Incremental Mountain Clustering Method to Find Building Blocks for Constructing Structures of Proteins

Ken-Li Lin*, Chin-Teng Lin, *Fellow, IEEE*, and Nikhil R. Pal, *Fellow, IEEE*

***Abstract*—In this paper we propose an algorithm named Incremental Structural Mountain Clustering Method (ISMCM) with a view to finding a library of building blocks for reconstruction of 3-D structures of proteins/peptides. The building blocks are short structural motifs that are identified based on an estimate of local "density" of 3-D fragments computed using a measure of structural similarity. The structural similarity is computed after the best-molecular-fit alignment of pairs of fragments. The algorithm is tested on two well known benchmark data sets. Following the protocols used by other researchers, for the first data set we reconstruct a set of 71 test peptides (up to first 60 residues) whereas for the second data set we reconstruct all 143 test peptides. The ISMCM algorithm is found to successfully reconstruct the test peptides in terms of both global-fit root-mean-square (RMS) error and local-fit RMS error. The low values of local-fit RMS errors suggest that these building blocks extracted by ISMCM are good quantizers, which can represent nearby fragments quite accurately. To further assess the quality of building blocks we use two alternative graphical ways. We also use Shannon's entropy to show the structural similarity of the clusters found by our algorithm. This is important as building blocks that represent clusters with structurally similar fragments will be very effective in reconstruction. The entropic analysis reveals a very interesting fact that the secondary structure of the central residue of the fragments in a cluster is most strongly conserved (minimum entropy) over the cluster, which might be an indicator that central residue of the structural motif plays a dominant role in local folding.**

***Index Terms*—Building blocks, incremental structural mountain clustering, protein structure, structural mountain clustering.**

I. Introduction

THE knowledge of the 3-D structure of a protein helps biologists in many ways including studying the functions of proteins, drug designing, and designing of novel proteins [1]–[3]. However, experimental methods based on X-ray crystallography or nuclear magnetic resonance imaging for finding the 3-D structure of a protein are very time consuming and expensive. Hence many attempts have been made to find faster alternative approaches. In this direction significant efforts have been put by researchers to find the relations between protein sequences and their 3-D structures. The various approaches to solve this problem include comparative modeling [4], [5], fold recognition [6], [7], *ab initio* prediction [8]–[10], and 3-D building blocks approach [11]–[21].

Machine learning techniques have been extensively used to predict local secondary structure as well as tertiary structure of proteins [22]–[31]. These methods fall in the category of comparative modeling. Of the various machine learning tools, neural networks (NNs) have been quite successful for protein structure prediction [24]–[26]. Multilayer perceptron (MLP) network and radial basis function (RBF) network have been used extensively to protein fold determination [24], [25]. Pal and Chakraborty [24] have used RBF neural networks with different feature sets to predict protein folds. RBF networks are also used by Chung *et al.* [25] and Huang *et al.* [7] for the same problem. Support vector machines (SVMs) have also been used as a tool of choice by many researchers for protein fold prediction [25]–[29]. Recently, Ghanti and Pal [31] used many novel features in conjunction with a neural feature selection technique and used several machine learning tools for protein fold prediction. In [31], the fold prediction accuracy is further improved using classifier fusion techniques. A basic problem with such approaches is that unless the target protein is homologous or has some structural similarity, such a method may not be useful.

A family of methods uses building blocks or short structural fragments, which appear frequently in different proteins and exhibit some sequence to structure relation. These methods are not dependent on homology of the proteins [11]–[21]. Such a method uses a set of proteins with known 3-D structures and constructs a library of building blocks or structural motifs. These structural motifs are then used to construct/reconstruct structures of new proteins. If a short structural fragment appears repeatedly in different proteins and if it is represented by the same or similar sequence of residues, then we may assume that such a fragment is a stand-alone unit, which folds independently [21]. Thus, such fragments are good candidates to be building blocks which can help reconstruction of 3-D structure of unknown proteins. A two-stage clustering algorithm is proposed by Unger *et al.* [11]. In this method a fragment having the highest number of neighboring fragments

Manuscript received January 14, 2010; revised October 10, 2010; accepted November 13, 2010. Date of current version February 02, 2011. This work was supported in part by the Aiming for the Top University Plan of National Chiao Tung University, the Ministry of Education, Taiwan, under Contract 99W906, and in part by the National Science Council, Taiwan, under Contract NSC 99-3114-E-009-167. *Asterisk indicates corresponding author.*

*K.-L. Lin is with the Computer Center, Chung-Hua University, Hsinchu 300, Taiwan(corresponding author, e-mail: kennylin@chu.edu.tw).

C.-T. Lin is with the Department of Electrical & Control Engineering, National Chiao Tung University, Hsinchu 300, Taiwan (e-mail: ctlin@mail.cn.nctu.edu.tw).

N. R. Pal is with the Indian Statistical Institute, Calcutta 700108, India (e-mail: nikhil@isical.ac.in).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNB.2010.2095467

within a cutoff distance is considered a building block. The RMS deviation between two fragments, after aligning them to the best possible extent, is taken as the distance between two fragments. A similar approach was also used by Micheletti *et al.* but they considered the largest number of nearby points within a similarity cutoff called "proximity score" [13] to find cluster centers. A modified k-means algorithm, called simulated annealing k-means was used in [12] to extract clusters/building blocks with the minimal total variance score. The tripeptides analyzed by Anishetty *et al.* were shown to be feasible and could be used to predict plausible structures for oligopeptides [32]. On the other hand, a structure alphabet consisting of 16 protein blocks is used by de Brevern *et al.* for prediction of protein structures [33], [34]. These building blocks represent approximately some structural motifs like central α-helices, central β-strands, β-strand-N-caps and so on. This is an interesting approach where the length of each protein block is only five residues and is described by eight dihedral angles. Thus it is easy to represent the 3-D structure of a protein by a string of alphabets. Here a self-organizing map type neural network is used for clustering.

The use of an appropriate length for the building blocks is an important factor defining the effectiveness of such an algorithm. If we use a too short fragment length then the sequence to structure specificity will be lost, whereas with too long fragments, it would be difficult to find good building blocks.

Here we first discuss a modified form of the mountain clustering/subtractive clustering method [35], [36] to find building blocks. Results of some preliminary investigation using the Structural Mountain Clustering (SMCM) are reported in [37]. In [37], we analyze the computational complexity of the SMCM and find that the SMCM is computationally expensive when the training data set size is large. To reduce the computational burden, we propose here an incremental version of the structural mountain clustering method. An incremental version of the two stage clustering algorithm of Unger *et al.* [11] is also proposed. In our simulations we have used two benchmark data sets. Our results demonstrate that incremental versions of both algorithms are quite effective, i.e., we do not lose much in terms of quality of quantization but we can drastically reduce the computing overhead. Moreover, the proposed incremental Mountain clustering algorithm can find better building blocks than the method in [11].

II. Materials and Methods

A. Datasets

Two datasets are used in this paper, we call them as Dataset A and Dataset B. Dataset A consists of the same set of 82 proteins as used in Unger *et al.* [11]. Dataset A is referred to as the "refined Brookhaven" database in [11]. Here, Dataset A has two versions, Dataset $\mathrm{A_{OLD}}$ and $\mathrm{A_{NEW}}$. The Dataset $\mathrm{A_{OLD}}$ is exactly the same database as used in [11]. The data in the Protein Data Bank (PDB) are updated continuously as more new experimental observations become available. The Dataset $\mathrm{A_{NEW}}$ contains the same set of proteins as that in Dataset $\mathrm{A_{OLD}}$ but with updated information. In Table I we list only the peptides in Dataset $\mathrm{A_{OLD}}$ that are updated in Dataset $\mathrm{A_{NEW}}$—i.e., the list of updated proteins. The new PDB numbers are indicated in parentheses. As in [37], here also we have used both Dataset $\mathrm{A_{OLD}}$ and $\mathrm{A_{NEW}}$ to evaluate the performance of our algorithms. Unger *et al.* [11] used four proteins (1BP2, 1PCY, 4HHBb, 5PTI) as the training data and the proteins of length larger than 60 as the test data; we also use the same protocols.

TABLE I
Updated List of Peptides in Dataset $\mathrm{A_{NEW}}$ With New PDB Number in Parentheses

1APR(2APR)	1GCR(4GCR)	3PTP(5PTP)	1GAPa(1G6Na)
1CPP(2CPP)	1HMQa(2HMQa)	3RXN(7RXN)	2FD1(5FD1)
1CPV(5CPV)	1INSa(4INSa)	3TLN(8TLN)	4FXN(2FOX)
1FB4h(2FB4h)	1PCY(1PLC)	4ADH(8ADH)	2APP(3APP)
1FBJl(2FBJl)	1SN3(2SN3)	4ATCa(6AT1a)	4CYTr(5CYTr)
1FDX(1DUR)			

Dataset B is derived from the dataset used by Kolodny *et al.* [12]. In [12], the training set had 200 peptides. This set has some peptides with sequence discontinuity, which we have excluded and considered 153 peptides for our investigation. Note that, since our method is an incremental one it will use only a few of these 153 peptides as the training data. In [12] the test dataset is the same as Park *et al.* [20] which had 149 peptides with six duplicate entries. Thus, we have used all of the 143 distinct peptides. In order to create the library of short fragments, only the c_α coordinates are used.

B. The Building Block Approaches

Given a set of proteins with known 3-D structures and their associated sequences, we divide the proteins into fragments of a fixed length (say six). Note that these fragments are represented by their 3-D representations. These structural fragments are then clustered by some clustering/data compression algorithm to divide these fragments into structurally similar subsets. The centroid (or a representative member) of each cluster is then used as the building block or prototype. These building blocks or quantizers can be used to represent the original fragments with some (tolerable) error and hence in turn can be used to reconstruct the 3-D structure of a whole protein. The use of any clustering algorithm requires defining similarity/dissimilarity between 3-D structures and that is what we do next.

C. Measuring Dissimilarities Between Fragments

A measure of dissimilarity between two fragments/structures should be position and rotation invariant—two helical fragments should have a very low value of dissimilarity irrespective of the positions and orientations of the fragments. A frequently used [11], [37] measure of dissimilarity between two fragments is computed in two steps. First, the two fragments are aligned to the best possible extent using the BMF (best molecular fit) algorithm [38], [39] and then the RMS deviation between the two aligned structures is computed. Thus, given two structures X and Y, the RMS deviation is calculated as

$$\mathrm{RMS}(X,Y) = \left[\left(\sum_{i=1}^{L} \|\mathbf{x}_i - \mathbf{y}_i\|^2 \right) \Big/ (L-2) \right]^{1/2} \qquad (1)$$

where $\mathbf{x}_i, \mathbf{y}_i$ are 3-D coordinates of the c_α atom of the i th residue of X and Y, respectively after best molecular fit of X on Y and L is the number of residues in the structure/fragment. For the ease of comparison with published results, as done in [11], to compute the RMS deviation, we divide by $(L-2)$.

D. Reconstruction of Proteins

For reconstruction using building blocks, we use the following procedure as in [11]. Suppose the fragments are of length L. Each fragment is represented by a building block, which is closest to it in terms of the best-fit RMS error. Then, since two consecutive original fragments overlap by $L-1$ residues, we align every two consecutive best-fit building blocks considering only the overlapped $(L-1)$ residues. More specifically, onto the last $(L-1)$ residues of the ith building block we align the first $(L-1)$ residues of the $(i+1)$st building block. In this way, the 3-D position of the Lth residue of the $(i+1)$st fragment is determined. We continue this till the entire protein is reconstructed.

E. Evaluation of Performance

The quality of the library of the building blocks depends on its ability to reconstruct proteins. As done in [12], this can be assessed at two levels.

1) Local-fit RMS (LRMS) error: It computes the average of all coordinate RMS deviations between every fragment and its associated BMF building block. This is actually the quantization error when each fragment is represented by its closest building block.
2) Global-fit RMS (GRMS) error: This index relates to the reconstruction error of a whole protein. It measures the RMS deviation of the reconstructed entire 3-D structure of proteins from the corresponding native structure of the targets.

In addition, we also use different ways for visual comparison of performance of algorithms used in this paper. Suppose we have two algorithms A1 and A2 both producing the same (or almost same) number of building blocks. We compute the histograms of local-fit RMS (LRMS) error of all fragments in the test data for A1 and A2 as H1 and H2, respectively. If the area under the curve (hence also frequencies) for H1 on the left side (lower LRMS error) is higher than that for H2, then A1 is a better performing algorithm because it can represent more fragments with lesser errors. Similarly, we can compare average LRMS error per fragment for every protein. For this we compute the average LRMS error per fragment for every protein by A1. Then sort these values, say, in descending order and plot. The average per fragment LRMS error for every protein produced by A2 is plotted using the same order of proteins as used for A1. If the curve for A1, in general is below the curve for A2, then we can infer about consistency of algorithm A1 that it outperforms A2. To assess the quality of the clusters, and hence of the associated building blocks, we also analyze the entropy at each position of fragments in the top five most populated clusters. This will be explained later.

III. Clustering Algorithms

For the sake completeness, we first describe the two stage clustering method of Unger *et al.* [11] and then discuss our method along with related algorithms.

A. Two Stage Clustering Algorithm (TSCA)

In the first stage of TSCA [11], a fragment is selected randomly, which acts as a cluster center. Then all fragments which are within 1 Å distance after the BMF alignment are included in that cluster. Every member of this cluster in turn considers itself a cluster center and adds all fragments which are within 1 Å. This process continues till no more fragments can be added to the cluster. Note that distance between a pair of members of this cluster could be much higher than 1 Å. After this, TSCA randomly selects another unused fragment as the cluster center for the next cluster and the cluster is grown as before. This cluster generation process is continued as long as there is any unused fragment. This part of the algorithm is deterministic as cluster assignment does not depend on the order of using the fragments. In the next stage of TSCA, these big clusters are split into smaller subclusters such that every member in a cluster is within a distance of 1 Å from a designated centroid, the building block. To find the center of a small cluster (subcluster), for each member in the big cluster, the number of fragments within 1 Å is counted. The fragment with the maximum number of neighbors within 1 Å is taken as the center of a new subcluster. The fragments within 1 Å are considered members of that cluster. The process is repeated until all fragments are assigned to some subcluster satisfying the constraint of 1 Å.

B. The Mountain Clustering Method (MCM)

The mountain clustering method proposed by Yager in [35] is briefly described here. Let $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset \Re^p$ be a set of n data points in p-dimension, and x_{jk} be the j th component of $\mathbf{x}_k$; $k = 1, 2, \ldots, n$; $j = 1, 2, \ldots, p$. We consider a hypercube $I = I_1 \times I_2 \times \cdots \times I_p$ of $\Re^p$ where the interval $I_j, j = 1, 2, \ldots, p$, is defined by the range of the j th set of coordinates in the data, so $X \subset I$. Now, we divide each interval I_j into q_j equidistant points. This quantization results in $N = \prod_{j=1}^{p} q_j$ grid points, say $\mathbf{v}_i, i = 1, 2, \ldots, N$. in $\Re^p$. The *grid points* are potential cluster centers for the given data set.

Let $d(\mathbf{x}_k, \mathbf{v}_i)$ be the distance between $\mathbf{x}_k$ and the grid point $\mathbf{v}_i$. We now compute the mountain potential [35] at each grid point $\mathbf{v}_i$ as

$$P_1(\mathbf{v}_i) = \sum_{k=1}^{n} e^{-\alpha d^2(\mathbf{x}_k, \mathbf{v}_i)}; \quad i = 1, 2, \ldots, N. \qquad (2)$$

In (2) α is a positive constant. Equation (2) reveals that the mountain potential computes an approximate measure of density in the neighborhood of a grid point. Therefore, grid points with high values of the mountain potential are good candidates for cluster centers.

Now the grid point with the maximum mountain potential is selected as the first cluster center. Let P_1^* be the maximum value of the mountain potential, and $\mathbf{v}_1^*$ be the grid node whose mountain potential is P_1^*. So $\mathbf{v}_1^*$ is picked up as the first cluster center.

To find other cluster centers, the mountain function values are "discounted" to reduce the effect of already detected centers. For this, Yager [35] suggested the following update rule:

$$P_k(\mathbf{v}_i) = P_{k-1}(\mathbf{v}_i) - P_{k-1}^* e^{-\beta d^2(\mathbf{v}_{k-1}^*, \mathbf{v}_i)}; \quad k = 2, 3, \cdots, c; \quad i = 1, 2, \cdots, N. \tag{3}$$

In (3) $P_k(\mathbf{v}_i)$ is the modified mountain function, $P_{k-1}(\mathbf{v}_i)$ is the previous mountain function, $\mathbf{v}_{k-1}^*$ is the most recently detected center, and β is a positive constant. The next cluster center $\mathbf{v}_k^*$ is obtained by finding the grid point $\mathbf{v}_i$ corresponding to the maximum discounted potential, $P_k(\mathbf{v}_i)$. We continue the process until the discounted potential becomes too small to look for useful clusters. For example, we may repeat up to c times, when P_{c+1}^* falls below a threshold, $\delta > 0, P_{c+1}^* < \delta$. And thus we shall have c cluster centers: $\{\mathbf{v}_1^*, \mathbf{v}_2^*, \ldots, \mathbf{v}_c^*\}$.

The algorithm involves three parameters α, β, δ whose choice can significantly influence the performance. Moreover the quality of the centers also depends on the fineness of the grid, and better resolution leads to more cost. The computational overhead increases rapidly with dimension p. To reduce the computational overhead of MCM Chiu [36] suggested a modification of MCM, known as the *Subtractive Clustering Method* (SCM).

C. The Subtractive Clustering Method (SCM)

In MCM we use a set of artificially generated grid points as potential centers. Instead of that in [36] authors used each *data point* as a potential cluster center. So the potential function is redefined as

$$P_1(\mathbf{x}_i) = \sum_{k=1}^{n} e^{-\alpha\, d^2(\mathbf{x}_k, \mathbf{x}_i)}; \quad i = 1, 2, \cdots, n; \tag{4}$$

and discounting the potential on subsequent steps becomes

$$P_k(\mathbf{x}_i) = P_{k-1}(\mathbf{x}_i) - P_{k-1}^*\, e^{-\beta\, d^2(\mathbf{x}_{k-1}^*, \mathbf{x}_i)}, \quad k = 2, \ldots, c;\ i = 1, \ldots, n. \tag{5}$$

Here $\mathbf{x}_{k-1}^*$ is the $(k-1)$th (most recently detected) cluster center, and α and β are positive constants. Except the choice of potential cluster centers, the SCM algorithm remains the same as that of mountain clustering method. Here the number of prospective cluster centers is equal to the number of data points. SCM is terminated when $(P_{k-1}^*)/(P_1^*) < \delta, 0.0 < \delta < 1.0$. Note that, the computational complexity of SCM does not grow like MCM with number of features and is not dependent on the fineness of the grid. However, SCM expects that the desired cluster centers (points corresponding to the maximum local density) will coincide with (or be close to) some of the data points, which may or may not be true. In this particular application, we need to choose some fragments as building blocks and hence the SCM framework is more appropriate.

D. Structural Mountain Clustering Methods (SMCM)

The subtractive mountain clustering algorithm in [35], [36] uses Euclidean distance and hence as such is not applicable for structural data such as protein fragments because the Euclidean distance between two fragments X and Y, where Y is a translated or rotated version of X, could be high. But for our purpose they are the same. The SMCM [37] is a modified form of MCM to deal with structural data. There have been several extensions of the mountain clustering method [40], [41] including one variant that can deal with similarity relations in general [41]. Let $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset \Re^p$ be set of fragments. SMCM considers each fragment (e.g., hexamer) a potential cluster center. The contribution of a fragment $\mathbf{x}_j$ to the potential at another fragment $\mathbf{x}_i; i \neq j$ depends not on the Euclidean distance, but on the structural similarity between $\mathbf{x}_j$ and $\mathbf{x}_i$. Hence, RMS deviation between $\mathbf{x}_j$ and $\mathbf{x}_i$ after best molecular fit alignment is taken as the distance between the two fragments [38], [39] to compute the potential in (4). Like MCM we obtain the fragment, $\mathbf{x}_k$, with the highest potential as the first cluster center. Once we identify a cluster center all fragments within 1 Å of RMS deviation after best molecular fit are considered members of the cluster and removed from the data.

The entire process is then repeated to find the next cluster center and associated cluster. Unlike MCM and SCM, in SMCM we do not discount the potential but remove the points in a cluster and then recompute the potential to ensure that every cluster center is at the center of a dense region. A schematic description of the algorithm is now provided [37]:

Algorithm SMCM:

```
Input: Dataset $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset \Re^p$
Choose: $\alpha$
Compute $d(\mathbf{x}_i, \mathbf{x}_j) = \mathrm{RMS}_{i,j}$, for all $i, j = 1, 2, \ldots, n$
    $\mathrm{RMS}_{i,j}$ is the root mean square distance between $\mathbf{x}_i$ and
    $\mathbf{x}_j$ after BMF alignment of the two fragments.
Repeat
    1. Compute the potential at each fragment $\mathbf{x}_i$ using (4).
    2. Find the fragment with the highest potential and pick it
       as a building block.
    3. Remove all fragments, which are within an RMS
       distance of 1 Å from the building block to form the
       cluster associated with the building block.
Until all fragments are assigned to some cluster.
```

To find the cluster centers SMCM considers the geometry of the data not just the count of points within a cutoff distance and thus it is likely to produce better building blocks than those by TSCA. We illustrate this with an example in 2-D consisting of 15 points divided into two clusters and an isolated point (named A) as shown in Fig. 1(a). In Fig. 1(a), the distribution of the points is such that 5 points of the left cluster and 4 points of the right cluster are within a distance of 1 Å from the central point A. Since TSCA identifies the point with the largest number of neighbors within the given threshold (here, 1 Å) as

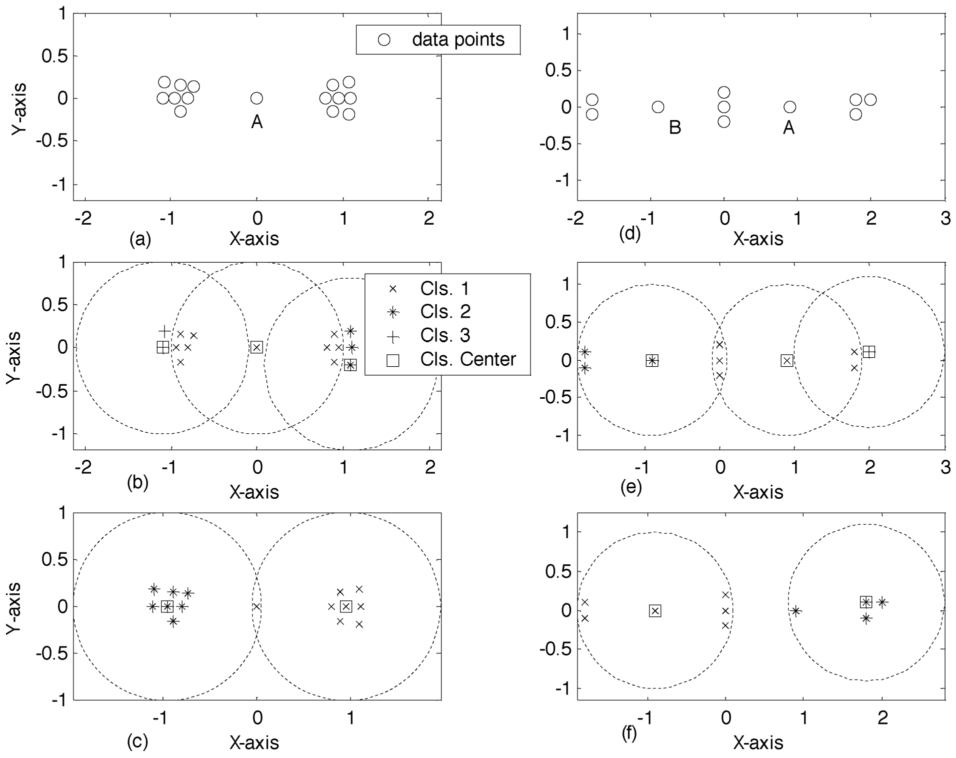

Fig. 1. (a) A set of 15 data points to illustrate different cluster centers by TCSA and SMCM. (b) Clusters generated by TSCA for 15 data points. (c) Clusters generated by SMCM for 15 data points. (d) A set of 10 data points to illustrate the different results depending on the cluster center which TSCA chooses first. (e) Clusters generated by TSCA for the 10 data points when the point A is chosen as the first cluster center. (f) Clusters generated by TSCA for the 10 data points when the point B is chosen as the first cluster center.

the cluster center, it chooses the point A as the cluster center or first building block. But this is a poor choice as A is not at the center of any cluster. Thus, TSCA building blocks will lead to more quantization/reconstruction error. Here, TSCA finally generates 3 clusters as displayed in Fig. 1(b). On the other hand, since SMCM takes into account the geometry of the distribution of the points, it generates two clusters as shown in Fig. 1(c). The building blocks are enclosed by rectangles. It is easy to see that SMCM produces much better building blocks than TSCA.

Another problem of TSCA is that it may generate different clustering results on the same dataset. Consider the 2-D dataset in Fig. 1(d) where point A and point B both have the same number of neighbors within the specified threshold. In this case, TSCA will generate two different results with different clusters. If it chooses point A as the first cluster center, it produces the clusters displayed in Fig. 1(e); whereas it produces a different result (Fig. 1(f)) if it chooses point B as the first cluster center.

E. Incremental Structural Mountain Clustering Methods (ISMCM)

Since the computational overhead for SMCM is quite high, it is very time-consuming to find clusters from a big dataset. To circumvent this problem, an incremental approach is proposed here. At first, we choose the longest protein in the training set and use it as the only protein for clustering to find the building blocks in the first step and evaluate the performance by checking the unassigned count of fragments (that cannot be assigned to any building block within 1 Å) for each protein in the whole training set. The protein with the largest count of unassigned fragments in this step is picked up and added to the selected set of proteins for clustering in the next step. Then, the two chosen proteins are used for clustering. As done before, we now find the third protein with the highest unassigned count and add it to the list of proteins to be clustered. The same process is repeated until the unassigned ratio (abbreviated as U_ratio, the percentage of fragments that are unassigned) on the whole set of training fragments is less than a threshold. Thus, we use only part of the original training dataset to cover the most occurring patterns and use them to find the building blocks accordingly.

Algorithm ISMCM:

```
Input: Dataset $P$ ={The complete list of proteins for training}
Choose: Threshold on unassigned ratio to stop the iteration
Initialization: Selected set: $P_1 = \{\}$, Remaining set: $P_2 = P$
Repeat
    1. Move the protein with largest unassigned count from $P_2$ into $P_1$. Note that $P_1$ and $P_2$ satisfy the conditions: $P_1 \cup P_2 = P$ and $P_1 \cap P_2 = \{\}$ (for the first iteration, the longest protein is chosen and moved into the selected set $P_1$)
    2. Find the building blocks from $P_1$ using SMCM described in Section III-D.
    3. Compute the unassigned count of fragments for each protein in $P_2$. These are the counts of fragments that cannot be represented by any building blocks derived from $P_1$ within an RMS error of 1 Å. Also, compute the unassigned ratio of the whole set of fragments.
Until unassigned ratio is less than the threshold.
```

TABLE II
SMCM RESULTS WITH FRAGMENT LENGTH EQUAL TO 6 USING (A) ORIGINAL DATASET AND (B) UPDATED DATASET

α	(a) Original Dataset A_{OLD}			(b) Newly updated Dataset A_{NEW}		
	library size	Test		library size	Test	
		LRMS	GRMS		LRMS	GRMS
8	106	0.75	7.72	108	0.73	7.53
7.5	106	0.75	7.72	108	0.73	7.53
7	106	0.75	7.70	107	0.73	7.48
6.5	105	0.75	7.82	107	0.73	7.48
6	106	0.74	7.64	107	0.72	7.32
5.5	106	0.74	7.64	107	0.72	7.32
5	104	0.75	7.19	107	0.73	7.59
4.5	104	0.75	7.27	107	0.73	7.69
4	105	0.75	7.30	106	0.73	7.75
3.5	104	0.75	7.62	105	0.73	7.76
3	103	0.75	7.92	106	0.72	7.71
2.5	104	0.75	8.00	105	0.73	7.94
2	105	0.75	7.95	106	0.72	7.48

The incremental version of TSCA (called here ITSCA) can be written exactly in the same manner.

IV. RESULTS

First we shall discuss the results with SMCM and TSCA on Dataset A only. In [37] we have reported some results on Dataset A. Here first we shall discuss an extended version of the results. This is required for the sake of comparison and completeness. Then we shall present the results by the incremental version of the two algorithms on both datasets.

For SMCM we have experimented with different choices of α for Dataset A_{OLD} and A_{NEW} using fragment length six. The Table II(a) displays the results obtained using the same dataset as used in [11]. Table II(a) reveals that $\alpha = 5$ is the best choice because it results in a global-fit RMS error of 7.19, which is less than 7.3 as reported in [11]. We also experimented with the newly updated dataset, Dataset A_{NEW} that is collected in December 2006. These results are depicted in Table II(b). For the new version of the data, the best GRMS error is 7.32 and is obtained for $\alpha = 5.5$ and 6.

We have also implemented the TSCA method and applied on the same data. Our implementation resulted in 55 main clusters and 102 subclusters (Unger *et al.* reported 103). On the other hand, SMCM extracted 104 building blocks. With a view to making a fair comparison of the two algorithms, we have discarded the trailing 2 building blocks from the SMCM list. Following the same protocols as used in [11], [37] we reconstruct the first 60 residues of 71 proteins whose lengths are larger than 60. We observe that with 102 building blocks, for SMCM, the LRMS error increases to 0.753 and global-fit RMS error increases to 7.23 (for $\alpha = 5$), which is still better than 7.3 [11]. Note that, with our implementation of TSCA, we have obtained an LRMS error of 0.77 and a GRMS error of 8.27. Comparing with the results reported in [11] we find that these error values are higher than the corresponding values in [11].

For the incremental version of the two algorithms, we have also varied the fragment length from 5 to 7. The choice of α is also varied from 3.5 to 6. For each fragment length, we report results with the best choice of α. Table III summarizes the results using the ISMCM algorithm for Dataset A_{OLD}. In Table III,

TABLE III
ISMCM RESULTS ON DATASET A_{OLD}

Protein count	PDB No.	Library Size	U_ratio	Train		Test	
				LRMS	GRMS	LRMS	GRMS
Fragment Length = 5, α =3.5							
1	4HHBb	20	20.49%	0.60	7.86	0.81	10.52
2	1PCY	38	2.93%	0.44	6.64	0.62	8.65
3	1BP2	40	1.46%	0.43	7.60	0.61	8.04
4	5PTI	44	0.00%	0.42	5.90	0.60	8.08
Fragment Length = 6, α =5							
1	4HHBb	35	34.98%	0.77	5.80	1.08	9.49
2	1PCY	73	9.36%	0.49	7.06	0.81	8.81
3	1BP2	93	2.96%	0.41	5.07	0.76	8.37
4	5PTI	104	0.00%	0.37	3.25	0.75	7.19
Fragment Length = 7, α =3.5							
1	4HHBb	51	43.78%	0.95	8.68	1.38	10.17
2	1PCY	117	16.42%	0.49	4.67	0.98	8.29
3	1BP2	153	6.22%	0.36	3.69	0.93	7.97
4	5PTI	173	0.00%	0.28	2.19	0.90	7.61

TABLE IV
ISMCM RESULTS ON THE UPDATED DATASET A_{NEW}

Protein count	PDB No.	Library Size	U_ratio	Train		Test	
				LRMS	GRMS	LRMS	GRMS
Fragment Length = 5, α =5.5							
1	4HHBb	20	20.73%	0.60	7.88	0.79	10.37
2	1PCY	35	3.90%	0.44	7.76	0.61	9.14
3	1BP2	39	0.73%	0.42	5.14	0.59	7.97
4	5PTI	45	0.00%	0.41	4.63	0.59	8.86
Fragment Length = 6, α =5.5							
1	4HHBb	35	34.98%	0.77	6.60	1.06	9.63
2	1PCY	74	9.85%	0.49	6.59	0.78	8.29
3	1BP2	94	3.20%	0.41	4.92	0.74	8.21
4	5PTI	107	0.00%	0.36	4.00	0.72	7.32
Fragment Length = 7, α =6							
1	4HHBb	51	43.78%	0.95	8.36	1.37	10.07
2	1PCY	117	16.67%	0.49	4.42	0.96	7.98
3	1BP2	151	6.22%	0.36	2.85	0.91	7.76
4	5PTI	175	0.00%	0.27	2.13	0.88	7.54

U_ratio denotes the percentage of total fragments that cannot be assigned to any building block within a distance of 1 Å. In this case too, we find that fragment length 6 with $\alpha = 5$ yields the best result of global-fit RMS error 7.19. Table III shows that with one protein in the training set, the test error is quite high. As we increase the number of proteins in the training set, the number of building blocks increases and the training and test errors decrease. Table III also reveals that increasing the number of proteins from 1 to 2 in the training set changes the number of building blocks and test error more drastically than those by increasing the number of training proteins from 3 to 4. This asymptotic behavior, which will be illustrated further with Dataset B, indicates the utility and consistency of the incremental version of SMCM.

Table IV depicts the performance of the ISMCM on Dataset A_{NEW}. We find from Table IV that when sequence length = 6 and $\alpha = 5.5$, we have the best reconstruction errors on the test set using 107 clusters: LRMS error = 0.72 and GRMS error = 7.32. On the other hand, in Table V, when we apply incremental version of the TSCA to the same dataset with sequence

TABLE V
ITSCA RESULTS ON THE UPDATED DATASET A_{NEW}

Protein count	PDB No.	Library size	U_ratio	Train		Test	
				LRMS	GRMS	LRMS	GRMS
Fragment Length = 5							
1	4HHBb	18	20.98%	0.69	8.28	0.86	10.81
2	1PCY	29	4.63%	0.57	7.48	0.70	9.07
3	1BP2	34	2.20%	0.50	8.00	0.64	9.04
4	5PTI	38	0.00%	0.49	5.64	0.62	8.21
Fragment Length = 6							
1	4HHBb	34	34.73%	0.81	7.39	1.08	10.12
2	1PCY	70	9.85%	0.53	7.33	0.82	8.83
3	1BP2	90	3.45%	0.46	6.63	0.77	8.29
4	5PTI	101	0.00%	0.43	5.94	0.76	8.14
Fragment Length = 7							
1	4HHBb	51	43.28%	0.96	8.62	1.37	10.05
2	1PCY	113	16.42%	0.51	4.67	0.98	8.31
3	1BP2	151	6.22%	0.38	3.47	0.92	7.95
4	5PTI	168	0.00%	0.31	2.12	0.90	7.59

length 6 and use the 4 training proteins to construct the building blocks, we obtain 101 clusters with no unassigned fragments (unassigned ratio $= 0\%$). The local-fit RMS error is 0.76 and global-fit RMS error is 8.14 on the test data. Since SMCM uses six more building blocks than TSCA, to make a fair comparison of ITSCA and ISMCM, we remove the trailing 6 building blocks from the 107 building blocks. Thus, for both methods we now use the same number of building blocks to represent all target fragments and reconstruct the first 60 residues of the 71 proteins whose lengths are larger than 60. For ISMCM, with 101 building blocks, the LRMS error very marginally increases to 0.73 and GRMS error increases to 7.55 from 7.32 (a 3% increase). But it is still better than 8.14 realized by ITSCA.

A. The Results on the Updated Dataset A_{NEW} Using Incremental TSCA (ITSCA)

If we apply the ITSCA to the updated Dataset A_{NEW}, we get the best result with fragment $\text{length} = 7$ and using all 4 proteins. However, the test global-fit RMS error is 7.59, which is still higher than the global-fit RMS error of 7.32 produced by the ISMCM with fragment length 6. The results are summarized in Table V.

B. The Results on the Dataset B

For this data too, we have experimented with fragment lengths 5, 6 and 7 as summarized in Tables VI and VII for the ISMCM and ITSCA respectively. For these two tables we find that ISMCM with fragment length 7 produces the best results of GRMS error of 14.56 which is better than the best GRMS error of 16.27 achieved by ITSCA with fragment length seven. However, the ISMCM usually finds more building blocks than the ITSCA. For example, Table VII shows that with fragment length 7 and six training proteins, the total number of building blocks found by ITSCA is 805 and this results in a GRMS reconstruction error of 16.27 whereas for ISMCM the number of building blocks is 858 yielding the best reconstruction error of 14.56. Just to compare the performance when ISMCM uses 710 building blocks (fragment length 7, number of proteins equal to 4) the GRMS error is 15.34 which is again smaller than 16.27. Thus the improvement in performance by the ISMCM is primarily not by the fact that it finds and uses more building blocks but because of quality of the building blocks that are placed at the center of dense areas of data points (here 3-D structures of length 5, 6, or 7).

TABLE VI
ISMCM RESULTS ON DATASET B

Protein count	PDB No.	Library Size	U_ratio	Train		Test	
				LRMS	GRMS	LRMS	GRMS
Fragment Length = 5, α =5							
1	1YGE	63	2.40%	0.51	16.96	0.56	19.20
2	1DMR	68	1.27%	0.50	16.77	0.55	19.10
3	1CZFa	80	0.98%	0.48	16.01	0.54	18.30
4	1LAM	84	0.74%	0.48	16.28	0.54	18.96
5	1SMD	91	0.56%	0.48	16.38	0.54	18.05
6	1DGFa	95	0.46%	0.47	16.83	0.52	18.05
Fragment Length = 6, α =5.5							
1	1YGE	171	11.2%	0.63	16.25	0.69	17.85
2	1DMR	216	6.6%	0.59	15.97	0.66	17.32
3	1CZFa	253	5.2%	0.58	15.26	0.65	16.78
4	1LAM	291	4.0%	0.57	14.90	0.64	16.73
5	1SMD	321	3.2%	0.56	14.55	0.63	16.32
6	1B4Va	349	2.8%	0.56	14.16	0.63	16.25
Fragment Length = 7, α =4							
1	1YGE	335	27.97%	0.76	15.68	0.83	17.67
2	1DMR	521	19.37%	0.70	14.52	0.77	15.94
3	1CZFa	612	16.50%	0.68	14.07	0.76	15.58
4	1SMD	710	14.26%	0.67	13.58	0.75	15.34
5	1QKSa	777	12.63%	0.66	13.76	0.74	15.72
6	1KAPp	858	11.05%	0.64	12.95	0.73	14.56

TABLE VII
ITSCA RESULTS ON DATASET B

Protein count	PDB No.	Library Size	U_ratio	Train		Test	
				LRMS	GRMS	LRMS	GRMS
Fragment Length = 5							
1	1YGE	56	3.15%	0.62	19.29	0.66	20.74
2	1DMR	56	2.01%	0.61	18.38	0.64	20.28
3	1CZFa	70	1.35%	0.56	17.36	0.60	19.19
4	1LAM	74	0.94%	0.60	18.88	0.63	20.18
5	1SMD	80	0.75%	0.60	19.19	0.63	20.14
6	1BXOa	84	0.61%	0.60	18.64	0.63	19.84
Fragment Length = 6							
1	1YGE	151	13.73%	0.68	16.68	0.73	18.75
2	1DMR	208	7.87%	0.68	15.43	0.73	17.54
3	1CZFa	239	6.20%	0.67	15.27	0.71	17.08
4	1SMD	274	4.86%	0.66	15.39	0.71	17.20
5	1LAM	292	4.27%	0.61	15.09	0.67	16.71
6	1QKSa	308	3.69%	0.60	15.20	0.66	16.49
Fragment Length = 7							
1	1YGE	327	29.72%	0.80	16.17	0.86	17.70
2	1DMR	496	20.40%	0.79	16.09	0.84	17.18
3	1CZFa	576	17.85%	0.77	15.37	0.83	16.71
4	1SMD	685	15.10%	0.75	15.57	0.82	16.84
5	1QKSa	729	13.98%	0.75	15.30	0.81	17.38
6	1KAPp	805	12.08%	0.73	15.40	0.80	16.27

Comparison of the GRMS errors in Tables VI and VII reveals that the ISMCM errors are usually less than those by the ITSCA. To compare the performance of both methods on the Dataset B, we proceed in the same way as we did for Dataset A. We remove the trailing clusters with smaller number of members to make both methods use the same number of building blocks. Thus, when we use only 805 clusters for ISMCM, the LRMS error very marginally increases to 0.734 and GRMS error increases to 14.83 which is still better than 16.27 realized by ITSCA.

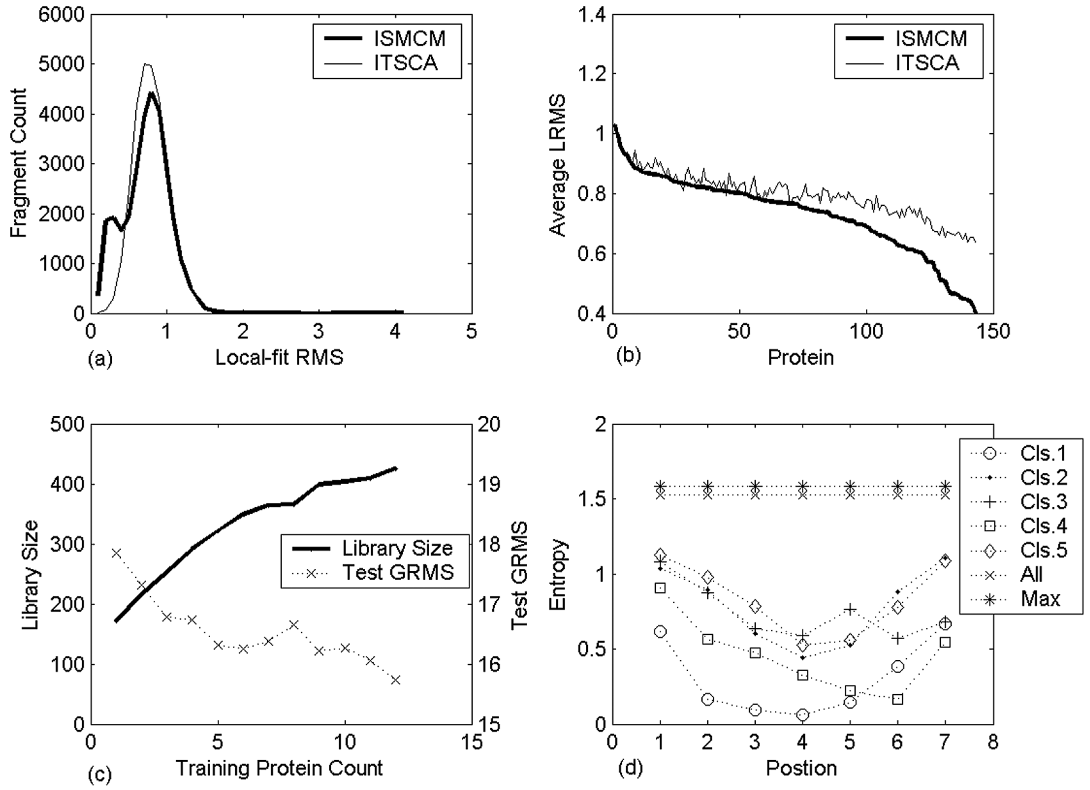

Fig. 2. (a) Comparisons of histograms of local-fit RMS error for ISMCM and ITSCA on Dataset B. (b) Comparisons of average local-fit RMS error per protein for ISMCM and ITSCA on Dataset B. (c) The variation of library size and that of reconstruction errors as functions of the number of training proteins. (d) Entropy plot for top 5 most populated clusters of ISMCM.

As mentioned earlier, we also assess the quality of the building blocks using two alternative graphical ways. In Fig. 2(a) and (b), we compare the histograms of LRMS errors and average LRMS error per protein, respectively. Here we find that ISMCM outperforms ITSCA with respect to these evaluation criteria. Similar behavior is also found while comparing SMCM and TSCA on Dataset A_{NEW} and A_{OLD}.

To investigate the effect of using more proteins for training, in Table VIII we report the results when we increase the number of proteins in the training set to 12 with fragment length 6. Table VIII depicts that the first six proteins generated 349 building blocks whereas another additional six proteins added only 76 building blocks. When the numbers of proteins used for training are equal to 3, 6, 9, and 12, then the numbers of building blocks are 253, 349, 399, and 425, respectively. As we increase the number of training proteins from 3 to 12 via 6 and 9, the increments in the number of building blocks are 96, 50, and 26, respectively. This indicates that as we increase the number of training proteins, the change in the number of building blocks becomes less and less. This is a very desirable property of any incremental algorithm. This behavior is clearly reflected in Fig. 2(c), which displays the variation of number of building blocks and global-fit RMS errors on the test data as a function of number of proteins in the training set. Thus use of more proteins in the training data increases the computational cost substantially, but the gain may not be significant. The computational cost increases with the number of proteins that are used for training and the marginal benefit decreases. The details are depicted in the Table VIII.

TABLE VIII
ISMCM RESULTS ON DATASET B USING 12 TRAINING PROTEINS WITH FRAGMENT LENGTH 6

Protein count	PDB No.	Library Size	U_ratio	Train		Test	
				LRMS	GRMS	LRMS	GRMS
Fragment Length = 6, α =5.5							
1	1YGE	171	11.2%	0.63	16.25	0.69	17.85
2	1DMR	216	6.6%	0.59	15.97	0.66	17.32
3	1CZFa	253	5.2%	0.58	15.26	0.65	16.78
4	1LAM	291	4.0%	0.57	14.90	0.64	16.73
5	1SMD	321	3.2%	0.56	14.55	0.63	16.32
6	1B4Va	349	2.8%	0.56	14.16	0.63	16.25
7	1QGUa	364	2.5%	0.55	14.17	0.62	16.38
8	1DGFa	366	2.3%	0.55	14.45	0.62	16.65
9	3SIL	399	1.9%	0.55	14.21	0.61	16.22
10	1QGXa	404	1.9%	0.55	13.98	0.61	16.26
11	1VNS	409	1.6%	0.54	14.03	0.61	16.05
12	1PHC	425	1.5%	0.54	13.65	0.61	15.74

C. How Well Do the Building Blocks Fit the Target Fragments?

Entropic Assessment: If each cluster represents homogeneous (similar) structures/fragments then every position of the fragments in a cluster should have some preference or bias to specific local secondary structures such as H'- helix, 'E'-strand, and 'C'-others which can be obtained by Dictionary of Protein Secondary Structure (DSSP) [42]. So we consider the local secondary structure of each residue for all fragments in a cluster and then compute the Shannon's entropy [43] at each position of the fragment. In Fig. 2(d), we summarize the

TABLE IX
THE ENTROPY FOR THE TOP 5 MOST POPULATED CLUSTERS

Entropy vs. Position	Pos.1	Pos.2	Pos.3	Pos.4	Pos.5	Pos.6	Pos.7
Cluster 1: SAQQQAQ	0.616	0.162	0.095	0.060	0.145	0.382	0.667
Cluster 2: GIKIYVS	1.037	0.892	0.601	0.439	0.527	0.881	1.104
Cluster 3: EPCLMHP	1.082	0.874	0.636	0.592	0.764	0.568	0.679
Cluster 4: VSWDQAL	0.908	0.563	0.473	0.326	0.222	0.162	0.545
Cluster 5: GNDILYG	1.126	0.976	0.785	0.526	0.558	0.779	1.089
All clusters	1.524	1.526	1.527	1.528	1.528	1.527	1.524
Max. entropy($\log_2 3$)	1.585	1.585	1.585	1.585	1.585	1.585	1.585

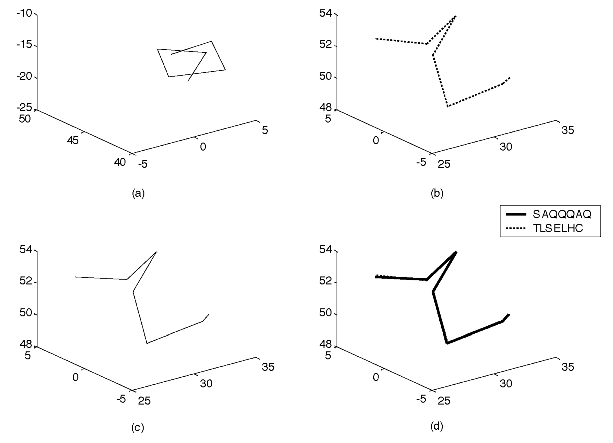

Fig. 3. Representation of a target fragment using a building block, a good fit case. (a) The original building block. (b) The target fragment. (c) The rotated and shifted building block. (d) The building block and target fragment superimposed after alignment.

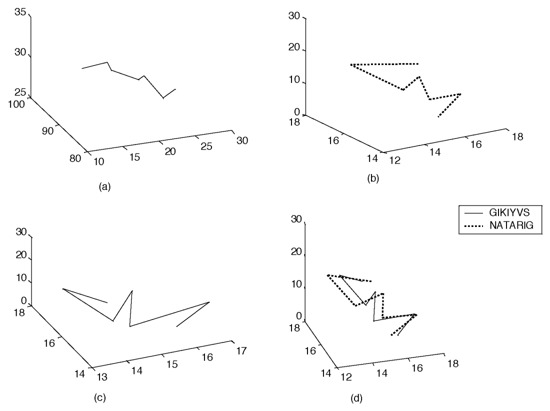

Fig. 4. Representation of a target fragment using a building block, a poor fit case. (a) The original building block. (b) The target fragment. (c) The rotated and shifted building block. (d) The building block and target fragment superimposed after alignment.

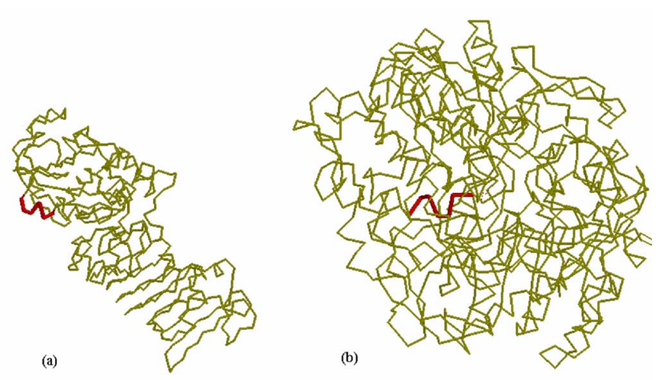

Fig. 5. (a) SMCM building block (SAQQQAQ) at residue 89–95 of 1KAPp. (b) TSCA building block (GAAQVIM) at residue 149–155 of 1DMR.

entropy distribution for the top five most populated clusters. The corresponding building blocks are listed in the Table IX. In order to get a better assessment of the position specific bias for a particular secondary structure we also compute the entropy over the entire set of fragments (i.e., taking all clusters together). Fig. 2(d) reveals a few interesting things. First, considering all clusters, there is no bias for any secondary structure for any position. Every position has almost the same entropy and it is almost equal to the maximum entropy. For the top five clusters, at each of the seven positions, the entropy is much lower than the corresponding entropy using all clusters. This suggests that every position has some bias for some particular structure. More interestingly the entropy is very low at the central residue implying that the secondary structure of the central residue for most fragments in a cluster is the same. This might suggest that the central residue has a stronger impact in deciding on the cluster and hence on the local folding of the fragment.

Visual Assessment: To examine visually how well the building blocks represent the target fragments, we consider two examples with fragment length seven: one with a very good fit (Fig. 3) and the other with a relatively poor fit (Fig. 4), but still within 1 Å threshold. Fig. 3(a) shows a building block (SAQQQAQ) whereas Fig. 3(b) represents a target fragment (TLSELHC). Apparently the two structures look quite different. But Fig. 3(c), the rotated version of the building block SAQQQAQ obtained after best molecular fit with the target, looks almost identical to Fig. 3(b). The superimposition of SAQQQAQ and TLSELHC shown in Fig. 3(d) clearly demonstrates an excellent fit between the two. The four panels in Fig. 4 show the representation of the target fragment EGVEIAC with the building block GIKIYVS. Although, in terms of the local-fit RMS error it is a poorer fit, yet the building block matches appropriately to the target [Fig. 4(d)].

We note that the two building blocks (fragment length seven) used in the illustration are associated with the two most populated clusters extracted by SMCM. We can also compare it with the building blocks of top two clusters generated by TSCA. The fragment SAQQQAQ represents a typical helical building block. This helical building block found by SMCM is located at residue 89–95 of 1KAPp; whereas the most populated building block found by TSCA is GAAQVIM and it is located at residue 149–155 of 1DMR. The fact that GAAQVIM is also of helical structure and is included in the cluster of SAQQQAQ, might be taken as an indicator that the TSCA cluster associated with GAAQVIM and the SMCM cluster associated with SAQQQAQ represent the same biological structural motif. To investigate this further we analyzed the two clusters. The SMCM cluster has 5683 members while those in the TSCA cluster is 5595 and these two clusters share 5349 common fragments. This suggests that the two clusters represent the same biological entity. Fig. 5(a) and (b) show these two building blocks. When we compute the root-mean-square deviation between these two building blocks, it is less than 1 Å, which further suggests that they represent the same structural cluster. Similarly, we find that the most typical extended strand GIKIYVS found by SMCM is exactly the same fragment GIKIYVS found by TSCA which is located at residue 464 of 1SMD. Thus the building blocks found by SMCM are likely to represent structures of biological significance.

V. Conclusion

We have first discussed the SMCM and TSCA and have demonstrated that since TSCA does not take into account the geometry of the data, it may extract poorer building blocks than the SMCM. The effectiveness of SMCM is demonstrated on the same dataset used by Unger *et al.* and on the updated version of the same data set. SMCM is found to outperform TSCA.

Both SMCM and TSCA are computationally very expensive when the size of training dataset is large. Hence we have proposed an incremental version of the SMCM. The same concept is also used to obtain an incremental version of the TSCA. We have made extensive experimentation with these two algorithms using two versions of the dataset used by Unger *et al.* as well as another dataset used by other researchers. The incremental SMCM is quite effective and is found to exhibit properties that are expected from an incremental algorithm. More specifically, as the number of proteins increases in the training set, the increase in the number of building blocks decreases and consequently the rate of decrease in the global reconstruction error both on the training and test data falls down. Moreover, the incremental SMCM is found to be more effective than the incremental TSCA. Although, the SMCM usually finds more building blocks than those found by the TSCA, we have demonstrated that the improved performance for SMCM comes from the quality of the building blocks, which are placed at the center of dense areas in the training data. The quality of the clusters (and hence of the associated building blocks) extracted by our algorithm is also assessed using entropy analysis. The entropy analysis has revealed a very interesting fact that the central residue possibly plays the most dominant role in the local folding of the fragments.

None of the algorithms discussed here can take into account fragments of variable length. To extend the algorithms for fragments of variable length, we need measures of similarity between fragments of different lengths. For example, if we have two fragments both are helix, but of different length, the structural similarity between the two should be very high; on a [0-1] scale, it should be 1. We plan to investigate this in near future.

References

[1] G. Klebe, "Recent developments in structure-based drug design," *J. Mol. Med.*, vol. 78, pp. 269–281, 2000.

[2] I. Belda, X. Llora, and E. Giralt, "Evolutionary algorithms and de novo peptide design," *Soft Comput.*, vol. 10, pp. 295–304, 2006.

[3] H. H. Tsai, C. J. Tsai, B. Ma, and R. Nussinov, "In silico protein design by combinatorial assembly of protein building blocks," *Protein Sci.*, vol. 13, no. 10, pp. 2753–2765, 2004.

[4] D. Baker and A. Sali, "Protein structure prediction and structural genomics," *Science*, vol. 294, pp. 93–96, 2001.

[5] F. Melo and A. Sali, "Fold assessment for comparative protein structure modeling," *Protein Sci.*, vol. 16, pp. 2412–2426, 2007.

[6] D. T. Jones, "GenTHREADER: An efficient and reliable protein fold recognition method for genomic sequences," *J. Mol. Biol.*, vol. 287, pp. 797–815, 1999.

[7] C. D. Huang, C. T. Lin, and N. R. Pal, "Hierarchical learning architecture with automatic feature selection for multiclass protein fold classification," *IEEE Trans. NanoBiosci.*, vol. 2, pp. 221–232, 2003.

[8] C. Bystroff and D. Baker, "Prediction of local structure in proteins using a library of sequence-structure motifs," *J. Mol. Biol.*, vol. 281, pp. 565–577, 1998.

[9] Y. Liu and D. L. Beveridge, "Exploratory studies of ab-initio protein structure prediction: Multiple copy simulated annealing, amber energy functions, and a generalized born/solvent accessibility solvation model," *Proteins*, vol. 46, pp. 128–146, 2002.

[10] P. Pokarowski, A. Kolinski, and J. Skolnick, "A minimal physically realistic protein-like lattice model: Designing an energy landscape that ensures all-or-none folding to a unique native state," *Biophys J.*, vol. 84, pp. 1518–1526, 2003.

[11] R. Unger, D. Harel, S. Wherland, and J. L. Sussman, "A 3D building blocks approach to analyzing and predicting structure of proteins," *Proteins*, vol. 5, pp. 355–373, 1989.

[12] R. Kolodny, P. Koehl, L. Guibas, and M. Levitt, "Small libraries of protein fragments model native protein structures accurately," *J. Mol. Biol.*, vol. 323, pp. 297–307, 2002.

[13] C. Micheletti, F. Seno, and A. Maritan, "Recurrent oligomers in proteins: An optimal scheme reconciling accurate and concise backbone representations in automated folding and design studies," *Proteins*, vol. 40, pp. 662–674, 2000.

[14] J. M. Bujnicki, "Protein-structure prediction by recombination of fragments," *ChemBioChem*, vol. 7, pp. 19–27, 2006.

[15] K. T. Simons, C. Kooperberg, E. Huang, and D. Baker, "Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions," *J. Mol. Biol.*, vol. 268, pp. 209–225, 1997.

[16] G. Chikenji, Y. Fujitsuka, and S. Takada, "A reversible fragment assembly method for de novo protein structure prediction," *J. Chem. Phys.*, vol. 119, pp. 6895–6903, 2003.

[17] D. Kihara and J. Skolnick, "The PDB is a covering set of small protein structures," *J. Mol. Biol.*, vol. 334, pp. 793–802, 2003.

[18] Y. Zhang and J. Skolnick, "The protein structure prediction problem could be solved using the current PDB library," *Proc. Nat. Acad. Sci. USA*, vol. 102, pp. 1029–1034, 2005.

[19] R. Kolodny and M. Levitt, "Protein decoy assembly using short fragments under geometric constraints," *Biopolymers*, vol. 68, pp. 278–285, 2003.

[20] B. H. Park and M. Levitt, "The complexity and accuracy of discrete state models of protein structure," *J. Mol. Biol.*, vol. 249, pp. 493–507, 1995.

[21] N. Haspel, C. J. Tsai, H. Wolfson, and R. Nussinov, "Hierarchical protein folding pathways: A computational study of protein fragments," *Proteins*, vol. 51, no. 2, pp. 203–215, 2003.

[22] B. Rost and C. Sander, "Prediction of protein secondary structure at better than 70% accuracy," *J. Mol. Biol.*, vol. 232, pp. 584–599, 1993.

[23] N. Qian and T. J. Sejnowski, "Predicting the secondary structure of globular proteins using neural network models," *J. Mol. Biol.*, vol. 202, pp. 865–884, 1988.

[24] N. R. Pal and D. Chakraborty, "Some new features for protein fold prediction," in *Proc. ICONIP 2003*, pp. 1176–1183.

[25] I. F. Chung, C. D. Huang, Y. H. Shen, and C. T. Lin, "Recognition of structure classification of protein folding by NN and SVM hierarchical learning architecture," in *Proc. ICONIP 2003*, pp. 1159–1167.

[26] C. H. Q. Ding and I. Dubchak, "Multi-class protein fold recognition using support vector machines and neural networks," *Bioinformatics*, vol. 17, no. 4, pp. 349–358, 2001.

[27] M. N. Nguyen and J. C. Rajapakse, "Multi-class support vector machines for protein secondary structure prediction," *Genome Informatics*, vol. 14, pp. 218–227, 2003.

[28] F. Markowetz, L. Edler, and M. Vingron, "Support vector machines for protein fold class prediction," *Biometrical J.*, vol. 45, no. 3, pp. 377–389, 2003.

[29] X. D. Sun and R. B. Huang, "Prediction of protein structural classes using support vector machines," *Amino Acids*, vol. 30, no. 4, pp. 469–475, 2006.

[30] A. V. Tendulkar, A. A. Joshi, M. A. Sohoni, and P. P. Wangikar, "Clustering of protein structural fragments reveals modular building block approach of nature," *J. Mol. Biol.*, vol. 338, no. 3, pp. 611–629, 2004.

[31] P. Ghanty and N. R. Pal, "Prediction of protein folds: Extraction of new features, dimensionality reduction, and fusion of heterogeneous classifiers," *IEEE Trans. NanoBiosci.*, vol. 8, no. 1, pp. 100–110, 2009.

[32] S. Anishetty, G. Pennathur, and R. Anishetty, "Tripeptide analysis of protein structures," *BMC Struct. Biol.*, vol. 2, no. 9, pp. 1–8, 2002.

[33] C. Benros, A. G. de Brevern, C. Etchebest, and S. Hazout, "Assessing a novel approach for predicting local 3D protein structures from sequence," *Proteins*, vol. 62, no. 4, pp. 865–880, 2006.

[34] A. G. de Brevern and S. Hazout, "Hybrid protein model for optimally defining 3D protein structure fragments," *Bioinformatics*, vol. 19, no. 3, pp. 345–353, 2003.

[35] R. R. Yager and D. P. Filev, "Approximate clustering via the mountain method," *IEEE Trans. Syst., Man Cybern.*, vol. 24, pp. 1279–1284, 1994.

[36] S. L. Chiu, "Extracting fuzzy rules for pattern classification by cluster estimation," in *Proc. 6th Int. Fuz. Syst. Assoc, World Congr. (IFSA'95)*, pp. 1–4.

[37] K. L. Lin, C. T. Lin, N. R. Pal, and S. Ojha, "Structural building blocks: Construction of protein 3-D structures using a structural variant of mountain clustering method," *IEEE Eng. Med. Biol. Mag.*, vol. 28, no. 4, pp. 38–44, Jul. 2009.

[38] W. Kabsch, "A solution for the best rotation to relate two sets of vectors," *Acta Crystallogr.*, vol. A32, pp. 922–923, 1976.

[39] W. Kabsch, "A discussion of the solution for the best rotation to relate two sets of vectors," *Acta Crystallogr.*, vol. A34, pp. 828–829, 1978.

[40] N. R. Pal and D. Chakraborty, "Mountain and subtractive clustering methods: Improvements and generalizations," *Int. J. Intell. Syst.*, vol. 15, pp. 329–341, 2000.

[41] K. Pal, N. R. Pal, J. Keller, and J. Bezdek, "Relational mountain (density) clustering method and web log analysis," *Int. J. Intell. Syst.*, vol. 20, no. 3, pp. 375–392, 2005.

[42] W. Kabsch and C. Sander, "Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features," *Biopolymers*, vol. 22, no. 12, pp. 2577–637, 1983.

[43] T. D. Schneider, "Information theory primer with an appendix on logarithms," National Cancer Institute, Apr. 14, 2007.

Ken-Li Lin received the B.S. and M.S. degrees from the Department of Control Engineering from National Chiao-Tung University (NCTU), Hsinchu, Taiwan, in 1990 and 1992, respectively, and the Ph.D. degree from the Department of Electrical and Control Engineering, National Chiao-Tung University, in 2008.

He worked as a System Engineer at AT&T Taiwan Telecommunication Corporation from 1994 to 1996, and at the Powerchip Semiconductor Corporation from 1996 to 1998. He is currently with the computer center of Chung Hua University, Hsichu. His research interests are in the areas of information technology, computational intelligence and bioinformatics.

Chin-Teng Lin (F'05) is currently the Chair Professor of Electrical and Computer Engineering, the Provost of National Chiao-Tung University (NCTU), Hsinchu, Taiwan, and Director of Brain Research Center at NCTU. He served as the Dean of Computer Science College from 2005 to 2007, Director of the Research and Development Office of NCTU from 1998 to 2000, the Chairman of Electrical and Control Engineering Department of NCTU from 2000 to 2003, and Associate Dean of the College of Electrical Engineering and Computer Science from 2003 to 2005. His current research interests are fuzzy neural networks, neural networks, fuzzy systems, cellular neural networks, neural engineering, algorithms, and VLSI design for pattern recognition, intelligent control, and multimedia (including image/video and speech/audio) signal processing, and intelligent transportation system (ITS). He has published over 110 journal papers in the areas of neural networks, fuzzy systems, multimedia hardware/software, and soft computing, including about 90 IEEE journal papers. He is the coauthor of *Neural Fuzzy Systems—A Neuro-Fuzzy Synergism to Intelligent Systems* (Prentice Hall), and the author of *Neural Fuzzy Control Systems with Structure and Parameter Learning* (World Scientific).

Dr. Lin currently serves as Associate Editor of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART I, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART II, IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, IEEE TRANSACTIONS ON FUZZY SYSTEMS, and *International Journal of Speech Technology.*

Nikhil R. Pal (F'05) is a Professor in the Electronics and Communication Sciences Unit of the Indian Statistical Institute, Calcutta. He has coauthored/edited and coedited several books. His current research interest includes bioinformatics, medical and satellite image analysis, pattern recognition, fuzzy sets theory, neural networks, and evolutionary computation. He has given plenary and keynote speeches at many international conferences. He serves the editorial/advisory board of several journals, including the *International Journal of Approximate Reasoning*, *International Journal of Hybrid Intelligent Systems*, *International Journal of Neural Systems and Fuzzy Sets and Systems*.

Dr. Pal is an Associate Editor of the IEEE TRANSACTIONS ON SYSTEMS MAN AND CYBERNETICS B and the Editor-in-Chief of IEEE TRANSACTIONS ON FUZZY SYSTEMS. He is an elected member of the IEEE-CIS AdCom (2010–2012).