i$:

$$P_a \longrightarrow (w_1, w_2, \ldots, w_{n-1})$$

Henceforth, all proteins of $P$ will be represented by this sequence of weights.

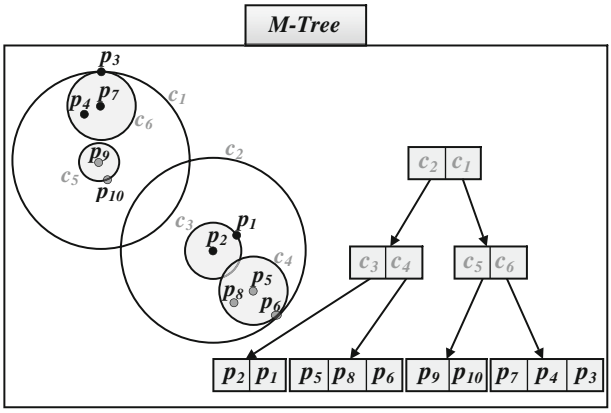


**Fig. 1** Data indexing with the M-Tree

### 2.2 Data indexing

As already mentioned, structure of biological molecules is a very important clue to understanding biological functions and for this reason robust tools for manipulating protein shapes are needed. Among the existing indexing tools for metric spaces, we use the Metric-Tree (M-tree) (Ciaccia et al. 1997) to index the protein data, as it combines both efficient query processing and dynamic data handling. The M-Tree is a balanced tree that can index objects with attributes in a metric space, compared by distance functions which satisfy the metric properties (symmetry, positivity, triangular inequality). A simple example of such an indexing is depicted in Fig. 1. As in any tree-based data structure, the M-tree is composed of nodes and leaves. In each node there is a data object that identifies it uniquely and a pointer to a sub-tree where its children reside. Every leaf has several data objects. For each node there is a radius ($r$) that defines a circle in the desired metric space, as shown in Fig. 1. Thus, every node $c_i$ and leaf $p_i$ residing in a particular node $C$, is at most distance $r$ from the node that belongs to ($C$), and every node $c_i$ and leaf $p_i$ with node parent $C$ keeps the distance from it.

Using the M-tree, we have the advantage of inserting, updating and deleting dynamically new and old protein structures, and supporting efficiently nearest neighbors and range queries. Moreover, with the defined similarity (or distance) measures, which satisfy the metric space properties, we can support similarity queries between large protein structures, by partitioning the proteins into sub-proteins and using incremental nearest neighbor queries for ranking. The next section describes the proposed methodology for such similarity queries.

## 3 Algorithm's description

### 3.1 Top-$k$ protein-to-protein similarity query (Multi-DISCO algorithm)

Let $P$ be a set of proteins. Each protein $P_i \in P$ is divided into $|P_i| - T + 1$ sub-proteins , with length $T$, which are indexed in an M-tree file. Let us consider an example. If a protein $P_i$ has length 10, depicted by the series [1 2 3 4 5 6 7 8 9 10] and $T = 3$ then the set of $|P_i| - T + 1$ sub-proteins that our algorithm will create will be the following: ([1 2 3], [2 3 4], [3 4 5], [4 5 6], [5 6 7], [6 7 8], [7 8 9], [8 9 10]). We are interested in ranking all

proteins in comparison with a given query protein $P_q$, and to retrieve the best top-$k$ results. The idea is as follows.

For any sub-protein $S_j$ of $P_q$ we retrieve incrementally its nearest neighbor sub-proteins. For that purpose we use an *M-tree cursor*, which we initialize to the first nearest neighbor of $S_j$, and we retrieve the next nearest neighbors of $S_j$ incrementally. An M-tree cursor (which has been implemented in the M-Tree project http://www-db.deis.unibo.it/Mtree/), provide sorted access from a specific object to the nearest neighbor order from that object and retrieve incrementally its nearest neighbors one by one. All existing sub-proteins into the M-tree will have a specific nearest neighbor order position from $S_j$. However, we do not leave the M-tree cursor to scan the entire indexed data-set by retrieving all nearest neighbors. When we retrieve at least one sub-protein from all existing proteins, we stop scanning the nearest neighbor order with the cursor. For any retrieved sub-protein we record its nearest neighbor order position as a ranking value of its corresponding protein. We repeat this process for any existing sub-protein $S_j$ of $P_q$ and we keep the ranking results in a table. Then, the final rankings of the proteins are computed in another table $R$ by taking the minimum or the maximum or the average of the recorded corresponding positions. The top-$k$ similar proteins are extracted from the table $R$ after an ascending sort. It is important to mention here that we can tune the parameter $T$, which is the length of each sub-protein, in order to have small or large sub-proteins. In this way we can detect either local similarity (small sub-proteins), either global similarity (large sub-proteins). Figure 2 presents the outline of the proposed algorithm for a top-$k$ protein-to-protein similarity query.

### 3.2 Pairwise similarity query (Pair-DISCO algorithm)

The pairwise similarity query is performed in a straightforward way. Each protein $P_i$ is again divided into $|P_i| - T + 1$ sub-proteins, as described before. The idea is to take any sub-protein $S1_i$ from the first protein $P_1$ and to compute its distance to any sub-protein $S2_j$ from the second protein $P_2$. Then, using a specific rule we compute the average of the minimum or maximum or average distances and return this value as a final score. When this score is close to 0, the proteins are similar. When this score is close to 1 the proteins are dissimilar. It is important to note that we retrieve the data of the proteins $P_1$, $P_2$, from the M-tree into memory, before the distance computations. This leads to a significant improvement in the performance of the query. In Fig. 3 we present the outline of this algorithm.

### 3.3 Motif extraction (Sub-structure query)

As already mentioned before, we divide each protein $P_i \in P$ into $|P_i| - T + 1$ sub-proteins, with length $T$. In this way, motif extraction is achieved as an intermediate step of Multi-DISCO algorithm, where we detect at least one sub-protein from all existing proteins to be the most similar to the each time examinant sub-protein. Because sub-structures of proteins are overlapping, for example if the first sub-structure is [1 2 3 4 5 6 7 8 9 10] then the second one will be [2 3 4 5 6 7 8 9 10 11], we have ranking results for all possible sub-proteins of the molecule. An example is shown in Table 1. Let us consider a dataset of 30 proteins. The numbers in the first column represent the protein and its sub-protein respectively. The second column contains the most similar sub-proteins from all twenty proteins of the example dataset in a row. That is to say, the first number of the second column corresponds to the most similar sub-protein of the first molecule, the second number (which is 120) corresponds to the most similar sub-protein of the second protein and so on.

**Algorithm** Multi-DISCO
**Input**
$P_q$: query protein
*rule*: min, max, average
$k$: number or top results
**Output**
$R[1...|P|]$: final rankings table
**Output to File**
$Pos[1...|S|, 1...|P|]$: sub-proteins positions table

01. initialize rankings table $R[]$ and positions table $Pos[,]$.
02. **for** any sub-protein $S_i$ of $P_q$, $(i = 1, ..., |P_q| - T + 1)$ **do**
03. allocate an M-Tree cursor $C$ to $S_i$ for incremental retrieval
04. $h = 1$ (initialize NN-position)
05. **while** exist any $Pos[S_i, P_j] = 0$, $(j = 1, ..., |P|)$ **do**
06. $S_j = C$.Get-Next-NN-Object()
07. $P_j$ = protein-ID of sub-protein $S_j$
08. **if** $Pos[S_i, P_j] = 0$ **then** $Pos[S_i, P_j] = h$
09. $h = h + 1$
10. **end-while**
11. **end-for**
12. **for** $j = 1, ..., |P|$ **do**
13. **if** $rule = \min$ **then**
$R[j] = \min_i\{Pos[S_i, P_j]\}$, $\forall i = 1, ..., |P_q| - T + 1$
14. **if** $rule = \max$ **then**
$R[j] = \max_i\{Pos[S_i, P_j]\}$, $\forall i = 1, ..., |P_q| - T + 1$
15. **if** $rule =$ average **then**
$R[j] = \frac{1}{|P_q|-T+1} \sum_{i=1}^{|P_q|-T+1} Pos[S_i, P_j]$
16. **end-for**
17. sort ascending the final rankings $R[]$ using an ID-index
18. return the Top-$k$ scores with their corresponding protein-ID's.

**Fig. 2** Outline of the Multi-DISCO algorithm (Top-$k$ protein-to-protein similarity query algorithm)

## 4 Experimental procedure

### 4.1 Parameters

It is known by this time that our methodology makes use of two parameters: $a$ and $T$. The first one is a real number between 0 and 1 which specifies how much the distances or the angles affect the final score and the second one refers to the sub-protein length. We focused on several groups of proteins that were previously used as test cases, e.g. Konagurthu et al. (2006), Micheletti and Orland (2009), in order to "train" our algorithm in terms of $a$ and $T$.

We tested parameter $a$, using at the same time various values of $T$ expressed as a percentage of the average protein length of the examined dataset. Specifically, we tested every value of $a$ with step 0.01 across the values of $T$ that correspond to a percentage [5–100 %] of the average protein length with step 5 %. For example, if the average protein length of a dataset is equal to 500, then we set the parameter $T$ equal to $5\,\% \times 500 = 25$, $10\,\% \times 500 = 50, \ldots$, etc. Precision was computed as the ratio of similar proteins in the top-10 ranking list to 10. Figure 4a depicts the average precision values for $a \in [0, 0.05]$ across parameter $T$,

**Algorithm** Pair-DISCO
**Input**
$P_1, P_2$: query proteins
$rule$: min, max, average
**Output**
$r$: final score

01. retrieve protein data of $P_1, P_2$ from M-tree.
02. $fdmin = fdmax = fdavg = 0$
03. **for** any sub-protein $S1_i$ of $P_1$, $(i = 1, ..., |P_1| - T + 1)$ **do**
04. $\quad dmin = \infty, dmax = 0, davg = 0$
05. $\quad$ **for** any sub-protein $S2_j$ of $P_2$, $(j = 1, ..., |P_2| - T + 1)$ **do**
06. $\quad\quad d = \text{distance}(S1_i, S2_j)$
07. $\quad\quad$ **if** $d < dmin$ **then** $dmin = d$
08. $\quad\quad$ **if** $d > dmax$ **then** $dmax = d$
09. $\quad\quad davg = davg + d$
10. $\quad$ **end-for**
11. $\quad davg = davg/(|P_2| - T + 1)$
12. $\quad fdmin = fdmin + dmin$
13. $\quad fdmax = fdmax + dmax$
14. $\quad fdavg = fdavg + davg$
15. **end-for**
16. $fdmin = fdmin/(|P_1| - T + 1)$
17. $fdmax = fdmax/(|P_1| - T + 1)$
18. $fdavg = fdavg/(|P_1| - T + 1)$
19. **if** $rule = \min$ **then** return $fdmin$
20. **if** $rule = \max$ **then** return $fdmax$
21. **if** $rule = \text{average}$ **then** return $fdavg$

**Fig. 3** Outline of the Pair-DISCO algorithm (Pairwise similarity query algorithm)

concerning a specific representative protein dataset. From Fig. 4 we see that as $a$ increases the performance of the algorithm decreases and for this reason we set the parameter $a$ in the interval [0, 0.05]. Also, we can see that best performance is achieved when $T$ equals the [20–30 %] of the average protein length. Similar results were obtained with other representative datasets as well, so consequently we decided to set our parameters $a = 2\,\%$ and $T = 20\,\%$. We chose a very low value for $a$, because distance between two C-alpha atoms is almost constant for the majority of proteins, but it can still be usefull and make the difference when we are talking about exact secondary structure information. $T$ was set to 20 % because this was the lowest value giving the best results. Of course these values can be changed and adapt to the each time user needs. For example parameter $a$ can be set equal to 0 for those who don't want to use it at all. We also conducted the same set of experiments for each proposed rule (Fig. 2, steps 13–15). Best results were derived using the min rule, which are the ones shown in Fig. 4. The same applies also when pairwise comparison is taking place.

## 4.2 Results

In the first section we talked about our algorithm's distinctiveness of combining many attributes together, about the reason why it is different from methods that have been proposed so far and about why it is difficult to be compared with these methods. We finally

**Table 1** Example of sub-structure (motif) query

| | |
|---|---|
| 1-1: | 1 120 6 45 103 1774 182 921 62 38 50 125 29 8 12 13 7 49 20 9 |
| 1-2: | 1 80 21 4 48 1028 212 450 88 23 14 78 11 36 13 28 310 34 5 |
| 1-3: | 1 56 16 4 52 1016 147 423 122 34 58 43 18 27 19 20 11 10 25 3 |
| 1-4: | 1 86 3 33 43 1143 174 298 11 101 4 7 10 8 23 52 13 32 39 20 |
| 1-5: | 1 66 22 25 19 866 67 136 10 74 13 15 16 5 24 30 4 96 61 8 |
| 1-6: | 1 59 23 13 21 1166 61 258 24 117 5 16 26 4 14 17 6 15 60 20 |
| 1-7: | 1 29 24 60 46 1126 144 226 32 193 13 47 58 9 10 6 5 12 27 18 |
| 1-8: | 1 8 17 108 65 1399 96 438 66 194 48 11 146 22 12 7 5 4 21 19 |
| 1-9: | 1 4 9 51 75 1461 55 406 77 165 33 11 159 32 28 12 2 8 10 40 |
| 1-10: | 1 2 11 35 25 1081 20 303 24 157 22 7 91 46 10 5 8 9 3 21 |
| 1-11: | 1 7 4 6 23 650 8 248 18 144 63 2 38 70 20 10 11 3 13 32 |
| 1-12: | 1 7 5 3 10 473 11 347 103 194 37 4 8 34 12 16 30 2 19 55 |
| 1-13: | 1 11 6 5 9 271 15 260 104 230 47 3 10 8 14 4 36 2 24 34 |
| 1-14: | 1 11 6 5 8 263 13 282 127 339 53 4 15 10 16 2 41 3 24 44 |
| 1-15: | 1 26 2 6 29 357 58 289 63 345 16 4 10 48 24 7 19 3 21 23 |
| 1-16: | 1 78 3 59 42 468 56 376 8 282 22 6 7 37 29 13 2 12 33 27 |
| 1-17: | 1 61 3 67 37 509 26 402 6 104 22 10 7 34 14 32 2 19 15 17 |
| 1-18: | 1 84 4 75 81 330 44 446 20 95 49 16 8 40 5 9 2 12 15 23 |
| 1-19: | 1 35 2 58 141 160 57 143 41 28 27 10 38 21 8 7 3 4 77 6 |
| 1-20: | 1 22 17 69 127 858 49 160 40 14 59 15 28 11 4 3 12 2 71 10 |
| 1-21: | 1 11 5 14 94 587 59 224 76 29 55 13 51 24 4 2 12 3 116 6 |
| 1-22: | 1 10 15 24 160 340 101 244 67 25 12 3 27 32 17 2 19 11 31 20 |
| 1-23: | 1 12 16 5 90 432 53 145 10 11 44 40 93 46 19 4 6 58 26 28 |
| 1-24: | 1 26 16 8 91 272 36 152 7 12 34 43 30 151 18 4 9 40 22 69 |
| 1-25: | 1 33 16 8 50 353 31 287 7 11 44 22 35 64 18 4 3 32 21 60 |
| 1-26: | 1 74 21 13 70 669 46 328 3 9 47 18 37 33 16 4 5 28 15 75 |
| 1-27: | 1 3 26 14 41 903 84 552 8 22 4 7 16 36 109 9 28 15 58 24 |
| 1-28: | 1 25 10 28 85 650 65 294 3 19 4 14 17 7 37 2 74 12 43 23 |
| 1-29: | 1 55 10 90 82 1556 84 689 39 17 9 14 11 35 68 5 16 179 61 6 |
| 1-30: | 1 36 3 58 56 1674 207 723 31 14 24 69 9 61 33 7 71 105 27 2 |

chose two algorithms in order to perform pairwise and multiple comparison with ours. We use the FATCAT-pairwise algorithm (Ye and Godzik 2003) to perform pairwise comparisons, which has been proved to provide good results between both flexible and rigid structures and also to outperform other known classical algorithms, such as RMSD and the ones provided by DALI and CE (Holm and Sander 1993; Veeramalai et al. 2008). We also use the CURVE algorithm (Zhi et al. 2006) to perform multiple comparisons as it uses a representation similar to ours and has also been proved to outperform many known algorithms (Can and Wang 2004).

### *4.2.1 Pairwise comparison*

As already mentioned, proteins that belong to the same family (a superfamily or a sub-family) typically have similar three-dimensional structures and functions. We first used Pair-DISCO algorithm to compare pairs of proteins that belong to same protein families and same protein

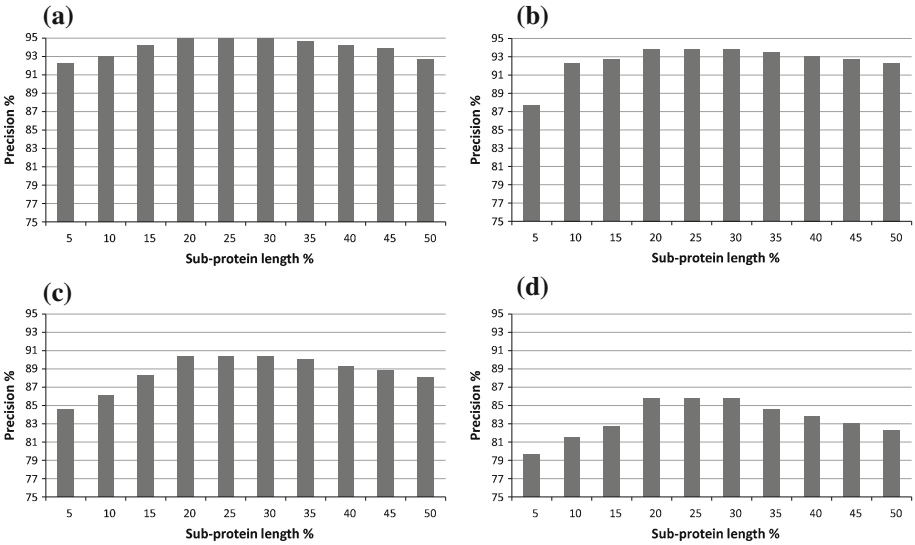


**Fig. 4** Average precision versus sub-protein length for $a$ values: **a** [0–0.05], **b** [0.05–0.1], **c** [0.1–0.15], **d** [0.15–0.2], with step 0.01

sub-families. Similarly we applied the algorithm to compare proteins that belong to different protein families. In order to be compared with FATCAT-pairwise, which reports its similarity measured as a P-value, we performed extensive experimentation to set a threshold for Pair-DISCO algorithm as well, with a view to distinguish significant similar proteins from simply similar ones. In general, values near zero indicate similar proteins. Significant similarity though is detected when the value of Pair-DISCO result is under 0.05.

We then applied Pair-DISCO algorithm in a representative sequence independent dataset (Fischer et al. 1996), obtained by using the Protein Data Bank (Berman 2007) and taking into account only structural criteria. Each structural protein family is equally represented in this dataset and also every chain within it has $< 30\,\%$ sequence identity.

Table 2 summarizes the results of the Pair-DISCO algorithm. The first two columns contain the PDB id of the compared proteins and the third column shows the obtained similarity score. The forth column uses the aforementioned defined threshold in order to discriminate the significant similarities. On the fifth column we appose the corresponding results from FATCAT-pairwise.

In our results we can see that proteins with PDB ids 1onc and 7rsa, which correspond to proteins P-30 and RNase-A respectively, are not signed to be significantly similar. This happens because the two proteins may be part of the same protein family (RNase), but the first one was found to perform functions and activities, that the second one does not possess (Mosimann et al. 1994). On the other hand, proteins with PDB ids 1gal and 3cox were found to be significantly similar and this is true because they are enzymes that belong to the same category (oxidases), which constitutes a subclass of the oxidoreductases protein family. They also perform a similar function which is to catalyze an oxidation-reduction reaction, involving molecular oxygen ($O_2$) as the electron acceptor. Generally, Pair-DISCO algorithm provides quite accurate results, similar to those of the FATCAT-pairwise algorithm and sometimes more precise ones, taking into account that FATCAT-pairwise considers two proteins to be significantly similar if the P-value is lower than 0.05, as well.

**Table 2** Pairwise results of Pair-DISCO algorithm

| Protein1 | Protein2 | Pair-DISCO | Signif. similar | FATCAT-pairwise |
|---|---|---|---|---|
| 1npx | 3grs | 0.0198731 | Yes | Yes |
| 1onc | 7rsa | 0.0809089 | No | Yes |
| 1osa | 4cpv | 0.0295322 | Yes | Yes |
| 2cmd | 6ldh | 0.0510457 | No | Yes |
| 1aba | 1ego | 0.0567986 | No | Yes |
| 1eaf | 4cla | 0.0234494 | Yes | Yes |
| 2sga | 4ptp | 0.0321825 | Yes | Yes |
| 1aaj | 1paz | 0.0259216 | Yes | Yes |
| 5fd1 | 2fxb | 0.0753062 | No | Yes |
| 1gal | 3cox | 0.0240907 | Yes | Yes |
| 1tlk | 2rhe | 0.0306053 | Yes | Yes |
| 1omf | 2por | 0.0508227 | No | Yes |
| 8i1b | 4fgf | 0.0645641 | No | Yes |
| 1mup | 1rbp | 0.0321935 | Yes | Yes |
| 1arb | 4ptp | 0.0330052 | Yes | Yes |
| 2pia | 1fnr | 0.0267139 | Yes | Yes |
| 3cd4 | 2rhe | 0.0690796 | No | Yes |
| 2mnr | 4enl | 0.0190777 | Yes | Yes |
| 2gbp | 2liv | 0.0325059 | Yes | Yes |
| 1fxiA | 1ubq | 0.03846 | Yes | Yes |
| 1ten | 3hhrB | 0.069999 | No | No |
| 2azaA | 1paz | 0.034002 | Yes | Yes |
| 1cewI | 1molA | 0.032229 | Yes | Yes |
| 1cid | 2rhe | 0.040133 | Yes | Yes |
| 1crl | 1ede | 0.031879 | Yes | Yes |
| 2sim | 1nsbA | 0.04458 | Yes | Yes |
| 1bgeB | 2gmfA | 0.056033 | No | Yes |
| 1tie | 4fgf | 0.033885 | Yes | Yes |

#### *4.2.2 Multiple comparison*

The authors in Zhi et al. (2006) proved that turning angles can be a representative descriptor of protein structures without any use of torsion angles and in our theory we adopted this idea. The basic differences though between CURVE and Multi-DISCO algorithm is that CURVE performs alignment and makes use of SSEs, while our algorithm doesn't and in this way, our method has an advantage over CURVE. Since we can not be compared in terms of alignment, we used the top-$n$ ranking list that these two algorithms produce and computed the precision measure. In the majority of cases Multi-DISCO is superior as it uses a precise and not an approximate protein molecule representation and of course because it uses a combination of attributes to detect protein similarities (see Sect. 1).

For our experiments we used representative datasets that were produced by utilizing the PDB select tool (Griep and Hobohm 2010) and especially the new—August 2012 version.

PDBselect (http://bioinfo.tg.fh-giessen.de/pdbselect/) provides the user with lists of representative protein chains with low mutual sequence identity ($< 25\,\%$) selected from the protein data bank (Berman 2007), which also includes gap-junction and ion-channel proteins (http://en.wikipedia.org). The user can also handle preselected ids and chains in order to get a representative set of PDB chains. In this way, we created several datasets of 100 representative structures (Can and Wang 2004; Zhi et al. 2006) from a database provided by the PDBfilter-select. Each of these structures was used as a query to search against the dataset and the precision measure was applied to evaluate the results. In our case precision is defined as the ratio of similar proteins in the top-$n$ ranking list to $n$. We used the top-1 and top-3 ranking lists to evaluate the results, because in representative datasets there are at most 3 proteins from the same family. We must also mention here that we include every time the query proteins in the examined datasets for reasons of verification.

The results of our experiments, which include the average values of precision for several random representative protein datasets are summarized as follows. In all cases both algorithms ranked the query protein correctly first in all of the top-1 lists (100 % precision). Things change though in top-3 ranking lists, were our method has again a performance of 100 % precision contrary to CURVE algorithm, which in that case has a performance of 86.4 % precision.

## 5 Discussion

In this paper we studied one of the most important issues in biology, which is to describe and compare biological structures. For this reason, we used a simple and comprehensive representation of protein data, which in the same time maintains all the necessary structural information of the data. Since these structures help us to understand and manipulate biological functions, consequently we need robust tools for comparing and classifying the universe of protein shapes. Towards this direction, we proposed two schemes (Multi-DISCO and Pair-DISCO algorithms) which are both based on the M-tree access method (Ciaccia et al. 1997).

Apart from the advantages of M-tree that we mentioned in Sect. 2.1, we need to recall that the M-tree is already equipped by the necessary tools to handle queries, as it has been reported in Ciaccia et al. (1997). The only requirement for the M-tree to work properly is that the distance used must satisfy the metric space properties. Since our metric measure satisfies these properties as shown in Sect. 2.1, it can be used as distance measure in the M-tree. Of course, other secondary memory schemes for metric spaces or any other metric access method could be applied equally well (Guttman 1984; Traina et al. 2000), but note that among all the metric indexing schemes we choose the M-tree because of its simplicity.

To sum up, we extensively presented our methodology for detecting similarity among or between 3D protein structures using a robust indexing technique. The novelty of our method is summarized as follows: it supports both pairwise/multiple comparisons and motif extraction, it is translation-rotation independent and non-sequential, it can detect both global and local similarity, it maintains the secondary structure information and flexibility of the protein molecule without performing any kind of alignment and in the same time it uses a robust indexing technique with a distance-based similarity measure that satisfies triangular inequality.

Due to our algorithm's distinctiveness it was difficult for our method to be compared with others. Nevertheless, we used two representative and well-known algorithms, one for pairwise comparison and one for multiple comparison, which as was shown in the paper, proved to be inferior to our method.

## References


Alexandrov NN (1996) SARFing the PDB. Protein Eng 9:727–732

Bachar O, Fischer D, Nussinov R, Wolfson H (1993) A computer vision based technique for 3D sequence-independent structural comparison of proteins. Protein Eng 6:279–288

Bashton M, Chothia C (2007) The generation of new protein functions by the combination of domains. Structure 15:85–99

Berman HM et al (2007) The protein data bank. Nucleic Acids Res 28:235–242

Budowski-Tal I, Nov Y, Kolodny R (2010) FragBag, an accurate representation of protein structure, retrieves stuctural neighbors from the entire PDB quickly and accurately. Proc Natl Acad Sci USA 107:3481–3486

Can T, Wang YF (2004) Protein structure alignment and fast similarity search using local shape signatures. J Bioinform Comput Biol 2:215–239

Carugo O, Pongor S (2002) Protein fold similarity estimated by a probabilistic approach based on C(alpha)-C(alpha) distance comparison. J Mol Biol 315:887–898

Ciaccia P, Patella M, Zezula P (1997) M-tree: an efficient access method for similarity search in metric spaces. In: Proceedings of the 23rd international conference on very large databases (VLDB)

Cohen FE, Sternberg MJE (1980) Use of chemically derived distance constraints in the prediction of protein structure with myoglobin as an example. J Mol Biol 137:9–22

Dror O, Benyamini H, Nussinov R, Wolfson HJ (2003) Multiple structural alignment by secondary structures: algorithm and applications. Protein Sci 12:2492–507

Fischer D, Elofsson A, Rice D, Eisenberg D (1996) Assessing the performance of fold recognition methods by means of a comprehensive benchmark. In: Pacific symposium on biocomputing, pp 300–318

Fong JH, Geer LY, Panchenko AR, Bryant SH (2007) Modeling the evolution of protein domain architectures using maximum parsimony. J Mol Biol 366:307–315

Gan HH et al (2002) Analysis of protein sequence/structure similarity relationships. Biophys J 83:2781–2791

Gibrat JF, Madej T, Bryant SH (1996) Surprising similarities in structure comparison. Curr Opin Struct Biol 6:377–385

Griep S, Hobohm U (2010) PDBselect 1992–2009 and PDBfilter-select. Nucleic Acids Res Database Issue 38:318–319

Guerler A, Knapp EW (2008) Novel protein folds and their nonsequential structural analogs. Protein Sci 17:1374–1382

Guttman A (1984) R-trees: a dynamic index structure for spatial searching. In: Proceedings of the ACM SIGMOD conference, p 4757

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

Koehl P (2001) Protein structure similarities. Curr Opin Struct Biol 11:348–353

Kolbeck B, May P, Schmidt-Goenner T, Steinke T, Knapp EW (2006) Connectivity independent protein-structure alignment. BMC Bioinform 7:510–510

Krissinel E, Henrick K (2004) Secondary-structure matching (SSM), a new tool for fast protein structure alignment in three dimensions. Acta Crystallogr 60:2256–2268

Konagurthu AS, Whisstock JC, Stuckey PJ, Lesk AM (2006) MUSTANG: a multiple structural alignment algorithm, proteins: structures. Funct Bioinform 64:559–574

Lesk AM (2004) Introduction to protein science: architecture, function and genomics. Oxford University Press, Oxford

Lichtarge O, Sowa ME (2002) Evolutionary predictions of binding surfaces and interactions. Curr Opin Struct Biol 12:21–27

Lupyan D, Leo-Macias A, Ortiz AR (2005) A new progressive-iterative algorithm for multiple structure alignment. Bioinformatics 21:3255–3263

Madej T, Gibrat JF, Bryant SH (1995) Threading a database of protein cores. Proteins 23:356–369

Micheletti C, Orland H (2009) MISTRAL: a tool for energy-based multiple structural alignment of proteins. Oxf Univ Press 20:2663–9

Mosimann SC, Ardelt W, James MNG (1994) Refined 1.7 a X-ray crystallographic structure of P-30 protein, an amphibian ribonuclease with anti-tumor activity. J Mol Biol 236:1141–1153

Needleman SB, Wunsch CD (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. J Mol Biol 48:443–453

Ortiz AR, Strauss CEM, Olmea O (2002) MAMMOTH: an automated method for model comparison. Protein Sci 11:2606–2621

Park C, Park S, Kim D, Park S, Sung M, Lee H, Shin J, Hwang C (2006) Fast protein structure alignment algorithm based on local geometric similarity. In: MICAI 2006, LNAI 4293, pp 1179–1189

Potestio R, Aleksiev T, Pontiggia F, Cozzini S, Micheletti C (2010) ALADYN: a web server for aligning proteins by matching their large-scale motion. Nucleic Acids Res 38:W41–W45
Rogen P, Fain B (2003) Automatic classification of protein structure by using Gauss integrals. Proc Natl Acad Sci 100:119–124
Shapiro J, Brutlag D (2004) FoldMiner: structural motif discovery using an improved superposition algorithm. Protein Sci 13:278–294
Shatsky M, Nussinov R, Wolfson HJ (2004) A method for simultaneous alignment of multiple protein structures. Proteins 56:143–156
Shindyalov IN, Bourne PE (1998) Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. Protein Eng 11:739–747
Stivala AD, Stuckey PJ, Wirth AI (2010) Fast and accurate protein substructure searching with simulated annealing and GPUs. BMC Bioinform 11:446–463
Traina C, Traina AJM, Seeger B, Faloutsos C (2000) Slim-trees: high performance metric trees minimizing overlap between nodes. In: Proceedings of the seventh international conference on extending database technology (EDBT), pp 51–65
Veeramalai M, Ye Y, Godzik A (2008) TOPS++FATCAT: fast flexible structural alignment using constraints derived from TOPS+ Strings Model. BMC Bioinformatics 9:358
Xie L, Bourne PE (2008) Detecting evolutionary relationships across existing fold space. Proc Natl Acad Sci USA 105:5441–5446
Ye Y, Godzik A (2003) Flexible structure alignment by chaining aligned fragment pairs allowing twists. Bioinformatics 19:246–255
Yuan X, Bystroff C (2005) Non-sequential structure-based alignments reveal topology-independent core packing arrangements in proteins. Bioinformatics 21:1010–1019
Zen A, Carnevale V, Lesk AM, Micheletti C (2008) Correspondences between low-energy modes in enzymes: dynamics-based alignment of enzymatic functional families. Protein Sci 17:918–929
Zhi D, Krishna S, Cao H, Pevzner P, Godzik A (2006) Representing and comparing protein structures as paths in three-dimensional space. BMC Bioinform 7:460–475
Zhang L, Bailey J, Konagurthu AS, Ramamohanarao K (2010) A fast indexing approach for protein structure comparison. BMC Bioinform 11:S46