On the Convergence of Protein Structure and Dynamics. Statistical Learning Studies of Pseudo Folding Pathways

Alessandro Vullo[1], Andrea Passerini[2], Paolo Frasconi[2], Fabrizio Costa[2], Gianluca Pollastri[1]

[1] School of Computer Science and Informatics
University College Dublin, Belfield, Dublin 4, Ireland
[2] Dipartimento di Sistemi e Informatica
Università degli Studi di Firenze, Via di S.Marta 3, 50139 Firenze, Italy

Abstract. Many algorithms that attempt to predict proteins' native structure from sequence need to generate a large set of hypotheses in order to ensure that nearly correct structures are included, leading to the problem of assessing the quality of alternative 3D conformations. This problem has been mostly approached by focusing on the final 3D conformation, with machine learning techniques playing a leading role. We argue in this paper that additional information for recognising native-like structures can be obtained by regarding the final conformation as the result of a generative process reminiscent of the folding process that generates structures in nature. We introduce a coarse representation of protein pseudo-folding based on binary trees and introduce a kernel function for assessing their similarity. Kernel-based analysis techniques empirically demonstrate a significant correlation between information contained into pseudo-folding trees and features of native folds in a large and non-redundant set of proteins.

1 Introduction

Accurate protein structure prediction is still an open and challenging problem for a vast subset of the protein universe. Experiments of blind prediction such as the CASP series [14] demonstrate that the goal is far from being achieved, especially for those proteins whose sequence does not resemble that of any protein of known structure (nearly half of the total) - the field known as *ab initio*. Difficulties in this case are well known: the choice of a reduced protein representation and the corresponding empirical potential function may allow for an efficient search of the conformational space, but generally the methods are not sensitive enough to differentiate correct native structures from conformations that are structurally close to the native state. On the other hand, techniques such as Comparative Modelling and Fold Recognition can be very successful at predicting accurate models, but success strongly depends on the quality of the alignment and the ability to reliably detect homologues. Moreover, models with severely unrealistic geometry can be produced, especially when using fully

automated prediction pipelines. As past and recent findings suggest, a practical way to obtain improvements in protein structure prediction consists of the integration of alternative techniques and sources of information. For instance, empirical elements (e.g. secondary structure predictions) are routinely used to constrain the space of allowed conformations, to correct and refine an alignment or to improve the sensitivity of remote homologue detection. Model quality assessment programs (MQAPs) are becoming increasingly important for filtering out wrong predictions [17]. A common theme between computational prediction techniques and most refinement methods is that they more or less directly depend on knowledge mined from existing protein structures and, to a smaller extent, on the available theory and principles of protein structure. In spite of the continuous increase in the amount of available structural data, progresses in protein structure prediction and model quality assessment have been slow. This may indicate that the goal of reaching reliable protein structure prediction requires new, alternative sources of information.

This paper is an attempt to investigate in this direction. We believe that novel algorithmic ideas may come from looking at the dynamics of protein folding simulations, instead of focussing solely on their final product. We assume that any plausible abstraction of the folding process may contain potentially valuable information about the final fold. Indeed, specific folding patterns are intimately related with the native structure. If deviations from these pathways occur, often this will yield incorrect (i.e. non native-like) contacts between residues that are more stable than the correct ones, resulting in structural deviations from the native fold [8]. Folding may then be viewed as the dynamical fingerprint of the resulting structure.

Modelling or understanding protein folding at the conceptual level remains beyond the scope of the present paper. Theoretical modelling of the dynamics of protein folding faces several difficulties: there is a much smaller body of experimental data than the PDB, which is typically at low resolution, and carrying out computations over long time scales requires either very large amounts of computer time or the use of highly approximate models [10]. Rather, we take the more pragmatic perspective of finding manageable representations of protein pseudo-folding simulations and evaluating their potential impact on protein structure prediction. In this study, we derive a representation called binary pseudo-folding tree (BPFT), borrowing ideas from other recent works [11, 22]. A BPFT expresses a hierarchy of timestamped pairing events involving secondary structure elements (SSEs) and is computed by inspecting the execution trace of a stochastic optimisation algorithm for structure reconstruction that explores a protein conformational space driven by spatial proximity restraints. Similar algorithms are common for example in the NMR structure determination literature and can be applied to recover protein structure from contact maps [18]. We empirically investigate the existence of a relationship between information provided by BPFTs and features of native folds for a large and non-redundant set of proteins. We first introduce a kernel function for measuring similarity between BPFTs, and compare its ability to detect similarities with respect to the

TM-score [23]. We then apply the kernel to cluster sets of optimisation traces associated with alternative reconstructions from contact maps.

2 Binary Pseudo-Folding Trees

Although the fine mechanisms that regulate protein folding are in principle extremely complex, hence nearly impossible to simulate and predict on current computational hardware, there is evidence that the essential elements of the process are much simpler and coarse-grained [2, 15]. In nature, the folding process appears to follow "pathways", involving hierarchical assemblies and intermediate states requiring doing and undoing of structures [9]. Rather than static, and driven by properties identifiable in the final fold, the folding process appears to be dynamic and driven by interactions whose nature and relative importance change during the process itself. Multiple pathways, with different transition states also appear to be possible [21]. The combination of experimental and computational techniques has revealed other properties of the folding process [7]. For instance, it appears that in some cases interactions among key elements in the protein form a core or nucleus that essentially constrains the protein topology to its fold [19]. Also, there is much evidence that folding is hierarchical; for some proteins it involves stable intermediates, called foldons, that consist of SSEs [12]. Folding routes can then be thought of as having an underlying tree structure [11] and clusters of interacting SSEs may form the tree labels [22].

Our aim is to derive representations of protein folding simulations which have to be simple yet informative, i.e. tractable by machine learning techniques. We borrow ideas from the work of other authors [11, 22], although with different premises and details. Neither we assume that the three-dimensional (3D) structure of a protein is known nor we want to identify real folding pathways for the protein under study. Rather we argue that regarding a predicted protein conformation as the result of a generative process may yield additional information about this conformation. Structures are generated by an algorithm that explores the conformational space of the protein. A labelled binary tree is built in an incremental fashion by observing notable intermediate events happening along the trajectory that is being followed. Since we are not dealing with the real process, we call the trajectory a pseudo-folding pathway and the resulting tree a binary pseudo-folding tree.

2.1 Pseudo-folding pathways

Protein folding simulations are carried out with an algorithm that models protein structures by exploring a protein's conformational space starting from an initial (random) configuration. Usually, this kind of algorithms are guided by some form of energy encoding structural principles or a pseudo-energy (statistical potential function) or a combination of the two. In this work, we employ 3Distill [3], a machine learning based system for the prediction of alpha carbon (C_α) traces. For a given input sequence, first a set of 1D features is predicted, e.g. secondary

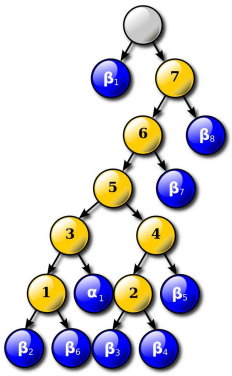

Fig. 1. BPFT for a protein (PDB code 1WITA) as a result of the application of Alg.1 to the trajectory followed by the reconstruction algorithm described in section 2.1.

structure and solvent accessibility. These features are then used as an input to infer the shape of 2D features like the contact map (binary or multi-class). In the last stage, protein structures are coarsely described with their backbone C_α atoms and are predicted by means of a stochastic optimisation algorithm using as pseudo-energy a function of the geometric constraints inferred from the underlying set of 1D and 2D predictions. The stochastic optimisation algorithm explores the configurational space starting from a random conformation, and refining this by global optimisation of the pseudo-potential function using local moves and a simulated annealing protocol. For more and complete details on the form of the cost function and the annealing protocol see [4].

2.2 Notation

Let $s_1 \dots s_m$ be the sequence of m secondary structure segments of the protein, where s_i is the i-th segment in the sequence, either a α-helix or a β-strand. Let $S_1 \dots S_T$ be the time ordered sequence of structures observed at discrete time steps during a simulation. Using this notation, S_1 is the initial configuration and S_T is the predicted model structure. We introduce a simple and synthetic representation of an execution trace based on binary trees.

A Binary Pseudo-Folding Tree (BPFT) is a rooted, unordered, leaf-labelled binary tree. Suppose we are given the pseudo-folding pathway $\mathcal{P} = S_1 \dots S_T$. The corresponding BPFT, called $\mathcal{T}$, expresses a hierarchy of timestamped pairing events involving sets of α-helices and β-strands[3]. Each leaf node has a label that represents the type and position of a SSE (e.g. β_2 means the second strand of the sequence, α_1 the first helix and so on). An internal node $n \in \mathcal{T}$ corresponds to a pairing event occurred at time $1 < t < T$ and that involved two SSEs

[3] Random coil fragments are not usually involved in major structure stabilisation events and are not considered here.

belonging to different clusters of interacting SSEs. Each of the two clusters is a child node of n, which in turn represents a larger set of SSEs that eventually joins another cluster in its parent. The recursive structure of $\mathcal{T}$ is inspired to other binary tree representations of folding pathways [22], but with a number of differences. In [22], a tree (the predicted folding pathway) is built by recursively applying a polynomial-time mincut algorithm to a weighted graph, this graph representing sets of interacting SSEs of the *known* experimental structure. Here, we do not assume to know the real 3D structure of a protein, unless we run the 3Distill reconstruction algorithm with experimental 1D and 2D restraints. Moreover, folding information is obtained by using a pseudo-folding trajectory, i.e. simulated dynamical data. For a given time step t of the simulation there is a node $n \in \mathcal{T}$ such that the subtree $\mathcal{T}_n$ rooted at n corresponds to the assembling history (from $t = 1 \dots t$) of a cluster of interacting SSEs in S_t, where the segments involved are given by the leaves dominated by n. Let $\mathrm{ch}_l[n]$ (resp. $\mathrm{ch}_r[n]$) be the assigned left (resp. right) child of n and let $\textsc{Leaves}(\cdot)$ be a function returning the set of leaf labels of a subtree of $\mathcal{T}$. The cluster of node n is formed because one or more segments in $\textsc{Leaves}(\mathcal{T}_{\mathrm{ch}_l[n]})$ interact with segments in $\textsc{Leaves}(\mathcal{T}_{\mathrm{ch}_r[n]})$, thus forming a larger cluster of pairwise interacting segments in n. An example BPFT can be seen in Fig. 1. For convenience, each internal node in the figure has a numerical index. The internal node 3 represents a cluster of interactions between the segments $\alpha_1\beta_2\beta_6$ in an intermediate fold S_t ($1 < t < T$). The cluster has formed because the first helix (α_1) started to interact with the β-sheet made by the second and sixth strands (node 1). Other portions of the tree can be similarly interpreted. The simulation ends in the predicted fold which is symbolically represented by the root node; its children indicate that the final structure was predicted by joining the first strand (β_1, left child) with one or more of the segments (i.e. leaves) dominated by node 7.

2.3 BPFT construction algorithm

The pseudo codes of Algorithms 1 and 2 describe the procedure that we apply to build BPFTs. Parameters of GenerateBPFT are the set of indexed SSEs of the protein and the ordered sequence of structures found along the whole simulation trajectory, from $t = 1 \dots T$. The BPFT $\mathcal{T}$ is built bottom-up, from the leaves to the root node. The structure returned by Algorithm 1 describes the assembling history of S_T as a hierarchical set of SSEs pairing events. Steps 1 to 6 initialise the partial tree with m leaf nodes, each one representing an isolated SSE not interacting with the others. This corresponds to the initial structural configuration before the configurational search starts. New nodes are then added whenever, moving from step t to $t + 1$ of the trajectory, we find that new SSEs interactions have been formed. If we add a new node (a new potentially larger cluster), its children (subclusters) are not necessarily searched among the last added nodes, because these might not longer represents clusters in S_{t+1} (i.e. at time $t + 1$, SSEs links in S_t may have been broken as well). For these reasons, $\mathcal{T}$ maintains a reference to a subset of its nodes, the 'frontier', each node pointing to a cluster of SSE interactions that are present in the fold at the current time

Algorithm 1 GENERATEBPFT($\{s_1 \dots s_m\}$, $\{S_1 \dots S_T\}$)

1: $\mathcal{T} \leftarrow \emptyset$
2: **for** $i \leftarrow 1 \dots m$ **do**
3: $v \leftarrow$ CREATENODE($\{s_i\}$)
4: ADDNODE($\mathcal{T}$, v, $\emptyset$)
5: $\mathcal{T}.frontier \leftarrow \mathcal{T}.frontier \cup \{v\}$
6: **end for**
7: $\mathcal{C}_T \leftarrow$ contact map of S_T
8: $\mathcal{NC} \leftarrow \emptyset$ {*Contacts of* S_T *formed so far*}
9: **for** $t \leftarrow 1 \dots T$ **do**
10: $(\mathcal{C}_t, \mathcal{CC}_t) \leftarrow$ (residue, coarse) contact maps of S_t
11: $\mathcal{NC}_t \leftarrow \mathcal{C}_t \cap \mathcal{C}_T$ {*Contacts of* S_T *in current fold*}
12: **if** $\mathcal{NC}_t \setminus \mathcal{NC} \neq \emptyset$ **then**
13: UPDATETREE($\mathcal{T}$, $\mathcal{CC}_t$) {*Update tree if there are new native contacts*}
14: **end if**
15: $\mathcal{NC} \leftarrow \mathcal{NC}_t$ {*Update the set of temporarily formed native contacts*}
16: **end for**

step. Whenever we add a new node, its cluster must describe pairings between smaller subclusters of the current fold, so that the children are always searched among the frontier nodes. At time step 0, the structure is assumed to contain only isolated segments (not forming any interaction), so that the frontier is made with only leaf nodes (Step 5). In order to build and complete the tree, the trajectory is monitored searching for events that involve SSEs interactions. This is accomplished by looking, at each step, at the formation of contacts among residues in different SSEs, with the constraint that these contacts exist in the final predicted fold S_T. We motivate this choice from the assumption that the topology of the protein, here represented by the contact map $\mathcal{C}_T$ of S_T in Step 7, has an influence on the corresponding pathway [1]. In Step 8, $\mathcal{NC}$ keeps trace of the set of contacts of S_T formed until a given time step of the simulation. From Step 9 to 16, the algorithm analyses the structure S_t of each time step t of the trajectory: $\mathcal{NC}_t$ is assigned to the set of contacts of S_T formed in S_t (Step 11) and if new contacts are formed with respect to those formed in steps $1 \dots t-1$ (step 12), the tree is updated by a call to UPDATETREE (Step 13) passing as parameter the coarse contact map of S_t[4]. Alg. 2 first updates $\mathcal{T}$'s frontier such that its nodes correctly represent clusters of SSE interactions of the last visited structure (steps 1-6). For each frontier node n, segments in LEAVES($\mathcal{T}_n$) form the vertexes of a graph with edges between interacting SSEs in the last coarse contact map. The nodes are partitioned into subsets of pairwise interacting SSEs[5] (Step 2). If there is only one component, the segments of n represent a portion of the interactions in the last fold. Hence the node is still in the frontier and will be searched for the next pairing operations. If this is not the case, the frontier is

[4] A coarse contact map represents SSEs interactions and is defined similarly to a residue contact map: SSEs are used instead of residues, see e.g. [16].
[5] PARTITION($\cdot$) is implemented by computing the connected components of the graph using a simple depth first search.

Algorithm 2 UPDATETREE($\mathcal{T}$, $\mathcal{CC}$)

1: **for** $n \in \mathcal{T}.frontier$ **do**
2: $C \leftarrow$ PARTITION(LEAVES($\mathcal{T}_n$), $\mathcal{CC}$)
3: **if** $|C| > 1$ **then**
4: $\mathcal{T}.frontier \leftarrow$ UPDATEFRONTIER($\mathcal{T}$, n, C)
5: **end if**
6: **end for**
7: **for** $(s_i, s_j) \in \mathcal{CC}$ **do**
8: $v \leftarrow \{x \in \mathcal{T}.frontier \mid s_i \in \text{leaves}(\mathcal{T}_x)\}$
9: $w \leftarrow \{x \in \mathcal{T}.frontier \mid s_j \in \text{leaves}(\mathcal{T}_x)\}$
10: **if** $v \not\equiv w$ **then**
11: $n \leftarrow$ CREATENODE()
12: ADDNODE($\mathcal{T}$, n, $\{v, w\}$) {LEAVES($\mathcal{T}_n$) = LEAVES($\mathcal{T}_v$)$\cup$ LEAVES($\mathcal{T}_w$)}
13: $\mathcal{T}.frontier \leftarrow \mathcal{T}.frontier \cup \{n\} \setminus \{v, w\}$
14: **end if**
15: **end for**

updated by a call to UPDATEFRONTIER (not shown) where $\mathcal{T}_n$ is visited and n is replaced by its first descendants that contain the clusters in C. In steps 7-14, we search for SSE interactions in the current fold (given by $\mathcal{CC}$) that are not represented by the partial tree built so far. The frontier nodes are searched for those containing two interacting SSEs (steps 8-9). If the corresponding nodes are distinct, it means that no node in $\mathcal{T}$ encodes the interaction so that a new node is formed as a parent of the two nodes; the frontier is updated accordingly.

2.4 Mining frequent pseudo-folding patterns

We briefly discuss an efficient procedure used to capture simple descriptions of the dominant features of pseudo-folding simulations, as represented by BPFTs, and then compare these descriptions with known experimental folding facts of a set of proteins considered in previous studies [22]. In this way, we test the protocol for its ability to mimic the real folding process.

We wish to discover patterns in pseudo-folding pathways represented by BPFTs. Since the simulator is stochastic, given the same set of restraints, any two runs could output different BPFTs varying both in shape and size. To tackle this, we represent a pseudo-folding landscape by the distribution of labelled subtrees in pseudo-folding pathways represented as BPFTs. Patterns can be naturally thought of as being the common subtrees of a set of BPFTs. We search for these patterns by mining the most frequent subtrees [5]. We have applied the methodology described to the set of proteins considered in [22]. For each protein, the reconstruction algorithm ran 200 times with the restraints defined by the native contact map, thus obtaining a sample of possible trajectories, hence BPFTs, leading to the correct native structure. From these trees we mined the most frequent subtrees and compared the events they describe with known facts about the folding of the protein under study. We have found significant correspondences between our artificial samples and the experimental evidence. Most

Table 1. Top 5 most frequent sub BPFTs mined from a sample of reconstruction traces (chain 1O6XA). Each subtree's support is the normalised frequency wrt to sample size.

Rank	Support	SubBPFT
1	0.85	$\beta_1\beta_2$
2	0.70	$(\alpha_2(\beta_1\beta_2))$
3	0.69	$(\beta_3(\alpha_2(\beta_1\beta_2)))$
4	0.63	$(\alpha_1(\alpha_3(\alpha_2(\beta_1\beta_2))))$
5	0.14	$(\beta_2(\beta_3(\beta_1\beta_2)))$

of the events described in the literature appear as encoded in one or more of the most frequent subtrees. For instance, Table 1 shows the top five frequent subtrees for one of the chains under study (PDB code 1O6XA). It is known that the folding nucleus of 1O6X is made by packing of the second helix with the β-sheet formed by $\beta_2\beta_1$ [22]. Indeed, we found the second most frequent subtree $(\alpha_2(\beta_1\beta_2))$ as perfectly describing this event, where the most frequent subtree indicates the formation of the β-sheet $\beta_2\beta_1$.

3 Kernels on BPFT

We develop kernels (i.e. similarity measures) between BPTFs to investigate the informative content of the proposed features by learning techniques. For efficiency issues, we turn BPFTs into ordered trees, by imposing a total order on the leaves according to the relative position of the SSEs in the protein sequence. We focus only on *complete* subtrees, that is subtrees that contain all descendants of the subtree root up to the leaves of the original tree. We can now apply a set kernel on complete subtrees by decomposing each BPFT into the set of its complete subtrees, and comparing two BPFTs by summing up all pairwise comparisons between elements of the two sets:

$$K(\mathcal{T}, \mathcal{T}') = \sum_{n \in \mathcal{T}} \sum_{m \in \mathcal{T}'} k(\mathcal{T}_n, {\mathcal{T}'}_m) \tag{1}$$

To keep things simple, we compare subtrees by the delta function $k(\mathcal{T}_n, {\mathcal{T}'}_m) = \delta(\mathcal{T}_n, {\mathcal{T}'}_m)$. The overall kernel computes the similarity between two BPFTs by counting the number of complete subtrees (i.e. partial pseudo-folding representations) they have in common. In the following, we refer to this kernel as *cluster-node* kernel. Note that by imposing a canonical ordering to BPFTs and having no timestamps in the internal nodes, we only care of the hierarchy of interactions between SSE clusters, ignoring differences due to the relative timestamp of events involving non-overlapping clusters. Such invariance aims at modelling cases in which separate portions of a chain fold independently, a situation which is known to take place in nature. Comparison of complete subtrees of size one (i.e. leaves) provides an informative contribution whenever two simulations rely on different SSE predictions. Note that the cluster-node kernel does not retain

information of temporary interactions which form during the process but are not preserved in the final structure. Moreover, the kernel compares SSE clusters, but it does not consider the specific SSE pairs responsible for the formation of a cluster, apart from those formed by exactly two SSEs.

By this, we also consider a variant where the description of internal BFPT nodes is enriched with three different sets of SSE pairs: those which began interacting when the cluster formed; those which preserved their interaction; those whose interaction was lost when the cluster formed. A new subtree kernel accounting for such information is defined as follows:

$$k(\mathcal{T}_n, \mathcal{T'}_m) = \delta(\mathcal{T}_n, \mathcal{T'}_m) + \sum_{\substack{i \in F(n) \\ j \in F(m)}} \delta(i,j) + \sum_{\substack{i \in P(n) \\ j \in P(m)}} \delta(i,j) + \sum_{\substack{i \in L(n) \\ j \in L(m)}} \delta(i,j) \quad (2)$$

where $F(n), P(n), L(n)$ represent the sets of pairwise SSE interactions which are respectively formed, preserved and lost in the cluster corresponding to node n. This kernel is dubbed *pairwise-interaction* kernel in the following. Note that the kernels described in this section are conceived for measuring similarities between BPFTs originating from simulations on the same protein sequence, even if with possibly different restraints. The extension to inter-protein similarities is subject of ongoing investigation.

4 Experiments and discussion

Given a predicted structure and its pseudo-folding pathway, we first test whether the corresponding BFPT retains some information about the distance between the predicted and native (unknown) fold. We thus generated a data set of pseudo-folding simulations for 250 non-redundant PDB chains (maximum 25% mutual sequence similarity for any two chains) considered in [4] by running 3Distill (see Sec. 2.1) using restraints obtained from four increasingly noisy contact maps: the native one, contact maps obtained from PDB templates with a max sequence identity threshold at 95% and 50% respectively, and an *ab initio* predicted map. For each of these maps, 200 simulations were run, resulting in 800 structures for each protein. The TM-score function [23] was used to measure the distance between the predicted and native fold. BPFTs were generated from the pseudo-folding processes using Alg. 1, and the two kernels defined in Section 3 were employed to measure pairwise BPFT similarities. The kernels were normalised as suggested in [6], i.e. the input vectors are normalised in feature space and centered by shifting the origin to their center of gravity. Figures 2(a) and 2(b) show the kernel matrices obtained averaging over structures with similar quality, for the cluster-node and pairwise-interaction kernel respectively. Each $([i, i+1], [j, j+1])$ bin in the maps represents the average kernel value between two structures whose TM-score to the native is in the $[i, i+1]$ and $[j, j+1]$ interval respectively. The kernel values increase with the TM-score to the native in both cases. Interestingly, the kernels discriminate pseudo-folding simulations when TM-score $\in [0.3, 0.4]$, a range of thresholds that separates poorly predicted

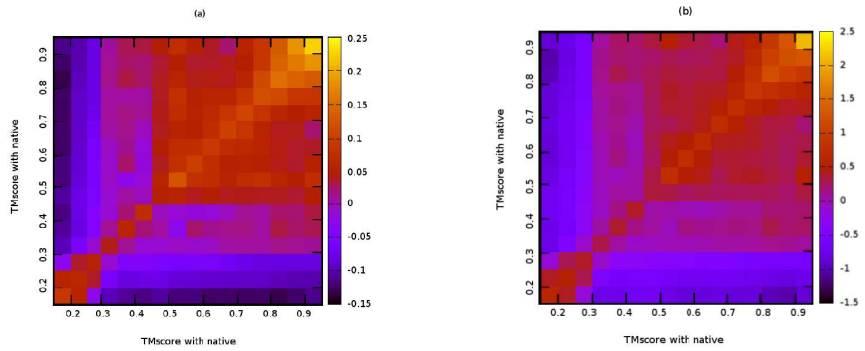

Fig. 2. Kernel matrix obtained averaging over structures with similar quality measured as TM-score with the native: (a) cluster-node kernel (b) pairwise-interaction kernel

and native-like folds [23]. This depends on the distribution of the scores, which presents (data not shown) a separation of the instances on the previous interval. The kernels are clearly modelling some aspects of the given distribution.

In a binary classification setting, the relatedness of a certain kernel function to the target can be measured by the Kernel Target Alignment (KTA) [6], defined as the normalised Frobenius product between the kernel matrix and the matrix representing pairwise target products. In our setting, a binary target can be obtained using a threshold on the TM-score with respect to the native structure (we chose 0.4, see above). Figure 3 (left) reports an histogram of KTA values for our two kernels. About half of the proteins show an alignment greater than 0.15. As expected, the more informed pairwise-interaction kernel has an overall better alignment.

As a final test for the discriminative power of our two kernels, we clustered protein structures and their simulations using spectral techniques [20]. Given a matrix S of pairwise similarities between examples, they compute the principal eigenvectors of a Laplacian matrix derived from S, and apply a simple clustering algorithm, like k-means or recursive bi-partitioning, on the rows of the eigenvector matrix. As suggested in [20], we employed the multicut algorithm [13], combined with a k-means with 5 runs initialized with orthogonal centers and 20 runs initialized with random centers. Since we are mainly focussing on separation between decoys and native-like structures, the number of searched clusters was set to two. We then measured the quality of clustering using the correlation between (1) a binary value that indicates the cluster assigned to the BPFT (2) the TM-score to the native of the corresponding predicted structure. Figure 3 (right) shows histograms of the correlations obtained by clustering with the two kernels. Albeit simple, the cluster-node kernel shows a significant correlation for a large fraction of tested proteins. For 80% of the proteins, the correlation is greater than 0.15. The average correlation per protein is 0.4, and goes up to 0.47 using the more informed pairwise-interaction kernel. With this kernel we see a consistent increase of the number of cases where the correlation is more than

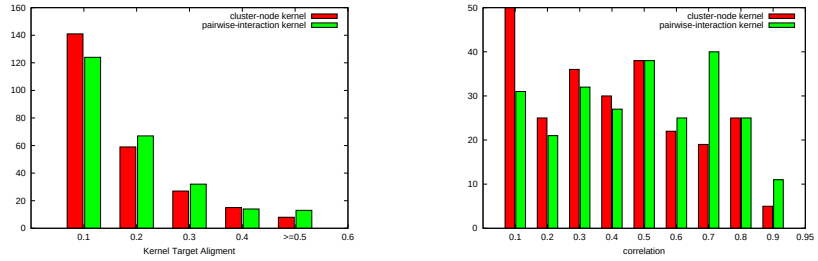

Fig. 3. Histogram of: (left) KTA values, binary targets obtained with TM-Score threshold with the native set to 0.4; (right) correlation between cluster assignment and TM-score with native structure. Results are for cluster-node and pairwise interaction kernel.

0.5. Noticeably, the ability of clustering the predicted models increases by using additional dynamical information, i.e. pairwise intermediate SSE interactions. Finally, the correlation between clustering quality and KTA value is about 0.6 for both cluster-node and pairwise-interaction kernel, thus showing a certain degree of match between the two analyses. An in-depth look at the results showed that high correlation is obtained when structures generated using the same restraints are assigned (with possibly few exceptions) to the same cluster. For the simple kernel, 42 proteins have correlation higher than 0.7. In 23 of these cases, structures generated from the native contact map are separated from all other structures, in 17 cases structures from native and 95% identity template maps are clustered together. In 1 case *ab initio* generated structures are clustered together with those from native maps, and all template-based structures are assigned to the other cluster. The last case (chain A of PDB entry 1OJH), is an interesting exception as indeed *ab initio* generated structures had a better TM-score with the native than all template-based ones.

5 Conclusions and Future Work

This study was motivated by the idea that reasonable computational abstractions of the protein folding process may contain useful information about the final protein structures. We focused on a specific pseudo-folding algorithm based on stochastic reconstruction from contact maps and empirically found that the information extracted from the pseudo-folding process does indeed allow us to define a discriminant measure of similarity (expressed by a kernel function) between the corresponding final protein structures. In particular, we found that (1) the folding abstraction used here agrees with availalble experimental evidence about the folding of some proteins, and that (2) our kernels are able to separate good and poor reconstructions of the same protein.

These findings pave the way towards the use of pseudo-folding features in the analysis and discrimination of protein structures. Attaining such a goal from a machine learning perspective requires a generalisation of the current kernel to compare pseudo-folding trees associated with different proteins.

References

1. Alm, E., Baker, D.: Prediction of protein-folding mechanisms from free-energy landscapes derived from native structures. PNAS **96** (1999) 11305–11310
2. Baker, D.: A surprising simplicity to protein folding. Nature **405** (2000) 39–42
3. Bau, D., Martin, A. J. M., Mooney, C., Vullo, A., Walsh, I., Pollastri, G.: Distill: A suite of web servers for the prediction of one-, two- and three-dimensional structural features of proteins. BMC Bioinformatics **7**:402 (2006)
4. Bau, D., Pollastri, P., Vullo, A.: Distill: a machine learning approach to ab initio protein structure prediction. Analysis of Biological Data: A Soft Computing Approach. Bandyopadhyay, S., Maulik U., Wang, J. eds., (2007) World Scientific
5. Chi, Y., Nijssen, S., Muntz R. R., Kok, J. N.: Frequent Subtree Mining–An Overview. Fundamenta Informaticæ **66**:1-2 (2005) 161–198
6. Cristianini N, Kandola J., Elisseef A., Shawe-Taylor J.: On kernel-target alignment, innovations in Machine Learning, (2006) 205–256
7. Dinner, A. R., Sali, A., Smith, L. J., Dobson, C. M., Karplus, M.: Understanding protein folding via free-energy surfaces from theory to experiments. Trends Biochem. Sci. **25**:7 (2000) 331–339
8. Dobson, C. M.: The structural basis of protein folding and its links with human disease. Phil. Trans. R. Soc. Lond. **356** (2001) 133–145
9. Dobson, C. M.: Protein folding and misfolding. Nature **426** (2003) 884–890
10. Friesner, R. A., Prigogine, I., Rice, A. S.: Computational methods for protein folding. Advances in Chemical Physics **120** (2002) John Wiley and Sons
11. Hockenmaier, J., Joshi, A. K., Dill, K. A.: Routes are trees: the parsing perspective on protein folding. Proteins **66** (2007) 1–15
12. Maity, H., Maity, M., Krishna, M., Mayne, L., Englander, S. W.: Protein folding: the stepwise assembly of foldon units. PNAS **102** (2005) 4741–4746
13. Meila, M., Shi J.: A random walks view of spectral segmentation. AISTATS (2001)
14. Abstracts of the CASP7 conference, Asilomar, CA, USA, 26-30/11/2007. Available online at: http://www.predictioncenter.org/casp7/Casp7.html
15. Plaxco, K. W., Simons, K. T., Ruczinski, I., Baker, D.L Topology, stability, sequence and length. Defining the determinants of two-state protein folding kinetics. Biochemistry **39** (2000) 11177–11183
16. Pollastri, G., Vullo, A., Frasconi, P., Baldi, P.: Modular DAG-RNN architectures for assembling coarse protein structures. J. Comp. Biol. **13**:3 (2006) 631–650
17. Tosatto, S.C.: The victor/FRST function for model quality estimation. J. Comp. Biol. **12**:10 (2005) 1316–1327
18. Vendruscolo, M., Kussell, E., Domany, E.: Recovery of protein structure from contact maps. Folding and Design **2** (1997) 295–306
19. Vendruscolo, M., Paci, E., Dobson, C., Karplus, M.: 3 key residues form a critical contact network in a protein folding transition state. Nature **409** (2001) 641–645
20. Verma, D., Meila, M.: A comparison of spectral clustering algorithms. TR 03-05-01, University of Washington (2001)
21. Wright, C. F., Lindorff-Larsen, K., Randles, L. G., Clarke, J.: Parallel protein-unfolding pathways revealed and mapped. Nature Struct Biol **10** (2003) 658–662
22. Zaki, M. J., Nadimpally, V., Bardhan, D., Bystroff, C.: Predicting protein folding pathways. Bioinformatics **20** (2004) i386–393
23. Zhang, Y. and Skolnick, J.: Scoring function for automated assessment of protein structure template quality. Proteins **57** (2004) 702–710