Region Matching on Protein Surfaces

Alper Sarikaya

May 14th, 2012

Abstract

Protein characterization *in silico* is a difficult problem that could have large repercussions for the field of biochemistry. Whereas the structure for a crystallized protein is easy to determine, it is a time-intensive task to characterize a protein. We propose a novel approach to characterizing functional interfaces on proteins by using region matching on a determined structure. These regions are comprised of descriptors that contain shape and chemical information. These local, multi-scale descriptors are grouped together into regions based on techniques from the computer vision based community. Using this model, we expect to see an advantage in functionality matching over the canonical protein shape matchers by incorporating chemical data at multiple scales.

1 Introduction

Proteins have a whole host of roles in the cell, and elucidating the roles of individual proteins is a very challenging problem. Once crystallized, biochemists can use the structure of the protein to help inform the functionality of the protein. However, much of the information gleamed from visual inspection of the protein surface is highly empirical, and depends on the set of assumptions that the inspector arrives with. One of the more traditional rules-of-thumb for characterizing functionality is the lock-and-key metaphor: the protein acts as the lock, and the incoming ligand to be bound acts as a key. The idea is that the protein surface lends itself to selectively binding the exact ligand, so the shape of the protein gives clues to which ligand it binds. Discovering which ligand binds to a protein is a critical step for assessing protein functionality, and is frequently a time-consuming step in discovering protein function.

While the lock-and-key metaphor holds for small ligands such as calcium atoms, NAD^+, and ATP, larger molecules such as DNA, protoporphyrin, and other proteins demand a significantly larger interface area on the protein. Contemporary methods for determining protein interfaces utilize the lock-and-key metaphor for divots on the protein surface, appropriately called pockets. Given an un-annotated protein structure, many solvers attempt to find pockets, and then attempt to classify the shape of the selected pockets based on a training set [1, 2]. Given small ligands, these solvers are very reliable at predicting pockets and classifying the ligands. For larger ligands (20+ atom molecules) and those with an affinity for significant charge, hydrogen bonds, or hydrophilicity, the canonical methods break down.

While many of these ligand predictors use just shape to classify, we propose using and extending our idea of multi-scale local descriptors. Based on the idea that a ligand only depends the local environment in which it is situated, we strive to maintain and collect chemical and physical information as well as shape information at regular points over the proteins surface. Additionally, we want to extend the idea of pocket matching to region matching. This breaks out of the limitation of limiting prediction to small pockets, and provides an avenue for accurate prediction of RNA, DNA, and protein binding.

2 Approach

An open question in biochemistry is how to determine the specificity of a protein binding to a specific DNA sequence. With the canonical ligand matchers (*cf.* [3]), using just shape to determine a fit does not lend itself to specifying specificity; it might be apparent that the protein binds DNA at that location, but lacks the information to inform the classifier exactly which bases and chemical interactions occur in that region. This is precisely the interaction that we propose to capture with both the multi-scale local descriptors and the region matching.

To generate the local descriptors across the surface of the protein surface, the solvent-excluded surface (SES) is first computed. This process is done by rolling a sphere of radius 1.5 Å(approx. the radius of a molecule of water) over the proteins surface to determine the solvent-reachable areas and pockets. With the SES computed, a triangle mesh is then constructed in order to provide reference points over the entire surface of the protein. At each of these reference points (vertices in the mesh), many physical and shape attributes are computed at different scales:

The ranges of potential interfaces for a specific ligand is extremely broad. Even a cross-section of PDB proteins that only bind, say, ATP, have incredible variation on how they latch onto their desired ligand. Using just a single descriptor to categorize the ligand interface seems like an insurmountable and intractable task. The advantage of our descriptors is their frequency; binding interfaces can be compared by combining these descriptors in a flexible way.

Our idea of generating and using local descriptors for categorization has become a canonical method in the computer vision community (survey: [5]). Small descriptors, given the correct dimensions and scales, can be made invariant to undesired effects, and can be aggregated together to represent a larger region. Local descriptor allegories can be drawn to image descriptors such as SIFT [6] and SURF [7], and many ways for building regions out of those descriptors have been studied, including template methods that enforce spatial relationships [8].

Item	Scales	Notes
% Visibility		Outside world visible from pt
Non-polar backbone		Distance to item
Aromatic sidechain		Distance to item
Aliphatic sidechain		Distance to item
N-backbone atom		Distance to item
O-backbone atom		Distance to item
S-backbone atom		Distance to item
Amide-N sidechain		Distance to item
Amide-O sidechain		Distance to item
W sidechain		Distance to item
OH sidechain		Distance to item
Charged O sidechain		Distance to item
Charged H sidechain		Distance to item
Patch anisotropy	1.6, 3.2, 4.8, 6.4, 8 Å	Sampled at different levels
Path curvature	1.6, 3.2, 4.8, 6.4, 8 Å	Sampled at different levels
Path curvature var.	1.6, 3.2, 4.8, 6.4, 8 Å	
Hydropathy	1.6, 3.2, 4.8, 6.4, 8 Å	
Charge	1.6, 3.2, 4.8, 6.4, 8 Å	
H-bond donor		Distance to nearest donor
H-bond acceptor		Distance to nearest acceptor

Table 1: Listing of the 40 descriptive features encoded in each mesh vertex [4]

To match and grow regions, we can use the computed local descriptors in conjunction with one another to define a region. Given a point's feature vector, and the feature vectors of its k neighbors in its vicinity, it is conceivable to find a similar region by matching the center of the given region with n points in the protein to compare. Those n points will be potential region matches, but those n points need to be expanded to see if the k neighbors of the given region match the k neighbors around the n points. The best match of the n center points will not necessarily match to the best region. Therefore, my algorithm depends on both of these parameters: k starting seeds for potential matched regions, and n vertex neighbors in each region. Varying these two parameters changes the sensitivity of generating region matches. The pseudo-code is below in Algorithm 1.

There are several aspects of this algorithm that need clarification. The first is finding the closest point in terms of descriptor distance. Here we are using a simple L_2 distance metric (Euclidean distance) that may not be representative or accurate distance metric depending on the chemical properties of the ligand (see Discussion section). Searching a 40-dimensional space of over ten thousand items for the nearest match is a very expensive computation, so we take advantage of an Approximate Nearest Neighbor (ANN) library. This library takes algorithmic advantages over standard iteration by utilizing data structures such as *kd-trees* and *box-decomposition* trees, allowing for compatible matches to be found in a fraction of the time.

For finding neighboring points in three-dimensional space, we again use the ANN library to organize the points for easy retrieval. The positions of the vertices are determined by fitting the solvent

Algorithm 1 Region-Matching(k, n) – Find region matches in two homologous proteins

```
Require: Input integers $k, n > 0$.
Require: Two proteins compareFrom, compareTo
Require: Seed center $p_c$ on compareFrom
 1: origNN, origNNDist $\leftarrow$ find closest $k$ points in space to $p_c$ on compareFrom
 2: $d_t \leftarrow \max(\text{origNNDist})$
 3: for each origNN do
 4:     $M \leftarrow$ find $n$ closest descriptor matches on compareTo
 5: end for
 6: $T \leftarrow$ build tree with 3D coordinates from points in $M$
 7: regCenter, regCenterDist $\leftarrow$ best $n$ possible $p_c$ descriptor matches on compareTo
 8: for each regCenter do
 9:     $D^i \leftarrow$ query $T$ with regCenter's 3D coordinates, limit to $k$ matches
10:     if $\max(D^i) < d_t$ then
11:         This regCenter is a probable region!
12:     else
13:         This is not a likely region
14:     end if
15: end for
```

excluded surface with a triangle mesh that is facilitated by *trimesh2*. This allows easy selection of neighboring vertices when growing/matching a region. See Figure 1 for an example.

For the testing of our hypothesis, we will use our in-house molecular visualization tool to quickly visualize, abstract, and color arbitrary protein molecules in the PDB format [4]. This application creates the multi-scale descriptors, allows for interaction with the protein, and can color points based on similarity. Although the code is of academic quality, there are hook-in points where differing distance metrics and methods can be placed. The application also reads in all protein chains into the viewer, and calculates the *shadows* that the ligand chains cast onto the protein domains. This is useful extra information for debugging, but is noted that shadow information will not be provided in a given uncharacterized protein structure.

The region matching functionality can be dropped in, and with some modification to the coloring code, the vertices in the protein surface can visualize the matched regions. I added a new interaction mode for the user to manipulate and compare two arbitrary proteins by region. The anticipated workflow (as a user) is as follows:

1. The user loads two proteins that bind similar ligands
2. The user then selects the “Region” interaction mode and selects a point in the *shadow* on one protein
3. Regions are highlighted:
    a. On the comparison protein, the top 3 similar regions are highlighted and colored according to rank

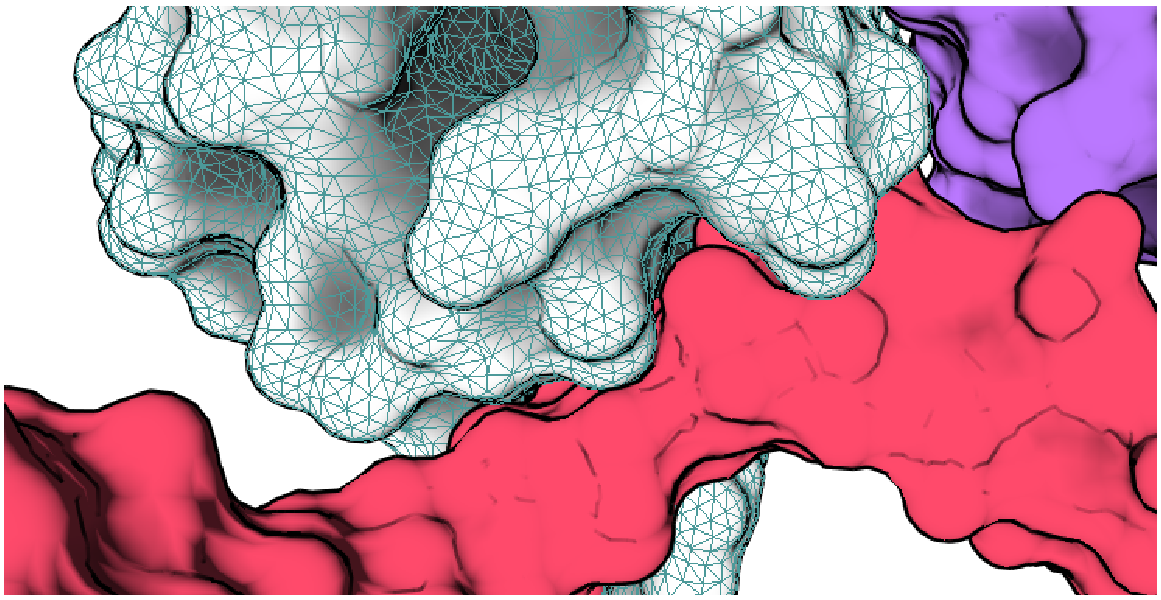

Figure 1: A representation of the triangle mesh defining the SES of 3DFV (the red chain is one strand of bound DNA, purple is another domain). At each vertex, the descriptors from Table 1 are computed

   b. On the original protein, the region grown from the user-picked point is highlighted

4. The user can interact with the two proteins to get a full three-dimensional picture of the matched regions

While the user workflow has been implemented, I've unfortunately run out of time to implement the coloring in the molecular visualizer. My plan over the summer is to implement this and experiment with different ways of growing and matching similar regions. Some additional discussion on this matter is in Section 4 (Discussion).

3 Emperical Evaluation

Algorithm 1 above was tested textually. The molecular visualization tool has a facility to dump the feature vector of the first chain in the protein (often the chain under test, and the white chain in all figures in this paper). I made some amendments to label just the ligands that we are interested in, ignoring other protein domains and smaller ligands (e.g. calcium atoms). This dumped feature vector has one vector on a line, each with its assigned ID number, x, y, and z coordinates in space, the 40 predictive dimensions outlined in Table 1, and several dimensions that allow for tracebacks to the original PDB file (e.g. residue ID, closest AA, etc.).

The python script as written takes two input feature vectors and a given vertex ID with which to grow the region to be matched. Some sample output is below (Region-Matching() with $k = 3, n = 8$):

```
Testing scipy implementation
```

```
Parsing file 1-1GAT
Parsing file 1-3DFV
Original point 1749 (-1.691, -5.004, 30.577)
Found neighbor 1749 (2.122, -0.383, 1.221)
Distance: 0.0
Found neighbor 10534 (2.63, -0.208, 1.253)
Distance: 0.538249941941
Found neighbor 6296 (1.551, -0.608, 1.376)
Distance: 0.633001579777
Distance threshold: 0.633001579777

Matching point 2257.0 (7.887, -9.078, 45.061)
Matching point 6946.0 (8.445, -9.227, 45.081)
...
Matching point 9760.0 (7.172, -8.545, 44.952)
Matching point 9761.0 (7.172, -7.911, 44.952)

Testing vertex 6948 for matches.
potRegionMatches: [2 3 1 0]
potRegionDists: [ 0.          0.57820585  0.62487199  0.87036429]
Vertex 6949.0 (7.777, -9.545, 45.54) distance: 0.0
Vertex 6946.0 (8.445, -9.227, 45.081) distance: 0.578205845699
Vertex 2257.0 (7.887, -9.078, 45.061) distance: 0.62487198689
Farthest point in NN match is 0.870364291547 ; wanted < 0.633001579777
...
Testing vertex 9760 for matches.
potRegionMatches: [7 5 6 8]
potRegionDists: [ 0.          0.61637489  0.63287044  0.634     ]
Vertex 6985.0 (7.758, -8.416, 44.811) distance: 0.0
Vertex 9759.0 (7.172, -9.115, 45.227) distance: 0.616374885926
Vertex 9761.0 (7.172, -7.911, 44.952) distance: 0.632870444878
Found matching region!!!!!!!
```

In the above output, two different DNA-binding proteins are compared. The first section corresponds to the first line of Algorithm 1; the `Distance threshold` line corresponds to line 2. The next section builds the closest matches on the other protein (lines 3-6), while the last couple of sections correspond to the iterations of the **for** loop that filters out region outliers (lines 7-15). Sections are eclipsed with elipses for berevity. In the output above, a matching region is found after generating matching region neighbors in the original protein (1GAT) to the potential region neighbors protein being compared (3DFV).

To verify the output and the conclusions reached by the region matcher, the textual tool is first given the same protein twice to see if it matches identical regions on both proteins. The main two proteins that I've been working with so far are two specific DNA binders — that is, they both only bind DNA sequence of a distinct sequence of bases. These two proteins are 1GAT and 3DFV and are shown below. Here, we try to rectify the textual match given by the above output to the visual surface of the two proteins.

In Figure 2, the little red dot on the finger of the protein is the matched point based on the mutual

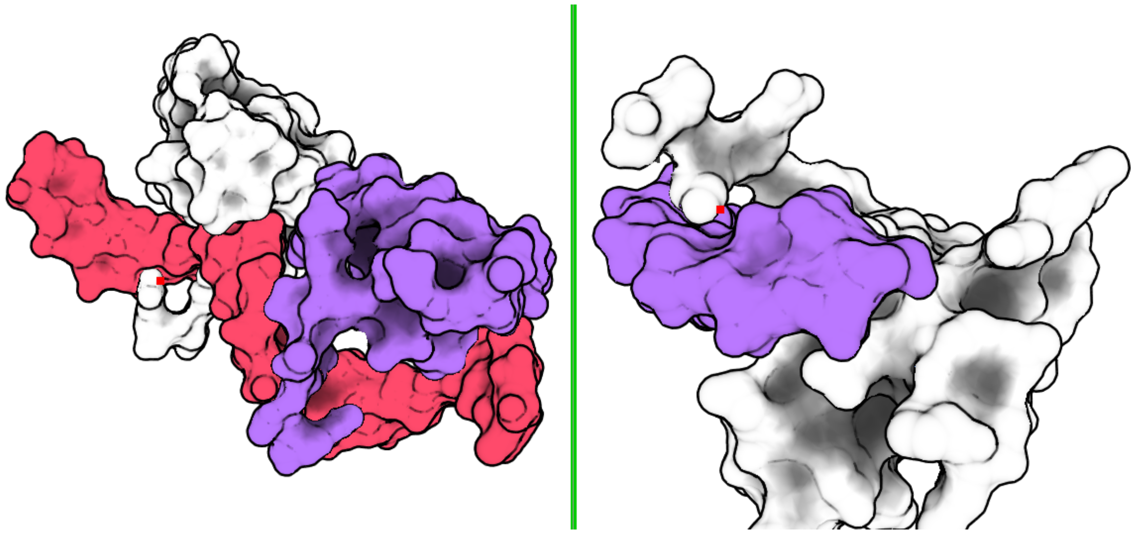

Figure 2: An example a match between PDB proteins 3DFV (left) and 1GAT (right). The white chain in both molecules is the one under test. One strand of DNA is shown (red on right, purple on left). In both structures, one helix of DNA is omitted as it occludes the matched point. The purple chain on 3DFV is another (identical) protein domain.

match of its surrounding environment. It has nothing to do with the position or interaction of the DNA chain directly interfacing both regions; both regions are matched solely on normalized feature vector. The results of the region mapper are pretty good – the 'finger' that wraps over the bound region of DNA has a functionally similar shape and chemical information, and on manipulation of both structures, this finger is wedged right between the two double-helices between bases ACA.

4 Discussion

The use of local, multi-scale descriptors has good promise of being a flexible interface matcher. It manages to break the paradigm of using just pockets to predict binding sites for protein classification, and shows promise for classifying and determining specific interactions with individual DNA bases.

This research is deep in the exploratory stage. There are a couple avenues I'd like to explore over the summer. The first is varying the fashion in which the region is grown. Instead of traveling over edges in the triangle mesh, I would like to use geodesic distance instead [9]. This metric is a technique for accurately calculating distance along a triangle mesh surface by following a line on the surface between vertices instead of tracing edges. A ripe tool for visualization would involve the use of exponential maps [10]. Exponential maps can be thought of as a way to flatten a complex, curved surface into a flat surface, in part by using the properties of the geodesic distance.

As seen in Figure 3, a point of interest is in the center. The geodesics are the directional vectors that radiate out from the picked center point. While the figure shows a simple sphere, one can imagine that on a more complicated surface the rays would accurately follow the proper geodesic distance from the center point. The *Hopf-Rinow theorem* that states that the exponential map is defined everywhere only if the surface is smooth manifolds. Unfortunately, this is not the case in

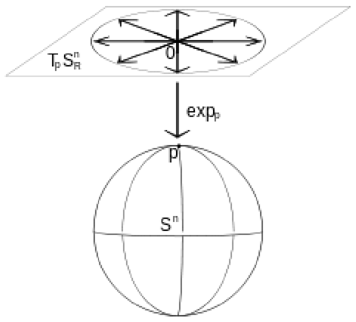

Figure 3: A sample representation of an exponential map centered at point p.

the solvent-excluded surfaces of the protein. Regions that are not smooth produce fractures on the exponential map that result in a discontinuous plane. Our anticipation based on preliminary exploration is that the exponential map is wholly defined in a local region (< 10 Å), which should be significant for our region matching needs.

There are a couple caveats to this approach of region matching, however. The first and formost assumption we must acknowledge is the fact that we are doing all this functional prediction here *in silico*. Our prediction of binding affinity and compatibility is only as good as our modeling, and by extension the descriptors that we've chosen. The chemistry of protein interaction can only be definitively proven in the wet laboratory – tools of this nature should therefore serve to be a useful tool in predicting and guiding time-consuming physical experiments.

Secondly, each descriptor is weighted equally. Realistically, each descriptor should be weighted for each specific protein-ligand and protein-protein interaction. For example, say a ligand is non-polar and has neutral charge; now the metric of charge makes no sense and should have no effect on binding affinity. However, when we try to find distance or the nearest comparable feature vector, dimensions that may not apply or apply less will still be taken into account, due to the fact that we utilize Euclidean distance (for lack of a comprehensive distance metric). This limitation has been brought up in previous work [4], and I propose to use machine learning techniques to learn an accurate model by weighting descriptor dimensions according to the specific binding interface.

There are several avenues to learning descriptor weights. The first is to do a per-interface learning of dimensional parameters. Given labeled interfaces that bind the same ligand (essentially the *shadow* referenced earlier), an expectation-maximization algorithm can gradually learn the corresponding relative weights of the predictive dimensions. Another possibility that utilizes labeled data is to perform a special kind of principal component analysis (Kernel PCA) in order to determine the key dimensions for each unique ligand [11]. However, an anticipated problem with these approaches echoes the original problem of weighting dimensions: regardless of the way these dimensional weights are constructed, the weights are still specific to the ligand that is interacting with the protein. Ideally, we should break out of this feedback loop and weight the feature dimensions of the protein's interface based either on the ligand composition, cellular environment, or other biological hints.

The lessons learned over the last couple of months will guide my research into the summer. From the textual region matcher, we see that the idea of utilizing and aggregating these local shape and chemical descriptors into regions to describe a binding interface is a tractable problem and a feasible way to match functionality. However, it does seem to have its limitations that have been detailed here, and solutions to those problems have been proposed and will be investigated over the coming months. As of now, this tool acts just as a functional surface 'highlighter,' but I hope it will develop into a functional unit and interface region predictor.

5 Acknowledgments

I thank my advisor, Prof. Mike Gleicher for his advice and guidance throughout this project. Many thanks are needed for Greg Cipriano, the student that produced the molecular visualization tool that is still immensely useful for visualization.

This term paper was submitted to Prof. Colin Dewey for Advanced Bioinformatics (CS/BMI 776) at the University of Wisconsin – Madison.

6 Bibliography

References

[1] L. Sael and D. Kihara. Binding ligand prediction for proteins using partial matching of local surface patches. *International Journal of Molecular Sciences*, 11(12):5009–5026, 2010.

[2] R. Gal and D. Cohen-Or. Salient geometric features for partial shape matching and similarity. *ACM Transactions on Graphics (TOG)*, 25(1):130–150, 2006.

[3] R.J. Morris, R.J. Najmanovich, A. Kahraman, and J.M. Thornton. Real spherical harmonic expansion coefficients as 3d shape descriptors for protein binding pocket and ligand comparisons. *Bioinformatics*, 21(10):2347–2355, 2005.

[4] Gregory Cipriano. Molecular surface abstraction, 2010.

[5] T. Tuytelaars and K. Mikolajczyk. Local invariant feature detectors: a survey. *Foundations and Trends® in Computer Graphics and Vision*, 3(3):177–280, 2008.

[6] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[7] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[8] P.F. Felzenszwalb and D.P. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, 2005.

[9] M.P. Do Carmo and M.P. Do Carmo. *Differential geometry of curves and surfaces*, volume 2. Prentice-Hall Englewood Cliffs, NJ, 1976.

[10] R. Schmidt, C. Grimm, and B. Wyvill. Interactive decal compositing with discrete exponential maps. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 605–613. ACM, 2006.

[11] B. Schölkopf, A. Smola, and K.R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.