Ryan Amos, Travis Peters, and Serena Liu
Bailey-Kellogg
CS75/175 Final Project Report
November 25, 2013

Introduction

Protein structure is known to inform protein function; thus, knowledge of a protein's 3D structure helps in predicting its biological function. Protein loops are defined as the amino acid stretches between secondary structure elements (SSEs), such as alpha helices or beta pleated sheets. Loop structures are highly variable and more flexible than SSE structures; some loops do not have a single fixed structure and may undergo conformational changes that impact their protein's overall function.[1] Rhodopsin, a G-protein coupled receptor involved in the light pathway, is one such example: significant changes in rhodopsin loop structures correspond to the protein's conversion between its activated and inactivated states.[2]

Loops play an instrumental role in a number of protein functions, helping to mediate immunoglobin binding of antigens and the interaction between serine proteases and their substrates.[3] Loop structure can also be essential for proper protein regulation, as with the calcineurin protein. The catalytic center of the calcineurin protein is composed of two β-sheets, which are connected by a loop ("loop 7"); mutations in loop 7 result in abnormally high levels of phosphatase activity, and Xiang et al. (2003) determined that loop 7 was instrumental for correct interaction between calcineurin's catalytic and regulatory subunits.[4] More generally, changes in loop structure can severely impact overall protein structure and function. Thus, accurate prediction of protein loop structures will enable better functional characterization of novel protein sequences.

Although loops play an important role in protein function, loop structures remain difficult to predict because of their irregularity and variability. Previous research into loop structure prediction has generally taken one of two approaches. *Ab initio* methods impute loop structure from the primary amino acid sequence by searching a set of sterically possible loop conformations and using a scoring function to identify the optimal structure for the loop sequence in a given environment. *Ab initio* search algorithms include global energy minimization, biased probability Monte Carlo search, and the minimum perturbation random tweak method, among many others.[3]

The second approach, database search, predicts a new loop structure based on similarity to previously known loop structures. The search scans through a database of known protein structures to find structure segments that will fit the SSE "stem regions" flanking the unknown loop structure; these segments are typically ranked by sequence similarity or some geometric criteria, and the best-fit structure is then transferred to the novel loop sequence. The caliber of database search predictions depends heavily on the quality and quantity of the source data, and the effectiveness of such methods was initially constrained by the limited availability of determined protein structures. The library of known protein structures has since expanded significantly, and more recent implementations of database search techniques, like FREAD, have been more successful and even outperform *ab initio* methods for some loop subsets.[5]

Our project aims to implement a database search algorithm to predict loop structure given a protein loop sequence. Most loops contain between 7 and 9 amino acids,[6] so we focus on classifying loops within this length range.

Methods

Overview

Our code and supporting materials are contained in the project LSP (short for "loop structure prediction). We acquired our loop data from Dr. Yoonjoo Choi; the provided dataset included protein sequence and structure information for over 32,000 proteins in PDB file format. For our prediction algorithm, we first extract loop sequence and structure data from the PDB files, along with information about each loop's anchor points and flanking SSEs. We use this loop data to cluster the loops hierarchically into groups of structurally similar loops, and then generate a representative probabilistic sequence for each of these loop families. Once our source loop data has been clustered, we have the infrastructure necessary to classify new loops. Given a new loop sequence, we first match it to the available loop clusters based on sequence similarity to each cluster's probabilistic sequence. To predict the loop's structure, we then transfer the structural characteristics of the most closely matched loop cluster to the new loop.

Data pre-processing

Before any of our proposed methodology could be carried out we first had to address the matter of extracting relevant sequence/structure data from the PDB files mentioned above.

Unfortunately, the PDB files that we had access to for this project were largely unannotated. To address this issue we use a combination of pre-existing code from *Computer Science 75/175* homework assignments (i.e. classes: Atom, SSE, etc.; functions: *read_pdb*, *get_ca_chain*, etc.), *DSSP*,[7] which is a program that calculates DSSP entries from PDB entries, and the *DSSP* module from *BioPython*,[8] which leverages the *DSSP* program to identify secondary structures within a protein sequence.

Together, these tools enable us to successfully locate the SSEs in a given PDB file; we then use the inverse space of the identified SSEs to define the loop structure elements (since we consider loop elements to be the protein structures that lie between a protein's SSEs). It should be noted that the DSSP program/BioPython returns an annotated representation of each residue in the PDB file, identifying the type of SSE that a particular residue belongs to – we then evaluate residue label patterns to determine where the SSEs are located. Upon recommendation from Dr. Choi,[6] we require that a minimum of three (3) sequential residues must conform to the same secondary structure label in order for us to consider the SSE to be valid. For simplicity, we treat varying types of helices or sheets as a single SSE class; that is, all helix variants are treated as alpha helices, while all sheet variants are labeled as beta sheets. We experimentally validated this approach by comparing these

results to annotated PDB files that we have previously used in course assignments; the resulting annotations were very similar, differing only by a residue or two at the beginning or end of some SSEs.

As we extract loop structure data from the PDB files we create representations of these loops which contain information about each loop's starting/ending residue position in a protein, the underlying amino acid sequence, the actual underlying atomic structure, and the 'types' of the flanking SSEs (i.e. alpha helices or beta sheets). How this data is used in our method for loop structure prediction is outlined in following sections.

Loop structure representation

Models

In LSP, we represent loop data using the Model class. We have designed the Model class to accommodate a variable number of loops, so that a model may represent either a single loop or group of loops. We do not set an upper limit on the number of loops a model may represent. A model has 5 primary components:

1. Sequence: a probabilistic sequence representing the amalgamation of the sequences of the model's component loops
2. θ: the angle between first (i.e. left) flanking SSE and the loop structure
3. ϕ: the angle between the second (i.e. right) flanking SSE and the loop structure
4. Anchor distance: the distance between the anchor (terminal) atom s of the loop structure
5. Backbone structure: the backbone C_α atom positions in 3D space, relative to the first loop anchor (adjacent to the first flanking SSE)

In order to obtain backbone atom positions relative to the first anchor position, we transform the atom positions from global space to a frame within global space that is oriented with respect to the protein loop. In order to establish a reference frame for the loop in 3D space, we use 2 reference vectors: the vector pointing along the body of the first flanking SSE and the vector pointing between the loop's two terminal anchor points.[9] We normalize the vector pointing from the first loop anchor point to the second loop anchor, then define our *x*-axis to point along this vector. We cross the *x*-axis vector with the vector along the first SSE and normalize that to obtain our *z*-axis unit vector. Finally, because our *x*-axis may not be orthogonal to the vector pointing along the first SSE, we cross the *x*-axis unit vector and the *z*-axis unit vector to determine the *y*-axis for the loop's reference frame. The origin for the frame is the loop's first anchor point (the loop residue adjacent to the first flanking SSE). We can then use this information to determine the relative position for each of the loop's backbone atoms. For an atom a and a reference frame f,

$$a' = f_{x-axis} \cdot (a - f_{origin}) + f_{y-axis} \cdot (a - f_{origin}) + f_{z-axis} \cdot (a - f_{origin}).$$[10]

Model comparison

We primarily measure structural similarity between two models based on the similarity of their loop backbone structures and the similarity of their loop anchor characteristics. We only

compare models representing loops of equal length, so backbone similarity is scored using the sum of the RMSD values for each corresponding atom pair. The three anchor metrics we consider are θ, ϕ, and anchor distance, as described in the preceding section. For every model, we normalize each of these values based on the theoretical range for the given metric; this range is $[0, 2\pi]$ for θ and ϕ, and $[0, l_{max}]$ for anchor distance, where l_{max} is the maximum anchor distance for a loop of given length. l_{max} is computed as follows:

$$l_{max}(n) = \begin{cases} \gamma * \left(\frac{n}{2} - 1\right) + \delta & if\ n\ is\ even \\ \gamma * \frac{n-1}{2} & if\ n\ is\ odd \end{cases}$$

where $\gamma = 6.046$ Å and $\delta = 3.46$ Å.[11] Individual values are normalized to $\frac{value - min_{theoretical}}{max_{theoretical} - min_{theoretical}}$. For example, a given θ value would be normalized to $\frac{\theta - 0}{2\pi - 0} = \frac{\theta}{2\pi}$. When two models are compared structurally, we consider the difference between their normalized scores for each of these anchor components, as well as the model pair's overall backbone RMSD value. The overall similarity score for a pair of models is the sum of each of the component anchor scores plus the backbone RMSD normalized to the standard RMSD similarity cutoff of 2Å.[6] We apply this comparison metric in both the clustering and classification steps of the project.

Model merging

To merge two models, we consider each of their five key components separately. For the anchor structure metrics (θ, ϕ, and anchor distance), the merged value is simply the weighted average the two component models' corresponding values. We weight based on the number of loops represented by each model. So, for example, if we merge models A and B to form model C, where model A contains 10 loops and model B contains 5 loops, model C's ϕ value would be

$$C_\phi = A_\phi * \frac{10}{10+5} + B_\phi * \frac{5}{10+5}.$$

To determine the backbone structure of the merged model, we also take a weighted average of the two component models: the (*x*,*y*,*z*) coordinates for each merged backbone atom position is given by the weighted average of each component model's atom coordinates for that backbone position. We only ever merge two loops if they are of the same length (see "Source loop clustering" below), so each atom in one component model will always have a corresponding atom in the other component model. Similarly, to determine the probabilistic sequence for the merged model, we simply calculate new amino acid frequencies for each loop position by taking a weighted average of the amino acid frequencies of each of the component models' sequences at that position.

Source loop clustering

Loop sequences were pre-binned based on length and flanking SSE type, and singleton models were constructed for each loop; loop models were then clustered hierarchically based on structural similarity. We measured structural similarity using the comparison metric described above (see "Model comparison"), and in each round of clustering we evaluated the loop pair with the lowest comparison score (indicating the closest structure similarity). If this comparison score fell beneath

our accepted threshold, we proceeded to merge the two models; if the loops represented by the two models were too different, we kept the groups separate.

In order to be considered "mergeable", two models were required to satisfy similarity thresholds for both backbone structure and the loop anchor metrics. For backbone structure, we applied the standard RMSD similarity threshold of 2Å.[6] For the anchor metrics of θ, ϕ, and anchor distance, we used a 0.02 similarity threshold, which was experimentally determined (see Results, "Preliminary 'model validation' testing"). We applied this 0.02 threshold to each anchor metric – that is, each of the component anchor scores for the model pair (i.e. the difference between the normalized anchor values for the two models) must differ by less than 0.02 in order for a model pair to be considered mergeable.

Model pairs that (1) were the most structurally similar for a given clustering round and (2) satisfied the similarity thresholds of 2Å for backbone RMSD and 0.02 for θ, ϕ, and anchor distance were subsequently merged. If a given model pair did not satisfy these thresholds, we assigned the models a similarity score of ∞ and kept the two models separate instead of merging them. We finish clustering when none of the remaining model pairs are similar enough to merge. Our final list of unmerged models comprises the list of possible structures that may be assigned to a novel loop sequence.

Loop classification

In the classification step we aim to leverage our previously gathered knowledge in order to take a given loop sequence of unknown structure and return information about its predicted structure.

As an initial step, we apply various filters to the space of all available loops, as previously discussed. From past conversations with Dr. Choi, we learned that loop length is a highly critical characteristic and can greatly impact the overall structure of a loop – thus we filter out any loops that do not have the same amino acid sequence length as the input sequence that we are trying to classify. Anchor structure – the type of SSE on either side of a loop structure – was yet another critical factor for determining loop structure, according to Dr. Choi. Thus, we apply yet another filter on the set of candidate loops whereby only loops with the same anchor structures as the input loop could possibly constitute an accurate representation of that loop. These filtering steps separate the initial loop dataset into their length-SSE type bins.

After completing this initial filtering, we cluster the candidate loops hierarchically (as described above) into loop clusters containing loops of similar structure. The representative models for these loop clusters constitute the set of possible loop representations. Only after the clustering is completed can we proceed with the actual classification.

For each of the representative models formed from hierarchical clustering, we proceed as follows:

1. Compare amino acid at index *i* in the input sequence with *each* observed amino acid at index *i* of the model's probabilistic sequence.
    a. Observe the frequency in which amino acid *k* appears in position *i* of the probabilistic sequence.

b. Determine the substitution score between the amino acid at index *i* of the input sequence and the amino acid *k* from part 1(a).
c. Compute the product of the results from 1(a) and 1(b).
d. Repeat 1(a)-(c) and sum the results.

2. Repeat step 1(a)-(d) for each index *i* of the input sequence, and sum the results. This number comprises the *sequence similarity score* for how well a given model matches the input sequence.

After computing the sequence similarity score for each model, we then sort the results and declare the maximum-scoring model to be most representative of the input sequence. Thus, we declare that the sequence's structure is well-represented by the representative structure of the model.

Cross-validation analysis

In order to rigorously test and assess the performance of our predictive models we performed *K*-fold cross-validation.

We performed cross-validation testing on loop datasets of varying sizes ranging from 200 to 4437. For each dataset, we performed 5-fold cross-validation with 10 repetitions. In each repetition, we first partitioned the loop set into 5 random partitions. For a given fold, we then trained our predictive model on the remaining 4 folds, clustering the loops and generating representative cluster models. Then, for each loop in the validation (test) set we (1) determined the best representative model by via sequence matching, and (2) computed a structure similarity score between the test instance and its best representative model. It should also be noted that for each fold we record both the median and the mean score of all of the scores. Doing this for each fold yields 5 medians/means; these values are then averaged for a given repetition and this data is recorded and averaged over the 10 repetitions that we perform on each dataset.

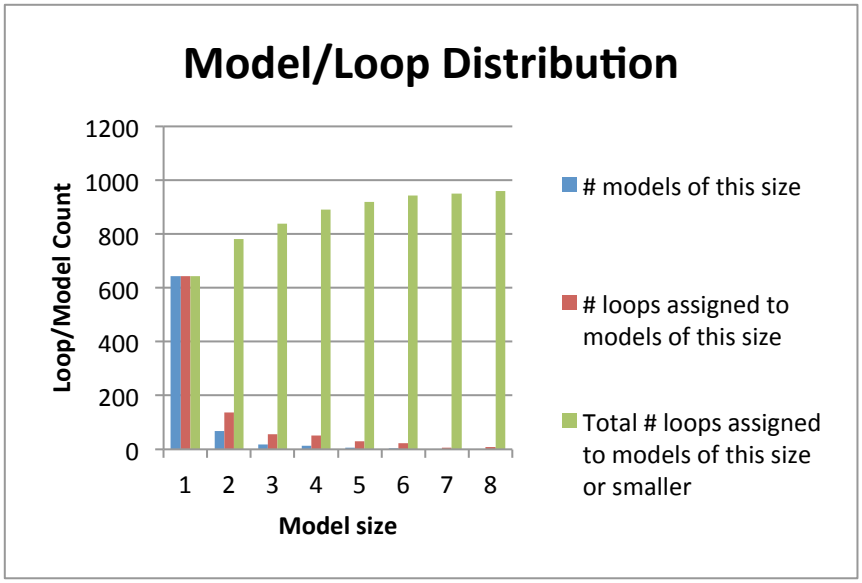
To examine the relative performance of our predictive model we use *random prediction* as our baseline (i.e. for each test instance, find all of the acceptable clusters/models that it *could* be represented by, and pick one at random as the representative cluster/model), and we record the same statistics for our random prediction model.

Results

Preliminary "model validation" testing

Before running our cross-validation tests, we first wanted to verify that the composite loop structure encoded in our models represented the structures of the models' component loops reasonably well. To accomplish this, we assembled datasets of soluble loops and applied our clustering algorithm to the entire dataset. We then classified each of the loops against the relevant model list and observed how often a loop was correctly classified into its original cluster. We found

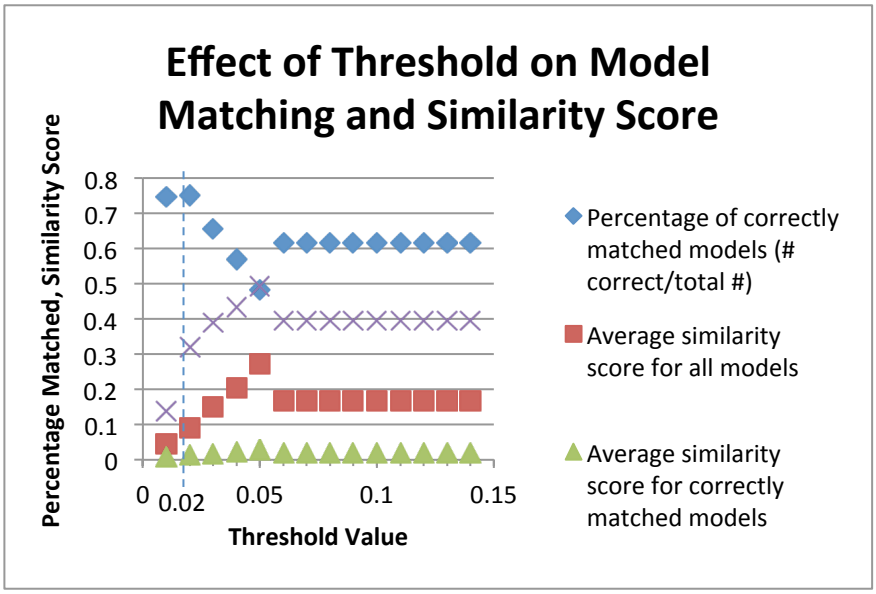
that for 78% of the loops, the top predicted model corresponded to the loops' original cluster.



Histograms charting model size (i.e. the number of loops represented by a single model) against the numbers of models and loops associated with models of that size for a dataset of 1,000 loops are shown in the figure on the left.

To get a sense of how closely the composite model loop structures matched the structures of the individual loops being classified, we also computed similarity scores between individual loop structures and the composite structure of the loop's top match model, for loops that were assigned to models of size 3 or larger. We did not compute similarity scores for loops assigned to models of size 1 or 2 in order to avoid over-fitting; if a model is representing only one loop, and that loop gets matched back to the model, then the similarity score between the two will indicate a perfect match, and these outlying scores would throw off our overall averages. Similarity scores (calculated using the metric described in "Model comparison") for models that were successfully matched back to their original cluster averaged at 0.0118, while the average score for unmatched models was 0.2386; the overall average similarity score for all models was 0.0779.

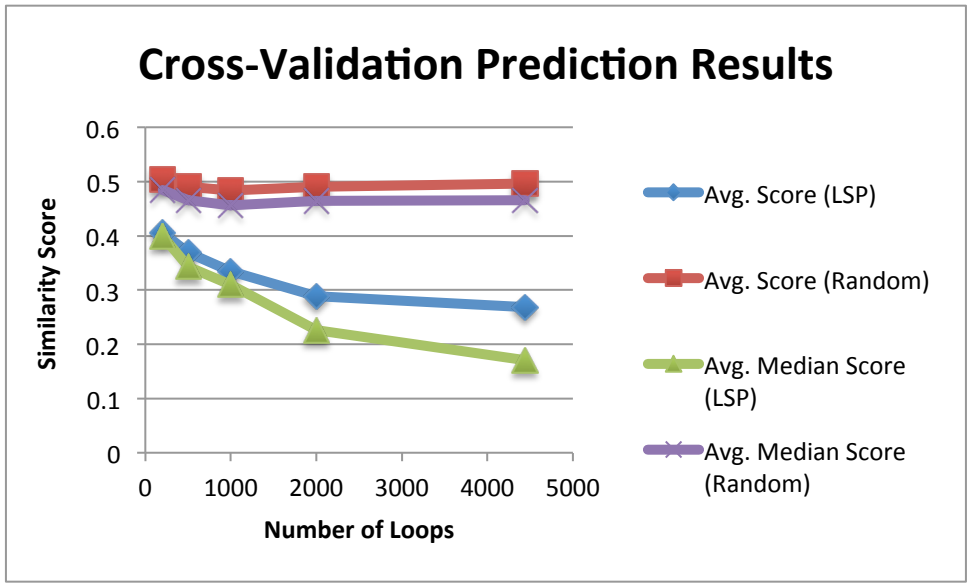
We also used this preliminary model-matching test to determine our similarity threshold for the loop anchor metrics (θ, ϕ, and anchor distance). We repeated the same test (train on all data, then classify each loop and determine which percentage of loops are classified correctly with the top predicted model) using anchor similarity threshold values ranging from 0.01 to 0.20, incrementing the threshold by 0.01 with each run. Plots of model-matching percentages and average similarity scores are shown in the figure to the right. Changing the similarity threshold alters which models are deemed "mergeable" and thus affects the final model lists, which are used in the classification step. The similarity threshold that yielded the highest model-matching percentage was 0.02; we thus use 0.02 as our similarity threshold



for loop anchor metrics in the cross-validation analysis. The average similarity scores between individual loop structures and the representative structure of the top match model are quite low for successfully matched models, indicating a high level of structural similarity. Interestingly, for thresholds of 0.06 and above, the model-matching percentages and similarity scores remain constant, suggesting that increasing the similarity threshold above 0.06 no longer affects the loop distributions.

Cross-validation analysis

We performed 5-fold, 10-repetition cross-validation analysis on loop datasets of varying size; the smallest loop set contained 200 loops, while the largest set contained 4437 loops. We performed the cross-validation analysis both using our sequence-matching algorithm and simply selecting a random model from the appropriate model bin (i.e. same loop length and flanking SSE types). Our cross-validation results are shown in the figure below; the precise numerical data are included in the Supplementary Materials.



For datasets of all sizes, our prediction algorithm (LSP) outperformed the random-chance baseline. As dataset size increased, the performance of the random-chance metric remained relatively constant, while the performance of our LSP metric improved (achieving a lower average similarity score, which indicates a greater degree of structural similarity between query loops and their predicted model matches).

Discussion

Preliminary testing

We conducted our preliminary "model-matching" tests in order to assess how well our method of representing composite loop structures in models captured the pertinent structural qualities of the models' constituent loops. In general, we observed that about 78% of the individual loops were correctly matched to their original clusters with their top predicted model. This finding suggests that loop sequence is reasonably predictive of loop structure; since we classified loops to models based on

sequence similarity but clustered the loops initially based on structure, we would not be able to consistently classify a loop back to its initial cluster if loop sequence were not predictive of loop structure.

In the preliminary testing, we also computed similarity scores for each loop-model match. On average, the similarity scores for successfully matched loops (average score 0.0118) were substantially lower than the scores for unsuccessfully matched models (average score 0.2386), indicating that successfully matched loops are structurally more similar to their predicted model structures as compared to unsuccessfully matched loops. This outcome is expected, since the initial loop clusters were formed on the basis of structural similarity, and so a loop that was successfully assigned back to its original cluster should be structurally more similar to that model's composite structure than to any other model's composite structure. However, this result does support the validity of our composite structure representation: presumably, our composite model structures must be capturing distinctive aspects of loop structure, or else the composite structure of a loop's original cluster would not match the loop's individual structure significantly better than the composite structures of other loop clusters. Thus, if our composite model structures did not represent the component loop structures well, we would expect the similarity scores for successfully matched loops and unsuccessfully matched loops to be much closer to one another.

Cross-validation analysis

We performed the cross-validation analysis both with our LSP prediction algorithm and with a random-chance algorithm, where the predicted model is selected randomly out of the appropriate model bin. Our prediction algorithm outperformed the random-chance algorithm at all the dataset sizes we tested, suggesting that our method does add value to the structure prediction process. Furthermore, while the prediction performance of random-chance remains fairly constant as dataset size increases, the performance of our prediction algorithm improves with larger datasets. This result is expected, since we implemented a database search approach: as the size of the database increases, there are more template structures available, and a novel sequence is more likely to locate a good match from within the available template options.

We also found that the difference between the median and mean score metrics were relatively small, on average. Since the mean score is affected by outlying scores more than the median score, this finding suggests that we did not encounter many extreme outliers over the course of our cross-validation analysis.

Future directions

Several components of our project could be improved upon. Firstly, our model comparison metric is central to both our loop clustering and our loop classification, and yet it still relies on hard thresholds for backbone RMSD and anchor metric values. With further experimentation, we may be able to discover more appropriate or more flexible thresholds for these values. Additionally, we currently assign each of these components equal weight in the overall similarity score, while in

actuality, these factors may not contribute equally to loop structure. Thus, future work could also investigate different weighting schemes for the individual comparison score components.

Another area for improvement is our method of merging models. Currently, our merged values are simply weighted averages of the component models' corresponding values. However, simple averages are a very naïve way to represent a combined metric, and further exploration may likely yield a better merging scheme.

We may also wish to investigate other means of forming the initial loop clusters. Hierarchical clustering is designed to locate small groups of similar items, and then join these groups with other similar groups. Thus, this approach may not be optimal for identifying groups whose members are all roughly related to one another. For our purposes, a network graph approach may have been a better solution: we could form a network graph by comparing each loop to every other loop in the dataset, establishing an edge between the two loops if they pass our similarity thresholds. After establishing the graph, we could locate our loop clusters by identifying the busiest regions in the graph.

Our project focused on classifying loops with lengths ranging from 7 to 9 amino acids, since the majority of protein loops fall into this length range. Future work could aim to expand our prediction algorithm to accommodate shorter and longer loops as well. Longer loop structures in particular are difficult to predict, since they are generally more flexible and may potentially adopt a much wider range of loop conformations. Another major question to investigate is the relationship between protein sequence and structure: there exist proteins with very similar sequences, and yet distinct physical structures. There are also proteins with divergent sequences and yet highly similar structures; convergent evolution is one example of how such proteins may arise. To tackle this question, we might first investigate sequence dependencies and interactions between amino acids, rather than assuming each sequence position to be independent. Discovering which elements of sequence inform structure could lead to improvements in protein structure prediction even for cases where proteins share both sequence and structure similarities.

References

1. Soto, C.S. et al. (2008) "Loop modeling: Sampling, filtering, and scoring." *Proteins*, *70(3)*: 834-43.

2. Nikiforovich, G.V. and G.R. Marshall (2005). "Modeling flexible loops in the dark-adapted and activated states of rhodopsin, a prototypical G-protein-coupled receptor." *Biophysical Journal*, *89(6)*: 3780-89.

3. Fiser, A. et al. (2008) "Modeling of loops in protein structures." *Protein Science*, *9(9)*.

4. Xiang, B.Q. et al. (2003) "The role of loop 7 in mediating calcineurin regulation." *Protein Engineering Design & Selection*, *16(11)*.

5. Choi, Y. and C.M. Deane. (2009) "FREAD revisited: Accurate loop structure prediction using a database search algorithm." *Proteins: Structure, Function, and Bioinformatics***,** *78 (6)*.

6. Choi, Yoonjoo. Private conversation

7. Wolfgang Kabsch and Chris Sander's DSSP implementation (http://swift.cmbi.ru.nl/gv/dssp/)

8. BioPython/DSSP module (http://biopython.org/DIST/docs/api/Bio.PDB.DSSP%27-module.html)

9. Denning, Jonathan. *Geometric Transformations*. Dartmouth College, Apr. 2013. Web. 24 Nov 2013.<http://www.cs.dartmouth.edu/~cs77/slides/04_transforms.pdf>.

10. Denning, Jonoathan. *Modeling*. Dartmouth College, Apr. 2013. Web. 24 Nov 2013. <http://www.cs.dartmouth.edu/~cs77/assignments/assignment02.zip>. Equation adapted from code in /src/vmath/transform.h.

11. Choi Y., S. Agarwal, and C.M. Deane. (2013) "How long is a piece of loop?" *PeerJ* 1:e1 http://dx.doi.org/10.7717/peerj.1

Supplementary Materials

Table 1. Cross-validation results

# Loops	Avg. Score (LSP)	Avg. Score (Random)	Avg. Median Score (LSP)	Avg. Median Score (Random)
200	0.40561	0.50502	0.39899	0.48393
501	0.36839	0.49003	0.343	0.46607
1000	0.33512	0.48347	0.31003	0.45585
2000	0.28874	0.49028	0.22614	0.4642
4437	0.26817	0.49559	0.17077	0.46568