RIGIDITY ANALYSIS FOR MODELING PROTEIN MOTION

A Dissertation

by

SHAWNA LYNN THOMAS

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

May 2010

Major Subject: Computer Science

RIGIDITY ANALYSIS FOR MODELING PROTEIN MOTION

A Dissertation

by

SHAWNA LYNN THOMAS

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

May 2010

Major Subject: Computer Science

ABSTRACT

Rigidity Analysis for Modeling Protein Motion. (May 2010)

Shawna Lynn Thomas, B.S., Texas A&M University

Chair of Advisory Committee: Dr. Nancy M. Amato

Protein structure and motion plays an essential role in nearly all forms of life. Understanding both protein folding and protein conformational change can bring deeper insight to many biochemical processes and even into some devastating diseases thought to be the result of protein misfolding. Experimental methods are currently unable to capture detailed, large-scale motions. Traditional computational approaches (e.g., molecular dynamics and Monte Carlo simulations) are too expensive to simulate time periods long enough for anything but small peptide fragments.

This research aims to model such molecular movement using a motion framework originally developed for robotic applications called the Probabilistic Roadmap Method. The Probabilistic Roadmap Method builds a graph, or roadmap, to model the connectivity of the movable object's valid motion space. We previously applied this methodology to study protein folding and obtained promising results for several small proteins.

Here, we extend our existing protein folding framework to handle larger proteins and to study a broader range of motion problems. We present a methodology for incrementally constructing roadmaps until they satisfy a set of evaluation criteria. We show the generality of this scheme by providing evaluation criteria for two types of motion problems: protein folding and protein transitions. Incremental Map Generation eliminates the burden of selecting a sampling density which in practice is highly sensitive to the protein under study and difficult to select. We also generalize the roadmap construction process to be biased towards multiple conformations of interest thereby allowing it to model transitions, i.e., motions between multiple known conformations, instead of just folding to a single known conformation. We provide evidence that this generalized motion framework models large-scale conformational change more realistically than competing methods.

We use rigidity theory to increase the efficiency of roadmap construction by introducing a new sampling scheme and new distance metrics. It is only with these rigidity-based techniques that we were able to detect subtle folding differences between a set of structurally similar proteins. We also use it to study several problems related to protein motion including distinguishing secondary structure formation order, modeling hydrogen exchange, and folding core identification. We compare our results to both experimental data and other computational methods.

To my husband, Nathan: you are my constant support

To my parents, Brig and Phyllis: you are my strong foundation

ACKNOWLEDGMENTS

I would first like to thank my advisor, Dr. Nancy Amato, for her unwavering support and encouragement. Even from the very beginning when I first met her, she saw in me what I did not. She managed to convince me as an undergraduate to explore research and my educational career, which has not been the same since. She continuously and tirelessly guided me through the years and I deeply appreciate all her guidance, care, and sacrifice. She has taught me not only the thrill of pursuing the unknown, but also the joy of mentoring others to do the same.

I would also like to thank my committee members, Dr. J. Martin Scholtz, Dr. Sing-Hoi Sze, and Dr. Jennifer Welch, for their support and feedback. I am grateful for the time they committed to helping me grow as a researcher.

I also thank the many collaborators I've enjoyed the opportunity to work with over the years, both on this work and on other research projects: Dr. Ken Dill, Dr. J. Martin Scholtz, Dr. Burchan Bayazit, Bryan Boyd, Dr. Lucia Dale, Dr. Jyh Ming Lien, Dr. Marco Morales, Roger Pearce, Sam Rodriguez, Dr. Guang Song, Dr. Lydia Tapia, Gabriel Tanase, Dr. Xinyu Tang, and Dawen Xie. I am especially thankful to Guang for his development of the protein folding framework, which is the foundation of this work and for his guidance during my early discoveries in computational biology; to Burchan for spending many hours helping me through my first project with the research group; and to Lucia for mentoring me in the very beginning as a young undergraduate with many questions. Thank you for your extreme patience as I learned the ropes. I am also grateful for working with Lydia through most of my graduate career. As we've shared an office together and collaborated on various research endeavors, we've also celebrated in each other's victories and helped each other through difficult challenges. We have accomplished much working together.

I would also like to thank several people for their assistance with the work presented in this dissertation. In particular, I would like to thank Dr. Lydia Tapia for her collaboration on distance metric alternatives based on rigidity theory and Cindy Yeh for help in running some of the experiments. Thank you to Kasia Leyk for assistance in running some of the protein transition experiments on the Morph Server. I would also like to thank Dr. Jyh Ming Lien, Dr. Marco Morales, Roger Pearce, and Dawen Xie for their work on Incremental Map Generation. Many thanks to Dr. Illeana Streinu for useful discussions on rigidity analysis, to Dr. A. J. Rader for advice on our folding core identification experiments, and to Dr. J. Martin Scholtz for his many conversations on hydrogen exchange and folding core identification. Finally, I would like to thank Dr. David Baker's group for their generous provision of the unpublished protein L mutant structures to work with.

Many others helped perhaps less directly, but no less significantly. Thank you to all the Parasol support members, both former and current, who worked around the clock to keep the lab up and running. All your work behind the scenes has not gone unnoticed — I am indebted to all of you. I would also like to thank Kay Jones for all her help over the years.

Thanks to the many funding agencies who have supported my graduate career and made all of this possible, including the National Science Foundation Graduate Fellowship Program, the Philanthropic Educational Organization (P.E.O.) Scholar Awards Program, the Department of Education Graduate Assistance in Areas of National Need Program, and the IBM Ph.D. Fellowship Program. I would also like to thank all the conferences and workshops that provided travel grants over the years. These include funding to attend the Grace Hopper Celebration of Women in Computing Conference, the IEEE International Conference on Robotics in Automation, the International Conference on Research in Computational Molecular Biology, the

Workshop on Flexibility in Biomolecules, and the Workshop on Modeling Protein Flexibility and Motions. My graduate career has been enriched by all of them.

Finally, I would like to give a very special thanks to my family who all saw me through this work. My husband, Nathan, has been my constant support in both the good times and the tough times. Your support means more to me than I could ever express, and I could not have done this without you. We have weathered this together, and I would not want to have done it any other way. I am deeply honored. I would also like to thank my parents, Brig and Phyllis, who taught me the importance of education and the value of hard work. They have been there every step of the way, and their love and encouragement has helped carry me through this long journey. Finally, I would like to thank my mother, Phyllis, and my mother-in-law, Belinda, for their sacrifice of time during the last several months after the birth of our first son, Daniel. Thank you for comforting him while I worked away from home; this would not have been possible without you.

TABLE OF CONTENTS

CHAPTER Page

I INTRODUCTION 1

A. Research Objective and Contributions 3

B. Outline 5

II PRELIMINARIES AND RELATED WORK 6

A. Proteins 6

1. Protein Structure 6
2. Degrees of Freedom 10
3. Potential Functions 11
4. Energy Landscapes 13

B. Protein Motion 14

1. Experimental Methods 14
2. Computational Methods 15

C. Probabilistic Roadmap Method Preliminaries 20

1. Motion Planning and C-space 20
2. The PRM Algorithm 21
3. Modifications to PRMs for Proteins 24

D. Rigidity Analysis 25

1. The Pebble Game 26
2. Rigid Cluster Decomposition 34
3. Dependent Hinge Sets Identification 39

III EXTENSIONS TO THE PRM FRAMEWORK FOR MOLECULAR MOTION 45

A. Detecting Model Stabilization when Building Roadmaps Incrementally 46

1. Incremental Roadmap Construction Details 47
2. Roadmap Evaluation 48
3. Results 50

B. Building Landscape Models for other Motions 52

1. Roadmap Construction with Multiple Conformation Biases 54
2. Results 56

CHAPTER Page

IV RIGIDITY FOR LANDSCAPE MODELING 63

A. Protein Rigidity Model 64

1. Single-body, Cα Model 65
2. Two-body, ϕ-ψ Model 69
3. Optimizing the Rigidity Analysis 73
4. Rigidity Model Comparison 75

B. Rigidity-Biased Sampling 82

1. Sampling Study 86

C. Rigidity-Based Distance Metric 96

1. Rigidity Maps of Proteins G, L, and their Mutants . . 101

V RIGIDITY FOR LANDSCAPE MODEL ANALYSIS 105

A. Identifying Structure Formation Order 105

1. Results 107
2. Case Study of Proteins G, L, and their Mutants . . . 117

B. Relative Hydrogen Exchange 120

1. From Residue Flexibility 121
2. From Rigid Cluster Decomposition 122
3. Related Computational Methods 127
4. Results 127

C. Folding Core Identification 159

1. Related Computational Methods 160
2. From Relative Hydrogen Exchange Rates 162
3. Results 165

VI CONCLUSION AND FUTURE WORK 172

REFERENCES 176

VITA 200

LIST OF TABLES

TABLE Page

I A comparison of protein motion models. 17

II Sizes of roadmaps for Protein G when built incrementally with varying increment size and stability thresholds, t. 51

III Pathway results for transitions studied. 62

IV Proteins studied in the rigidity model performance comparison. . . . 76

V Proteins used in rigidity-based sampling study. 87

VI Sequences of protein G mutants [114]. 102

VII Sequences of protein L mutants [113]. 103

VIII Proteins studied for secondary structure formation order based on rigidity analysis. 109

IX Rigidity-based sampling and rigidity-based secondary structure formation order results for the proteins in Table VIII. 111

X Comparison of secondary structure formation orders for proteins in Table VIII using rigidity-based sampling and the rigidity-based secondary structure formation definition with known experimental results. 116

XI Comparison of secondary structure formation orders for proteins G, L, and their mutants with known experimental results. 118

XII Proteins studied for relative hydrogen exchange and folding core identification. 128

XIII Comparison of average relative exchange rate for both EX_{RS} and EX_{CS} between the entire protein and the experimentally defined folding core. 154

TABLE Page

XIV Comparison between rigidity scores, cluster scores, and COREX/BEST of their ability to predict the most stable residues. 158

XV Summary of folding core identification performance. 169

LIST OF FIGURES

FIGURE Page

1 Amino acid structure. 7

2 Linking of amino acids to form a protein. 8

3 Examples of secondary structure: (a) an α helix and (b) a two-stranded, antiparallel β sheet. 9

4 Example of tertiary structure of protein G: (a) ball-stick view and (b) ribbons view. 10

5 Visualization of the protein's energy landscape [38]. 13

6 2D pebble game example. 27

7 Pebble rearrangement example. 28

8 Rigidity models for a sample molecule: (a) bar-joint model and (b) body-bar model. 33

9 The final pebble game graph and rigid cluster decomposition after all constraints are considered for the example in Fig. 6(a). 38

10 Example of how rigid clusters are merged in Alg. D.6. 40

11 The reachability of a vertex is the set of vertices that are reachable by directed paths from that vertex in the graph. 41

12 Example of how dependent hinge sets are identified in Alg. D.8. . . . 43

13 Flow diagram for Incremental Map Generation (IMG). 47

14 Pathway distribution profile of protein G as a function of roadmap size. Stopping points for incremental map generation are shown for various increment sizes and stability thresholds, t. 53

FIGURE Page

15 Conformational changes of calmodulin: (a) calcium-free state (1CFD) to (b) bound state (1CLL) and of the N-terminal domain: (c) calcium-free to (d) bound. 56

16 The potential vs. RMSD distribution of samples for calmodulin, N-terminal domain. 57

17 Pathway profiles for the calmodulin N-terminal domain comparing our method to ‘morphs’ of various resolution: (a) contacts present, (b) coarse potential energy, (c–d) all-atoms potential energy. 59

18 Pathway profiles for the calmodulin N-terminal domain comparing our method to ‘morphs’ of various resolution, continued: (e) degrees of freedom computed by rigidity analysis, and (f) RMSD to both target states. 60

19 Example turn protein fragment (FANGST) where hydrogen bonds are indicated with dashed lines and hydrophobic interactions with dotted lines. 66

20 Rigidity model examples of varying granularity for the protein fragment in Fig 19. 67

21 Single-body, Cα model example of the turn segment F-A-N-G-S-T in Fig. 19. 68

22 Example helical protein fragement (E-V-L-A-Y-A-D-T) where hydrogen bonds are indicated with dashed lines and hydrophobic interactions with dotted lines. 70

23 Single-body, Cα model example of the helical segment in Fig. 22. . . 71

24 Two-body, ϕ-ψ model example of the turn fragment (FANGST) in Fig. 19. 72

25 Two-body, ϕ-ψ model example of the helical fragment (EVLAYADT) in Fig. 22. 74

26 Correlation between protein length and average contacts present and average rigidity model constraints present in the sample set. . . . 78

FIGURE Page

27 Running time to perform rigidity analysis on 1000 proteins ranging in length from 56 to 372 residues. 78

28 Running time to perform rigidity analysis on 1000 proteins ranging in length from 56 to 372 residues for each model individually. . . 79

29 Running time breakdown of the 1000 rigidity analysis computations on proteins ranging in length from 56 to 372 residues for the (a) single-body, Cα rigidity model and the (b) two-body, $\phi - \psi$ rigidity model. 80

30 Speedup gained from the backbone optimization. 81

31 Correlation of the average number of attempts, A_{avg}, and the average time required to generate a valid sample for rigidity-based sampling. 88

32 Comparison of iterative Gaussian sampling and rigidity-based sampling statistics for protein A. 90

33 Comparison of iterative Gaussian sampling and rigidity-based sampling statistics for CTXIII. 91

34 Comparison of iterative Gaussian sampling and rigidity-based sampling statistics for protein G. 92

35 Comparison of iterative Gaussian sampling and rigidity-based sampling statistics for ApoMb. 93

36 Comparison of iterative Gaussian sampling and rigidity-based sampling statistics for Fibronectin. 94

37 Comparison of iterative Gaussian sampling and rigidity-based sampling statistics for RNase H. 95

38 Comparison of running time and the number of nodes "missing" from the targeted distribution for iterative Gaussian sampling and rigidity-based sampling on protein A, CTXIII, and protein G. 97

FIGURE Page

39 Comparison of running time and the number of nodes “missing” from the targeted distribution for iterative Gaussian sampling and rigidity-based sampling on ApoMb, Fibronectin, and RNase H. . . . 98

40 Rigidity maps for the native state of protein G (PDB: 1PGA). 100

41 Native state ribbons diagrams of proteins G, L, and their mutants NuG1, NuG2, L1 – L4, and S1 – S2. 102

42 Rigidity maps of proteins G, L, and their mutants NuG1, NuG2, L1 – L4, and S1 – S2 using the $\phi - \psi$ model. 104

43 Example of the rigidity-based secondary structure formation definition for the first β-strand (shown in red) in protein G. 108

44 Comparison of rigidity-based sampling to previous work for several proteins. 114

45 Comparison of sampling methods as a function of protein length. . . 115

46 Example unfolding pathway for protein G. 123

47 Corresponding relative exchange rates from the rigidity scores (red) and the cluster scores (green) for the example unfolding pathway for protein G. 124

48 Protein coloring based on relative exchange rates for the example unfolding pathway for protein G. 125

49 Comparison of simulated exchange rates from rigidity scores and cluster scores to experimental data. 131

50 Visual comparison of simulated exchange rates to experimental data. 149

51 Minimal variance thresholds (dashed lines) to cluster EX_{RS} for Tendamistat using (a) $k = 2$, (b) 3, and (c) 4 clusters. 163

52 Example of the elbow criterion for selecting an appropriate number of clusters k. 164

53 Folding core identification sensitivity and specificity by EX_{RS} and EX_{CS} compared to GNM-G slow modes and GNM-H fast modes. . . 167

FIGURE Page

54 Folding core identification sensitivity and specificity by EX_{RS} and EX_{CS} compared to FIRST-H3 and FIRST-H1. 168

CHAPTER I

INTRODUCTION

Proteins are essential to nearly all forms of life. Every protein is composed of a string of amino acids. This amino acid sequence, unique to the protein, determines the protein's final, stable three dimensional structure. This final structure is a large determinant of the protein's function. The protein's final structure is so crucial that when a protein misfolds into a different structure, it can ultimately result in a devastating disease. Misfolded proteins have been implicated in a number of such diseases including Alzheimer's, Mad Cow, and Creutzfeldt-Jakob [1]. Surprisingly little is known about how and why a protein misfolds in this way. Thus, understanding how a protein folds from an unstructured string of amino acids into its final, stable, three dimensional structure could highlight causes (and potential treatments) of misfolded proteins.

In addition to folding, other protein motions, ranging from molecular flexibility to large-scale conformational change, play a critical role in many biochemical processes. For example, local conformational change often occurs in binding interactions between proteins and between proteins and ligands, sugars, or other small molecules. While no consensus has been reached regarding models for protein binding, the importance of protein flexibility in the process is well established by the ample evidence that the same protein can exist in multiple conformational states and can bind to structurally different molecules [2].

Our understanding of molecular movement is still very limited and has not kept pace with the explosion of knowledge regarding protein structure and function. There

The journal model is *IEEE Transactions on Automatic Control.*

are several reasons for this. First, the structural data in repositories like the Protein Data Bank (PDB) [3] consists of the spatial coordinates of each atom in the protein. Generally, the experimental methods currently used to collect this data, e.g., X-ray crystallography [4] and nuclear magnetic resonance (NMR) [5], do not operate at the time scales necessary to record detailed large-scale protein motions. Second, traditional simulation methods such as molecular dynamics [6, 7, 8, 9] and Monte Carlo methods [10, 11] are computationally too expensive to simulate long enough time periods for anything other than small peptide fragments. Nevertheless, there has been some recent attention focused on developing methods for modeling and studying protein flexibility and motion.

One research effort has stemmed from work in robotic motion planning and is the foundation for the work presented in this dissertation. The motion planning problem is to find a valid path between some starting placement to an ending placement for a movable object. While traditionally intended for the field of robotics, solutions to this problem can be applied to a wide variety of domains such as computer animation [12, 13], computer-aided design (CAD) [14, 15], and computational biology [16, 17, 18, 19], including protein folding. This is done simply by changing the definitions for the movable object and path validity.

These motion planning solutions are based on the Probabilistic Roadmap Method (PRM) [20]. PRMs work by first building a graph, or roadmap, of the motion space. The roadmap nodes correspond to specific, valid, placements of the movable object, and an edge is placed between two nodes if a feasible motion is identified to take the movable object from one placement to the other. The resulting roadmap models the connectivity of the motion space and can be used to answer questions such as the existence of a valid path between a starting placement and ending placement.

Previously, we applied this methodology to study protein folding [18, 21, 22, 23]. The resulting roadmap contained thousands of energetically feasible pathways to the protein's known, stable, three-dimensional structure. We obtained promising results for several small proteins (60–100 amino acids) and validated our pathways by comparing secondary structure formation order with known experimental results [18]. However, our previous work is limited in the size of problems it can efficiently model and in the types of problems it can study.

A. Research Objective and Contributions

Our research objective is to extend our existing protein folding framework to handle larger proteins and to study a broader range of problems related to protein motion such as transition intermediates, allostery, and misfolding. We present techniques based on rigidity theory that allow us to study larger proteins by more efficiently characterizing the protein's energy landscape with fewer, more realistic conformations. Specifically, we develop a new sampling scheme exploiting rigidity analysis (Chapter IV, Section B) and describe new rigidity-based distance metrics for identifying connectable conformations (Chapter IV, Section C). These resulted in the dramatic reduction of roadmap sizes for a set of 26 proteins studied for stable secondary structure formation order (Chapter V, Section A.1, [24]). In addition, it is only with these rigidity-based techniques that we are able to detect the subtle folding differences of structurally similar proteins G, L, and their mutants (Chapter V, Section A.2, [24]).

We provide new methods for exploiting multiple known conformations of interest, such as the endpoints of a known protein transition, during roadmap construction. In particular, we bias conformation sampling around multiple conformations specified by the user, instead of only around a single specified conformation. This generalized methodology allows us to study other protein motion problems related to transitions between bound and unbound conformations, folding intermediates, misfolding, and allosteric interactions. We present results for several different transitions (Chapter III, Section B, [24]) and provide evidence that the transitions mapped by our approach are more realistic than those given by the computationally less expensive Morph Server [25].

We also present an Incremental Map Generation framework (IMG) [26] and provide metrics for identifying when the roadmap model is sufficient for answering motion questions for both protein folding problems (Chapter III, Section A.2 and Chapter V, Section A) and protein transition problems, i.e., motion between specific conformational states such as in binding interactions, (Chapter III, Section A.2). This allows the roadmap to be built automatically instead of relying on the user to specify a sampling density. We use roadmaps for protein G as an example to illustrate how the stabilization threshold affects roadmap size (Chapter III, Section A.3). We compare this approach, coupled with rigidity-based sampling and distance metrics, to previous methods [18, 21, 22, 23] and show that it produces significantly smaller roadmaps (and higher connectivity) with the same structure formation properties, e.g., secondary structure formation order distribution, (Chapter V, Section A.1, [24]). The pathway distributions also agree with experimental data when available for all proteins studied except for chymotrypsin inhibitor 2.

Finally, we use rigidity theory and analysis to study specific problems related to protein motion, such as secondary structure formation order (Chapter V, Section A), hydrogen exchange (Chapter V, Section B), and folding core identification (Chapter V, Section C). We compare our simulated exchange rates and ability to identify the folding core to available experimental data and other computational methods over a set of 21 different proteins of varying size and structure [27]. Our method is more successful in identifying the slowest exchanging (and therefore most stable) residues than COREX/BEST [28, 29] (Chapter V, Section B.4). It also performes better than the other computational approaches [30, 31] for folding core identification in terms of sensitivity and better than all but one class in terms of specificity (Chapter V, Section C.3, [27]).

B. Outline

The dissertation is outlined as follows. In Chapter II, we provide a background on proteins and existing techniques, both experimental and computational, for studying their motion. We discuss one of these computational methods, the Probabilistic Roadmap Method, in detail as it is the basis of this work. We conclude the chapter with a presentation of rigidity theory and analysis which is applied to many different parts of this research. In Chapter III, we present two new extensions to the original Probabilistic Roadmap Method framework for molecular motion: automatic, incremental roadmap construction and augmenting the model to study other motions besides folding (e.g., transitions). Chapter IV uses rigidity analysis to guide more efficient roadmap construction during both the sampling phase and the connection phase. We introduce a new sampling method that restricts conformation perturbation based on rigidity information, and we develop new distance metrics for judging conformation similarity with respect to their rigidity composition. Chapter V uses rigidity analysis to analyze the landscape model produced by the roadmap. We present three applications: distinguishing secondary structure formation order in roadmap pathways, modeling the relative hydrogen exchange of a residue during an unfolding experiment, and folding core identification. We conclude with some final remarks in Chapter VI.

CHAPTER II

PRELIMINARIES AND RELATED WORK

In this chapter we discuss related work on protein motion, both experimental and computational. We also discuss probabilistic roadmap methods for protein folding in detail as this is the foundation for much of this work. Finally, we provide background on rigidity analysis, a key technique used throughout this work.

A. Proteins

Proteins are essential to nearly all forms of life. These organic molecules are involved in nearly every process within the cell including catalyzing reactions, maintaining cell structure, cell signaling, and transporting molecules in and out of the cell. Proteins are made up of a sequence of amino acids. This sequence determines the protein's properties including its three dimensional shape, called the *native state*. The native state is important as it largely determines a protein's function and how it interacts with other proteins.

In the following we describe the protein's structure in more detail, energy functions for evaluating a protein conformation (or shape), and the protein's energy landscape.

1. Protein Structure

A protein's structure may be described at three levels: the primary structure, the secondary structure, or the tertiary structure [32]. We describe each below.

Primary Structure. A protein is made of a sequence of amino acids. This amino acid sequence is known as the protein's *primary structure*. There are 20 different types of amino acids. An amino acid may be partitioned into two parts, the backbone and the side chain, see Fig. 1. The backbone is identical in all amino acids. It is composed of a central carbon atom (C_α) (which has a hydrogen attached), an amino group (NH_2), and a carboxyl group ($COOH$). The side chain is attached to the C_α atom and is unique to the 20 different types of amino acids.

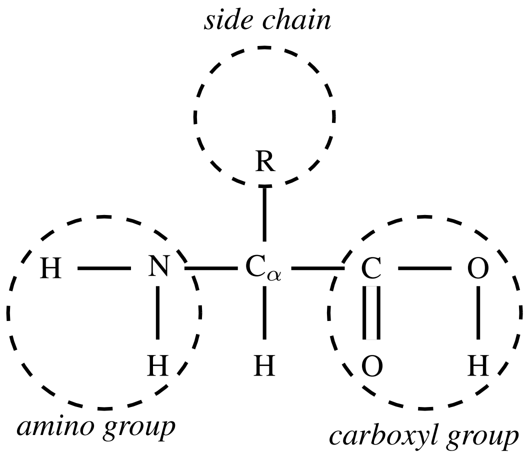


Fig. 1.: Amino acid structure. Every amino acid is composed of a central C_α atom (with attached H), an amino group and a carboxyl group, forming the backbone. Different amino acids are distinguished by the side chain, denoted by R.

Amino acids are joined together through a peptide bond between the carboxyl group of one amino acid and the amino group of the other. This joining releases a water molecule. The resulting chain has the backbone pattern $NH - C_\alpha H - CO - \ldots$ all along its length, see Fig. 2. Linked in this way, amino acids are commonly referred to as *residues*.

Secondary Structure. The sequence of amino acids in a protein folds into a

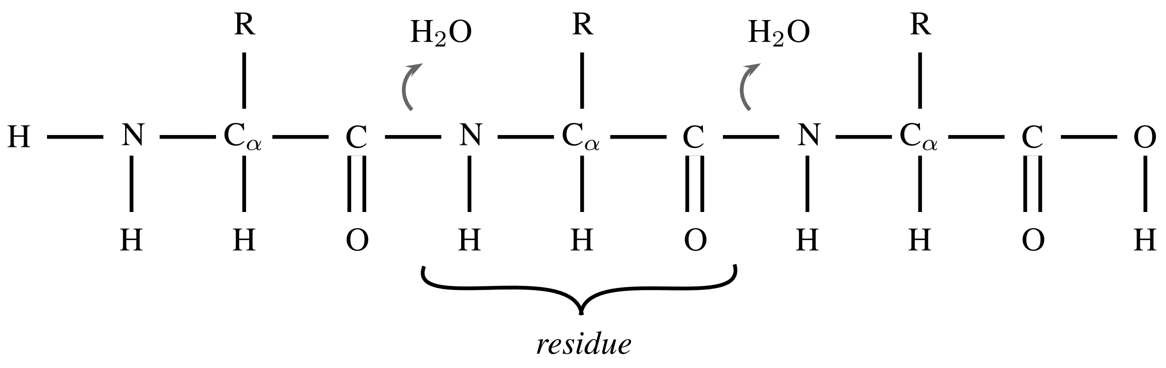
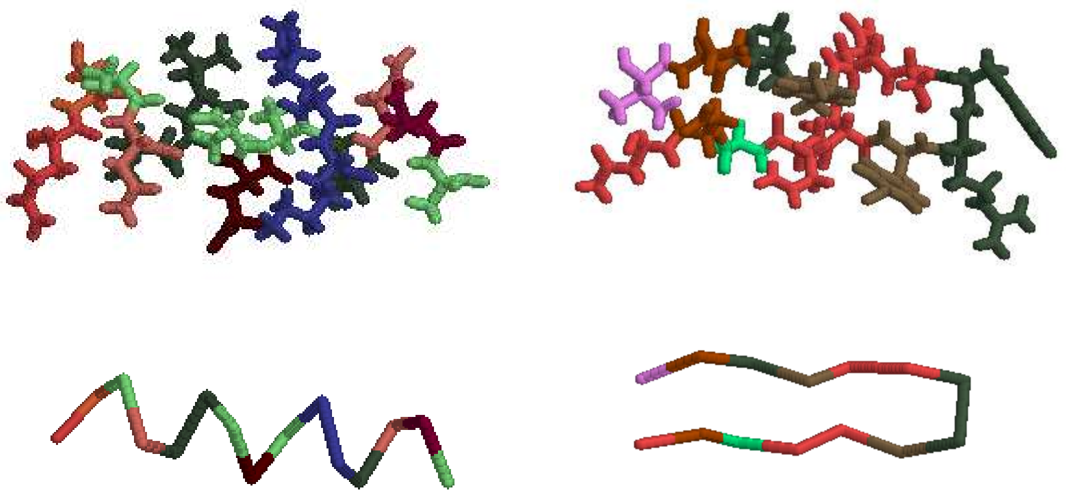

R
H_2O
R
H_2O
R
H — N — C$_\alpha$ — C — N — C$_\alpha$ — C — N — C$_\alpha$ — C — O
H H O H H O H H O H
residue

Fig. 2.: Linking of amino acids to form a protein. Three amino acids are joined by two peptide bonds and the release of two water molecules.

three dimensional shape, or conformation, called the *native state*. While native states between proteins vary widely, they are typically composed of a few repeated building blocks, called the *secondary structure*. There are two major types of secondary structure: α helices and β strands. These structures can be identified by their regular hydrogen bonding patterns.

The backbone of an α helix resembles a spring or coil, see Fig. 3(a), with the side chains on the outside. It may be either right-handed or left-handed in terms of the orientation of the coil. Most α helices exhibit hydrogen bonds between the amino group of the ith residue and the carboxyl group of the $i-4$th residue ($i \leftarrow i-4$). Residues in such helices create a 100^o turn about the helical axis resulting in 3.6 residues per coil/revolution. Other types of α helices (e.g., 310 helices and π-helices) vary this pattern (e.g., $i \leftarrow i-3$ and $i \leftarrow i-5$).

The β sheet is formed by one or more β strands, see Fig. 3(b). A β strand is a consecutive set of amino acids with extended backbones. β strands join together by lateral hydrogen bonds, alternating between the amino group of one strand and



(a) (b)

Fig. 3.: Examples of secondary structure: (a) an α helix and (b) a two-stranded, antiparallel β sheet. Both the wireframe (above) and backbone (below) views are shown. Residues are colored by type.

the carboxyl group of the other. They may either be oriented in the same direction (i.e., parallel) or in opposite directions (i.e., antiparallel). In antiparallel β sheets, the sequence of residues connecting two consecutive β strands is referred to as a β turn. Side chains in a β sheet are aligned with the sheet but alternate in their orientation (i.e., alternate between strands i and $i-1$ and strands i and $i+1$).

Tertiary Structure. Elements of secondary structure are linked together by segments of random coil to form the protein's *tertiary structure*, see Fig. 4. The tertiary structure describes the three dimensional placement of all the atoms in the protein. Folded proteins adopt a stable shape (or state) known as the *native state*.

This native state is uniquely determined by the amino acid sequence, or primary structure [33]. The tertiary structure is responsible in large part for the protein's function and how it interacts with other proteins. For example, enzymes catalyze reactions by binding to substrates at their active site. The active site is the area of the protein that interacts and binds to other molecules. This active site must provide both a good physical fit (e.g., no collisions) and a good chemical fit (e.g., attractively charged) to the substrate. It is the tertiary structure that brings distant amino acids along the protein backbone together to form the active site.

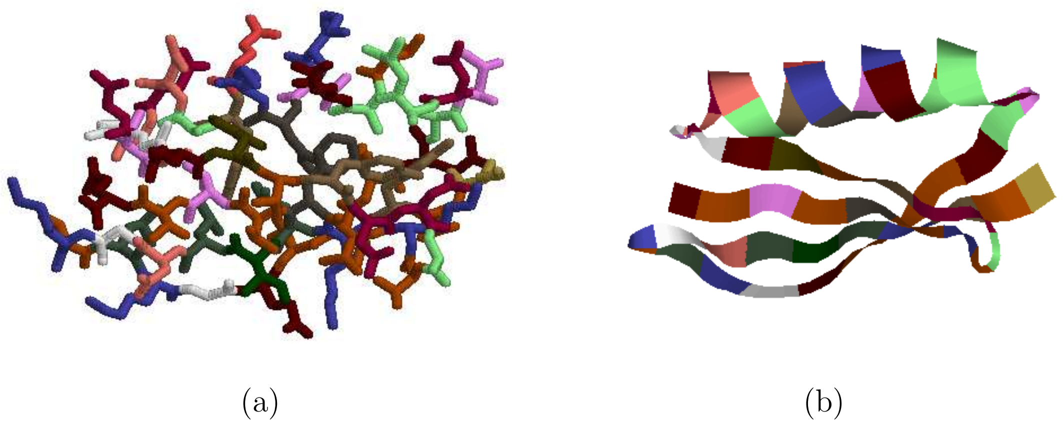


Fig. 4.: Example of tertiary structure of protein G: (a) ball-stick view and (b) ribbons view. Residues are colored by type.

2. Degrees of Freedom

While a protein folds into a single tertiary structure, it is not a rigid molecule. It can move and flex in response to changes in its environment such as temperature or the presence of other proteins or chemicals.

The atoms in a protein are connected by bonds. The lengths and angles of these bonds are not fixed, but in practice they vary very little. In many cases, they are treated as fixed [34]. In the results presented in this work, bond lengths and angles are considered fixed. However, this is not always an appropriate assumption (see Chapter III, Section B) and in the future we would like to eliminate this limitation.

The protein's flexibility is derived from rotations along a subset of the bonds. Each amino acid has two major rotational degrees of freedom, ϕ and ψ. The ϕ dihedral angle is the rotation about the $N - C_\alpha$ bond, and the ψ dihedral angle is the rotation about the $C_\alpha - C$ bond. Typically the dihedral angle ω between amino acids along the $C - N$ bond remains fixed at 180^o.

The protein's degrees of freedom can then be expressed as a set of ϕ and ψ angles, one pair for each amino acid. This greatly reduces the protein's degrees of freedom from $3X$ to $2N$, where X is the number of atoms in the protein and N is the number of amino acids.

3. Potential Functions

Proteins are composed of interacting atoms, both with the surrounding solvent and with each other. These interactions range from very strong (e.g., peptide bonds) to moderately strong (e.g., hydrogen bonds) to relatively weak (e.g., hydrophobic interactions). These interactions cause the protein to fold or unfold depending on environmental conditions (e.g., temperature or the presence of denaturant).

These interactions may be modeled by potential energy functions. Potential energy functions aim to quantify the forces present in a given protein conformation and evaluate the likelihood of observing that conformation under the given environmental conditions. In general, a potential energy function may be expressed as a summation of various terms including bond length flexing, bond angle flexing, dihedral angle flexing, van der Waals interactions, and electrostatic interactions [6].

Potential functions vary in the approximations and assumptions made and consequently in their accuracy and expense. In this work, we use a coarse potential function introduced in [18]. We use a step function approximation of the van der Waals potential component and model side chains as spheres with zero degrees of freedom. All spheres have the same radii. If any two side chain spheres are too close (i.e., less than 2.4Å during sampling and 1.0Å during connection), a very high potential is returned. Otherwise, the potential is:

$$U_{tot} = \sum_{constraints} K_d\{[(d_i - d_0)^2 + d_c^2]^{1/2} - d_c\} + E_{hp} \tag{2.1}$$

where K_d is 100 kJ/mol, d_i is the distance between the endpoints of the ith constraint, and $d_0 = d_c = 2$Å as in [6]. The first term represents constraints favoring known secondary structure through main-chain hydrogen bonds and disulphide bonds. The second term is the hydrophobic effect. The hydrophobic effect (E_{hp}) is computed as follows: if two hydrophobic residues are within 6Å of each other, then the potential is decreased by 20 kJ/mol.

In previous work, we compared the results of our method using this coarse potential function and using a detailed all-atoms model [22]. We showed that our method could detect the subtle folding differences between structurally similar proteins G and L with either energy function. While the all-atoms energy function had a greater percentage of folding pathways matching the experimentally determined formation order, it required two weeks of computation time versus less than 1 day with the coarse potential.

4. Energy Landscapes

The energy landscape is the set of all protein conformations and their associated energies. Fig. 5 gives a simplified 3D visualization of this high-dimensional space, where the xy-plane represents the parameters defining a protein conformation (e.g., the ϕ and ψ angles for each amino acid in our case) and the z-axis is the potential energy function. It is theorized that the protein's native state is at the bottom of this energy landscape and that the landscape is funnel-shaped to explain the rapid nature of protein folding [35, 36, 37]. The landscape may contain many local minima as illustrated here. Protein folding can then be viewed as a ball rolling down this energy landscape.

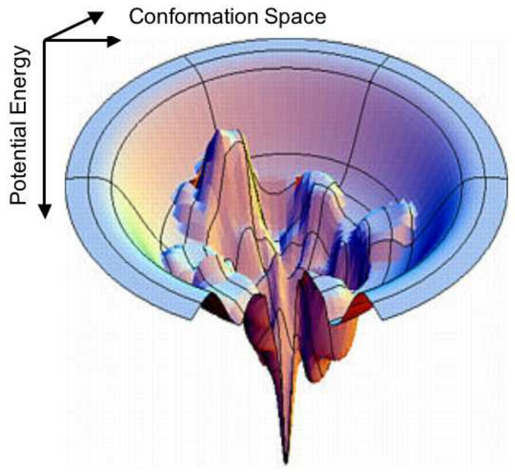


Fig. 5.: Visualization of the protein's energy landscape [38]. The xy-plane corresponds to the protein's conformation space, and the z-axis is the potential energy.

Different proteins will have different energy landscapes. These energy landscapes characterize the folding process. Thus, to model protein folding, we aim to build a model of this energy landscape.

B. Protein Motion

As described above, proteins are not static structures. They move and flex with changes in their environment. There have been both experimental and computational methods developed to study protein motion. Many of these methods focus specifically on protein folding. Here we discuss both experimental and computational techniques, highlighting their strengths and weaknesses.

1. Experimental Methods

There have been several advances in experimental techniques to study protein motion including circular dichroism, fluorescence experiments, hydrogen exchange and pulse labeling, NMR spectroscopy, and time-resolved X-ray crystallography.

Circular Dichroism. Circular dichroism (CD spectra) measures the absorption of polarized light for the entire population of protein conformational states as a function of thermal stability [39]. There are two main methods [40]: near UV and far UV. Near UV experiments use wavelengths between 250nm and 350nm to examine the formed tertiary structure. Far UV uses wavelengths between 190nm and 250nm to probe the formation of secondary structure. CD experiments had been limited to 10 milliseconds but recently have been extended to 400 microseconds [41].

Fluorescence Experiments. Fluorescence experiments monitor change in fluorescence as a function of denaturant. Three primary categories of fluorescence experiments are stopped-flow methods, continuous flow methods, and independent equilibrium methods [40]. During stopped-flow experiments, denaturant is added over a series of timesteps, and fluorescence is measured after each addition. Continuous flow methods instead monitor fluorescence during a continuous addition of denaturant. Independent equilibrium methods measure the fluorescence intensity in different denaturant conditions.

Hydrogen Exchange. Hydrogen exchange mass spectrometry and pulse labeling experiments can investigate protein folding by identifying which parts of the structure are most exposed or most protected [42]. From this data, one can infer which portions of the protein fold first and which are last to form, up to the millisecond timescale.

NMR Spectroscopy. NMR spectroscopy is another experimental tool well-suited to study protein dynamics because it can acquire site-specific, detailed information on a variety of timescales, ranging from picoseconds [43] to milliseconds [44]. It has been used to study both side-chain motion and backbone motion. See [45] for a recent review of current techniques.

Time-resolved X-ray Crystallography. Time-resolved Laue X-ray diffraction has been used to identify intermediate structures along a reaction pathway. This technique aims to not only study intermediate structures, but to also gather their rates of transition. The first work on myoglobin [46] and photoactive yellow protein [47] identified motions on the picosecond to microsecond timescale.

2. Computational Methods

Several computational approaches have been applied to study protein motions and folding. These include lattice models, molecular dynamics, Monte Carlo methods, simulated annealing, statistical mechanical models, roadmap-based methods, and ‘morphing’ between known conformational states. We describe them each below.

Table I summarizes their properties.

Lattice Models. Lattice models [48, 36, 49] represent the protein as a chain of beads constrained to a rigid lattice. Each bead corresponds to a single amino acid. They are typically divided into just two types: hydrophobic and polar. These simplifications allow them to be computationally studied in full detail. While these models have provided many theoretical insights, they are not used on actual proteins in practice.

Molecular Dynamics. Molecular dynamics [6, 7, 8, 9] simulates the forces on all the atoms at each timestep to produce a motion trajectory. It requires the definition of a detailed and accurate potential function to model the atomic forces. Each run of this trajectory-based method provides a single, high resolution transition pathway but is computationally intensive. For example, it is limited to studying proteins with less than 130 amino acids [50], even when it uses massive computational resources, such as tens of thousands of PCs in the Folding@Home project [51, 52] or large supercomputers [53]. Thus, it is computationally infeasible to study global properties of the folding landscape using them.

Monte Carlo Simulations. Monte Carlo simulations [10, 11] are random walks on the protein's energy landscape that favor lower energy transitions. At each timestep, the next conformational state is chosen from a set of neighboring states (including the current state) based on the transition probabilities between them. Such methods can suffer by becoming trapped in local minima. Similar to molecular dynamics, it provides a high resolution pathway at a large computational cost. Therefore, it is not practical to apply it to large proteins or to study the folding landscape though the computation of many pathways.

Statistical Mechanical Models. Statistical mechanical models [54, 55, 56] compute statistics about the global energy landscape. From these statistics, they

Table I.: A comparison of protein motion models.

Approach	**Folding Landscape**	**# Paths Produced**	**Path Quality**	**Compute Time**	**Native State Needed**
Lattice Model	*Not used on real proteins*				
Molecular Dynamics	No	1	Good	Long	No
Monte Carlo	No	1	Good	Long	No
Simulated Annealing	No	1	Good	Long	No
Statistical Mechanical Model	Yes	0	n/a	Short	Yes
Roadmap-Based	Yes	Many	Approx.	Short	Yes
Morphing	No	1	Approx.	Short	Yes[a]
Flexibility Models	Yes	0	n/a	Short	Yes

[a]The 'morph' between two conformational states requires the three dimensional structures of both states.

can infer ensemble properties about the protein's motion. They are not designed to produce individual pathway trajectories. They typically rely on simple energy functions heavily biased by the native state. Thus, their accuracy degrades as the protein moves away from this state. These methods are computationally efficient but can only provide results for global averages of the energy landscape and the underlying kinetics. This is undesirable for some proteins where the energy landscape is known to be partitioned into two or more types of kinetic pathways, such as hen egg-white Lysozyme [57].

Roadmap-Based Methods. Several computational methods have been proposed based on the Probabilistic Roadmap Method (PRM) [20] originally developed for robotic motion planning. PRMs model their motion space (e.g., the energy landscape for proteins) by building a network, or graph, of valid motions in that space. This graph is referred to as the *roadmap*. PRMs first randomly sample valid conformations and add them to the roadmap. Then, neighboring conformations are connected if there exists a feasible transition between them. Often these connections are weighted to reflect the energetic feasibility of such a transition.

Singh, Latombe and Brutlag first applied PRMs to protein/ligand binding [16]. In subsequent work, our group used a PRM variant [58] on this problem [59]. Our group was the first to adapt the PRM framework to model protein folding pathways [18, 21, 22, 23]. Apaydin et al. [17, 60] have also applied PRMs to proteins, however their work differs from ours in several aspects. First, they model the protein at a much coarser level considering each secondary structure to be rigid. Second, while our focus is on studying the transition process, their focus has been to compare the PRM approach with other computational methods such as Monte Carlo simulation. In recent work, Cortes and Simeon used a PRM-based approach to model long loops in proteins [61, 62]. Finally, we adapted the PRM framework to study RNA folding kinetics [19].

Database of Macromolecular Movements. Gerstein et. al. have developed the Database of Macromolecular Movements [63, 25] to classify protein motions. Their Morph server takes two conformations of interest and produces a 'morph' movie between them. They can produce 'morph' movies in just a few minutes on a desktop PC. Their database currently includes more than 240 distinct protein motions [64].

To create a 'morph' between two target conformations, they first perform an alignment. This allows the comparison of proteins with different sequences. Then, an iterative 'sieve-fit' procedure [65, 66] produces a superposition of the target conformations. A conformational 'morph' is created by interpolating the Cα atom positions between the two superimposed conformations. Each intermediate conformation along the interpolation is energy minimized. This interpolation method, called adiabatic mapping, was selected because it has modest computational requirements yet produces chemically reasonable morphs. Adiabatic mapping methods have problems with some kinds of large deformations [67]. More recently, they have also added a different interpolation method based on the Framework Rigidity Optimized Dynamics Algorithm (FRODA) [68] that may be used upon request. FRODA determines the rigid regions that are mutually present in both target conformations, and then performs a Monte Carlo simulation while keeping these regions fixed.

Protein Flexibility and Rigidity Models. Several computational approaches have studied rigidity and flexibility in proteins. One approach infers the protein's flexibility/rigidity by comparing different known conformations of the protein [69, 70, 71]. Molecular dynamics has been used to extract flexibility information from simulated motion [72, 73, 74, 75]. A third approach studies rigidity and flexibility of a single protein conformation [76, 77, 78, 79, 80]. In particular, many methods have used a constraint counting technique called the pebble game [81, 82, 83, 84] to better simulate motion [85, 86, 87, 30, 88]. We discuss this last class of methods in more detail later in this chapter in Section D.

C. Probabilistic Roadmap Method Preliminaries

As discussed previously, the Probabilistic Roadmap Method (PRM) [20] has been highly successful in solving motion problems for objects with many degrees of freedom. Here we describe the method in more detail.

1. Motion Planning and C-space

The motion planning problem is to find a valid path for a movable object from a start placement to a goal placement in a given environment. We will refer to object placements in an environment as *conformations.* A conformation simply provides values for all the parameters needed to uniquely specify an object placement (i.e., its degrees of freedom). For example, a rigid body robot in a three-dimensonal workspace has 6 degrees of freedom: 3 for the placement of the body along the x-axis, y-axis, and z-axis and 3 for rotations about each axis. Its conformation is a 6-dimensonal vector: $\langle x, y, z, \alpha, \beta, \gamma \rangle$. A protein may be modeled with 2 degrees of freedom per amino acid. Thus a conformation for a protein containing N amino acids has a $2N$-dimensional vector: $\langle \phi_1, \psi_1, \ldots \phi_N, \psi_N \rangle$.

The *conformation space* (C-space) is the set of all conformations, feasible or not [89]. The set of feasible conformations is the *free C-space* (C_{free}), and the set of unfeasible conformations is referred to as the *C-space obstacles* (C_{obst}). Motion planning then becomes a search for a continuous trajectory in C_{free} from the start conformation to the goal conformation. In practice, it is too computationally expensive to compute C_{obst} explicitly. Instead, conformations are determined feasible or not based on some validity checker, typically collision detection for robotics problems and an energy calculation for proteins.

The C-space is a useful abstraction so motion planning methods may be applied to many types of movable object. Its dimension is the same as the number of degrees of freedom in the movable object. From the examples above, the rigid body robot in a three-dimensional workspace as a 6 dimensional C-space while the protein has a $2N$-dimensional C-space.

2. The PRM Algorithm

PRMs work by first sampling random points in the movable object's C-space. Only samples meeting certain feasibility requirements are added as nodes to the roadmap. Then edges are added between samples in the roadmap if there exists a feasible path or transition between them that can be determined by some simple *local planner*. A *local planner* is any method that computes a sequence of conformations between two endpoints. For example, a straight line in C-space is a local planner where conformations are linearly interpolated between the two endpoints. In practice, only "near-by" samples are attempted for connection as determined by some *distance metric*. One commonly used *distance metric* is the Euclidean distance in C-space between the two samples. Roadmap construction is outlined in Algorithm C.1.

Algorithm C.1 consists of the following primitive operations: sampling a point in C-space, determining the feasibility of points in C-space (e.g., checking collision), and computing distances in C-space. Let C-space have d dimensions, one for each degree of freedom of the movable object. Sampling a point in C-space uniformly at random (line 2) takes $O(d)$ time. Let the time to check a sample's feasibility take $O(f)$ time (line 3). This largely depends on the application domain. For example, when using collision detection as a feasibility requirement, the time to preform this

Algorithm C.1 The Probabilistic Roadmap Method

Input: A description of a movable object and its environment, the number of samples n to add to the roadmap, and the number of neighbors k to examine during sample connection.

Output: A roadmap R modeling the feasible portions of the C-space.

```
1:  for $i = 1 \ldots n$ do
2:      Let $q$ be a randomly sampled conformation.
3:      if $q$ is feasible then
4:          Add $q$ to $R$.
5:      end if
6:  end for
7:  for each conformation $r \in R$ do
8:      Let $Q$ be the set of $k$-nearest neighbors to $r$ in $R$.
9:      for each conformation $q \in Q$ do
10:         if there exists a feasible transition between $r$ and $q$ then
11:             Add the edge $(r, q)$ to $R$.
12:         end if
13:     end for
14: end for
return $R$.
```

check is a function of the number of triangles describing the geometry of the movable object and its environment. Computing the distance between two points in C-space (e.g., Euclidean distance) usually takes time linear in the dimension of C-space, d.

The running time of the algorithm is as follows. Sampling n nodes for the roadmap (lines 1–6) takes $O(ndf)$ time using these primitive operations. A typical method for determining the k-nearest neighbors of a sample (line 8) requires computing the distance between it and all the other samples in the roadmap. This brute force approach takes $O(nd)$ time in each iteration of the `for` loop in line 7. For each sample, there are then k local planning attempts (lines 9–10). The cost of a local planning attempt depends on the local planning method selected and the number of feasibility checks it needs to accept or reject the edge. For example, a straight line local planner will interpolate between the two samples at a fixed resolution. In the worst case, it must check every intermediate conformation along this straight line in C-space. Assume that the average number of feasibility checks is $O(l)$. Thus, the expected cost of a local planning attempt is $O(lf)$. The total time taken to connect a sample to the roadmap is $O(nd + klf)$. Therefore, the time required for the connection phase (lines 7–14) is $O(n^2d + nklf)$. The running time of the entire PRM algorithm becomes $O(ndf + n^2d + nklf)$. Typically, k, d, and f are much smaller than n resulting in a final complexity of $O(n^2)$.

The roadmap can be used to answer questions about the planning space, such as the existence of a feasible path between a start and goal conformation. To answer this type of query, for example, the start and goal conformations are added to the roadmap and connected to existing samples using the local planner as before. Then, a simple graph search can extract a path (i.e., a sequence of nodes and edges in the roadmap) if the start and goal are connected in the roadmap. A major strength of PRMs is that they are simple to apply, even for high degrees of freedom problems, only requiring the ability to randomly sample points in C-space and test them for feasibility.

3. Modifications to PRMs for Proteins

The overall strategy follows the general methodology presented above with only a few modifications to adapt the framework for modeling protein motion.

Protein Model. The protein is modeled as an articulated linkage. Using a standard modeling assumption for proteins that bond angles and bond lengths are fixed [34], the only degrees of freedom in our model are the backbone's ϕ and ψ torsional angles. These are modeled as revolute joints taking values $[0, 2\pi)$.

Conformation Sampling. Due to the high dimensionality of the protein's C-space, uniform sampling would take too long to provide sufficiently dense coverage of the region surrounding the native state. Instead, we bias our sampling around the native state by iteratively applying small Gaussian perturbations (i.e., random perturbations following a Gaussian distribution) to existing conformations.

Sample Retention. The traditional collision-free requirement is replaced with a potential energy calculation. A conformation q is added to the roadmap probabilistically based on its potential energy $E(q)$ as follows:

$$P(\text{accept} \quad \text{q}) = \begin{cases} 1 & \text{if } E(q) < E_{\text{min}} \\ \frac{E_{\text{max}} - E(q)}{E_{\text{max}} - E_{\text{min}}} & \text{if } E_{\text{min}} \leq E(q) \leq E_{\text{max}} \\ 0 & \text{if } E(q) > E_{\text{max}} \end{cases} \tag{2.2}$$

where E_{min} is the potential energy of the open chain and E_{max} is $2E_{\text{min}}$.

Edge Weighting. Edges in the roadmap are weighted to reflect the transition's energetic feasibility. The weight for the edge (q_1, q_2) is a function of all the intermediate conformations along the edge $\{q_1 = c_0, c_1, c_2, \ldots, c_{n-1}, c_n = q_2\}$. For each pair of consecutive conformations c_i and c_{i+1}, the probability P_i of transitioning from c_i to c_{i+1} depends on the difference in their potential energies $\Delta E_i = E(c_{i+1}) - E(c_i)$:

$$P_i = \begin{cases} e^{\frac{-\Delta E_i}{kT}} & \text{if } \Delta E_i > 0 \\ 1 & \text{if } \Delta E_i \leq 0 \end{cases} \tag{2.3}$$

The edge weight $w(q_1, q_2)$ is then $\sum_{i=0}^{n-1} -\log(P_i)$. This weighting scheme allows us to extract low energy paths in the roadmap using graph algorithms for computing shortest paths.

Path Extraction. The roadmap contains thousands of folding pathways. For many applications, shortest paths are extracted as the most energetically feasible. However, in some instances a more stochastic approach is warranted. One such approach is Map-based Monte Carlo simulation (MMC) [90]. MMC is similar to traditional Monte Carlo simulation [10, 11] except that it is a walk on an approximate landscape model (i.e., the roadmap) instead of on the complete energy landscape. The likelihood of transitioning between conformations is probabilistically biased by their Boltzmann transition probabilities. These transition probabilities are based on the edge weight in the roadmap. Unlike traditional Monte Carlo simulation, it can be applied to larger structures because it operates on a simplified landscape.

D. Rigidity Analysis

Here, we use a rigidity analysis technique belonging to the third class of approaches called the pebble game [81, 82, 83, 84, 91] to better simulate motion. The pebble game has been an effective tool for studying protein flexibility and rigidity [85, 86, 87, 30, 88].

1. The Pebble Game

The pebble game [81, 82, 83] is a constraint counting algorithm based on Laman's theorem [92] which determines the degrees of freedom of a set of distance constraints between point masses in the plane, along with its rigid and flexible regions. Fig. 6(a) gives an example set of constraints. In 2D, the pebble game creates a graph where every vertex represents a point mass from the input set. It assigns each vertex in the graph two pebbles, representing the two degrees of freedom of a point mass in the plane, see Fig. 6(b). Then each constraint is iteratively examined to determine if it is independent or redundant. If two free pebbles can be placed on both endpoints of the constraint, then the constraint is marked independent and is covered by a pebble from one of its incident vertices, see Fig. 6(c). Covering a constraint $c = (u, v)$ can be represented by adding a directed edge $e = (u \rightarrow v)$ between the vertices and removing a free pebble from u. Once a constraint is covered by a pebble, it remains covered through the rest of the game, although which vertex the pebble comes from may change (i.e., the direction of the edge may change).

Pebbles may be rearranged by moving a free pebble from an incident vertex onto a covered constraint and moving the covering pebble from the other incident vertex off (i.e., reversing the direction of the edge $e = u \rightarrow v$, removing a free pebble from u, and adding a free pebble to v). Fig. 7 provides an example of pebble rearrangement when collecting an additional free pebble on vertex 6. If pebbles cannot be rearranged to get two free pebbles on both of a constraint's endpoints, then the constraint is marked redundant and indicates a rigid region in the graph. After every constraint is examined, the remaining free pebbles indicate the graph's degrees of freedom.

The pebble game algorithm is given in Algorithm D.1 and follows the presentation in [83, 91]. It is parameterized by two variables k, the number of degrees of freedom

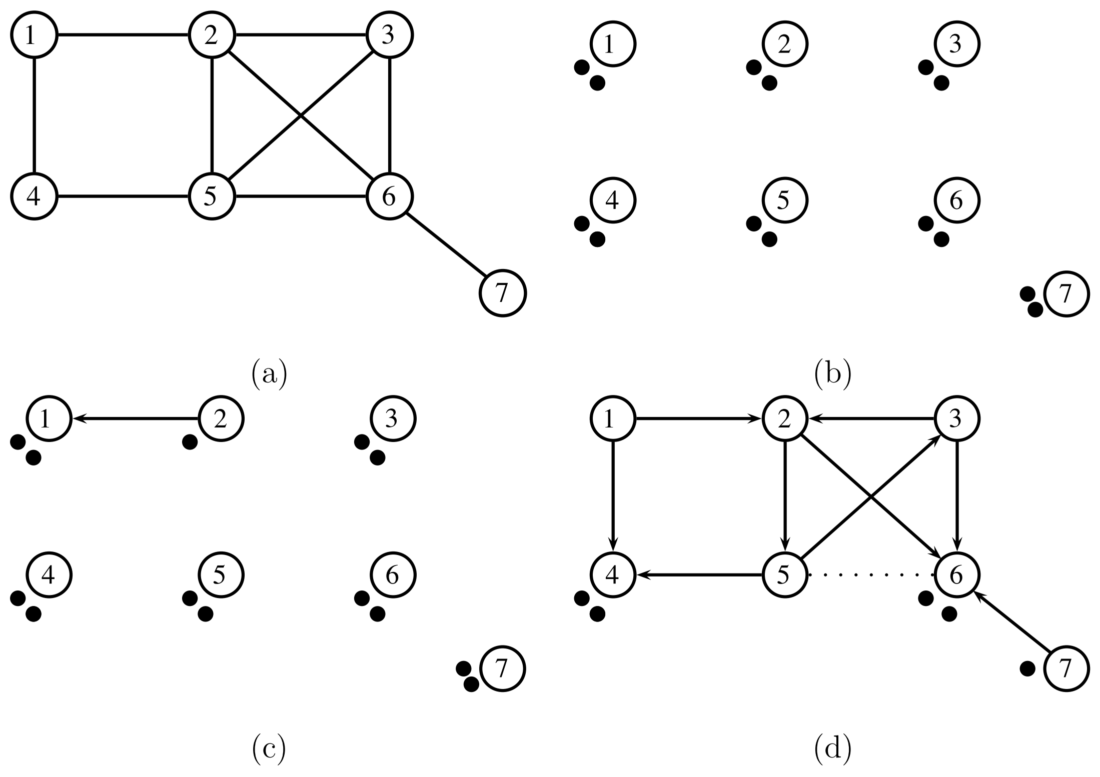
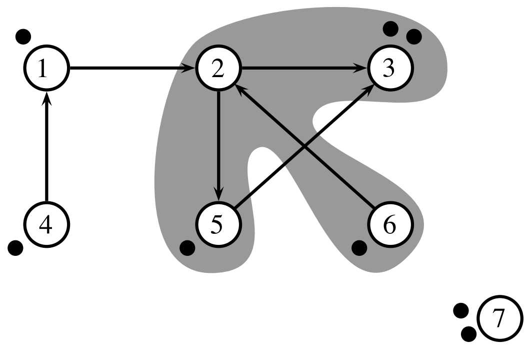
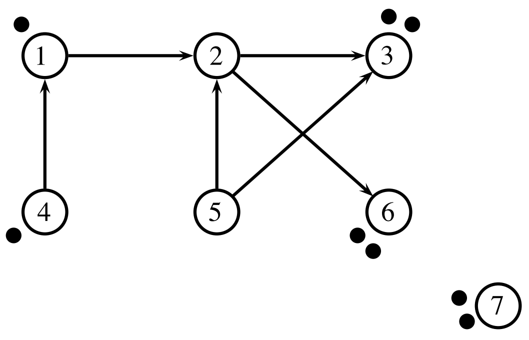


Fig. 6.: 2D pebble game example. (a) The set of constraints to consider. (b) The pebble game graph initializes each vertex with 2 pebbles. (c) Constraints are added to the graph if 4 pebbles can be collected on the constraint's endpoints indicating it is independent. The constraint between vertex 1 and vertex 2 is independent. A pebble is removed from vertex 2 and a directed edge ($2 \rightarrow 1$) is added. (d) The final graph after all constraints are considered. Redundant constraints indicate overstressed/rigid regions of the graph and are marked by dotted lines.



(a)



(b)

Fig. 7.: Pebble rearrangement example. (a) When considering the constraint (3,6), two free pebbles must be collected on each endpoint. Searching for a free pebble proceeds as a depth-first-search indicated by the shaded region where the constraint endpoints (3 and 6) are ignored. A free pebble is found on vertex 5. (b) After pebble rearrangement. Here, the pebble was removed from vertex 5 and traveled the depth-first-search path, reversing edges along the way.

Algorithm D.1 The Pebble Game

Input: A list of points V, a list of constraints C between points in V, the number of pebbles k per point, and the number of pebbles l required to mark a constraint as independent.

Output: A graph G that the pebble game has been played on, a list of independent edges E_I, and a list of redundant edges E_R.

Uses: Algorithm D.2.

```
 1: Let $E_I = \{\emptyset\}$.
 2: Let $E_R = \{\emptyset\}$.
 3: Let $G$ be the pebble game graph. Add a vertex to $G$ for every $v \in V$. Give each
    vertex in $G$ $k$ free pebbles.
 4: for each constraint $c = (v, w) \in C$ do
 5:     Collect_Pebbles($G$, $v$, $k$).
 6:     if Collect_Pebbles($G$, $w$, $l - k$) is successful then
 7:         Constraint $c$ is independent. Add $c$ to $E_I$.
 8:         Release blocked pebbles associated with $v$ and $w$.
 9:         Cover $c$ in $G$ with a free pebble from $w$.
10:     else
11:         Constraint $c$ is redundant. Add $c$ to $E_R$.
12:         Release blocked pebbles associated with $v$ and $w$.
13:     end if
14: end for
return $G$, $E_I$, and $E_R$.
```

associated with a vertex, and l, the number of collected free pebbles required to label a constraint as independent. For a planar graph in 2D as described above, $k = 2$ and $l = 4$. The algorithms Alg. D.2 – D.4 describe the method for collecting pebbles on a vertex as illustrated in Fig. 7. They are based on a breadth-first-search traversal of the pebble game graph. Because there are at most $O(2n)$ edges in the graph [83], each pebble search takes $O(n)$. The entire pebble game consists of l pebble searches for each edge. Because $l << n$ and there are $O(2n)$ edges in the graph, there are $O(n)$ pebble searches. The resulting complexity of the entire pebble game algorithm is therefore $O(n^2)$. Proofs of correctness can be found in [83, 91].

The 2D pebble does not generalize to three dimensions for arbitrary constraint sets, but it can be applied to 3D bond-bending networks [84]. A bond-bending network is a truss structure with constraints between nearest neighbors and next-nearest neighbors. A protein, with fixed bond lengths and bond angles, forms a bond-bending network where atoms are modeled as vertices with 3 degrees of freedom and bonds are modeled as edges. This is called the bar-joint model, see Fig. 8(a). In the 2D pebble game, edges/constraints may be placed in any order, however, in the 3D pebble game, order matters for correctness. The first constraint must be a nearest neighbor constraint (i.e., a bond length constraint). Then all the associated next-nearest neighbor constraints (i.e., associated bond angle constraints) must be placed before placing another nearest neighbor constraint. For proteins, the first constraint must be a bond length constraint and all the associated bond angle constraints must be placed before another bond length constraint is placed. The bar-joint model has been successfully used by several applications to study protein rigidity and flexibility [85, 86, 87, 30, 88].

Algorithm D.2 Collect_Pebbles

Input: A pebble game graph $G = (V, E)$, a vertex v, and the number of pebbles to collect n.

Output: True or false. Collected pebbles marked as blocked in G.

Uses: Algorithms D.3 and D.4.

```
1: for $i = 1 \dots n$ do
2:     Let $V_{seen} = \{\emptyset\}$ be the set of vertices visited.
3:     Let $Path$ be an array of size $|V|$ with all entries initialized to -1.
4:     if Find_Pebble($G$, $v$, $V_{seen}$, $Path$) is successful then
5:         Rearrange_Pebbles($G$, $v$, $Path$).
6:         Mark the resulting free pebble as blocked.
7:     else
return false.
8:     end if
9: end for
return true.
```

Algorithm D.3 Find_Pebble

Input: A pebble game graph $G = (V, E)$, a vertex v, a set of visited vertices V_{seen}, and an array *Path* of size $|V|$ recording the path to the free pebble.

Output: True or false. *Path* updated to store path to the free pebble.

```
1: Add $v$ to $V_{seen}$.
2: $Path[v] = -1$.
3: **if** $v$ has a free pebble **then**
return true.
4: **end if**
5: **for** each unblocked pebble covering edge $e = (v, w)$ **do**
6:     **if** $w \notin V_{seen}$ **then**
7:         $Path[v] = w$.
8:         **if** Find_Pebble($G$, $w$, $V_{seen}$, $Path$) **then**
return true.
9:         **end if**
10:    **end if**
11: **end for**
return false.
```

Algorithm D.4 Rearrange_Pebbles

```
Input: A pebble game graph $G$, a vertex $v$ to rearrange the free pebble to, and an
    array $Path$ recording the path to the free pebble found by Free_Pebble.
Output: The pebbles in $G$ rearranged such that a free pebble appears on $v$.
1: Let $w = Path[v]$.
2: if $w \neq -1$ then
3:     Rearrange_Pebbles($G$, $w$, $Path$).
4:     Uncover the edge $(v, w)$ with a free pebble from $v$.
5:     Cover the edge $(w, v)$ with a free pebble from $w$.
6: end if
```

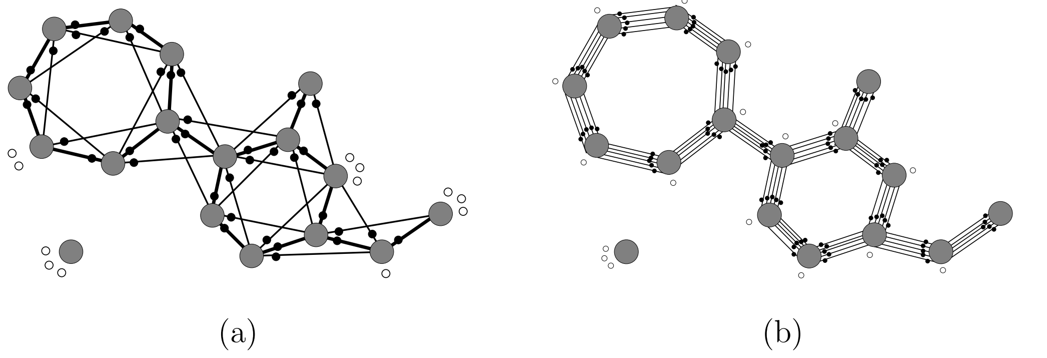


Fig. 8.: Rigidity models for a sample molecule: (a) bar-joint model and (b) body-bar model. For the bar-joint model, nearest neighbor constraints (i.e., bond lengths) are shown with thick lines and next-nearest neighbor constraints (i.e., bond angles) are shown with thin lines. Both models yield the same degrees of freedom and rigid/flexible regions.

An alternative model, the body-bar model, represents atoms as rigid bodies with 6 degrees of freedom and the torsional bonds between them as 5 bars/constraints [93], see Fig. 8(b). This model is conjectured to be equivalent to the 3D bond-bending network. There exists an exact mapping between the two models, provided that the body-bar model allows sub-dimensional rigid bodies[1][84]. For 3D bond-bending networks using the body-bar model, the pebble game parameters are $k = 6$ and $l = 7$. Recall that the pebble game takes $O(n^2)$ time, where n is the number of vertices in the pebble game graph. If the pebble game is played on an all-atoms model of the protein, then n is the number of atoms the protein contains (which is directly proportional to its length). Thus the playing the pebble game on a protein takes quadratic time with respect to the protein's length.

2. Rigid Cluster Decomposition

In addition to labeling constraints as independent or redundant, rigidity analysis can partition the vertices in the graph into *rigid clusters*. A *rigid cluster* is a set of vertices such that any two vertices in the set are rigid with respect to each other. The only degrees of freedom in a rigid cluster are rigid body translations and rotations. Constraints within a rigid cluster are rigid while constraints between rigid clusters are flexible.

With a little bookkeeping added to the pebble game algorithm, rigid cluster decomposition can be done at the same time. Algorithm D.5 augments the original pebble game algorithm (Algorithm D.1) to perform incremental rigid cluster decomposition. It follows the presentation in [91]. The original pebble game algorithm is

[1]Isolated bodies contain 3 degrees of freedom, bodies with 1 nearest neighbor (e.g., dimers) contain 5 degrees of freedom, and bodies with more than 1 nearest neighbor contain 6 degrees of freedom.

Algorithm D.5 The Pebble Game with Rigid Cluster Decomposition

Input: A list of points V, a list of constraints C between points in V, the number of pebbles k per point, and the number of pebbles l required to mark a constraint as independent.

Output: A graph G that the pebble game has been played on, a list of independent edges E_I, a list of redundant edges E_R, and a list of rigid clusters RCD.

Uses: Algorithms D.2 and D.6.

```
 1: Let $E_I = \{\emptyset\}$.
 2: Let $E_R = \{\emptyset\}$.
 3: Let $G$ be the pebble game graph and $V$ be the vertices in it. Give each vertex $k$
    free pebbles.
 4: Let $RCD$ be a set of rigid clusters. For each vertex $v \in V$, add a rigid cluster
    $rcd = \{v\}$ to $RCD$.
 5: **for** each constraint $c = (v, w) \in C$ **do**
 6:     **if** there is a rigid cluster $rcd \in RCD$ such that $v \in rcd$ and $w \in rcd$ **then**
 7:         Constraint $c$ is redundant. Add $c$ to $E_R$.
 8:     **else**
 9:         Collect_Pebbles($G$, $v$, $k$).
10:         Collect_Pebbles($G$, $w$, $l - k$).
11:         Constraint $c$ is independent. Add $c$ to $E_I$.
12:         Release blocked pebbles associated with $v$ and $w$.
13:         Cover $c$ in $G$ with a free pebble from $w$.
14:         Update_Rigid_Cluster_Decomposition($G, RCD, c$).
15:     **end if**
16: **end for**
**return** $G$, $E_I$, $E_R$, and $RCD$.
```

Algorithm D.6 Update_Rigid_Cluster_Decomposition

Input: A partially played pebble game graph G, a list of rigid clusters RCD found so far in G, and a constraint $c = (v, w)$ just found to be independent.

Output: RCD is updated considering the independent constraint c.

Uses: Algorithm D.7.

```
1: **if** there are fewer than $l$ free pebbles on $v$ and $w$ **then**
2:     Let $R_{v,w}$ = Reachability($v$) $\cup$ Reachability($w$).
3:     **if** there is not a vertex $u \neq v, w \in R_{v,w}$ with a free pebble in $G$ **then**
4:         Let $rcd = \{\emptyset\}$ be a new rigid cluster. Add $rcd$ to $RCD$.
5:         **for** each vertex $u \in R_{v,w}$ **do**
6:             Let $rcd_u$ be the rigid cluster containing $u$ in $RCD$.
7:             $rcd = rcd \cup rcd_u$.
8:             Remove $rcd_u$ from $RCD$.
9:         **end for**
10:        Let $Q = \{\emptyset\}$ be a queue.
11:        Add to $Q$ all vertices $x$ such that $e = (x, y) \in G$ and $y \in R_{v,w}$.
12:        **while** $Q \neq \{\emptyset\}$ **do**
13:            $Q \rightarrow x$.
14:            $R_x$ = Reachability($x$).
15:            **if** there is not a vertex $y \neq v, w \in R_x$ with a free pebble in $G$ **then**
16:                **for** each vertex $u \in R_x$ **do**
17:                    Let $rcd_u$ be the rigid cluster containing $u$ in $RCD$.
18:                    $rcd = rcd \cup rcd_u$.
19:                    Remove $rcd_u$ from $RCD$.
20:                **end for**
```

Algorithm D.6 continued.

```
21:                 Add to $Q$ all the vertices $a$ such that $e = (a, b) \in G$ and $b \in R_x$
    that have not previously been enqueued.
22:             end if
23:         end while
24:     end if
25: end if
```

Algorithm D.7 Reachability

Input: A graph G and a vertex w.

Output: A set of vertices R that are reachable from w via directed paths from w in G.

```
1: Let $Q = \{w\}$ be a queue.
2: Let $R = \{\emptyset\}$.
3: while $Q \neq \{\emptyset\}$ do
4:     $Q \rightarrow u$.
5:     for each covered edge $e = (u, v) \in G$ do
6:         if $v \notin R$ then
7:             $Q \leftarrow v$.
8:             Add $v$ to $R$.
9:         end if
10:     end for
11: end while
return $R$.
```

changed in the following way. First, it initializes a set of rigid clusters RCD when the pebble game graph is initialized (line 4). It adds a rigid cluster for each vertex in the pebble game graph. As the algorithm progresses, rigid clusters will be merged. Then, before collecting pebbles for a constraint, it first checks if the endpoints are already in the same rigid cluster (line 6). If so, the constraint is automatically redundant. Finally, after an independent constraint is found, the rigid cluster decomposition is updated (line 14, see Algorithm D.6). The resulting pebble game output rigid cluster decomposition for the example in Fig. 6(a) is given in Fig 9. Dependent edges are rejected in constant time and updating the rigid cluster decomposition can be done in linear time, resulting in an algorithm still requiring only $O(n^2)$ time as before [91]. As with the original pebble game, the running time is quadratic with respect to the protein's length. Proofs of correctness can be found in [91].

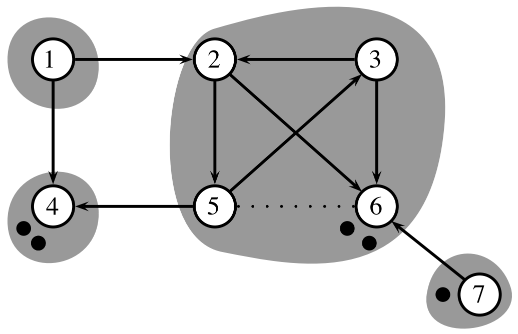


Fig. 9.: The final pebble game graph and rigid cluster decomposition after all constraints are considered for the example in Fig. 6(a).

Fig. 10 shows an example of how rigid clusters are merged by Alg. D.6. Constraint (3,6) has just been found independent and added to the graph. The current rigid cluster decomposition is shown in Fig. 10(a). First the reachability of vertices 3 and 6 are computed (line 2, see Alg. D.7). The reachability of a vertex u is the set of vertices that are reachable by directed paths from u in the graph. For example, Fig. 11 shows the reachability of vertex 5 is $\{2, 3, 6\}$. Here, the reachability of vertex 3 is $\{\emptyset\}$ and the reachability of vertex 6 is $\{3\}$, thus $R_{3,6} = \{3\}$. Since there is not a vertex in $R_{3,6}$ with a free pebble besides the constraint endpoints (line 3), a new rigid cluster *rcd* is created (lines 4–9). Here, *rcd* is initialized to $\{3, 5\}$ (since vertex 5 was already in the same rigid cluster as vertex 3). The queue is initialized to all the incoming vertices into $R_{3,6}$: vertices 2, 5, and 6 (lines 10–11). In the first iteration of the while loop on line 12, vertex 2 is removed from the queue. Its reachability (R_2) is $\{3, 6\}$. Since there is not a vertex with a free pebble besides the constraint endpoints, the rigid clusters containing these vertices are merged with *rcd* (lines 15–20). The resulting rigid cluster decomposition is shown in Fig 10(b). The vertices with edges into R_2 are 2 and 5. These have already been enqueued before so they are not added to the queue (line 21). This process repeats until the queue is empty. The final rigid cluster decomposition is given in Fig. 10(c).

3. Dependent Hinge Sets Identification

From the pebble game and rigid cluster decomposition, each constraint is labeled as rigid (i.e., its endpoints belong to the same rigid cluster) or flexible (i.e., its endpoints belong to different rigid clusters). Flexible constraints are often referred to as *hinges.* However, there are two types of flexible constraints (or hinges): *independently flexible* and *dependently flexible. Independently flexible* constraints can be moved without requiring the movement (or flexing) of any other constraints aside from rigid body mo-

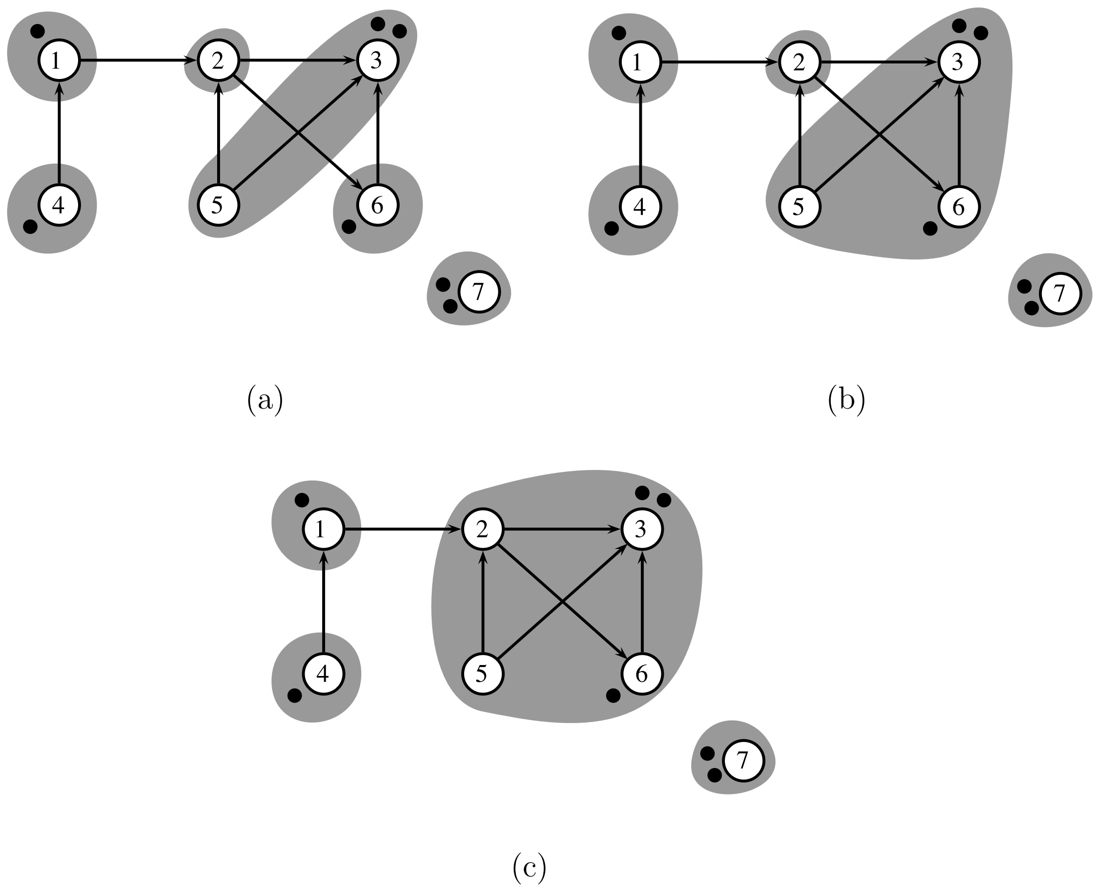


Fig. 10.: Example of how rigid clusters are merged in Alg. D.6. (a) Constraint (3,6) has just been found independent and added to the graph. Shaded regions indicate the current set of rigid clusters. (b) The rigid cluster containing vertex 6 is merged into the cluster containing vertices 3 and 5. (c) After updating the rigid cluster decomposition, vertices 2 and 6 are added to the cluster containing vertices 3 and 5.

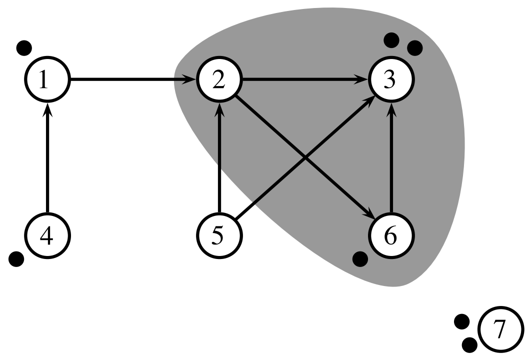


Fig. 11.: The reachability of a vertex is the set of vertices that are reachable by directed paths from that vertex in the graph. For example, the reachability of vertex 5 is indicated by the shaded region.

tions of the rigid clusters. *Dependently flexible* constraints form sets called *dependent hinge sets* such that all the vertices within a set move in a coordinated fashion.

After the pebble game is played and rigid cluster decomposition is performed on a graph, Algorithm D.8 identifies the dependent hinge sets present [85]. It first initializes a hinge set for each vertex in the graph (lines 3–5). Then it examines each flexible constraint and attempts to add an extra constraint between its endpoints (lines 8–9). If unsuccessful, the endpoints are in the same dependent hinge set, along with all the vertices traversed on the failed free pebble search (lines 14–18). Otherwise, they are not yet in the same set, and the extra constraint is add to the graph (lines 10–12) and remains until all of the flexible constraints are examined (lines 22–24).

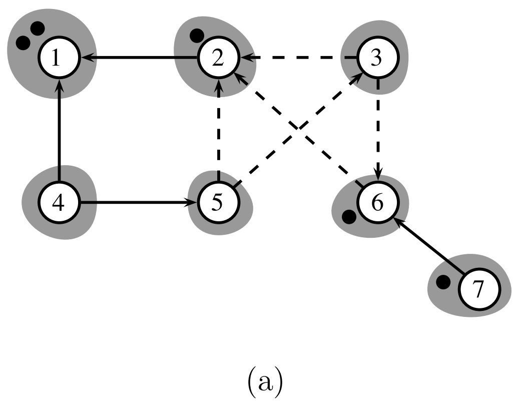
Fig. 12(a) shows the initial hinge sets for the input graph from Fig. 6(a). Here, the rigid constraints are marked with dashed lines since they are not considered by this algorithm. The first constraint examined is (1,2). Fig. 12(a) shows the pebble placement after collecting 2 pebbles on vertex 1 and attempting to collect 2 pebbles

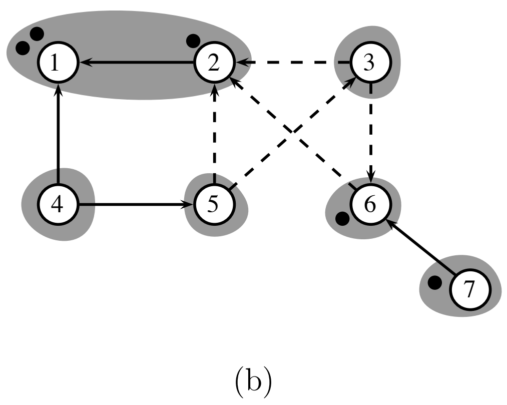
Algorithm D.8 Dependent Hinge Sets Identification

```
Input: A graph $G$ that the pebble game has been played on, the number of pebbles $k$
    per vertex, and the number of pebbles $l$ required to mark an edge as independent.
Output: A list $H$ of dependent hinge sets.
Uses: Algorithms D.2 and D.7.
 1: Let $C = \emptyset$ be a set of covered edges.
 2: Let $H = \emptyset$.
 3: for each vertex $v \in G$ do
 4:     Let $h = \{v\}$ be a new hinge set. Add $h$ to the list of hinge sets $H$.
 5: end for
 6: for each flexible edge $e = (v, w) \in G$ do
 7:     if $v$ and $w$ do not already belong to the same hinge set $h \in H$ then
 8:         Collect_Pebbles($G$, $v$, $k$).
 9:         if Collect_Pebbles($G$, $w$, $l - k$) is successful then
10:             $v$ and $w$ are not in the same hinge set.
11:             Release blocked pebbles associated with $v$ and $w$.
12:             Cover $e$ in $G$ with a free pebble from $w$. Add $e$ to $C$.
13:         else
14:             Let $h_v$ be the hinge set containing $v$ in $H$.
15:             Let $h_w$ be the hinge set containing $w$ in $H$.
16:             Let $h = h_v \cup h_w \cup$ Reachability($G$, $w$).
17:             Remove $h_v$ and $h_w$ from $H$. Add $h$ to $H$.
18:             Release blocked pebbles associated with $v$ and $w$.
19:         end if
20:     end if
21: end for
```

Algorithm D.8 continued.

```
22: for each edge $e \in C$ do
23:     Uncover $e$ in $G$.
24: end for
return $H$.
```





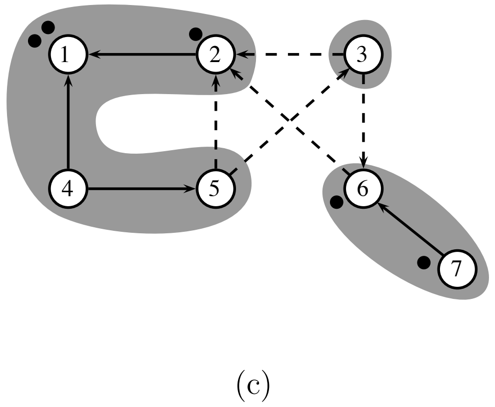


Fig. 12.: Example of how dependent hinge sets are identified in Alg. D.8. Rigid constraints are indicated with dashed lines since they are ignored by the algorithm. (a) The initial hinge set decomposition. When considering constraint (1,2), the algorithm attempts to collect 4 pebbles on the endpoints. (b) Because 4 pebbles are not collected on the endpoints, vertices 1 and 2 belong to the same hinge set and are merged. (c) The final dependent hinge set decomposition.

on vertex 2. Because the pebble search is unsuccessful, vertices 1 and 2 are part of the same dependent hinge set. The hinge sets containing them are merged in Fig. 12(b). The algorithm continues until all flexible constraints are considered. Fig 12(c) displays the final dependent hinge set decomposition.

CHAPTER III

EXTENSIONS TO THE PRM FRAMEWORK FOR MOLECULAR MOTION[*]

The initial application of the probabilistic roadmap method to protein folding demonstrated that this simple method could be used to study complex problems with many degrees of freedom [18, 21, 22, 23]. However, it is limited to studying protein folding instead of other motions such as large-scale transitions between particular conformations, e.g., when studying folding intermediates, allostery (i.e., when binding a molecule to one area of the protein affects the binding affinity of another, typically distant, area of the protein), or misfolding. In addition, it required the user to specify a sampling density *a priori* which can be problematic. In particular, if the original sampling density is insufficient, the entire computation must be performed again. In this chapter, we discuss two extensions to the original PRM methodology to combat these weaknesses. First, we propose an incremental strategy for roadmap construction that automates the process to eliminate the requirement of a user-specified sampling density [26]. Second, we present a generalized methodology for handling multiple conformations of interest, instead of just the single native state for protein folding problems [24]. We provide evidence that the transitions mapped by our approach are more realistic than those provided by the computationally less expensive Morph Server [25], especially for transitions requiring large conformational changes.

[*]Part of the data reported in this chapter is reprinted with the kind permission of Springer Science+Business Media from "Incremental map generation (IMG)" in *Algorithmic Foundation of Robotics VII* by D. Xie, M. Morales, R. Pearce, S. Thomas, J.-M. Lien, and N. M. Amato, 2008, Springer, Berlin. Copyright 2008 by Springer. [26] Part of the data reported in this chapter is reprinted with permission from "Simulating protein motions with rigidity analysis" by S. Thomas, X. Tang, L. Tapia, and N. M. Amato, 2007. *J. Comput. Biol.*, vol. 14, no. 6, pp. 839–855, Copyright 2007 by Mary Ann Liebert, Inc. [24]

A. Detecting Model Stabilization when Building Roadmaps Incrementally

Roadmap accuracy is related to the sampling density. Previously [18, 21, 22, 23], sampling density was user specified and difficult to tune. We had no systematic way of building a roadmap and determining if it was large enough to accurately capture the main features of the energy landscape. Thus, roadmaps were built much larger than required. We need a framework for automating roadmap construction such that the resulting roadmap is large enough to be accurate but small enough to be quick to construct and query. This requires a mechanism for determining when the roadmap has approximated the folding landscape (e.g., the area of the energy landscape that determines folding behavior) accurately enough.

To automate roadmap construction, we interleave sampling and connection to incrementally build a roadmap [26]. We first generate a sparse set of samples around each conformation of interest (e.g., the folded state for protein folding, the transition endpoints for protein transitions). Then we connect 'nearby' samples together and compute edge weights as before. After each round of sampling and connection, we evaluate the roadmap and determine if it is accurate enough to stop. We continue this process until the roadmap adequately represents the portion of the protein's energy landscape we are trying to model. Fig. 13 shows the Incremental Map Generation (IMG) framework.

IMG has several important features, including:

- *Automatic determination of roadmap size.* The most important feature of IMG is that it provides a mechanism to incrementally construct roadmaps and to automatically determine when construction should be halted.

- *Evaluation criteria.* A key requirement for IMG is effective evaluation criteria that can be efficiently tested during roadmap construction. We propose eval-

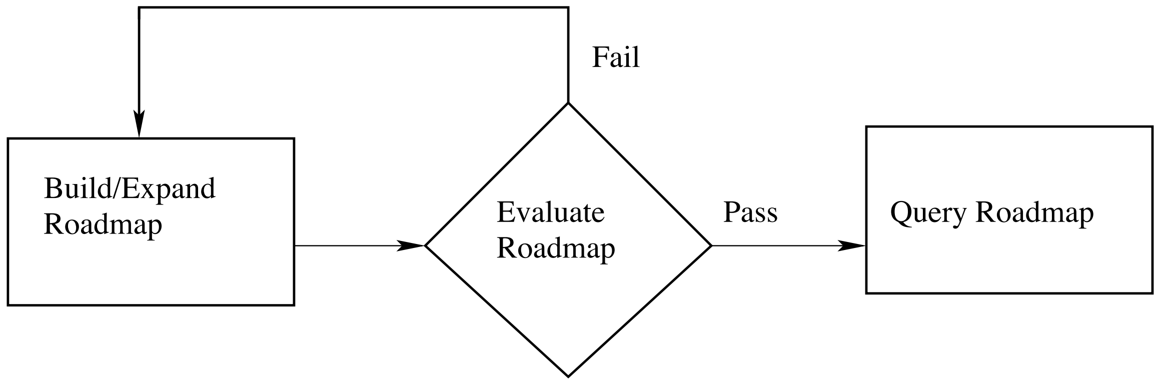


Fig. 13.: Flow diagram for Incremental Map Generation (IMG).

uation criteria for detecting model stabilization for protein folding landscapes and protein transitions.

- *Compatibility with existing sampling-based planners.* IMG is *not* a new sampling method; instead, it is a general strategy that can be used with any sampling-based planner. For example, we can use it with the iterative Gaussian sampling strategy described in Chapter II, Section C.3 or the new sampling strategy based on rigidity analysis proposed in Chapter IV, Section B.

Algorithm A.1 describes IMG. This framework is simple and general. It can be customized for a particular application domain or problem by simply varying the node generation and connection strategies used and the evaluation criteria.

1. Incremental Roadmap Construction Details

To build the roadmap incrementally, we first divide roadmap construction into "sets" of size n; the size, or target number of nodes for each set, is specified by the user. Then, for each iteration, IMG performs the following steps:

Algorithm A.1 Incremental Map Generation.

```
Input: An existing roadmap $R$, a roadmap evaluator $E$, the size of a node set $n$.
Output: A roadmap $R$ that meets the criteria indicated by $E$.
1: repeat
2:     Initialization. Set parameters for this iteration.
3:     Sampling. Generate the new node set ($n$ nodes) and add them to roadmap $R$.
4:     Connection. Perform connection.
5: until $R$ meets criteria in $E$
```

Initialization. In line 2, Algorithm A.1, in order to ensure the independence of each set, we seed the random number generator. The seed s is a polynomial function of the *base seed* of the program (e.g., the time execution starts), the type of node generation method used, and the number of sets completed by that node generation method so far. Calculating the seed in a deterministic way based on a (possibly random) base seed supports reproducibility given the same base seed.

Sampling. In line 3, Algorithm A.1, the sampling strategy selected for that iteration is applied. Recall that IMG is *not* a new sampling method, but rather is a general strategy that can be applied to any sampling-based planner.

Connection. In line 4, Algorithm A.1, the connection strategy chosen by the user is applied to connect the new set of nodes to the existing roadmap.

2. Roadmap Evaluation

The other key component enabling automatic determination of roadmap size is the stopping or evaluation criteria. The IMG framework can accept a broad range of stopping or evaluation criteria customized for particular applications or user preferences.

In [26], several stopping criteria are presented for general robotics motion planning. Here, we give two examples of application-specific evaluation methods.

Stable Secondary Structure Formation Order. For protein folding, we want to build a roadmap until the secondary structure formation order along its pathways stabilizes. We can define 'formation' of a piece of secondary structure in several ways. In these results, we consider a piece of secondary structure 'formed' when at least 80% of its contacts (i.e., residue pairs within 7Å of each other in the native state) are present. Later we will define 'formation' based on rigidity analysis (see Chapter IV, Section A). The pathway's secondary structure formation order is then the order at which pieces are 'formed.'

To determine model stabilization for folding, we first examine every pathway in the roadmap from an unstructured conformation to the folded state and group them by their secondary structure formation ordering. As in [18, 21, 22, 23], we consider a conformation as unstructured when no piece of secondary structure has greater than 30% of its contacts present. We consider the roadmap stable when the percentage of each group does not vary from the previous roadmap by more than some threshold t.

Max-flow Evaluation. Some applications require many paths between two conformations. For example, to study how a protein transitions between two specific conformations of interest, we can examine the probable paths between them in the roadmap. To determine if the roadmap is sufficient for modeling such protein transition pathways, we use a different type of evaluation criteria than the one for protein folding. Here we want to examine the probable paths between the transition endpoints in the roadmap.

Characterizing the pathways between the transition endpoints in the roadmap is similar to looking at the flow between a source and a sink in a flow network [94]. A flow network is a directed graph where edges are assigned a capacity, i.e., the maximum rate that material can flow along that edge. Flow networks can model a wide variety of phenomena such as water moving through a set of pipes, current traveling along an electrical network, or packages shipped by the postal service. The source produces material (e.g., water, current, or packages) and the sink consumes the material. The maximum network flow is the greatest rate that material can be moved from the source to the sink given the underlying directed graph.

Thus, we can interpret the problem of characterizing the probable paths between transition endpoints in the roadmap as a maximum flow problem where the roadmap becomes the flow network graph, one transition becomes the source, and the other transition becomes the sink. If a roadmap edge weight, $w(e)$, reflects the likelihood that the protein will move from one configuration to the next, then we can define the edge capacity in our flow network $c(e)$ as $1/w(e)$. Thus, the maximum network flow between two conformations approximates the transition rate between them [94]. The evaluator returns success if the max-flow between the two configurations is above some user specified threshold f.

3. Results

Here we demonstrate how to use the stable secondary structure formation order evaluator for protein folding to automatically determine roadmap size. Later, in Section B.2 we use the max-flow evaluator to automatically build roadmaps for protein transition problems.

We built several roadmaps incrementally for protein G. Protein G is a 56 residue protein consisting of a central α-helix and a 4-stranded β-sheet: β-strands 1 and 2 form the N-terminal hairpin (β1-2) and β-strands 3 and 4 form the C-terminal hairpin (β3-4). It is known that the C-terminal hairpin forms before the N-terminal hairpin from hydrogen exchange experiments [95] and Φ-value analysis [96]. We used several different increment sizes: 100, 500, and 1000. During map generation, we tracked when different stability thresholds, t, would stop the construction process. We varied t between 0.01 and 0.20.

Table II summarizes the results. As the increment size is increased for a given value of t, the roadmap size also increases. Also, as expected, the roadmap size required for stable pathway distributions increases as the stability threshold t decreases and becomes more strict.

Table II.: Sizes of roadmaps for Protein G when built incrementally with varying increment size and stability thresholds, t.

	Increment Size		
t	**100**	**500**	**1000**
0.01	900	4500	3000
0.05	600	1000	3000
0.10	300	1000	2000
0.20	300	1000	2000

It may appear from these results that it is desirable to pick the smallest increment size possible as they yield smaller roadmaps. However, this could lead to premature stopping of the generation process. Consider the pathway distribution changes in Figure 14. To simplify the display, we grouped all pathways into two types: those which follow experimentally determined order (i.e., β3-4 forms before β1-2) and those that do not. This artificial clustering of pathways was not used to determine stable secondary structure formation order distributions. Initially, the pathways do not match the experimentally determined order. It is not until 400 nodes that these distributions switch to the correct ordering. Even then, it takes two to three times as many more nodes for these distributions to stabilize. Building the roadmap with an increment size of 100 would stop roadmap construction before stabilization is reached for all vales of t.

Figure 14 also demonstrates the danger of setting the threshold t too low. This could cause a much larger roadmap to be built than necessary. For example, with an increment size of 500 and $t = 0.01$, roadmap construction continues long after stabilization.

In practice, we found that $t = 0.10$ strikes a good balance between the two extremes. It allows the construction process to continue long enough to stabilize, but not too long to waste resources. We benchmarked this threshold on a set of proteins for which the secondary structure formation order is experimentally known: protein G, protein L, and two mutants of protein G. We found that for all four proteins, $t = 0.10$ produced secondary structure formation order distributions that matched with experiment, while smaller values tended to produce distributions contrary to experimental findings, particularly for protein G and its mutants.

B. Building Landscape Models for other Motions

We would like to study motion problems in addition to protein folding such as transitions between known folding intermediates, transitions between bound and unbound conformations to a ligand, misfolded proteins, and allostery interactions. For exam-

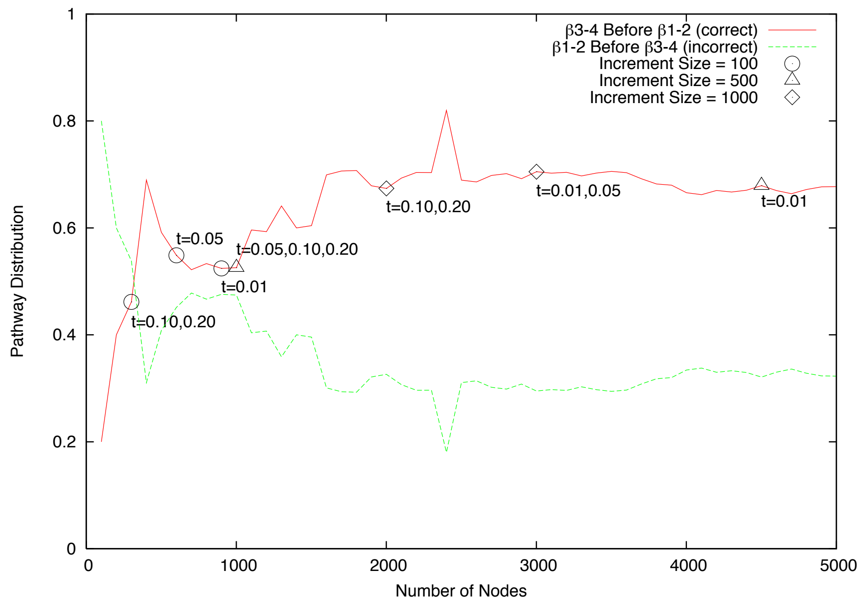


Fig. 14.: Pathway distribution profile of protein G as a function of roadmap size. Stopping points for incremental map generation are shown for various increment sizes and stability thresholds, t. Note that pathways are grouped into two types for the plot: those which follow experimentally determined order (i.e., β3-4 forms before β1-2) and those that do not. However, stable secondary structure formation order evaluation is not determined on theses two groupings but on the full orderings found in the roadmap pathways.

ple, calmodulin undergoes two large-scale conformational changes when binding to Ca^{2+} thereby regulating many cellular processes [97, 98, 99]. Changing conformations allows it to bind to over 100 different proteins [98, 100]. Several devastating diseases such as scrapie in sheep and goats, bovine spongiform encephalopathy (Mad Cow disease), and Creutzfeldt-Jakob disease in humans are caused by misfolded proteins called prions [101, 102]. The prion protein has a significantly different structure in the diseased state than in the normal state [101, 103, 104]. Insight into how these proteins misfold could help develop better treatments.

We extend our PRM framework to study specific large-scale conformational changes by iteratively sampling around each target conformation and connecting samples together as described earlier in Chapter II, Section C.3. Thus our roadmaps contain the target conformations, as well as transitions between them, and approximate the energy landscape encompassing the transition under study.

1. Roadmap Construction with Multiple Conformation Biases

To map specific large-scale transitions, we interleave sampling and connection to incrementally build a roadmap as in Section A. The only difference here is we sample around each target conformation during each round of roadmap construction. Then we connect samples together and compute edge weights as before. We continue until the roadmap adequately represents the protein's energy landscape near the target conformations and between them. We use the maximum flow evaluation described earlier in Section A.2. From this roadmap, we can extract multiple low energy transition pathways between target conformations and characterize the energy barriers between them. Algorithm B.1 describes the process.

One challenge in using multiple conformations of the same protein is that the bond lengths and bond angles typically assumed to be fixed may vary between the conformations. There are several contributors to this. First, structures determined from experimental data, such as those deposited in the Protein Data Bank (PDB) [3], may contain noise or experimental error. This is especially true of low resolution structures. Second, bond length and bond angle stretching may legitimately occur during a transition.

In the results presented in this work, we maintain our fixed bond length and

Algorithm B.1 Roadmap Construction with Multiple Conformation Biases.

Input: A set of target conformations T, the size of a node set n, and a flow threshold f.

Output: A roadmap R that maps the transitions between conformations in T.

```
1: repeat
2:     Set parameters for this iteration.
3:     for each conformation $t \in T$ do
4:         Iteratively apply small Gaussian perturbations to existing samples biased by $t$ and add them to $R$.
5:     end for
6:     Connect newly generated samples to each other and to existing samples in $R$. Update $R$ accordingly.
7: until The maximum flow between all pairs $(t_i, t_j) \in T$ is at least $f$.
return $R$.
```

bond angle assumption. To reconcile the varying lengths and angles, we average them from the input structures and use these new values for our protein model. However, when applying these averaged values back to the original input structures, some atom collisions may occur. We resolve these by applying an energy minimization to the structure. In the results presented here, we use the energy minimization function from the EEF1 all-atoms potential function [105]. We allow the energy minimizer to alter side chain placement and backbone torsional angles, but it must keep the bond lengths and bond angles fixed. In the future, we intend to interpolate these values from the input structures for each residue based on the relationship between its ϕ and ψ angles and the ϕ and ψ angles of the input structures.

2. Results

Calmodulin is a 148-residue signaling protein that binds to Ca^{2+} to regulate several processes in the cell [97, 98, 99]. It is composed of 4 EF-hands joined by a flexible central α helix [106, 107]. Each calmodulin domain binds 1 calcium ion in each EF-hand pair [108], with the two domains acting independently [109]. When binding to Ca^{2+}, it undergoes two large-scale conformational changes: (1) the central α helix linking the C-terminal and N-terminal domains unravels to bring the protein from a dumbbell conformation to a more globular conformation [110] (Fig. 15(a–b)) and (2) the α helices in each domain reorganize [111] (Fig. 15(c–d)).

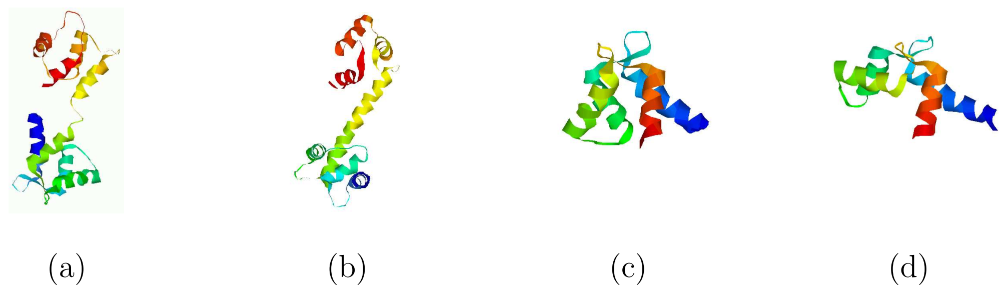
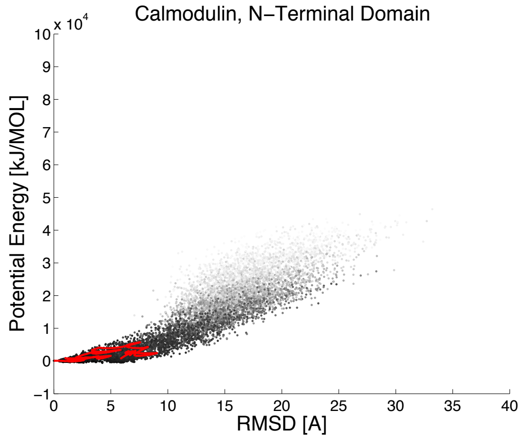
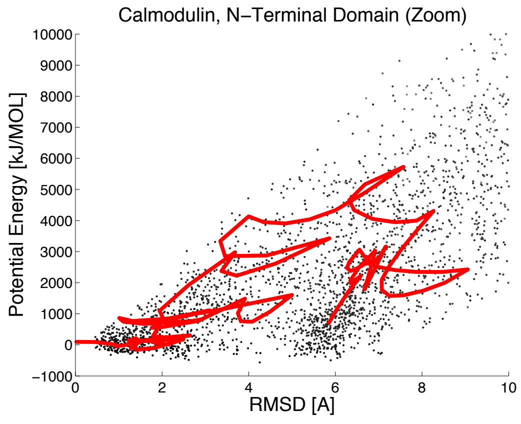


Fig. 15.: Conformational changes of calmodulin: (a) calcium-free state (1CFD) to (b) bound state (1CLL) and of the N-terminal domain: (c) calcium-free to (d) bound.

We built a roadmap biased towards both target states. Fig. 16 indicates the distribution of samples in the roadmap, colored by rigidity/flexibility from rigidity analysis. The most energetically feasible transition in the roadmap between the bound and unbound states is also plotted in red. Calmodulin traverses one large energy barrier (Fig. 16(b)) to transit from the unbound state to the bound state without traveling far in terms of RMSD. Notice that the transition pathway is fairly ‘rough’ or ‘jagged’. This demonstrates an opportunity for path refinement/smoothing by adding additional samples near the extracted transition pathway. However, some ‘roughness’ may be a function of the reaction coordinate (RMSD) used to view the pathway.



(a)



(b)

Fig. 16.: The potential vs. RMSD distribution of samples for calmodulin, N-terminal domain. Rigidity is indicated by color from most rigid (black) to most flexible (light gray). The most energetically feasible transition in the roadmap between the bound and unbound states is displayed as a solid red line.

Figs. 17 and 18 compare pathway profiles of the most energetically feasible transition between the two states in our roadmap and ‘morphs’ of various resolution obtained from the Database of Macromolecular Movements (Morph server) [63, 25]. Recall, that the 'morphs' are generated using an interpolation method called adiabatic mapping [67] that was chosen as a computationally efficient method which produces chemically realistic 'morphs.' We examined pathway profiles for energy, contacts present (i.e., when two residues are within 7Å of each other in the target state), degrees of freedom computed by rigidity analysis, and RMSD distance to the target states. Note that since the Morph server alters the original target conformations, their profile endpoints do not always align with our pathways. One striking observation is the regularity of the concavities for the 'morphs' corresponding to the various resolution levels across all the profiles except for the RMSD profiles in which the RMSD to the target states seems to change monotonically with the path step. These regularities in the 'morphs' would not be expected in actual transition pathways, e.g., one would not expect a monotonic increase in RMSD from 1CFD to 1CLL. In contrast, our roadmap pathways profiles are more plausible — they exhibit trends, but also have reasonable fluctuations. Indeed, this type of behavior has also been observed by other researchers, e.g., in [112], Monte Carlo simulations indicate a wide range of transition pathways and event durations.

Figs. 17(a) and 18(e) show the contacts present and degrees of freedom computed by rigidity analysis along the pathway. Note that the protein does not completely unfold, but maintains a large number of contacts and loses few degrees of freedom. Generally, the actual degrees of freedom is inversely proportional to the number of contacts present. It is interesting to note, however, that we see a slight break in this relationship on the second half of the pathway where the peaks in degrees of freedom do not match up with the peaks in number of contacts. Regions of the protein become stressed when the number of contacts increases without a corresponding decrease in degrees of freedom.

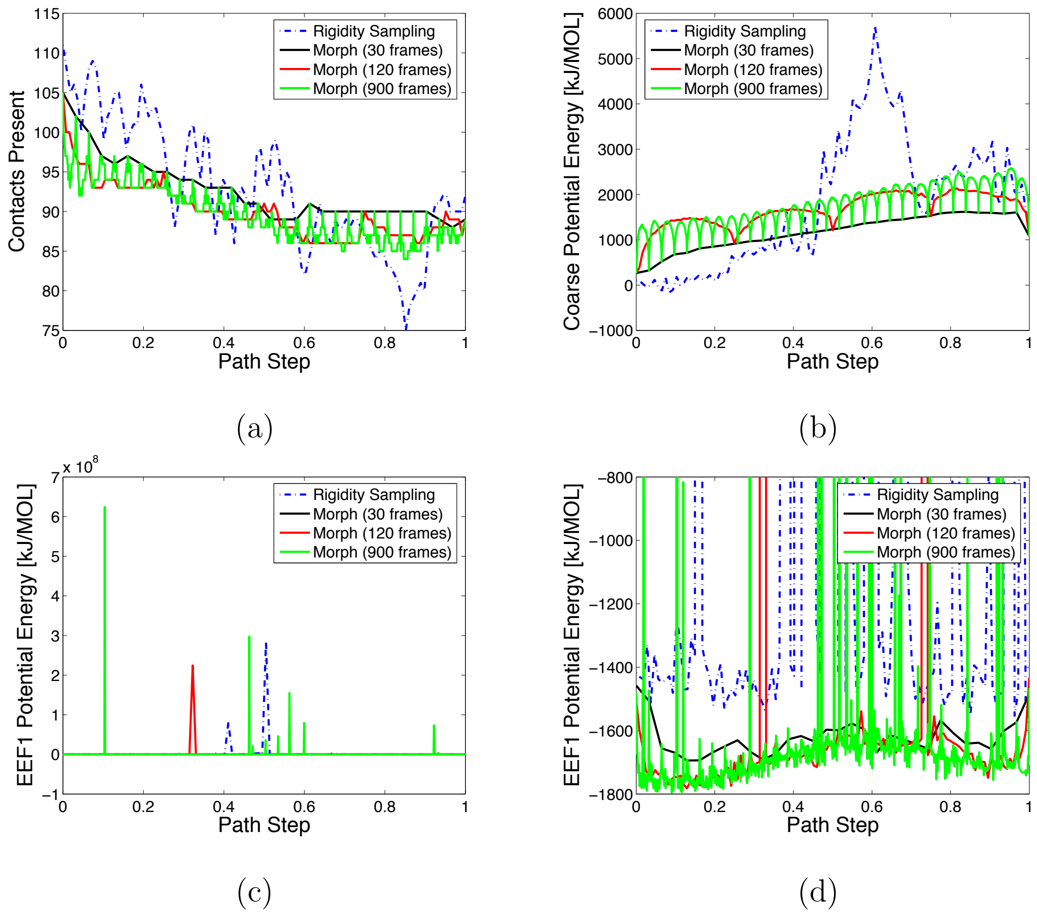
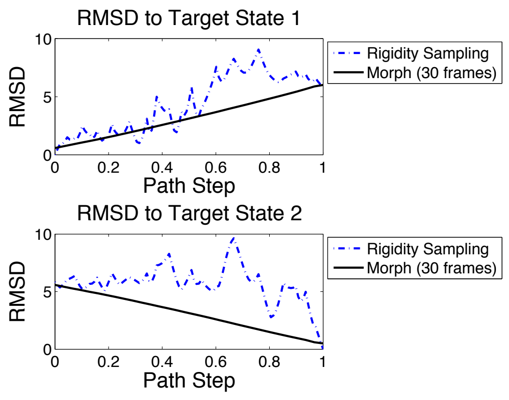
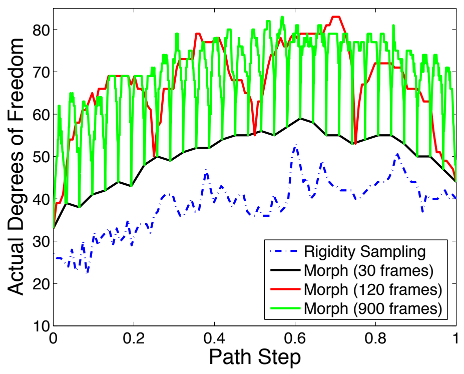


Fig. 17.: Pathway profiles for the calmodulin N-terminal domain comparing our method to 'morphs' of various resolution: (a) contacts present, (b) coarse potential energy, (c–d) all-atoms potential energy.



(e)



(f)

Fig. 18.: Pathway profiles for the calmodulin N-terminal domain comparing our method to 'morphs' of various resolution, continued: (e) degrees of freedom computed by rigidity analysis, and (f) RMSD to both target states. For RMSD, only the 30 frame 'morph' shown because all resolutions are nearly identical.

We investigated several other protein transitions in a similar way, see Table III. We measure the percentage of degrees of freedom gained as the difference between the maximum degrees of freedom along the pathway and the minimum degrees of freedom of the starting/ending conformations, as a percentage of the total degrees of freedom possible (2*length). Most transitions do not involve a complete unfolding of the protein. In fact, several have percentage of degrees of freedom gain less than 10%. We also captured different types of transitions including smooth transitions without any significant energy barriers (i.e., 1PRV, 1BMR, and 1FOX) and those with multiple energy barriers (i.e., 2VGH and 1CMF). Finally, we measured the RMSD difference between the two target conformations and the maximum RMSD along the transition pathway to each target. Some transitions (e.g., 1PRV) remain near both targets and some transitions (e.g., 1PRF) travel significantly far from both targets. Interestingly, this measure is not an indicator of the amount of unfolding involved. For example, the transition pathway for 1FOX contains a large RMSD to the targets but has very little unfolding (i.e., only gains 3.9% degrees of freedom).

We also compared 'morphs' of various resolutions to our transition pathways when possible. (The Morph server was not able to produce some higher resolution 'morphs' for transitions 1BMR–1FH3 and 1PRV–1PRU.) Across all transitions, we observed the same concavity pattern phenomenon for the 'morph' transitions as seen in calmodulin (Figs. 17 and 18) for energy, contacts, and degrees of freedom. Here also, the RMSD to the target states essentially changed monotonically with the path step. Again, our pathways did not exhibit these unrealistic regularities.

Table III.: Pathway results for transitions studied. Most do not involve large unfolding of the protein.

Transition IDs		**Length**	**Structure**	**% Degrees of Freedom Gained**	**# Barriers**	**RMSD between Targets**	**Max. RMSD to Target 1**	**Max. RMSD to Target 2**
2VGH	1VGH	55	$1\alpha + 4\beta$	21.8	2	3.97	6.41	8.36
1PRV	1PRU	56	3α	5.4	0	2.93	2.93	3.19
1BMR	1FH3	67	$1\alpha + 3\beta$	32.8	0	6.87	14.57	15.72
1CFD	1CLL	72	$4\alpha + 2\beta$	18.1	1	5.83	9.06	9.64
1CMF	1CMG	73	$5\alpha \rightarrow 4\alpha$	24.7	2	7.68	13.91	15.54
1FOX	2FOW	76	$3\alpha + 2\beta$	3.9	0	7.32	17.22	15.48
1PFH	1HDN	85	$3\alpha + 4\beta$	43.5	1	2.51	25.84	26.06

CHAPTER IV

RIGIDITY FOR LANDSCAPE MODELING*

The original probabilistic roadmap method for protein folding [18, 21, 22, 23] can construct approximate landscape models for a variety of proteins. These models were validated by comparing the secondary structure formation order of the folding pathways in the roadmap to experimental data [18]. However, this original method is limited in the size of problems it can efficiently model. For example, modeling landscapes for proteins with 60–100 amino acids requires many samples (e.g., 10,000) and takes several hours on a desktop machine with a coarse energy function and up to two weeks with an all-atoms energy function [23].

Here we present several techniques based on rigidity analysis to extend the size of problems we can study. We first describe how to express a protein conformation as a graph, which we call the *rigidity model*, where nodes represent points on the protein and edges represent constraints between them. We can analyze this rigidity model using standard techniques to identify which portions of the conformation are the least constrained (e.g., flexible) and which portions are over-constrained (e.g., rigid). We then discuss how to use this labeling to improve both sampling and connection in the original roadmap method. We show later in Chapter V, Section 1 that these improvements significantly reduce the roadmap size required to effectively model the energy landscape. This in turn will allow us to study larger problems than previously.

*Part of the data reported in this chapter is reprinted with permission from "Simulating protein motions with rigidity analysis" by S. Thomas, X. Tang, L. Tapia, and N. M. Amato, 2007. *J. Comput. Biol.*, vol. 14, no. 6, pp. 839–855, Copyright 2007 by Mary Ann Liebert, Inc. [24]

A. Protein Rigidity Model

In order to perform rigidity analysis on a protein conformation, we must first express the conformation as a graph where nodes correspond to points in space and edges correspond to distance constraints between them. We call this graph representation of the protein conformation the *rigidity model*. To determine which portions of the conformation are rigid and flexible, we can simply perform rigidity analysis directly on the corresponding rigidity model. In this work, we use the "pebble game" [81, 82, 83, 84], a constraint counting algorithm, to perform the rigidity analysis. We follow the implementation of the pebble game outlined in [91].

Recall from Chapter II, Section D that there are two types of rigidity models for which the pebble game applies: the bar-joint model and the body-bar model, see Fig. 8. These models are conjectured to be equivalent [84]. While the bar-joint model is closer to the all-atoms representation of the protein (e.g., nodes in the rigidity model directly correspond to atoms in the protein and edges are present for each bond length and bond angle, see Fig. 8(a)), we instead employ the body-bar model for the analysis. The body-bar model allows us to represent groups of atoms as single bodies (i.e., nodes in the graph) instead of representing each atom explicitly. Thus, we can adjust the granularity of our rigidity analysis by changing the number of atoms that each body represents.

Fig. 19 shows an example protein conformation containing 6 amino acids: F-A-N-G-S-T. This protein fragment is one of the β-turns in the native state of a protein L, B1 domain mutant (PDB: 1KH0) [113]. Hydrogen bonds are indicated with thick black lines and hydrophobic interactions with thick dashed lines. Fig. 20(b–d) presents three possible body-bar rigidity models for this protein fragment with varying granularity. In each body-bar rigidity model, atom groups for each body are circled. Fig. 20(a) shows the all-atoms bar-joint rigidity model for reference, conjectured to be equivalent to the all-atoms body-bar rigidity model in Fig. 20(b).

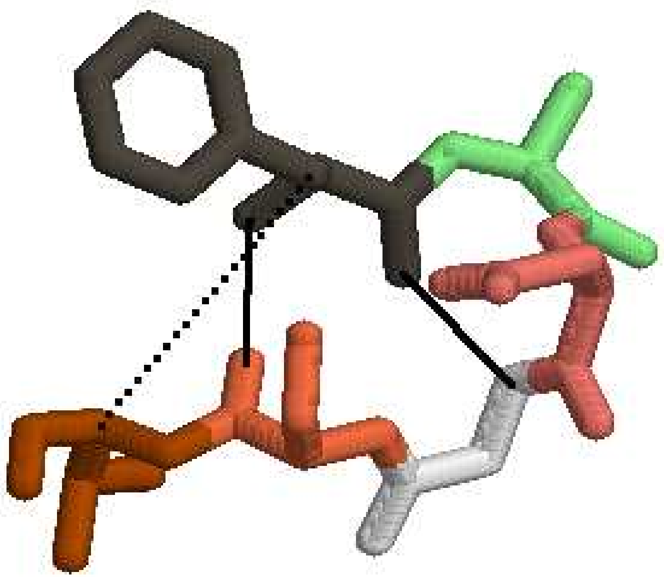
This scheme allows us to match the granularity of our rigidity model to the granularity of our protein model (from Chapter II, Section C). Below we explore two different granularity levels: the single-body, Cα model and the two-body, ϕ-ψ model.

1. Single-body, Cα Model

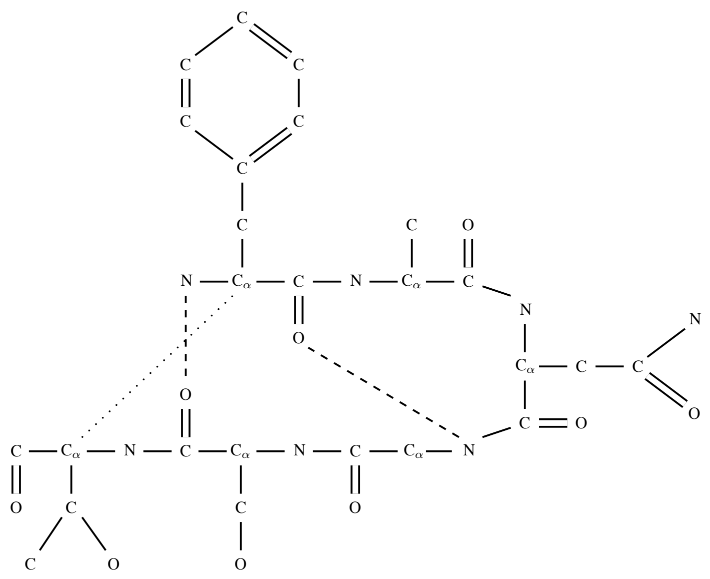
The coarsest level of granularity occurs when we consider each residue in the protein as a rigid body. In this rigidity model, we represent each residue as a rigid body centered at the Cα atom. Thus, the rigidity model simply becomes a sequence of Cα rigid bodies with constraints representing peptide/backbone bonds, disulphide bonds, hydrogen bonds, and hydrophobic interactions. Fig. 21(a) shows the resulting body-bar rigidity model for the protein fragment in Fig. 19. We model each peptide bond with 4 bars. For the open chain conformation, this yields $6+2N$ total degrees of freedom, where N is the length of the protein, corresponding to the 6 trivial rotational and translational degrees of freedom of the entire protein and the ϕ and ψ torsional degrees of freedom for each residue.

To mimic the relative strengths between peptide bonds, disulphide bonds, hydrogen bonds, and hydrophobic interactions, we model them with different numbers of bars. The weakest constraint, hydrophobic interactions, are modeled with 1 bar. Hydrogen bonds, in between peptide bonds and hydrophobic interactions in terms of strength, are modeled with 2 bars. Disulphide bonds are modeled similarly to peptide bonds with 4 bars.

Fig. 21(b) shows the rigidity analysis results of the rigidity model in Fig. 21(a). Rigidity analysis partitions the constraints into three categories: rigid (in red), independently flexible (in blue), and dependently flexible sets (in purple). In this example,



Wireframe View



Chemical View

Fig. 19.: Example turn protein fragment (FANGST) where hydrogen bonds are indicated with dashed lines and hydrophobic interactions with dotted lines. In the wireframe view, atoms are colored by amino acid type.

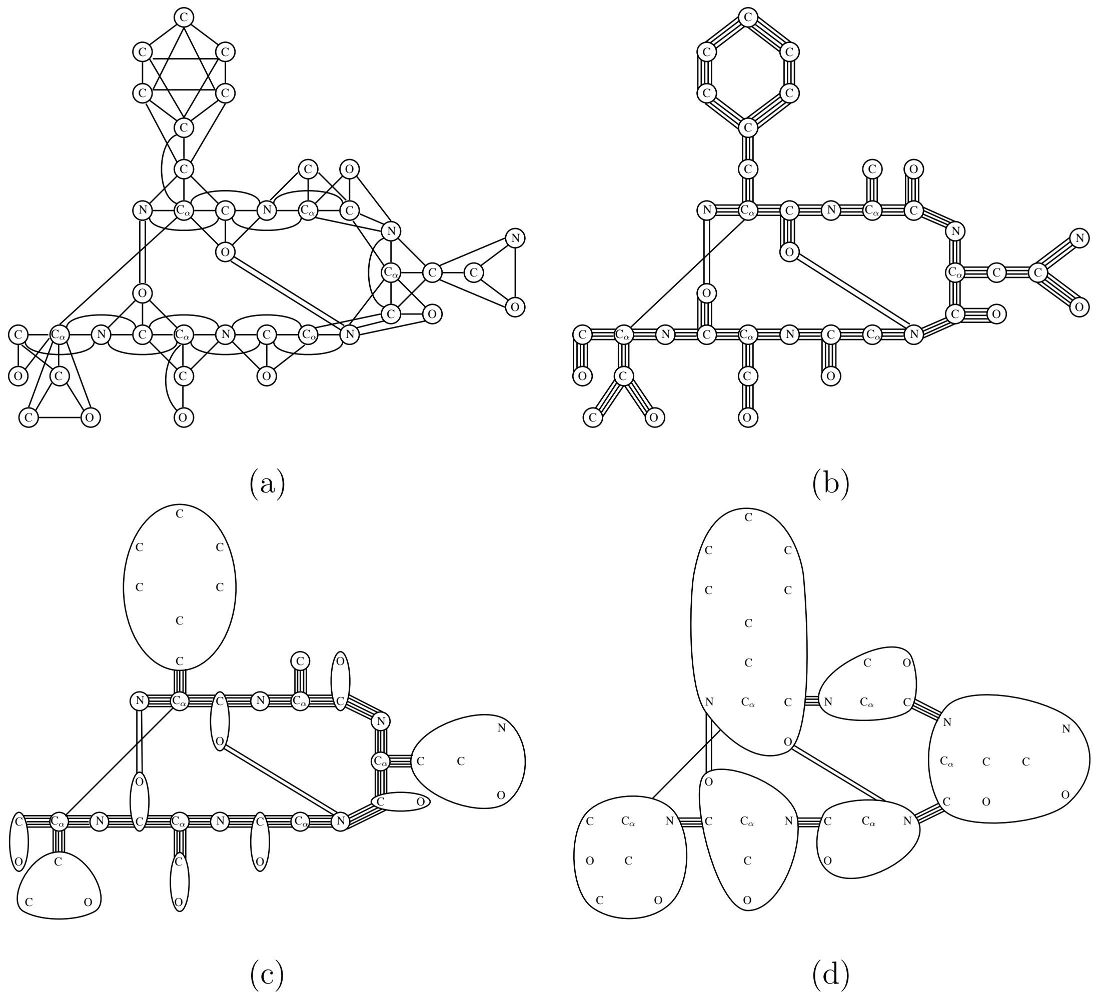


Fig. 20.: Rigidity model examples of varying granularity for the protein fragment in Fig 19. (a) The all-atoms bar-joint model. (b) The all-atoms body-bar model. (c) A coarser body-bar model where side chain atoms are grouped together. (d) The coarsest body-bar model where each residue is grouped together. In each body-bar rigidity model, bodies may contain 1 or more atoms and are indicated with circles. Relative strength is reflected in the number of bars placed to represent the bond/interaction. Note that in the all-atoms body-bar model, peptide double bonds, as typically occurs between C and O atoms, have one additional constraint than the other peptide bonds resulting in 6 bars instead of 5.

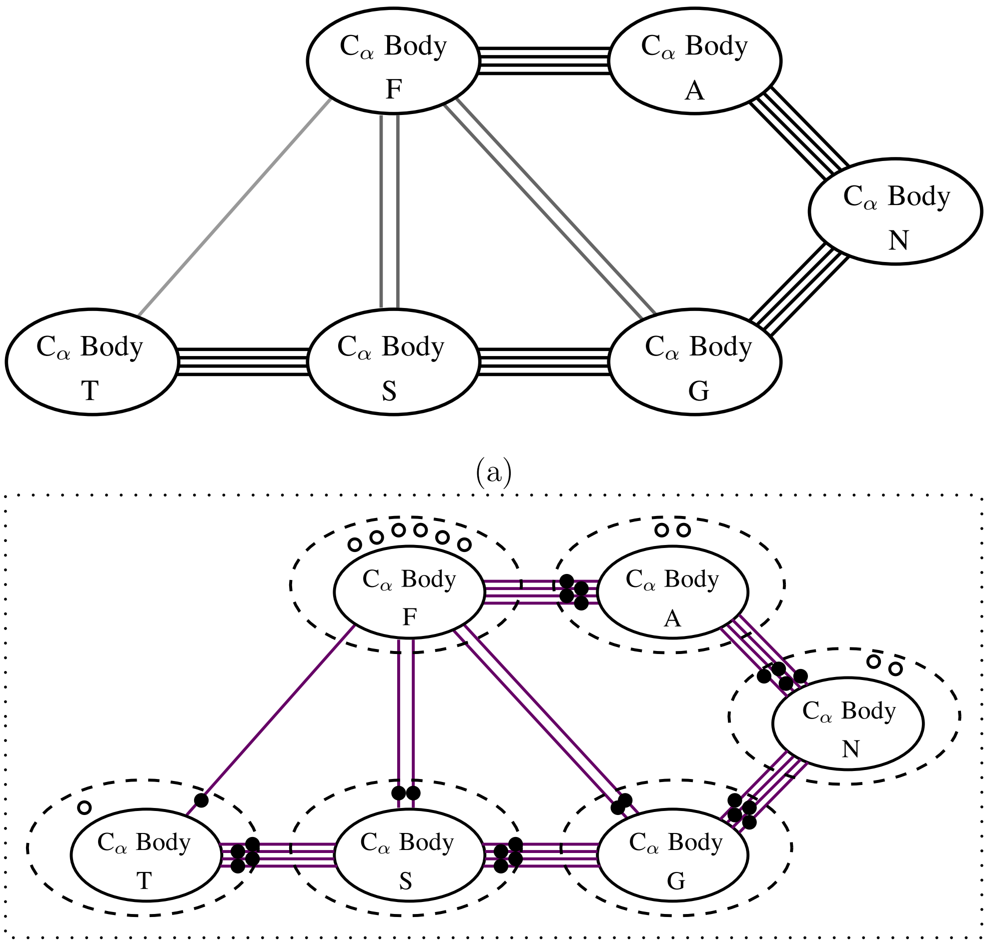


Fig. 21.: Single-body, Cα model example of the turn segment F-A-N-G-S-T in Fig. 19. (a) The resulting single-body, Cα model where hydrogen bonds are indicated with double dark grey lines and hydrophobic interactions with single light grey lines. (b) Pebble game output and rigidity analysis results. Each body has 6 pebbles which are either free (unfilled) or constrained to a bar (filled). The turn segment resulted in a single dependently flexible set (purple constraints, dotted box). Rigid clusters are indicated with dashed circles.

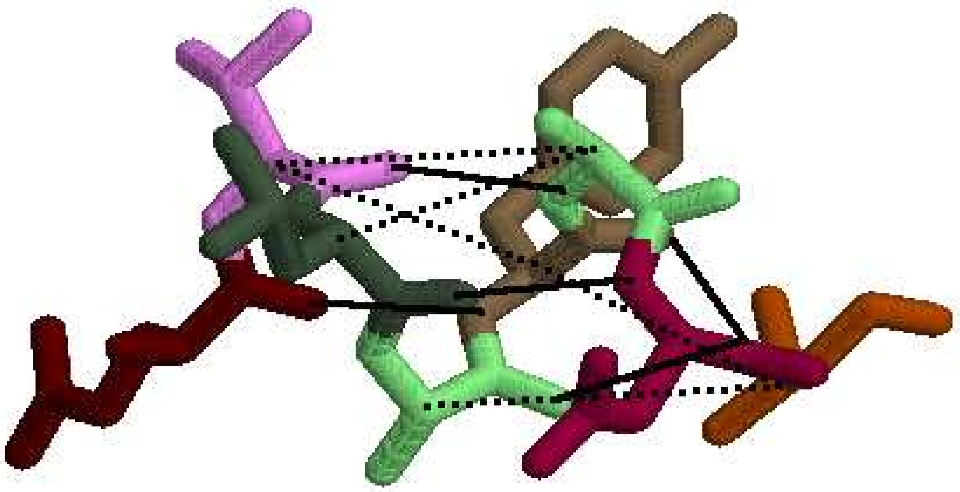
there is a single dependently flexible set indicated by the dotted box.

Fig. 22 shows a helical example resulting in a single rigid cluster. This protein fragment is part of the central α-helix in the native state of a protein L, B1 domain mutant (PDB: 1KH0) [113]. It contains 8 amino acids: E-V-L-A-Y-A-D-T. Hydrogen bonds are indicated with thick black lines and hydrophobic interactions with thick dashed lines. Fig. 23(a) shows the corresponding single-body, Cα rigidity model, and Fig. 23(b) shows the resulting pebble game output and rigidity analysis. Again, rigidity analysis partitions the constraints into three categories: rigid (in red), independently flexible (in blue), and dependently flexible sets (in purple). In this example, there is a single rigid cluster indicated by the dashed box. One of the constraints was found to be redundant during the analysis (red dashed line).

2. Two-body, ϕ-ψ Model

We can also model each residue with 2 rigid bodies to distinguish between the 2 torsional degrees of freedom in each residue. Fig. 24(a) shows the resulting body-bar rigidity model for the protein fragment in Fig. 19. Intuitively, the first body for each residue represents the first half of the residue backbone, and the second body represents the second half of the residue including the side chain. We model each peptide bond with 5 bars. For the open chain conformation, this yields $6 + 2N$ total degrees of freedom, where N is the length of the protein, corresponding to the 6 trivial rotational and translational degrees of freedom of the entire protein and the ϕ and ψ torsional degrees of freedom for each residue.

As with the single-body, Cα model, we mimic the relative strengths between the different types of bonds by varying the numbers of bars. Here disulphide bonds have 5 bars, hydrogen bonds have 2 bars, and hydrophobic interactions have 1 bar. For disulphide bonds, the bars are placed between the second bodies in the residue because


Wireframe View

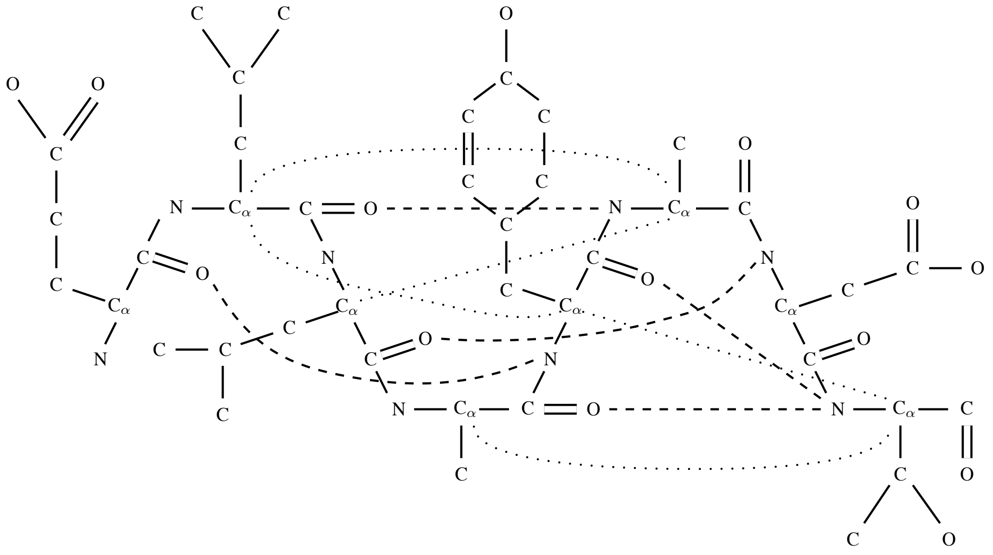
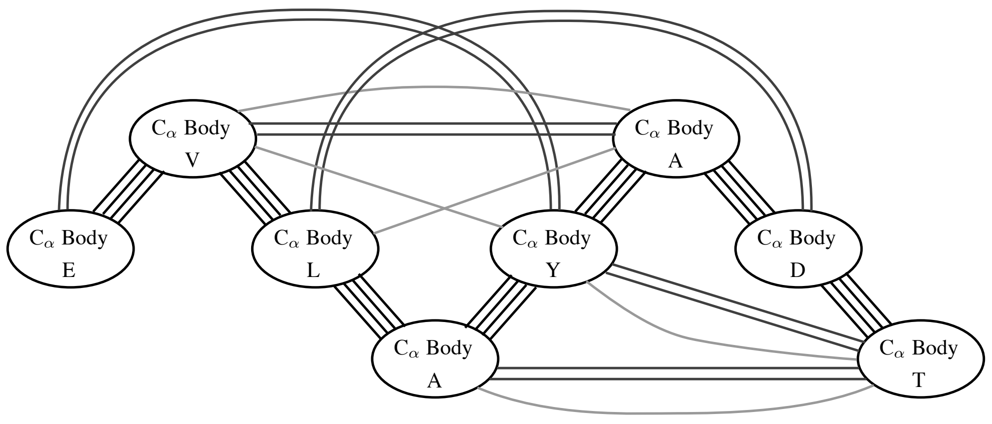
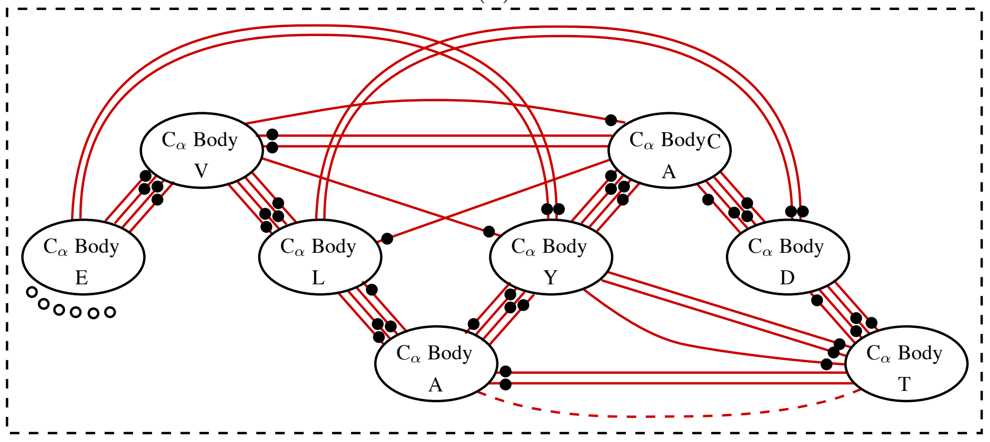

Chemical View

Fig. 22.: Example helical protein fragement (E-V-L-A-Y-A-D-T) where hydrogen bonds are indicated with dashed lines and hydrophobic interactions with dotted lines. In the wireframe view, atoms are colored by amino acid type.

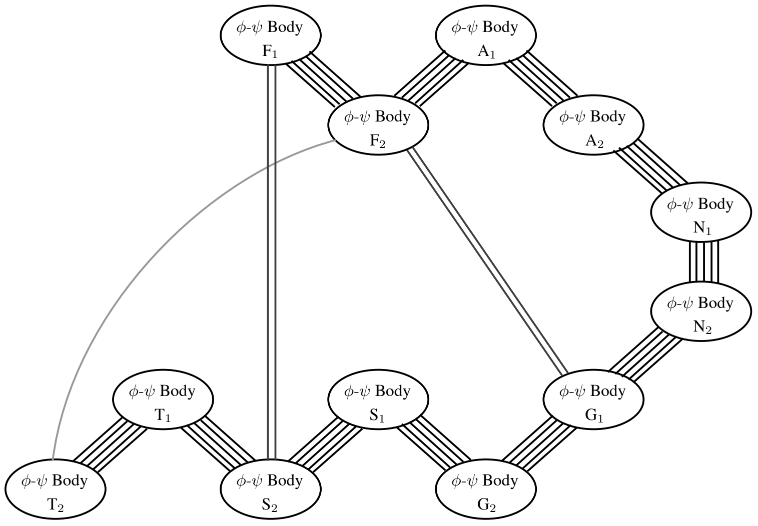


(a)

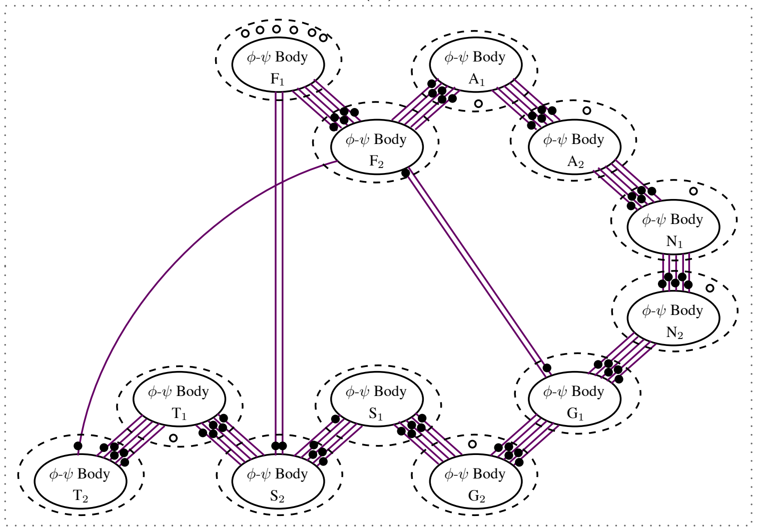


(b)

Fig. 23.: Single-body, Cα model example of the helical segment in Fig. 22. (a) The resulting single-body, Cα model where hydrogen bonds are indicated with double dark grey lines and hydrophobic interactions with single light grey lines. (b) Pebble game output and rigidity analysis results. Each body has 6 pebbles which are either free (unfilled) or constrained to a bar (filled). The helical segment resulted in a single rigid cluster (red constraints, dashed box). One constraint was found to be redundant (red dashed line).



(a)



(b)

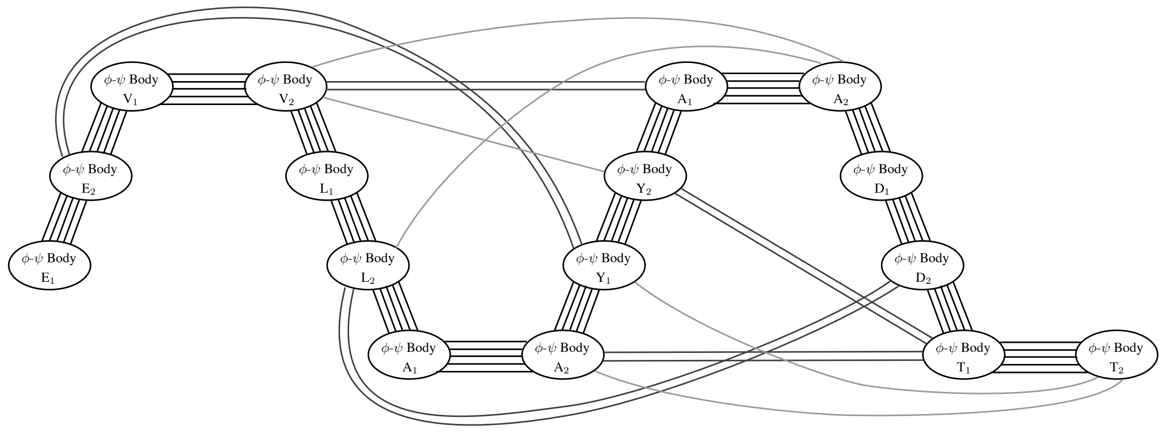
Fig. 24.: Two-body, ϕ-ψ model example of the turn fragment (FANGST) in Fig. 19. (a) The resulting rigidity model where hydrogen bonds are indicated with double dark grey lines and hydrophobic interactions with single light grey lines. (b) Pebble game output and rigidity analysis results. Each body has 6 pebbles which are either free (unfilled) or constrained to a bar (filled). The turn segment resulted in a single dependently flexible set (purple constraints, dotted box).

these interactions occur between the side chains. For hydrogen bonds, the donor residue (i.e., where the bond occurs at the N atom) is constrained at the first body and the acceptor residue (i.e., where the bond occurs at the O atom) is constrained at the second body. This is illustrated in Fig. 24(a) where the donor/acceptor residue for the first hydrogen bond is phenylalanine (F)/serine(S) and the donor/acceptor residue for the second hydrogen bond is glycine(G)/phenylalanine (F). Finally, recall that we measure hydrophobic interactions by the distance between the Cα atoms. Thus, we place these bars between the second bodies in the residue.

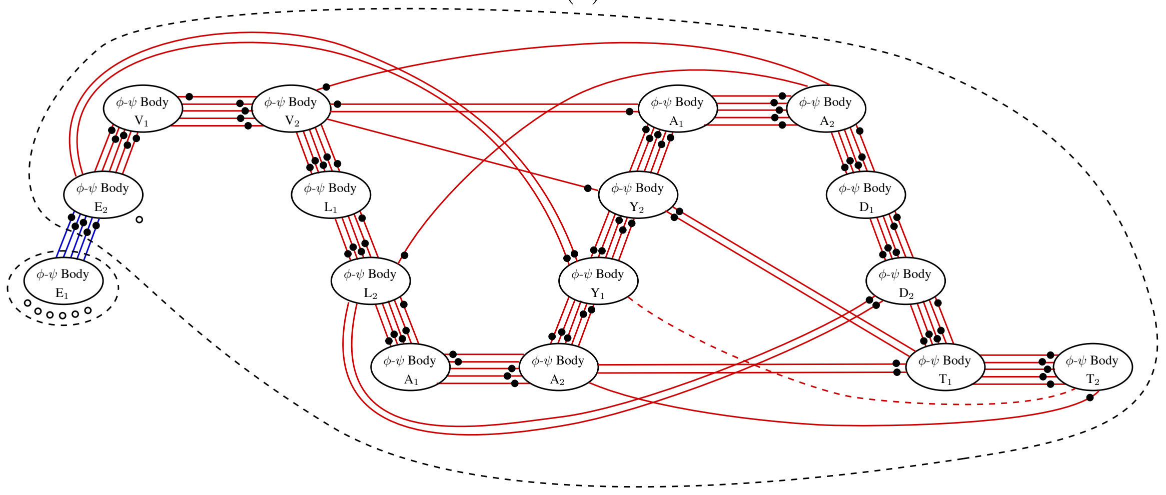
Figs. 24(b) and 25(b) show the rigidity analysis results of the rigidity model in Figs. 24(a) and 25(a), partitioning the constraints into three categories: rigid (in red), independently flexible (in blue), and dependently flexible sets (in purple). The turn segment (b) resulted in a single dependently flexible set indicated by the dotted box. The helical segment (d) resulted in two rigidity clusters indicated in dashed circles, one independently flexible constraint (in blue), and one redundant constraint (red dashed line).

3. Optimizing the Rigidity Analysis

As discussed in Section D, the pebble game, when applied to proteins, first considers all the backbone related constraints before any others (such as hydrogen bonds and hydrophobic interactions). Because all of our conformations have an intact backbone (i.e., we do not allow the peptide bonds along the backbone to break during unfolding), we can optimize the pebble game computation by pre-computing the portion relating to the backbone constraints. Typically, more than half of the constraints for a given conformation are backbone constraints. Thus, this simple optimization can result in a considerable savings, trading the time to play the pebble game on the backbone constraints for the time to copy the results from a pre-computed version.



(a)



(b)

Fig. 25.: Two-body, ϕ-ψ model example of the helical fragment (EVLAYADT) in Fig. 22. (a) The resulting rigidity model where hydrogen bonds are indicated with double dark grey lines and hydrophobic interactions with single light grey lines. (b) Pebble game output and rigidity analysis results. Each body has 6 pebbles which are either free (unfilled) or constrained to a bar (filled). The helical segment resulted in two rigidity clusters (dashed circles), one independently flexible constraint (in blue), and one redundant constraint (red dashed line).

We demonstrate below in Section A.4 that the optimized version runs over 50% faster for some proteins.

4. Rigidity Model Comparison

We first compare the computational performance of the two rigidity models. We studied a set of 18 proteins ranging in length from 56 to 372 residues and of varying secondary structure makeup. Table IV gives the set of proteins studied. Unless otherwise stated, all results employ the backbone optimization for the pebble game. These results were run on a single 2.5GHz processor in the Brazos Cluster at Texas A&M University. The Brazos Cluster contains 126 computing nodes: 96 with 16 GBytes RAM each, and 30 with 32 GBytes RAM each. Each node is a Dell PowerEdge 1950 with two quad-core 2.5 GHz Intel Xeon processors.

For each protein, we randomly generate a set of 1000 samples, from folded to unfolded. To get a similar distribution for each protein, we partition the energy landscape based on contacts present, just as for iterative Gaussian sampling described in Chapter II, Section C.3. Fig. 26 shows that the relationship between protein length and average contacts present is linear across the entire sample set. Fig. 26 also indicates the relationship between protein length and average constraints (rigidity model bars) present for each model is also linear.

Fig. 27 presents the running time to perform rigidity analysis on 1000 randomly sampled conformations for proteins ranging in length from 56 to 372 residues. Computing rigidity analysis for the single-body, Cα rigidity model is faster than the two-body, $\phi - \psi$ rigidity model. The running time is dependent on the number of constraints the pebble game has to evaluate, and the two-body, ϕ-ψ model has 2 to 2.5 times as many constraints as the single-body, Cα model. Both models are fit to a quadratic function of the form $ax^2 + bx + c$, where x is the average number of con-

Table IV.: Proteins studied in the rigidity model performance comparison. The average number of rigidity model bars present in a conformation from the 1000 randomly generated input samples is given for both rigidity models.

PDB				**Avg. Constraints Present**	
ID	**Length**	**# α**	**# β**	**Cα Model**	**ϕ-ψ Model**
1BDD	60	3	0	280.1	640.1
351C	82	5	0	402.4	894.4
2CRS	60	0	6	280.7	641.7
2AIT	74	0	6	352.8	797.2
1PGA	56	1	4	277.4	613.4
2PTL	62	1	4	304.9	676.9
1ALU	124	5	0	603.4	1347.5
1A6M	151	10	0	749.7	1655.7
1GXE	130	0	7	604.6	1384.6
1MFN	184	0	15	874.8	1978.8
2AFG	129	4	10	638.8	1412.8
2RN2	155	5	5	793.0	1723.0
1AII	322	19	0	1807.2	3739.2
1JK0	334	22	0	1933.3	3937.3
1FNH	269	0	23	1343.0	2957.0
2JQY	280	0	17	1317.8	2997.8
1A8P	257	12	11	1292.6	2834.6
1QLP	372	12	14	1912.5	4144.5

straints present. This provided the best fit of the data (i.e., the smallest variance of the residuals).

Fig. 28 shows the individual models separately. They also indicate the type of protein for each data point: all α (triangles), all β (inverted triangles), or mixed secondary structure (diamonds). The running time performance is not dependent on secondary structure makeup.

The rigidity analysis computation consists of four steps: detecting which constraints are present (e.g., hydrogen bonds, hydrophobic interactions), the pebble game, rigid cluster decomposition, and dependent hinge set identification. Fig. 29 displays the running time breakdown of the 1000 rigidity analysis computations for each protein. As the average number of constraints present increases, the time becomes dominated by the pebble game portion of the computation. For the smaller proteins, constraint detection consumes a moderate portion of the running time. The time spent in constraint identification is independent of the rigidity model. However, because the single-body, Cα model uses a smaller graph to play the pebble game on (i.e., only 1 vertex per residue instead of 2), the constraint identification consumes a larger percentage of the running time.

Finally, we look at the impact of the backbone optimization. Fig. 30(a) compares the rigidity analysis running time of the optimized and non-oprimized versions. For some proteins, the optimized version runs over 50% faster. As the number of remaining constraints increases (i.e., those not associated with the backbone and not pre-computed), the performance gain drops. Fig. 30(b) shows that the average number of non-backbone constraints increases with protein length. (Note that it is the same for both models since both models represent hydrogen bonds and hydrophobic interactions with the same number of constraints.) Thus, as the protein gets larger, not only does the number of remaining constraints increase, but the size of the under-

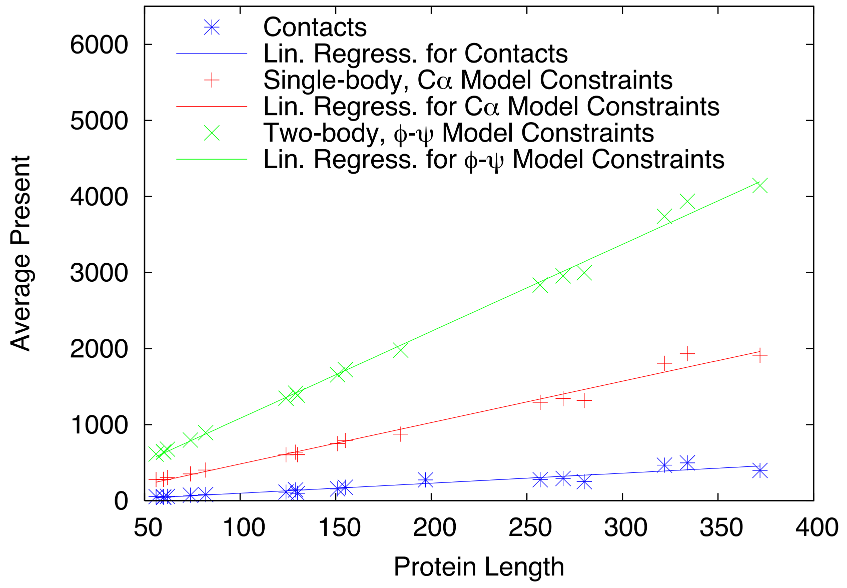


Fig. 26.: Correlation between protein length and average contacts present and average rigidity model constraints present in the sample set.

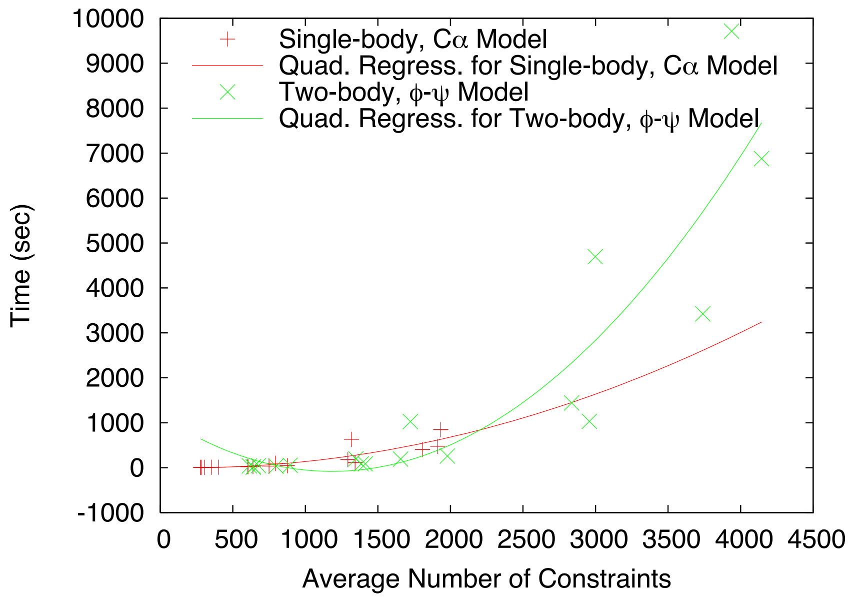
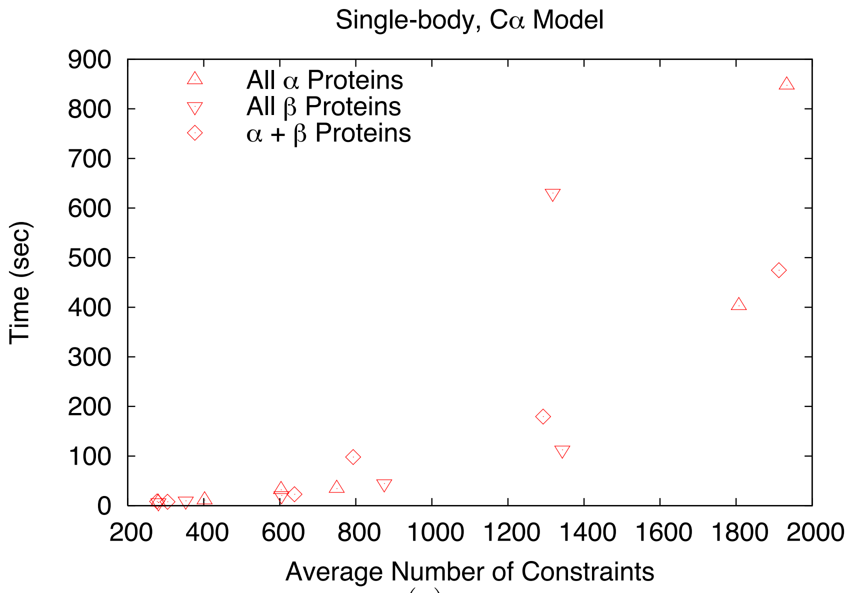
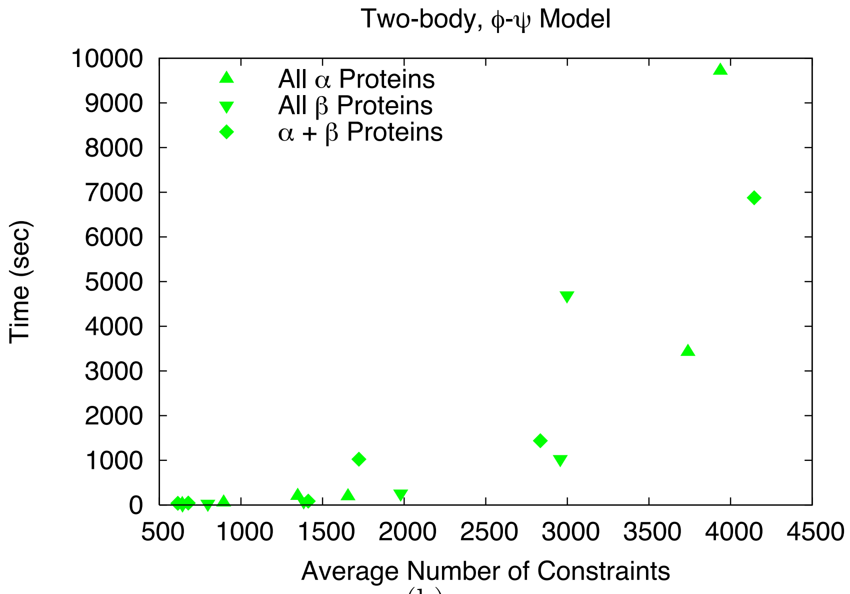


Fig. 27.: Running time to perform rigidity analysis on 1000 proteins ranging in length from 56 to 372 residues. Running times are fit to a quadratic regression on the average number of constraints (rigidity model bars) present.

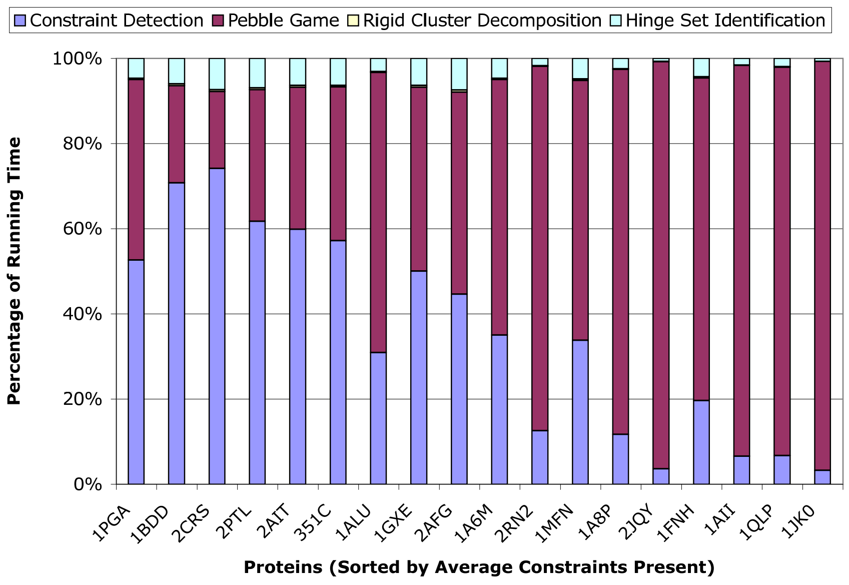


(a)

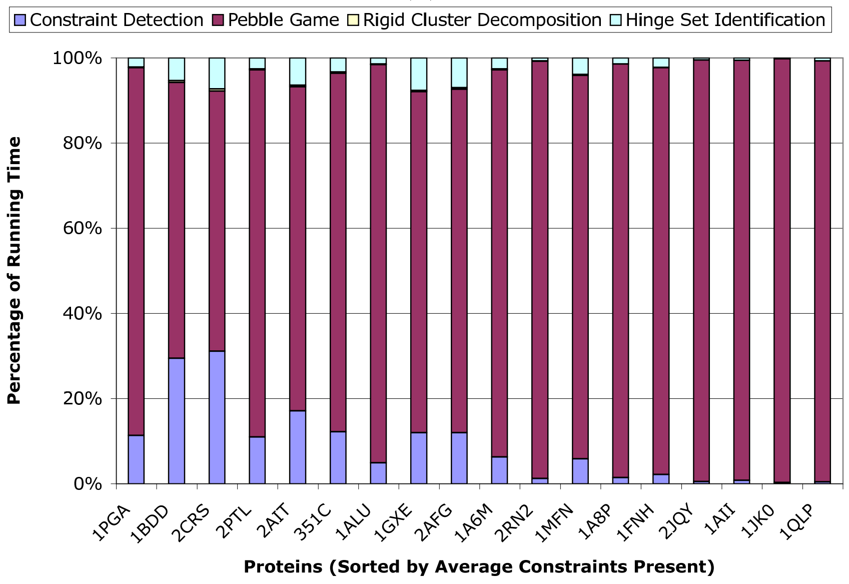


(b)

Fig. 28.: Running time to perform rigidity analysis on 1000 proteins ranging in length from 56 to 372 residues for each model individually. (a) Single-body Cα model only. (b) Two-body ϕ-ψ model only.

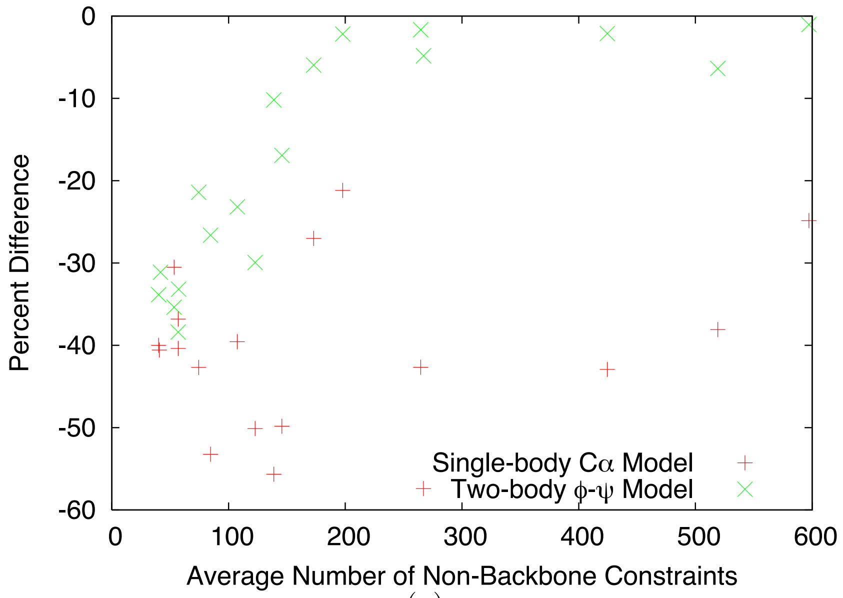


(a)

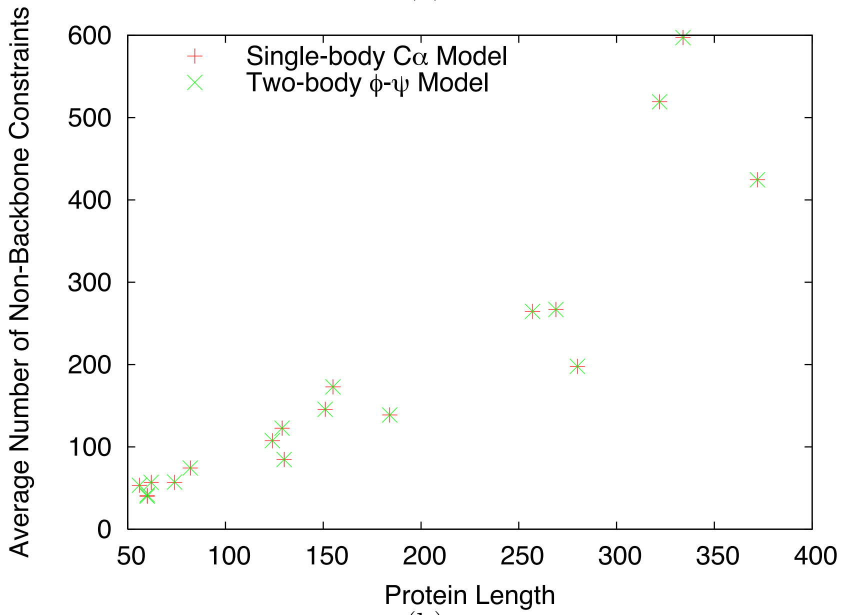


(b)

Fig. 29.: Running time breakdown of the 1000 rigidity analysis computations on proteins ranging in length from 56 to 372 residues for the (a) single-body, Cα rigidity model and the (b) two-body, $\phi - \psi$ rigidity model.



(a)



(b)

Fig. 30.: Speedup gained from the backbone optimization. (a) The percentage difference in running time between the optimized and non-optimized versions. (b) The average number of remaining constraints (i.e., not associated with the backbone) grows with protein length. Note that it is the same for both models since both represent hydrogen bonds and hydrophobic interactions with the same number of constraints.

lying graph on which the pebble game is played also increases. Both of these factors cause the performance gain to drop seen in (a).

B. Rigidity-Biased Sampling

The roadmap produced by our technique is an approximation of the protein's energy landscape. Roadmap quality is measured both by how realistic (as compared to experimental data) are the pathways it contains and by how many samples are required to achieve the desired level of accuracy. The latter is important because it determines how much computation is required and thus what size molecules can be analyzed.

Only a relatively small portion of the conformation space 'near' the target conformation(s) for study is of interest in modeling motions. This implies that we should not use uniform sampling of conformation space — it would require very dense sampling to adequately cover the region near the target conformation(s) and is therefore infeasible for all but very small molecules.

In previous work [18, 21, 22, 23], we obtained a denser distribution of samples near the target conformation through an iterative sampling process where we apply small perturbations to existing conformations, beginning with the target conformation. The perturbations were generated according to a Gaussian distribution centered around the existing conformation. This approach works fairly well, but still requires many samples (e.g., 10,000) for relatively small proteins (e.g., 60–100 residues). To apply our method to larger proteins, we need strategies to generate 'better' samples. The conformations sampled should be more physically realistic and moreover, since we are interested in modeling motion, they should represent 'stepping stones' for conformational transitions.

To better model how a protein transitions from one conformation to another, we follow the same iterative sampling strategy as before but use rigidity analysis to guide how a conformation is perturbed. To perturb a given conformation, we first use rigidity analysis to determine which constraints are independently flexible, dependently flexible, and rigid, see Figs. 21(b), 23(b), 24(b), and 25(b). Independently flexible constraints can be perturbed without affecting the rest of the bodies in the system. Dependently flexible constraints form a set of constraints such that perturbing any one of these results in a corresponding perturbation in the rest of the set.

Recall that peptide constraint sets reflect the backbone torsional ϕ and ψ degrees of freedom in the protein model. For the single-body, Cα rigidity model, a peptide constraint set (i.e., the 4 bars between each body in Fig. 21(a)) represents both ϕ and ψ torsional angles associated with that residue. For the two-body, ϕ-ψ rigidity model, a peptide constraint set (i.e., the 5 bars between each body in Fig. 24(a)) represents either the ϕ or ψ torsional angle associated with that residue. Thus, we perturb each torsional degrees of freedom in the conformation based on the rigidity/flexibility of the corresponding peptide constraint set as follows. If the peptide constraint set is independently flexible, we perturb the corresponding torsional angle(s) with a high probability, P_{flex}. If the peptide constraint set is rigid, we perturb the corresponding torsional angle(s) with a low probability, P_{rigid}. Perturbing rigid degrees of freedom improves the coverage of the space. For each set of dependently flexible peptide constraint sets, we randomly select d torsional angles to perturb with the probability P_{flex} and perturb the remaining torsional angles with the probability P_{rigid}, where d is the internal degrees of freedom to the set. This reflects the d true flexible degrees of freedom in the dependently flexible set. Note that because we cannot distinguish between the ϕ and ψ torsional angles in the single-body, Cα rigidity model, we perturb them both if the peptide bond constraint set selected for perturbation. Algorithm B.1 presents the overall sampling approach.

Algorithm B.1 Rigidity-biased sampling algorithm.

Input: A protein conformation c, probabilities P_{rigid} and P_{flex}, and an angle δ.

Output: A protein conformation c'.

```
Require: Let RAND() return a uniformly distributed random number between 0
    and 1 and GAUSS($x$) return a Gaussian distributed random number centered at
    $x$.
1: Let $B_{peptide}$ be the set of peptide bonds present in $c$.
2: Compute the bonds present in $c$. Let $B_{disulphide}$ be the set of disulphide bonds,
   $B_{hydrogen}$ be the set of hydrogen bonds, and $B_{hydrophobic}$ be the set of hydrophobic
   interactions.
3: Construct a rigidity model $r$ from $B_{peptide}$, $B_{disulphide}$, $B_{hydrogen}$, and $B_{hydrophobic}$.
4: Perform rigidity analysis on $r$. Let $R$ be resulting set of rigid peptide constraints,
   $I$ be the resulting set of independently flexible peptide constraints, and $D$ be the
   resulting set of dependently flexible peptide constraint sets.
5: for each $b \in B_{peptide}$ do
6:     if ($b \in I$ and RAND() $\leq P_{flex}$) or ($b \in R$ and RAND() $\leq P_{rigid}$) then
7:         Set $c'(b) = c(b)$+GAUSS($\delta$).
8:     end if
9: end for
```

Algorithm B.1 continued.

```
10: **for** each $d \in D$ **do**
11:     Let $dof$ be the actual degrees of freedom present in the dependent set $d$.
12:     Let $S$ a set of randomly selected peptide bonds from $d$. $|S| = dof$.
13:     **for** each $b \in \{B_{peptide} \cap d\}$ **do**
14:         **if** ($b \in S$ and RAND() $\leq P_{flex}$) or ($b \notin S$ and RAND() $\leq P_{rigid}$) **then**
15:             Set $c'(b) = c(b)$+GAUSS($\delta$).
16:         **end if**
17:     **end for**
18: **end for**
**return** $c'$.
```

1. Sampling Study

One goal of the rigidity-based sampling algorithm is to restrict how conformations are perturbed in such a way as to find similar low-energy conformations to the source conformation. For various values of P_{rigid} and P_{flex}, we perturb a set of 100 randomly generated input conformations, including the native state. We examine all pairs of probabilities from the set $\{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$ such that $P_{rigid} \leq P_{flex}$ and $P_{flex} > 0$. Perturbation angles are selected from the following set: $\{1.0^o, 2.0^o, 5.0^o, 10.0^o, 20.0^o\}$. These parameters are typically used during iterative sampling in previous work [18, 21, 22, 23]. We study the 6 proteins in Table V and measure the following statistics:

A_{avg} — Average number of attempts to generate a conformation with energy below the node generation maximum threshold, E_{max}.

ΔE_{avg} — Average energy difference between the input conformation and the perturbed conformation.

ΔC_{avg} — Average difference in the number of contacts between the input conformation and the perturbed conformation.

Δd^{Eucl}_{avg} — Average Euclidean distance in $\phi - \psi$ space between the input conformation and the perturbed conformation.

Δd^{RMSD}_{avg} — Average RMSD distance between the input conformation and the perturbed conformation. RMSD is measured over the backbone atoms.

Fig. 31 displays the correlation between A_{avg} and the time required to generate a valid sample. Results are partitioned based on the number of native contacts present in the parent conformation: unfolded parents have $< 25\%$ present, partially folded parents have $\geq 25\%$ and $< 75\%$ present, and folded parents have $\geq 75\%$ present.

Table V.: Proteins used in rigidity-based sampling study.

Protein	**PDB**	**Length**	**Secondary Structure Makeup**
All α Proteins			
Protein A, B domain (Protein A)	1BDD	60	3α
Apo-Myoglobin (ApoMb)	1A6M	151	10α
All β Proteins			
Cardiotoxin analogue III (CTXIII)	2CRS	60	6β
Mouse fibronectin (Fibronectin)	1MFN	184	15β
Mixex α and β Proteins			
B1 domain of protein G (Protein G)	1PGA	56	$1\alpha + 4\beta$
Ribonuclease H (RNase H)	2RN2	155	$5\alpha + 5\beta$

(Note that no "unfolded" parents were generated in the input set for ApoMb.) Each data point represents the average value for a pair of input probabilities (P_{flex}, P_{rigid}) for each type of parent conformation.

As expected, there is a direct relationship between the time spent and the number of attempts made, and this relationship holds regardless of the type of parent conformation or the input probabilities. For all of the proteins except protein A, unfolded conformations typically require fewer attempts to generate a valid sample, and the number of attempts increases with the "foldedness" of the protein. This can be attributed to the overall shape of the energy landscape. At the bottom of the energy

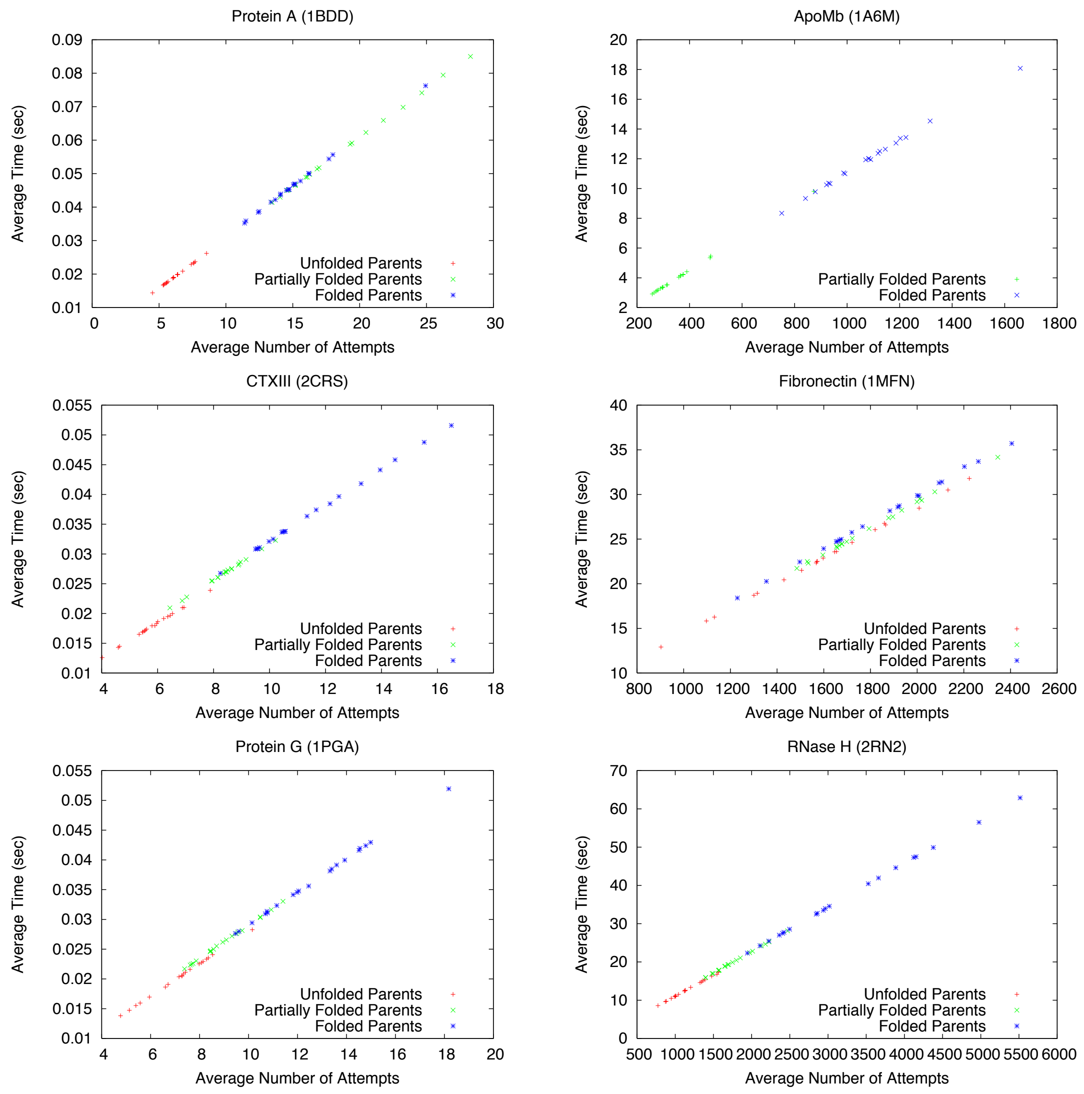
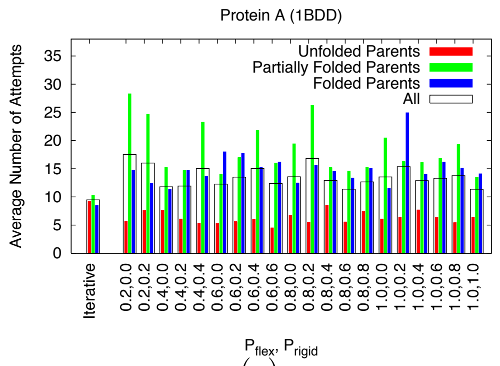


Fig. 31.: Correlation of the average number of attempts, A_{avg}, and the average time required to generate a valid sample for rigidity-based sampling. Results are partitioned based on the number of native contacts present in the parent conformation: unfolded parents have $< 25\%$ present, partially folded parents have $\geq 25\%$ and $< 75\%$ present, and folded parents have $\geq 75\%$ present. Note that no “unfolded” parents were generated in the input set for ApoMb.

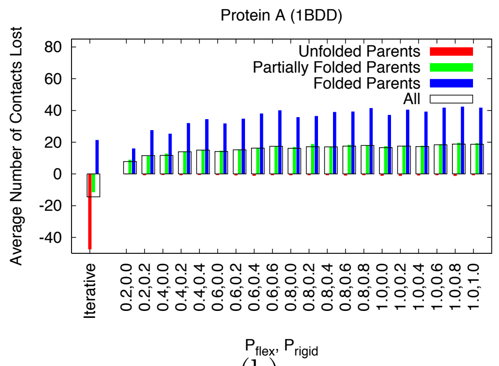
landscape near the native state, the energy landscape is much more "narrow" and the protein is more tightly packed. Here, even small perturbations in the torsional angles can result in collisions and high energies. Thus, it requires more perturbation attempts to generate a valid sample. At the top of the energy landscape, the protein is loosely packed and can better tolerate these perturbations, therefore needing fewer attempts. It is interesting that for Fibronectin the "foldedness" of the parent conformation has less of an effect on the attempts required. This could suggest an energy landscape with a more uniform "width" than the traditional funnel shape.

Figs. 32–37 compare the different statistics for each sampling method and protein. In general, iterative sampling requires fewer attempts (and is faster, e.g., Fig. 31) than rigidity-based sampling for the smaller proteins (protein A, CTXIII, and protein G), but this advantage disappears as the proteins increase in size. Also, as the perturbing probabilities P_{flex} and P_{rigid} increase, there is a corresponding increase in ΔC_{avg}, Δd_{avg}^{Eucl}, and Δd_{avg}^{RMSD}. Rigidity-based sampling preserves the overall shape of the protein better than iterative Gaussian sampling as demonstrated by smaller Δd_{avg}^{RMSD} values.

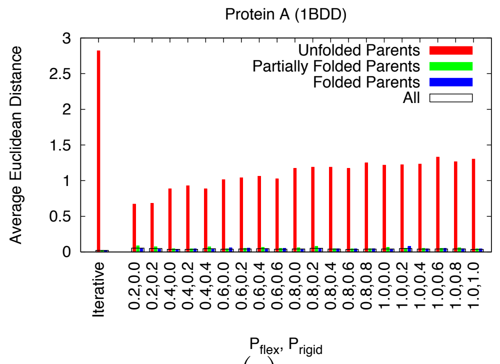
We also look at rigidity-based sampling in the context of iteratively sampling the energy landscape starting from a target state, such as the native state for protein folding problems. Recall from Chapter II, Section C.3 that sampling is biased around a target state by iteratively applying small Gaussian perturbations to existing conformations. In order to obtain an even distribution of samples in the bottom of the energy landscape funnel, in the middle, and in the top, the energy landscape is partitioned into b bins [18, 21, 22, 23]. Then, as sampling progresses, conformations are placed into the different bins. After sampling completes, n/b conformations from each bin are inserted into the roadmap, where n is the total number of conformations requested. Not every bin is always filled with at least n/b conformations resulting in



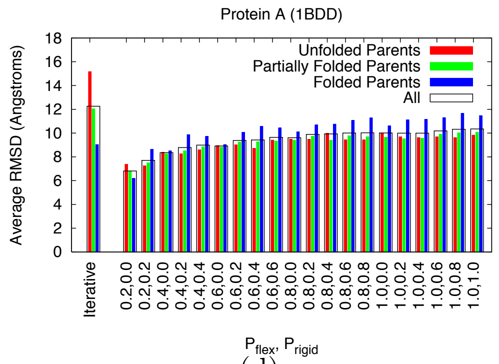
(a)



(b)

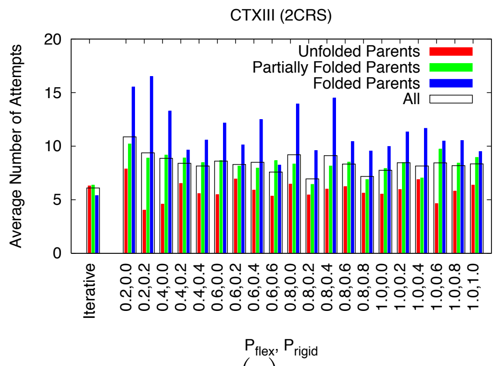


(c)

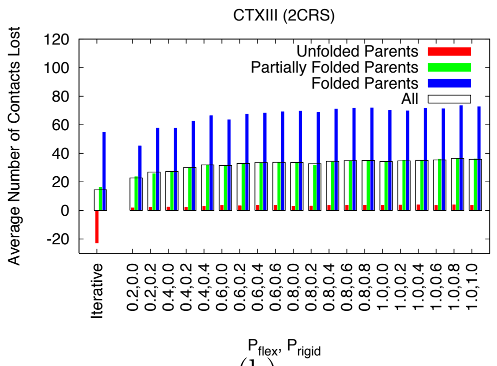


(d)

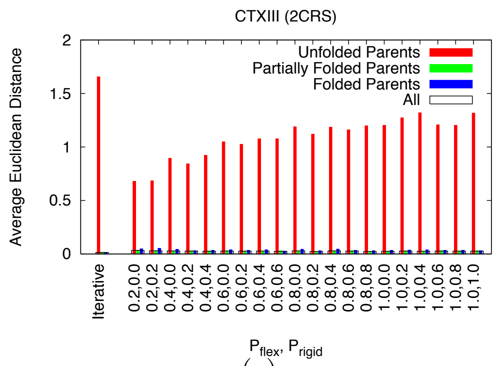
Fig. 32.: Comparison of iterative Gaussian sampling and rigidity-based sampling statistics for protein A. (a) Average number of attempts, A_{avg}. (b) Average number of contacts lost, ΔC_{avg}. (c) Average Euclidean distance, Δd_{avg}^{Eucl}. (d) Average RMSD distance, Δd_{avg}^{RMSD}.



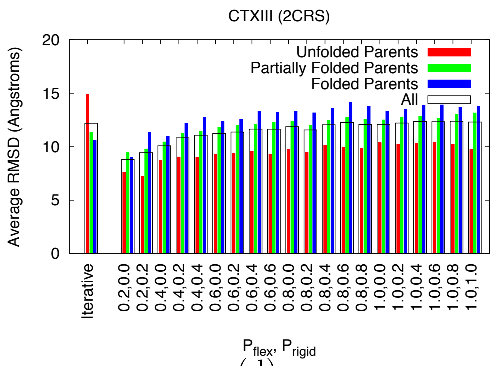
(a)



(b)

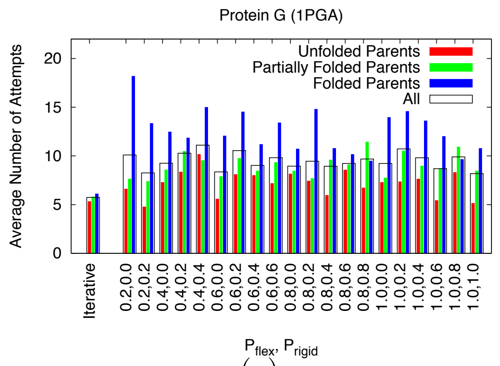


(c)

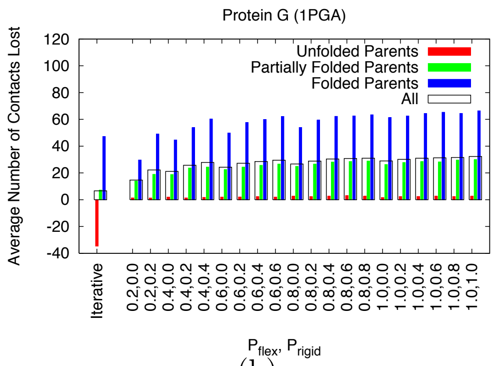


(d)

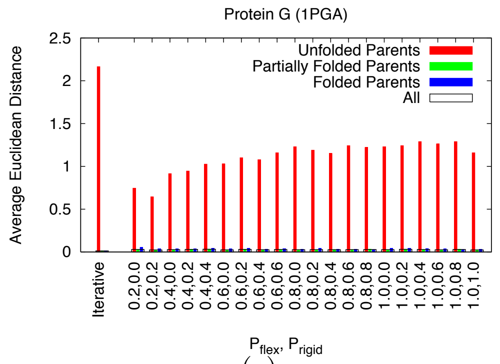
Fig. 33.: Comparison of iterative Gaussian sampling and rigidity-based sampling statistics for CTXIII. (a) Average number of attempts, A_{avg}. (b) Average number of contacts lost, ΔC_{avg}. (c) Average Euclidean distance, Δd_{avg}^{Eucl}. (d) Average RMSD distance, Δd_{avg}^{RMSD}.



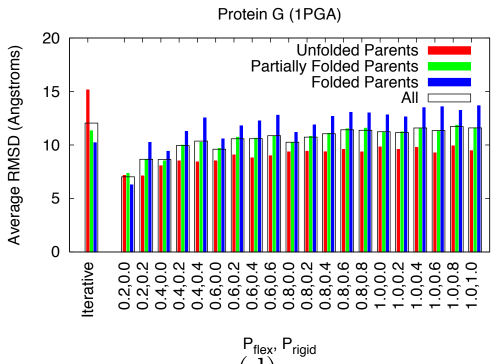
(a)



(b)

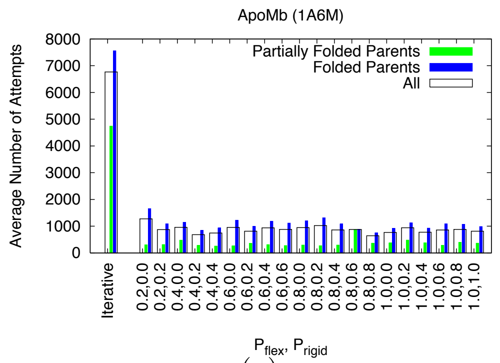


(c)

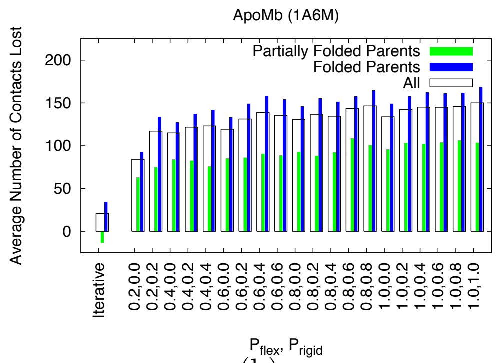


(d)

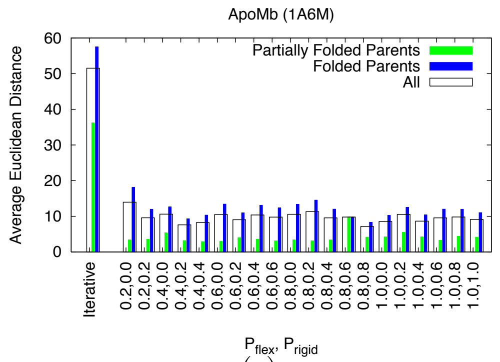
Fig. 34.: Comparison of iterative Gaussian sampling and rigidity-based sampling statistics for protein G. (a) Average number of attempts, A_{avg}. (b) Average number of contacts lost, ΔC_{avg}. (c) Average Euclidean distance, Δd^{Eucl}_{avg}. (d) Average RMSD distance, Δd^{RMSD}_{avg}.



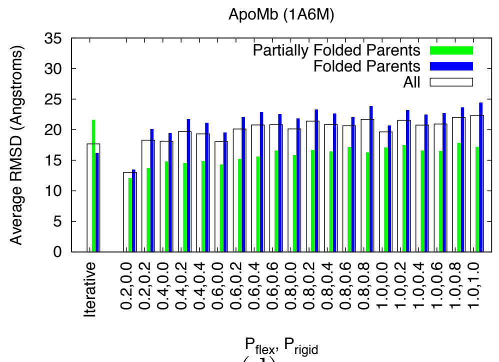
(a)



(b)



(c)



(d)

Fig. 35.: Comparison of iterative Gaussian sampling and rigidity-based sampling statistics for ApoMb. (a) Average number of attempts, A_{avg}. (b) Average number of contacts lost, ΔC_{avg}. (c) Average Euclidean distance, Δd_{avg}^{Eucl}. (d) Average RMSD distance, Δd_{avg}^{RMSD}.

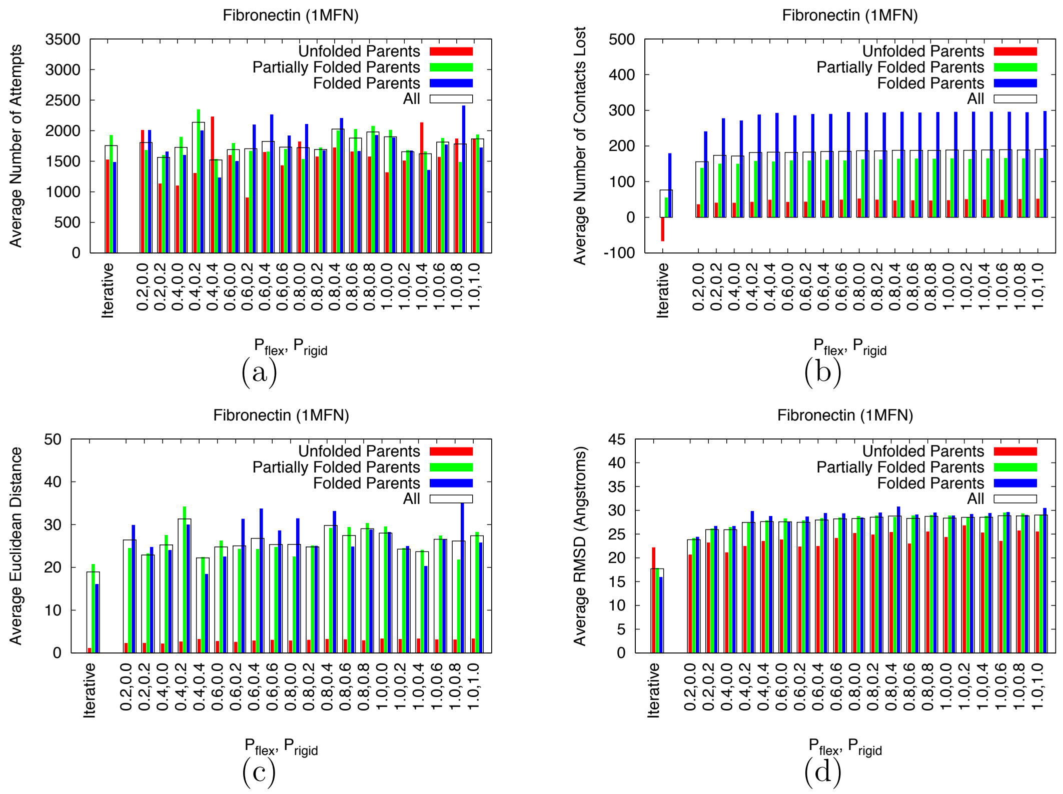


Fig. 36.: Comparison of iterative Gaussian sampling and rigidity-based sampling statistics for Fibronectin. (a) Average number of attempts, A_{avg}. (b) Average number of contacts lost, ΔC_{avg}. (c) Average Euclidean distance, Δd_{avg}^{Eucl}. (d) Average RMSD distance, Δd_{avg}^{RMSD}.

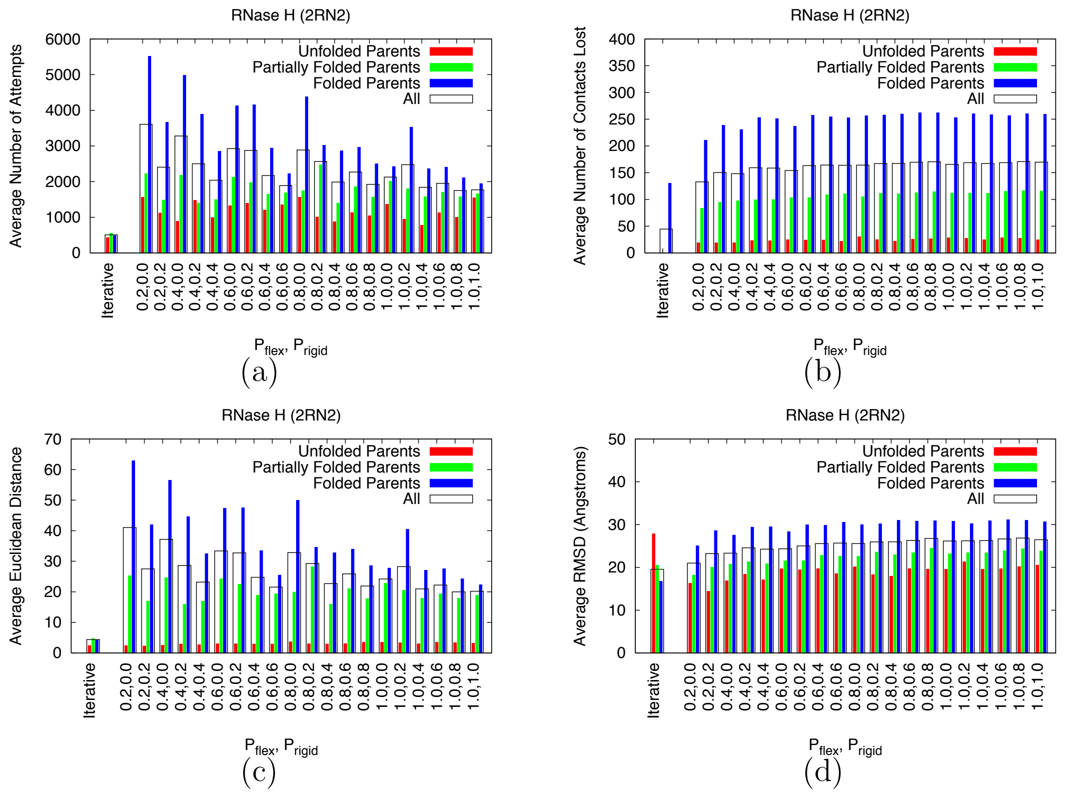


Fig. 37.: Comparison of iterative Gaussian sampling and rigidity-based sampling statistics for RNase H. (a) Average number of attempts, A_{avg}. (b) Average number of contacts lost, ΔC_{avg}. (c) Average Euclidean distance, Δd_{avg}^{Eucl}. (d) Average RMSD distance, Δd_{avg}^{RMSD}.

a less than even distribution of conformations in the roadmap.

Here we compare the ability of iterative Gaussian sampling and rigidity-based sampling to fill all the bins while generating a set of conformations. As in [18, 21, 22, 23], we use equal-sized bins that are based on the contacts present. Each bin represents 10% of the total possible contacts (i.e., $[0\%, 10\%), [10\%, 20\%), \ldots$). For each of the proteins in Table V, we generate 1000 samples, either from iterative Gaussian sampling or rigidity-based sampling. We measure the time required to generate the set of samples as well as the number of nodes "missing" from each bin in order to obtain an even distribution with n/b nodes in each bin.

Figs. 38 and 39 display the results. While iterative Gaussian sampling is faster at attempting to fill the bins than rigidity-based sampling for these proteins, it is not as successful in filling all of the bins. Except for protein A and RNase H, there is at least one combination of P_{flex} and P_{rigid} that have fewer "missing" samples in the distribution. (For protein A and RNase H, they are similar.) We also observe that lower values of P_{flex} and P_{rigid} tend to run faster but produce more "missing" samples. (Note that high values of P_{flex} and P_{rigid} do not always produce even distributions, e.g., protein G, CTXIII, Fibronectin, and RNase H). The values of $P_{flex} = 0.8$ and $P_{rigid} = 0.2$ provide a good trade-off between running time and distribution uniformity for many of the proteins.

C. Rigidity-Based Distance Metric

As samples become farther apart, it is increasingly important which pairs of samples are attempted for connection for two reasons: (1) larger distances mean longer computation times for determining connection feasibility and (2) larger distances mean that it is more likely connections will encounter larger energy barriers since they will

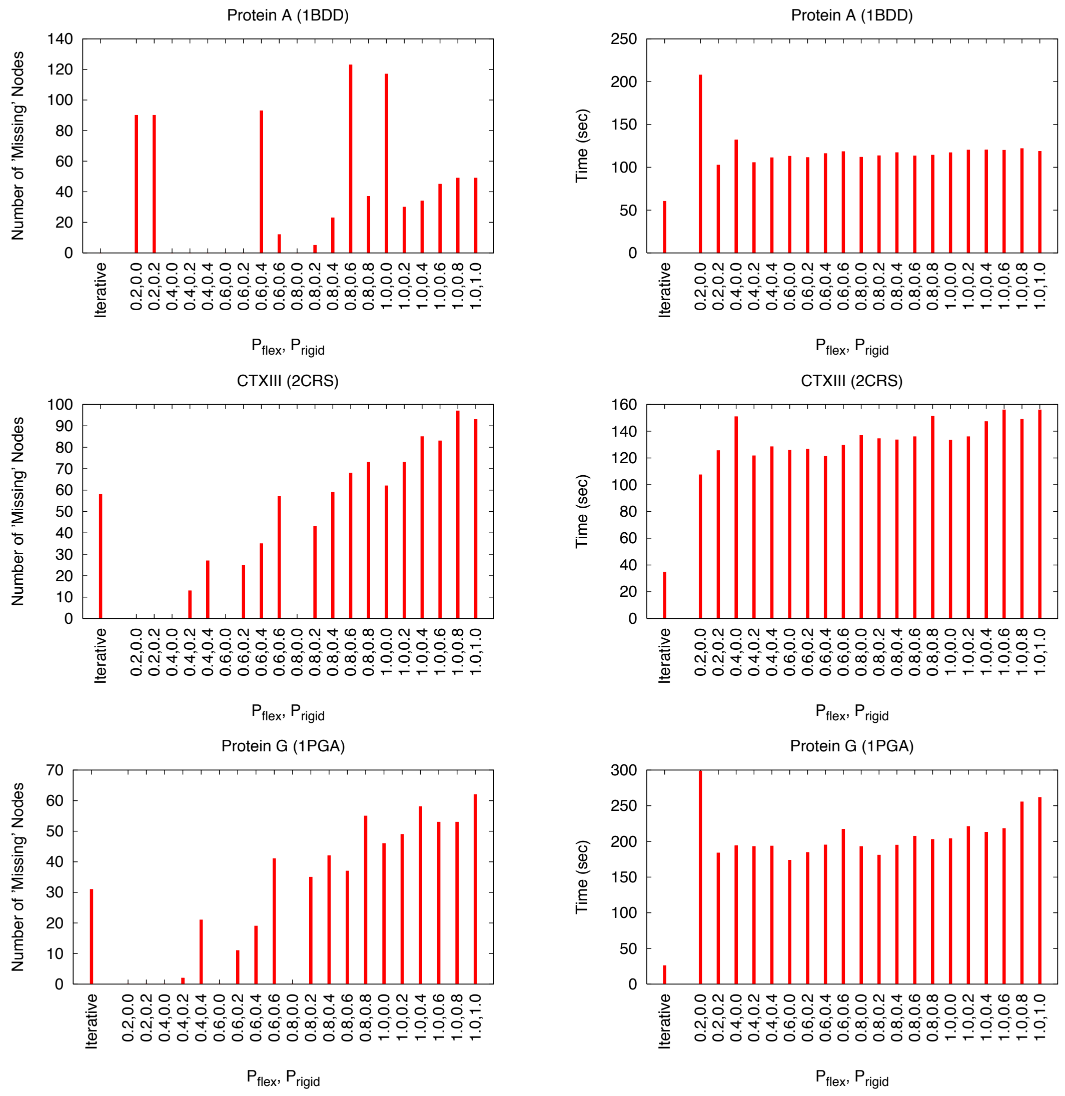


Fig. 38.: Comparison of running time and the number of nodes "missing" from the targeted distribution for iterative Gaussian sampling and rigidity-based sampling on protein A, CTXIII, and protein G.

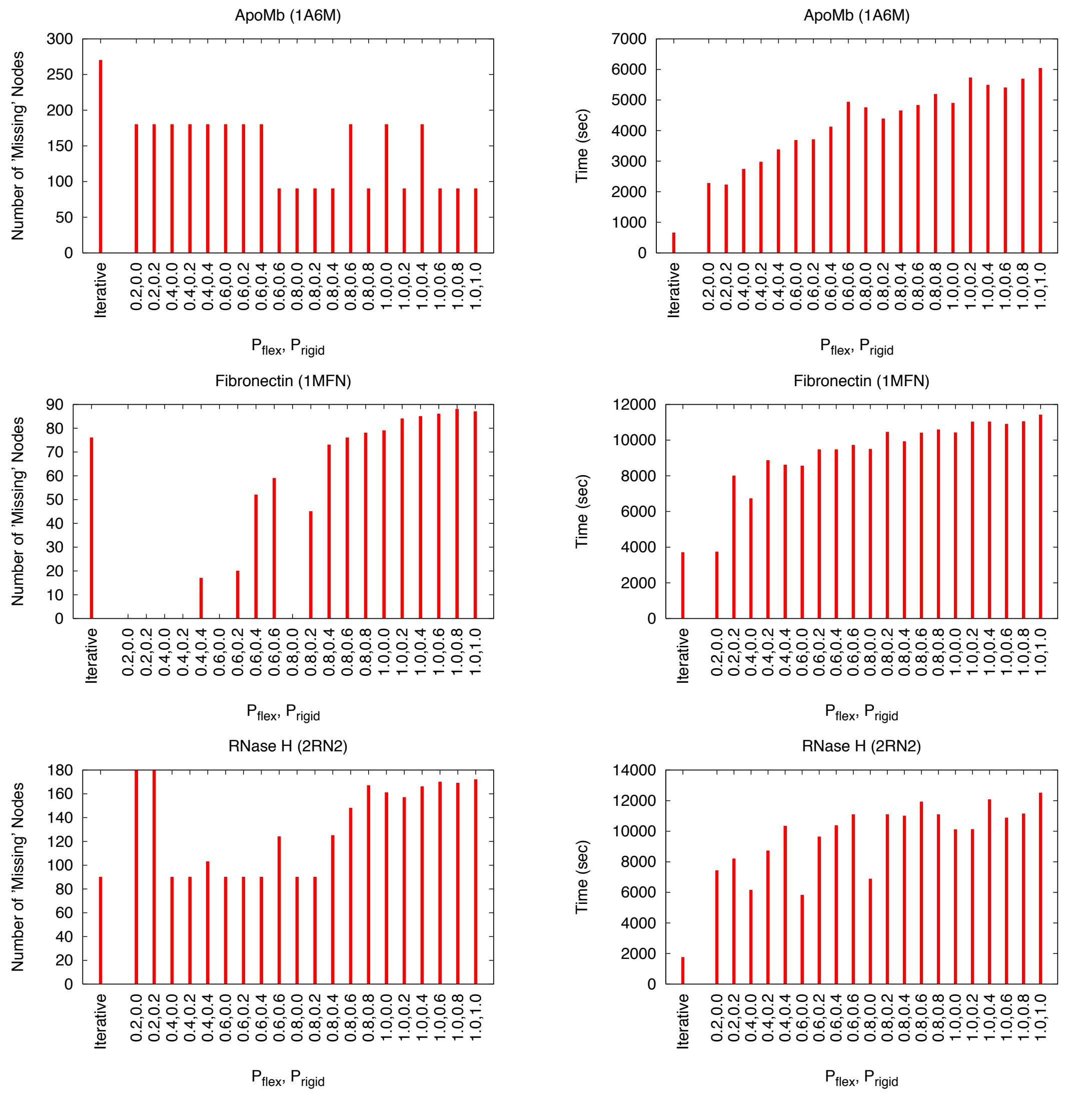


Fig. 39.: Comparison of running time and the number of nodes “missing” from the targeted distribution for iterative Gaussian sampling and rigidity-based sampling on ApoMb, Fibronectin, and RNase H.

traverse larger amounts of the energy landscape. We present a new distance metric for identifying 'nearby' samples based on rigidity analysis. We can then use this distance metric for identifying which samples to connect during roadmap construction instead of the Euclidean distance in ϕ-ψ space as done previously.

We first define a new concept, a *rigidity map*. A rigidity map, r, is similar to a contact map. Rigid body pairs (i, j) from the rigidity model are marked if they have the same rigidity relationship. We mark the residue pairs as follows:

$$r(i,j) = \begin{cases} 2 & \text{if } i \text{ and } j \text{ are in the same rigid cluster} \\ 1 & \text{if } i \text{ and } j \text{ are in the same dependent hinge set} \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

Fig. 40 shows the rigidity map of (a) the native state for protein G for both (b) the single-body, Cα rigidity model and (c) the two-body ϕ-ψ rigidity model. Rigid clusters (i.e., score of 2) are colored black and dependent hinge sets (i.e., score of 1) are colored green.

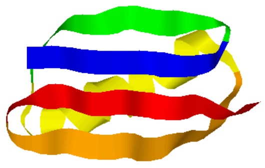
Rigidity maps provide a convenient way to define two rigidity distance metrics, $d_{rig}(q_1, q_2)$ and $d_{clust}(q_1, q_2)$, between two conformations q_1 and q_2 where n is the number of rigid bodies in the model, r_{q_1} is the rigidity map for conformation q_1, and r_{q_2} is the rigidity map for conformation q_2. The first distance metric looks at the entire rigidity map:

$$d_{rig}(q_1, q_2) = \left(\frac{1}{n(n-1)/2}\right)\left(\sum_{0 \le i < j \le n} \delta_{rig}(r_{q_1}(i,j) - r_{q_2}(i,j))\right) \tag{4.2}$$

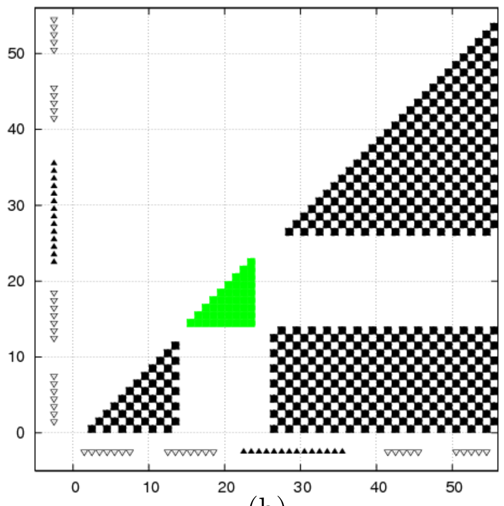
where

$$\delta_{rig}(x) = \begin{cases} 0 & \text{if } x = 0 \\ 1 & \text{otherwise} \end{cases} \tag{4.3}$$

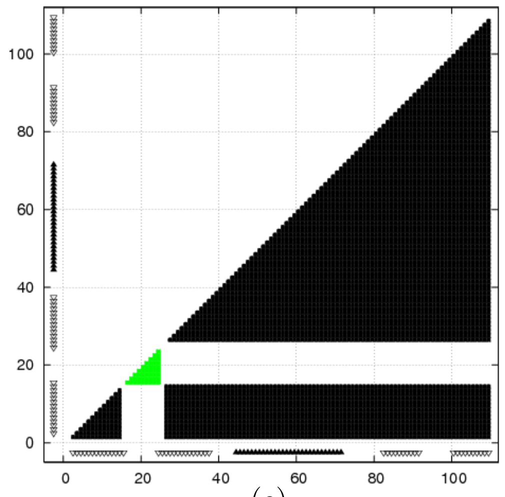
The second distance metric filters out the dependent hinge set information and only



(a)



(b)



(c)

Fig. 40.: Rigidity maps for the native state of protein G (PDB: 1PGA). (a) The native state of protein G. (b) The rigidity map resulting from the single-body, Cα rigidity model. (c) The rigidity map resulting from the two-body, ϕ-ψ rigidity model. For both rigidity maps, rigid clusters are colored black and dependent hinge sets are colored green.

considers the rigid cluster results:

$$d_{clust}(q_1, q_2) = \left(\frac{1}{n(n-1)/2}\right)\left(\sum_{0 \le i < j \le n} \delta_{clust}(r_{q_1}(i,j) - r_{q_2}(i,j))\right) \tag{4.4}$$

where

$$\delta_{clust}(x) = \begin{cases} 1 & \text{if } ||x|| = 2 \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

1. Rigidity Maps of Proteins G, L, and their Mutants

Proteins G, L, mutants of protein G, NuG1 and NuG2 [114], and mutants of protein L, L1 – L4 and S1 – S2 [113], are a unique set of proteins. They are known to fold differently despite having similar structure, see Fig. 41. All proteins are composed of a central α-helix and a 4-stranded β-sheet: β-strands 1 and 2 form the N-terminal hairpin (β1-2) and β-strands 3 and 4 form the C-terminal hairpin (β3-4). The mutants of proteins G and L were computationally designed to switch the folding behavior of the two β-hairpin turns by altering the relative stabilities of the hairpins [114, 113]. Tables VI and VII indicate the substitutions made.

Our rigidity analysis can help to explain the stability shift in NuG, NuG2, L1 – L4, and S1 – S2. For example, consider their native state rigidity maps shown in Fig. 42. In all proteins, the majority of the central α-helix is inside the same rigid cluster. We also see increased rigidity in β1-2 from protein G to NuG1 and NuG2 due to mutating the β-turn from type I to more stable types II' and I' [114]. In all the protein L mutants, there is increased rigidity in both β-hairpins. In these mutants, mutating β3-4 from a non-canonical β-turn type to more stable types (I, II, I', and II') [113] not only increases the rigidity of the local turn, but also helps stabilize the entire structure.

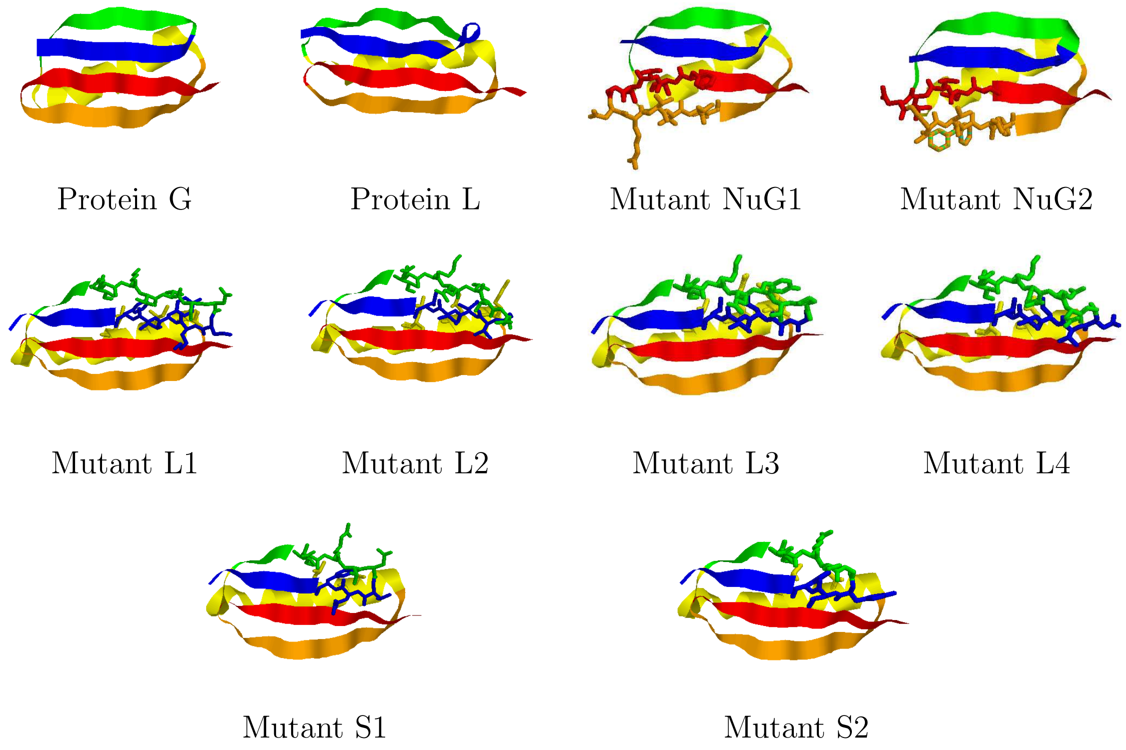


Fig. 41.: Native state ribbons diagrams of proteins G, L, and their mutants NuG1, NuG2, L1 – L4, and S1 – S2. Mutated residues are shown in wireframe.

Table VI.: Sequences of protein G mutants [114]. Residues 1–5 and 17–57 are identical to wildtype. Turn types I' and II' are more favorable over type I' in β-strands.

Protein	**Sequence (6–16)**	**β Turn Type**
WT	`ILNGKTLKGET`	I
NuG1	`FIVIGDRVVVV`	II'
NuG2	`VIVLNGTTFTY`	I'

Table VII.: Sequences of protein L mutants [113]. In the sequences, a '`.`' indicates the same residue as in the wildtype, and a '`-`' indicates a deletion. Residues 1–25 and 35–48 are identical to wildtype. Two-residue turn types (I, II, I', and II') are more favorable over the wildtype 4-residue turn. Residues in the helix (26–34) were mutated in addition to the second β-hairpin turn to create good packing between the new turns and the helix.

Protein	**Sequence (26–34)**	**Sequence (49–60)**	**β Turn Type**
WT	`FEKATSEAY`	`VDVADKG--YTL`	none
L1	`A...I...L`	`IEKVVSDNKYIF`	I
L2	`K...L..VL`	`IDKRVTNGVIIL`	I'
L3	`Y...R...L`	`IDKRYTPGALIL`	II
L4	`....L..VL`	`IDKRQDGNVLVL`	II'
S1	`....V...L`	`IDR--TDT--RF`	I
S2	`....V...L`	`IDR--DGY--LF`	II'

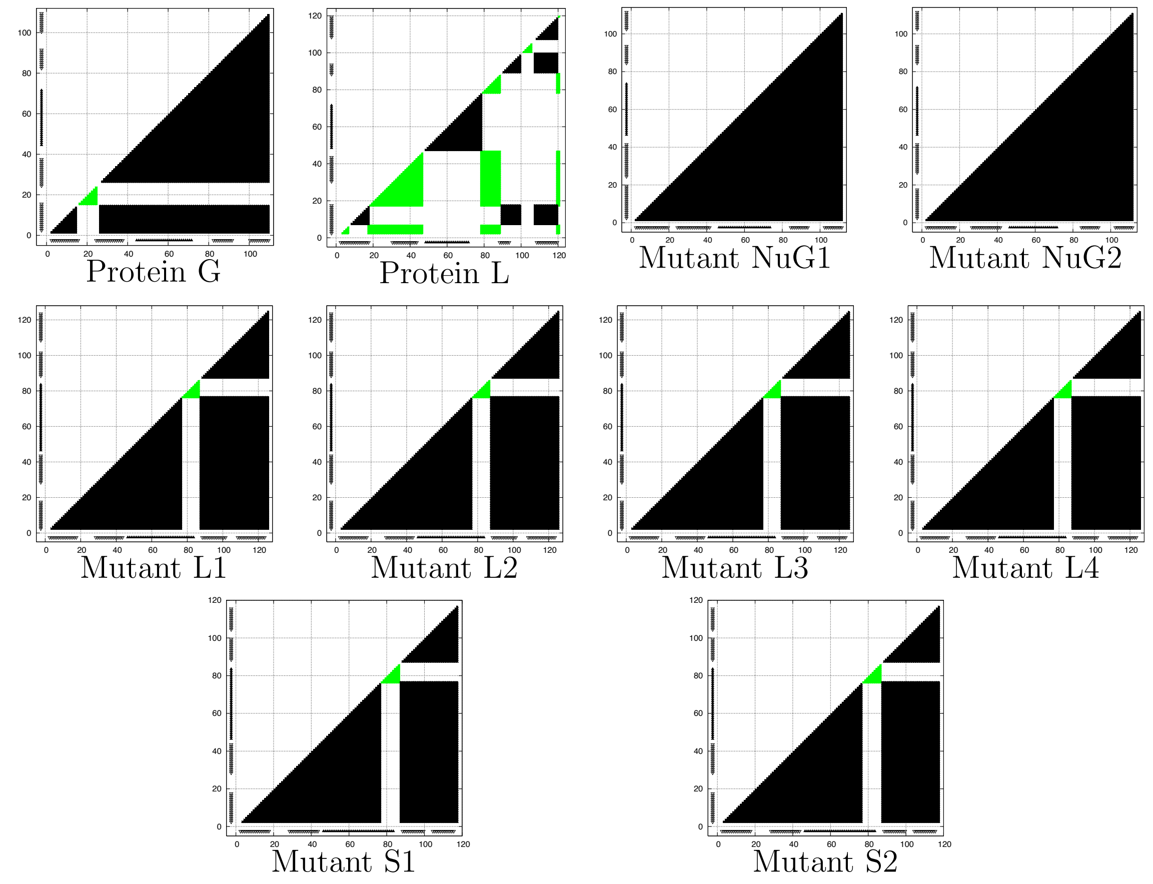


Fig. 42.: Rigidity maps of proteins G, L, and their mutants NuG1, NuG2, L1 – L4, and S1 – S2 using the $\phi - \psi$ model. Rigid clusters are colored black and dependent hinge sets are colored green. α-helices and β-sheets are indicated with solid triangles and hollow inverted triangles for reference.

CHAPTER V

RIGIDITY FOR LANDSCAPE MODEL ANALYSIS*

Rigidity analysis can be used to extract data from our approximate landscape models, or roadmaps, in addition to aiding in their construction as demonstrated in Chapter IV. In this chapter, we explore several different applications of rigidity analysis to extract and analyze landscape properties. We first describe how to use rigidity analysis to identify the secondary structure formation order distribution of folding pathways in the roadmap. With this technique we are able to detect the subtle folding differences of structurally similar proteins G, L, and their mutants that we were unable to detect before with our previous definition of secondary structure formation. We then develop a new method for simulating relative hydrogen exchange rates of a set of input pathways using rigidity analysis. We compare our results to experimental data when available. Finally, we use the relative hydrogen exchange rates from our rigidity analysis to detect folding core membership (i.e., the subset of the protein's structure to form first during folding and break last during denaturation). We compare our results to other computational methods and to experiment when available.

A. Identifying Structure Formation Order

The first application we study is using rigidity analysis to determine the secondary structure formation order distribution of the folding pathways in the roadmap. Recall from Chapter II, Section C.3, we can extract folding pathways from the roadmap by

*Part of the data reported in this chapter is reprinted with permission from "Simulating protein motions with rigidity analysis" by S. Thomas, X. Tang, L. Tapia, and N. M. Amato, 2007. *J. Comput. Biol.*, vol. 14, no. 6, pp. 839–855, Copyright 2007 by Mary Ann Liebert, Inc. [24]

finding the most energetically feasible (i.e., shortest) pathway from every unstructured conformation to the native, folded conformation. Here we use the same definition of unstructured as used in Chapter III, Section A.2. We then can analyze each pathway to determine its secondary structure formation order. This results in a distribution of secondary structure formation orders with which we can compare to experimental data.

Previously, we labeled a piece of secondary structure as 'formed' when $x\%$ of its native contacts (i.e., residue pairs within 7Å of each other in the native state) are present, with x typically set to 80%. We can also use rigidity analysis to define when a piece of secondary structure is 'formed'. Rigidity analysis provides more structural information than simply counting native contacts. In addition, it allows us to define 'formation' for pieces of secondary structure that do not contain internal native contacts but may have native contacts to other residues in the structure (e.g., β-strands).

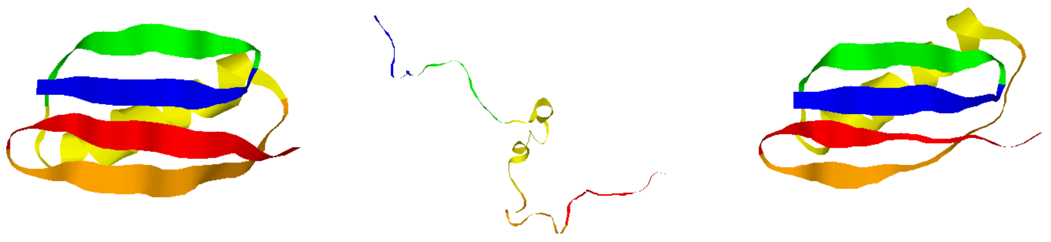
For a given conformation c and a set of residues s corresponding to a piece of secondary structure, we label s as 'formed' in c if the rigidity distance of s between c and the native, folded state is less than a threshold d. As before, each pathway's secondary structure formation order is the order at which pieces are 'formed'.

For example, consider protein G in the two different conformations in Fig. 43(b,c), and let $d = 0.6$. Protein G has 5 pieces of secondary structure: β-strand 1 (residues 2–8, shown in red), β-strand 2 (residues 13–19, shown in orange), α-helix 1 (residues 23–36, shown in yellow), β-strand 3 (residues 42–46, shown in green), and β-strand 4 (residues 51–55, shown in blue). Fig. 43(d–f) shows the corresponding rigidity maps of β-strand 1 using the $\phi - \psi$ rigidity model. (Note that β-strand 1 contains bodies [2,16) in the $\phi - \psi$ rigidity model.) The rigidity distance, $r_{\text{dist}}(c_{\text{native}}, c_1)$, of β-strand 1 between c_{native} and c_1 is 0.857. Thus, β-strand 1 in c_1 would *not* be labeled as 'formed'.

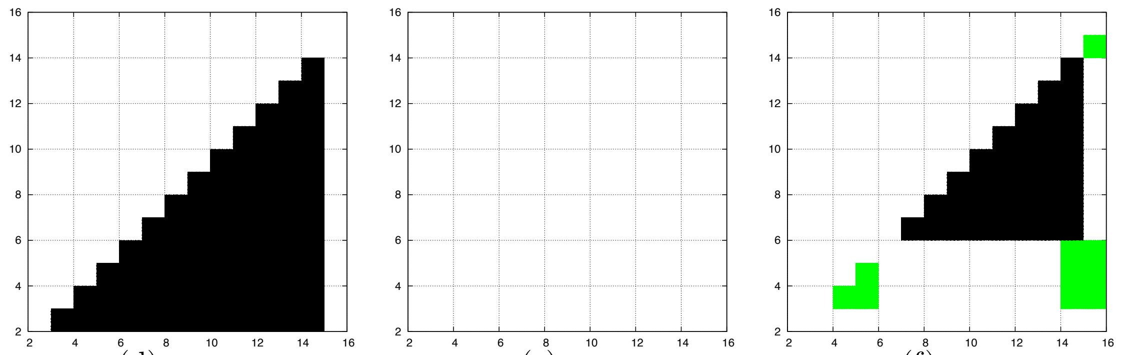
Conversely, the rigidity distance, $r_{\text{dist}}(c_{\text{native}}, c_2)$, of β-strand 1 between c_{native} and c_2 is 0.505. In this case, β-strand 1 in c_2 *would* be labeled as 'formed'.

1. Results

We study how the rigidity-based definition of secondary structure formation order compares to experimental data for the proteins in Table VIII. For each protein, we build two roadmaps, one using iterative Gaussian sampling [18, 21, 22, 23] and one using iterative rigidity-based sampling (see Chapter IV, Section B). The roadmaps are built incrementally using sets of 2000 nodes and stopped when the secondary structure formation order distribution of the roadmap's pathways stabilizes (within 10%), as discussed in Chapter III, Section A. We selected these values because they performed well on several "benchmark" proteins where we knew the experimentally determined secondary structure formation order to validate against. We found that smaller sets of nodes for incremental generation, such as 100, prematurely stabilized their secondary structure formation order to incorrect or unstable distributions. We selected 10% as a stabilization threshold because this allows a little bit of fluctuation in the pathway distributions while still requiring some amount of stability. Smaller values, such as 1% or 5%, would sometimes never stabilize due to the noisiness of the map generation process. For the iterative Gaussian sampling roadmaps, secondary structure pieces are considered 'formed' when 80% of the native contacts are present, as described in Chapter II, Section C.3. For the iterative rigidity-based sampling roadmaps, secondary structure pieces are considered 'formed' when the rigidity distance is ≤ 0.8, as described above. This value requires the pieces to exhibit very similar rigidity properties without requiring them to be identical. Larger values for the rigidity distance typically required the complete structure to be present.



(a) c_{native} (b) c_1 (c) c_2



(d) $r_{c_{\text{native}}}$ (e) r_{c_1} (f) r_{c_2}

Fig. 43.: Example of the rigidity-based secondary structure formation definition for the first β-strand (shown in red) in protein G. (a) Native, folded conformation of protein G, c_{native}. (b) Protein G in a mostly unfolded conformation, c_1. (c) Protein G in a partially folded conformation, c_2. (d–f) Rigidity maps using the $\phi - \psi$ rigidity model of the first β-strand in c_{native}, c_1, and c_2, respectively.

Table VIII.: Proteins studied for secondary structure formation order based on rigidity analysis.

Protein	**PDB**	**Length**	**Secondary Structure**
Crambin, Ser22/Ile25 form	1AB1	46	$2\alpha + 2\beta$
Crambin, Pro22/Leu25 form	1CCM	46	$2\alpha + 2\beta$
Desulfovibrio vulgaris Miyazaki F rubredoxin (RdDvMF)	1RDV	52	$2\alpha + 3\beta$
Murine epidermal growth factor (mEGF)	1EGF	53	3β
Albumin-binding GA module	1PRB	53	4α
Clostridium pasteurianum rubredoxin (RdCp)	1SMU	54	$3\alpha + 3\beta$
Clostridium acidurici ferredoxin (FdCa)	1FCA	55	$2\alpha + 4\beta$
Heparin-binding domain of vascular endothelial growth factor (VEGF)	1VGH	55	$1\alpha + 4\beta$
B1 domain of protein G (Protein G)	1GB1	56	$1\alpha + 4\beta$
α-Spectrin SH3 domain (α-Spectrin)	1SHG	57	1α+5β
Bovine pancreatic trypsin inhibitor (BPTI)	1BPI	58	$2\alpha + 2\beta$
Trypsin inhibitor	4PTI	58	$2\alpha + 2\beta$
Fyn src SH3 domain (fSH3)	1NYF	58	5β
Human complement control factor H (hCCPh)	1HCC	59	7β
Protein A, B domain (Protein A)	1BDD	60	3α
Tick anticoagulant peptide (TAP)	1TCP	60	$2\alpha + 2\beta$
Saccharomyces cerevisiae ADR1 DNA-binding domain	2ADR	60	$2\alpha + 2\beta$

Table VIII Continued

Protein	**PDB**	**Length**	**Secondary Structure**
B1 domain of protein L (Protein L)	2PTL	62	$1\alpha + 4\beta$
Chymotrypsin inhibitor 2 mutant (CI2 mutant)	1COA	64	$1\alpha + 5\beta$
Chymotrypsin inhibitor 2 (CI2)	2CI2	65	$2\alpha + 5\beta$
Escherichia coli cold shock protein (eCSPA)	1MJC	69	7β
α-amylase inhibior Hoe-467A (Hoe)	1HOE	74	7β
Ubiquitin	1UBQ	76	$1\alpha + 5\beta$
Activation domain of human procarboxypeptidase A2 (ADA2h)	1O6X	81	$2\alpha + 3\beta$
Activation domain of porcine procarboxypeptidase B (ADBp)	1PBA	81	$4\alpha + 3\beta$
Bovine acyl-coenzyme A binding protein (bACBP)	2ABD	86	5α

Table IX summarizes the results. Connectivity is the average number of neighbors a sample has in the roadmap. In all cases, the rigidity-based roadmaps produce equivalent folding pathways as the previous method with smaller, more efficient roadmaps, i.e., average change in the number of samples needed is -79.01%. Rigidity-based roadmaps also increase connectivity, i.e., average change in connectivity is 175.67%. Because both techniques attempt the same number of connections, increased connectivity indicates that the rigidity-based roadmaps identified better, more easily connectible, sample pairs as connection candidates.

Table IX.: Rigidity-based sampling and rigidity-based secondary structure formation order results for the proteins in Table VIII. N is the number of nodes, E is the number of edges. In all cases, rigidity-based sampling significantly reduces the required roadmap size ($N + E$) to produce equivalent pathways. It also increased roadmap connectivity (E/N).

	Gaussian Sampling				**Rigidity Sampling**			
PDB	N	E	$N+E$	E/N	N	E	$N+E$	E/N
1AB1	24206	386974	411180	15.99	6000	158286	164286	26.38
1CCM	43646	728964	772610	16.70	10000	456080	466080	45.61
1RDV	33691	457392	491083	13.58	4000	166702	170702	41.68
1EGF	27356	391146	418502	14.30	4000	164902	168902	41.23
1PRB	44551	696708	741259	15.64	4000	126562	130562	31.64
1SMU	35501	557416	592917	15.70	4000	158852	162852	39.71
1FCA	38216	489840	528056	12.82	4000	162526	166526	40.63
1VGH	38216	631936	670152	16.54	4000	157454	161454	39.36
1GB1	34236	912908	947144	26.66	4000	160552	164552	40.14
1SHG	24696	270232	294928	10.94	18000	654884	672884	36.38
1BPI	28426	399418	427844	14.05	4000	112010	116010	28.00
4PTI	39121	389468	428589	9.96	4000	160100	164100	40.03
1NYF	23921	262376	286297	10.97	6000	249450	255450	41.58
1HCC	33691	453628	487319	13.46	28000	1079904	1107904	38.57

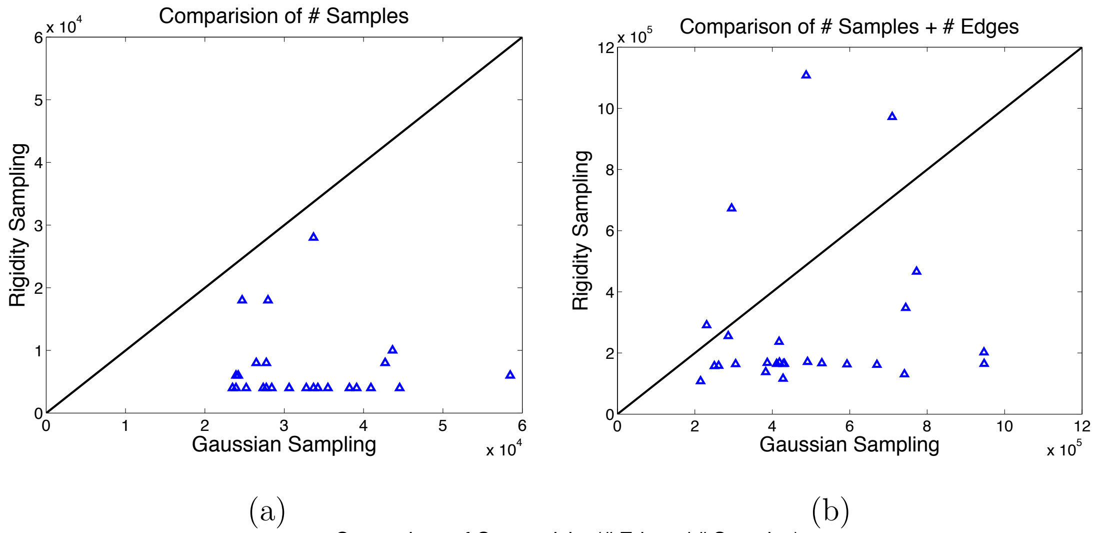
Table IX Continued

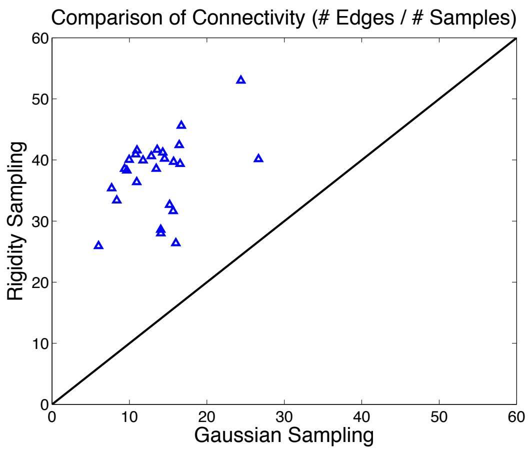
PDB	**Gaussian Sampling** N	E	$N+E$	E/N	**Rigidity Sampling** N	E	$N+E$	E/N
1BDD	58486	888298	946784	15.19	6000	195950	201950	32.66
1TCP	32786	354262	387048	10.81	4000	163692	167692	40.92
2ADR	42723	701942	744665	16.43	8000	339498	347498	42.44
2PTL	23921	281334	305255	11.76	4000	159728	163728	39.93
1COA	27746	403438	431184	14.54	4000	160838	164838	40.21
2CI2	27746	389670	417416	14.04	8000	228706	236706	28.59
1MJC	23481	226942	250423	9.66	4000	153140	157140	38.29
1HOE	30626	184012	214638	6.01	4000	103668	107668	25.92
1UBQ	25206	236216	261422	9.37	4000	154192	158192	38.55
1O6X	40931	342138	383069	8.36	4000	133544	137544	33.39
1PBA	26476	203974	230450	7.70	8000	282960	290960	35.37
2ABD	27956	681796	709752	24.39	18000	953900	971900	52.99
Average	32983	458555	491538	13.68	6923	269157	276080	33.70
Average Percent Change					-79.01	-41.30	-43.83	175.67

Fig. 44 provides a graphical view of the data in Table IX. It compares the two methods in terms of (a) number of samples needed, (b) total size of the resulting roadmap (i.e., the number of samples and edges), and (c) roadmap connectivity. Each data point corresponds to a different protein/row in Table IX. For example, protein 1AB1 appears at point (24206, 6000) in part (a) of the figure, indicating that it required 24,206 samples when using our previous sampling strategy and only 6,000 samples when using the new rigidity based strategy. From these plots, we see that rigidity-based sampling both reduces roadmap size (a, b) and increases roadmap connectivity (c). Both of these properties are highly desirable when constructing roadmaps. They allow our method to extend to larger proteins than before.

Fig. 45 shows the performance of both methods as a function of protein length for the same statistics: (a) number of samples needed, (b) total roadmap size, and (c) roadmap connectivity. There are two data points for each protein, one for each method. For instance, protein 1AB1, which has 46 residues, appears as a blue circle at (46, 24206) for the previous sampling method and as a red cross at (46, 6000) for the new rigidity-based sampling method. Note that in some cases more than one protein has the same length (e.g., 1AB1 and 1CCM both are 46 amino acids long). These plots indicate that the performance gains in Fig. 44 are not dependent on protein length. Thus, with rigidity analysis, we can study much larger proteins than before.

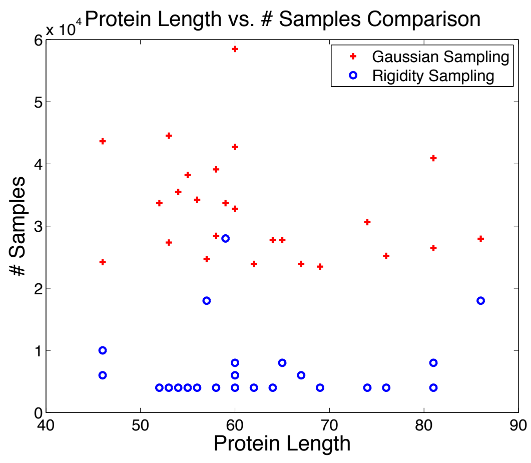
Table X compares the secondary structure formation order of the rigidity roadmaps to experimental data when available. Experimental results come from various methods including continuous labeling hydrogen exchange [42], pulse labeling hydrogen exchange [42], saturation transfer hydrogen exchange[115, 116], Φ-value analysis [117, 118], and Ψ-value analysis [119]. For most proteins, there is good agreement between experimental data and the secondary structure formation order found in our roadmaps. The one exception is wild-type chymotrypsin inhibitor 2 (2CI2). Here, experimental data suggests that α-helix 2 forms early during folding while our results indicate the opposite.

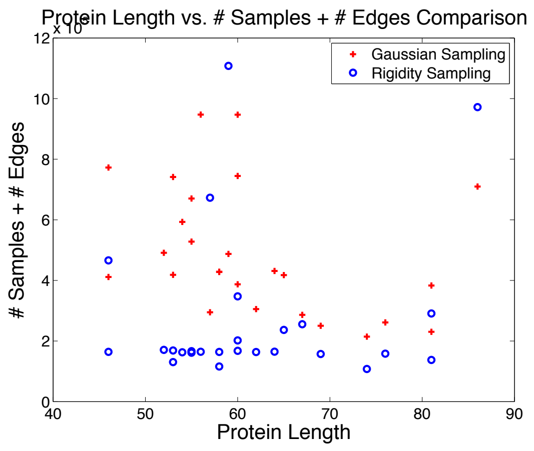




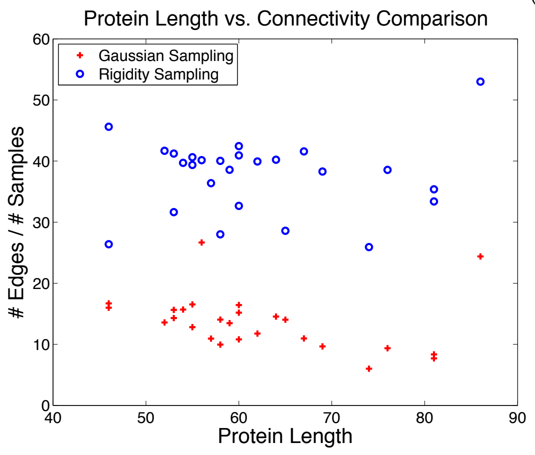
(c)

Fig. 44.: Comparison of rigidity-based sampling to previous work for several proteins. Rigidity-based sampling gives improved performance in terms of both (a, b) reduced roadmap size and (c) increased roadmap connectivity.





(a) (b)



(c)

Fig. 45.: Comparison of sampling methods as a function of protein length. Performance gains are not dependent on protein length.

Table X.: Comparison of secondary structure formation orders for proteins in Table VIII using rigidity-based sampling and the rigidity-based secondary structure formation definition with known experimental results. Brackets indicate no clear order. Only formation orders greater than 1% are shown.

PDB	**Experimental Order**	**Rigidity Order**	**%**
1PRB	$\alpha2$, $\alpha4$, $\alpha3^a$ [120]	$\alpha2$, $\alpha1$, $\alpha4$, $\alpha3$	99.4
1SMU	[$\alpha3$,$\beta1$,$\beta2$,$\beta3$], [$\alpha1$,$\alpha2$] [121]	$\beta3$, $\alpha3$, $\alpha2$, $\alpha1$, $\beta2$, $\beta1$	99.9
1GB1	[α,$\beta1$,$\beta3$,$\beta4$], $\beta2$ [122]	α, $\beta3$-4, $\beta1$-2	99.4
	[α,$\beta4$], [$\beta1$,$\beta2$,$\beta3$] [123]		
1SHG	[α,$\beta2$,$\beta4$,$\beta5$], $\beta3^b$ [124, 125]	$\beta5$, α, $\beta4$, $\beta3$, $\beta2$, $\beta1$	98.6
1BPI	[$\alpha2$,$\beta1$,$\beta2$]^c [126]	$\alpha2$, $\beta2$, $\beta1$, $\alpha1$	99.3
	[$\beta1$,$\beta2$], [$\alpha2$]^c [127, 128]		
1NYF	[$\beta3$,$\beta4$], $\beta2$, [$\beta1$,$\beta5$] [129, 130]	$\beta4$, $\beta3$, $\beta2$, $\beta5$, $\beta1$	99.2
1BDD	[$\alpha2$,$\alpha3$], $\alpha1$ [131]	$\alpha2$, $\alpha1$, $\alpha3$	99.9
	[$\alpha1$,$\alpha2$,$\alpha3$] [131]		
	$\alpha2$, [$\alpha1$,$\alpha3$] [132, 133]		
2PTL	[α,$\beta1$,$\beta2$,$\beta4$], $\beta3$ [134]	$\beta1$-2, α, $\beta3$-4	100.0
	[α,$\beta1$], [$\beta2$,$\beta3$,$\beta4$] [135]		
2CI2	[$\alpha2$,$\beta2$,$\beta3$], [$\alpha1$,$\beta4$]^d [136]	$\beta5$, $\beta4$, $\beta3$, $\beta2$, $\alpha1$, $\beta1$, $\alpha2$	98.5
	[$\alpha2$], [$\alpha1$,$\beta1$,$\beta2$,$\beta3$,$\beta4$,$\beta5$] [137]		
	[$\alpha2$,$\beta2$,$\beta3$], [$\alpha1$,$\beta1$,$\beta4$,$\beta5$] [138, 139]		
1MJC	[$\beta3$,$\beta4$], [$\beta1$,$\beta2$,$\beta5$]^e [140]	$\beta2$, $\beta4$, $\beta3$, $\beta7$, $\beta6$, $\beta5$, $\beta1$	97.2
		$\beta6$, $\beta4$, $\beta3$, $\beta2$, $\beta7$, $\beta5$, $\beta1$	1.1
1UBQ	[α,$\beta1$,$\beta2$,$\beta3$], [$\beta4$,$\beta5$] [141]	$\beta4$, α, $\beta2$, $\beta1$, $\beta5$, $\beta3$	99.9

Table X Continued

PDB	Experimental Order	Rigidity Order	%
	[α,β1,β2,β3,β4], [β5] [142]		
	[α,β1,β2], [β3,β4,β5] [143, 144]		
1O6X	[α2,β2], [α1,β1,β3] [145]	α1, α2, β3, β2, β1	100.0
2ABD	[α1,α4,α5], [α2,α3] [146]	α4, α5, α2, α3, α1	95.9
		α5, α2, α4, α3, α1	1.3

[a]α1 was not measured by experiment.
[b]β1 was not measured by experiment.
[c]α1 was not measured by experiment.
[d]β1 and β5 were not measured by experiment.
[e]β6 and β7 were not measured by experiment.

2. Case Study of Proteins G, L, and their Mutants

As discussed in Chapter IV, Section C.1, proteins G, L, and their mutants (NuG1, NuG2, L1–L4, and S1–S2) present a good test case for our technique because they are known to fold differently despite having similar structure, see Fig. 41. Recall that all proteins are composed of a central α-helix and a 4-stranded β-sheet: β-strands 1 and 2 form the N-terminal hairpin (β1-2) and β-strands 3 and 4 form the C-terminal hairpin (β3-4). The mutants of proteins G and L were computationally designed to switch the folding behavior of the two β-hairpin turns by altering the relative stabilities of the hairpins [114, 113]. Hydrogen exchange experiments, both continuous labeling and pulse labeling, for proteins G and L indicate that β1-2 forms first in protein L, and β3-4 forms first in protein G [95]. This is consistent with Φ-value analysis on G

[96] and L [147]. In [114], protein G is mutated in the first β-hairpin turn to increase the relative stability of β1-2 over β3-4 (see Table VI for the specific mutations). Φ-value analysis indicates that the hairpin formation order for both NuG1 and NuG2 is switched from the wild type. In [113], the second β-hairpin turn is replaced with more stable two-residue turns (and the central α-helix is mutated to accommodate the new turn conformation) to increase the relative stability of β3-4 over β1-2 (see Table VII for the specific mutations). Φ-value analysis indicates that the hairpin formation order is switched from wild type for mutant L2. Refolding kinetics suggests that the hairpin formation order may be also switched for the other mutants (L1, L3, L4, S1, and S2).

Our previous sampling strategy [22] was able to capture the folding differences between proteins G and L, but not between protein G and its mutants or between protein L and its mutants. Our new rigidity-based sampling and analysis is able to also capture the correct folding behavior of the mutants, see Table XI.

Table XI.: Comparison of secondary structure formation orders for proteins G, L, and their mutants with known experimental results. Brackets indicate no clear order. In all cases, our technique predicted the secondary structure formation order seen in experiment. Only formation orders greater than 1% are shown.

Protein	**Experimental Order**	**Rigidity Order**	%
G	[α,β1,β3,β4], β2 [122] [α,β4], [β1,β2,β3] [123]	α, β3-4, β1-2	99.4
L	[α,β1,β2,β4], β3 [134]	β1-2, α, β3-4	100.0

Table XI Continued

Protein	**Experimental Order**	**Rigidity Order**	%
	[α,β1], [β2,β3,β4] [135]		
NuG1	β1-2, β3-4 [114]	α, β1-2, β3-4	97.6
		β1-2, α, β3-4	1.6
NuG2	β1-2, β3-4 [114]	α, β1-2, β3-4	96.6
		β1-2, α, β3-4	1.1
		β3-4, β1-2, α	1.1
L1	β3-4 may be formed early[a] [113]	α, β2-α, β3-4, β1-2, β1-α, β1-4	100.0
L2	β3-4 may be formed early[a] [113]	α,β3-4, β2-α, β1-2, β1-4, β1-α	83.3
		α, β2-α, β3-4, β1-2, β1-4, β1-α	9.0
		β3-4, α, β2-α, β1-2, β1-4, β1-α	7.7
L3	β3-4 may be formed early[a] [113]	α, β3-4, β2-α, β1-2, β1-4, β1-α	96.5
		α, β2-α, β3-4, β1-2, β1-4, β1-α	3.5
L4	β3-4, β1-2 [113]	α, β2-α, β3-4, β1-2, β1-4, β1-α	93.4
		β2-α, α, β3-4, β1-2, β1-4, β1-α	6.6
S1	β3-4 may be formed early[a] [113]	α, β3-4, β1-2, β2-α, β1-4, β1-α	100.0
S2	β3-4 may be formed early[a] [113]	α, β3-4, β2-α, β1-2, β1-α, β1-4	70.8
		α, β2-α, β3-4, β1-2, β1-α, β1-4	29.2

[a]Refolding kinetics experiments [113] indicate that the mutant folds faster than wild-type and on the same order of magnitude as mutant L4 suggesting that the folding behavior is similar to mutant L4. No other experiments were performed by the authors to confirm this hypothesis.

B. Relative Hydrogen Exchange

Hydrogen exchange investigates protein folding by identifying which parts of the structure are most exposed or most protected [42]. From this data one can infer which portions of the protein fold first and which are last to form, up to the millisecond timescale. Since hydrogen exchange identifies specific proton exchanges during folding, it has become a prominent experimental technique when studying folding intermediates [148]. Here we use rigidity analysis and our approximate landscape models (i.e., roadmaps) to compute relative hydrogen exchange rates. The goal is to validate against experimental data and to provide information for residues that were unable to be measured experimentally.

There are two main types of hydrogen exchange experiments: continuous labeling and pulse labeling. In continuous labeling experiments, proteins are exposed to Deuterium (D_2O) during folding. This causes exposed hydrogens to swap with D_2O. In pulse labeling, folding is induced and brought to an equilibrium point. After this, the proteins are exposed to D_2O for a short pulse. This allows identification of the exposed hydrogens at that equilibrium point because they have been swapped with D_2O.

Hydrogen exchange analysis can then proceed either locally or globally. In local analysis, the protein is broken up into short subsections, and these subsections are studied for D_2O exposure. Global analysis gives an overall view of how the protein behaved in the D_2O environment. Both the local and global results are analyzed with tools such as Mass Spectrometry [42] or NMR [149].

We have developed two new techniques to extract relative hydrogen exchange rates from our approximate landscape models. Our methods can compute relative exchange rates from any input pathway. We use MMC (described in Chapter II, Section C.3) to extract multiple pathways, analyze each one individually for its relative exchange rates, and then average the results over all input pathways.

1. From Residue Flexibility

Our first method is based on the idea that at a given conformation (or path-step), flexible residues are more likely to experience hydrogen exchange while rigid residues are less likely to exchange. We can label every residue at every path-step along an input pathway as rigid, independently flexible (i.e., can move without requiring movement of other residues), or dependently flexible (i.e., can only move in a coordinated motion with other residues) using rigidity analysis [81, 84]. Using these labels, we then assign each residue at every path-step a score based on its rigidity classification: 0 for independently flexible, 0.5 for dependently flexible, and 1 for rigid. For a residue i, we define its *rigidity score*, $RS(p,i)$, for a particular pathway p, as the average of its rigidity scores at each path-step:

$$RS(p,i) = \frac{1}{|p|}\sum_{c\in p}\begin{cases} 0 & \text{if residue } i \text{ at conformation } c \text{ is independently flexible} \\ 0.5 & \text{if residue } i \text{ at conformation } c \text{ is dependently flexible} \\ 1 & \text{if residue } i \text{ at conformation } c \text{ is rigid} \end{cases} \tag{5.1}$$

To compare the rigidity scores to experimental data, we define the relative exchange rate $ex_{\mathrm{RS}}(p,i)$ for residue i along a pathway p, as

$$ex_{\mathrm{RS}}(p,i) = 1 - \frac{RS(p,i) - RS_{min}(p)}{RS_{max}(p) - RS_{min}(p)} \tag{5.2}$$

where $RS_{min}(p)$ is the smallest $RS(p,i)$ obtained over p for all residues i and $RS_{max}(p)$ is the largest. Thus, residues with large rigidity scores (i.e., most rigid along the pathway) will have low relative exchange rates, and residues with small rigidity scores (i.e., most flexible along the pathway) will have high exchange rates.

MMC provides a set of stochastic pathways whose rigidity scores can be averaged for an overall view of the rigid formation during the folding process. For a set of MMC pathways P, we define the average relative exchange rate, $EX_{\mathrm{RS}}(i)$, for residue i as the average of the relative exchange rates, $ex_{\mathrm{RS}}(p, i)$, for all pathways $p \in P$:

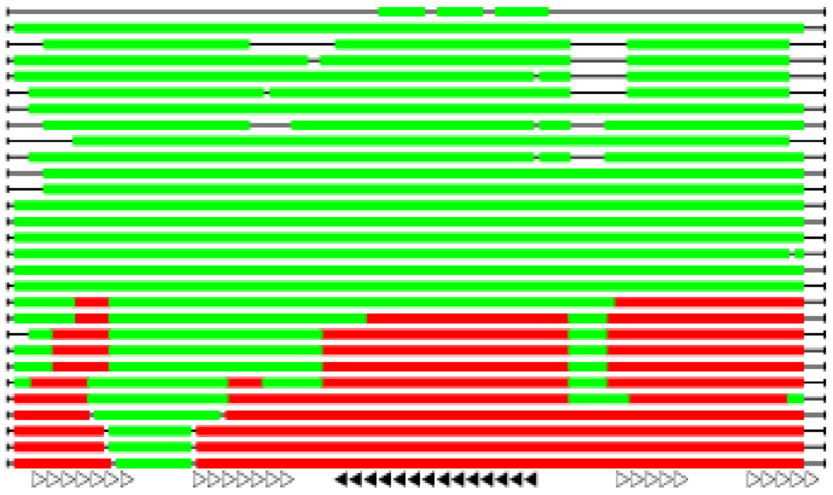
$$EX_{\mathrm{RS}}(i) = \frac{1}{|P|} \sum_{p \in P} ex_{\mathrm{RS}}(p, i) \tag{5.3}$$

Fig. 46(a) shows the rigidity analysis along an example pathway for protein G (a 56 residue protein with a central α-helix flanked by two β-hairpin turns), and Fig. 47 and Fig. 48(a) shows the corresponding relative exchange rates. The second β-hairpin is experimentally known to form before the first β-hairpin [95, 96]. This behavior is reflected in both the rigidity scores and the relative exchange rates: β-hairpin 2 remains more rigid longer than β-hairpin 1 along the pathway in Fig. 46(a), and its corresponding relative exchange rates Fig. 47 are lower.

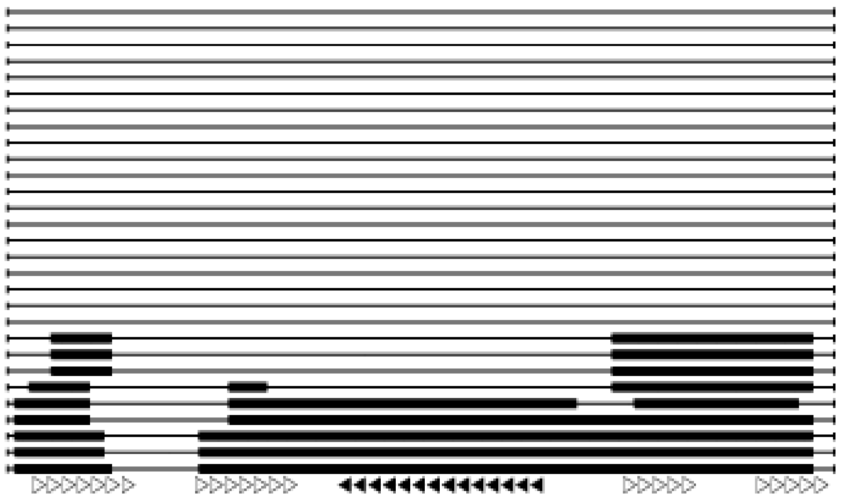
2. From Rigid Cluster Decomposition

Previous experimental [95] and simulation [30, 31] techniques have helped clarify the folding core definition, a related application to hydrogen exchange. Their work suggests that the slowest exchanging residues also include those involved in the formation of tertiary structure, as identified by long-range contacts, rather than simply those involved in secondary structure formation. Inspired by this, we define another score called the *cluster score* based on rigid cluster decomposition. Rigidity analysis can group rigid residues together in *rigid clusters*. Inside a rigid cluster, all the residue positions are fixed with respect to each other. Two rigid clusters may, however, move relative to each other. This partitioning is called the rigid cluster decomposition.

First, we identify subsequences of rigid residues that are more than 1 residue long. For all subsequence pairs identified, if they contain tertiary contacts between



(a)



(b)

Fig. 46.: Example unfolding pathway for protein G. In all plots, α-helices (filled triangles) and β-sheets (empty triangles) are indicated along the bottom for reference. (a) Rigidity analysis results for every path-step from folded (bottom) to unfolded (top). At each path-step, residues are labeled rigid (red), dependently flexible (green), or independently flexible (not colored). (b) Contact subsequence presence at every path-step from folded (bottom) to unfolded (top). At each path-step, residues are colored black if they have a cluster score of 1.

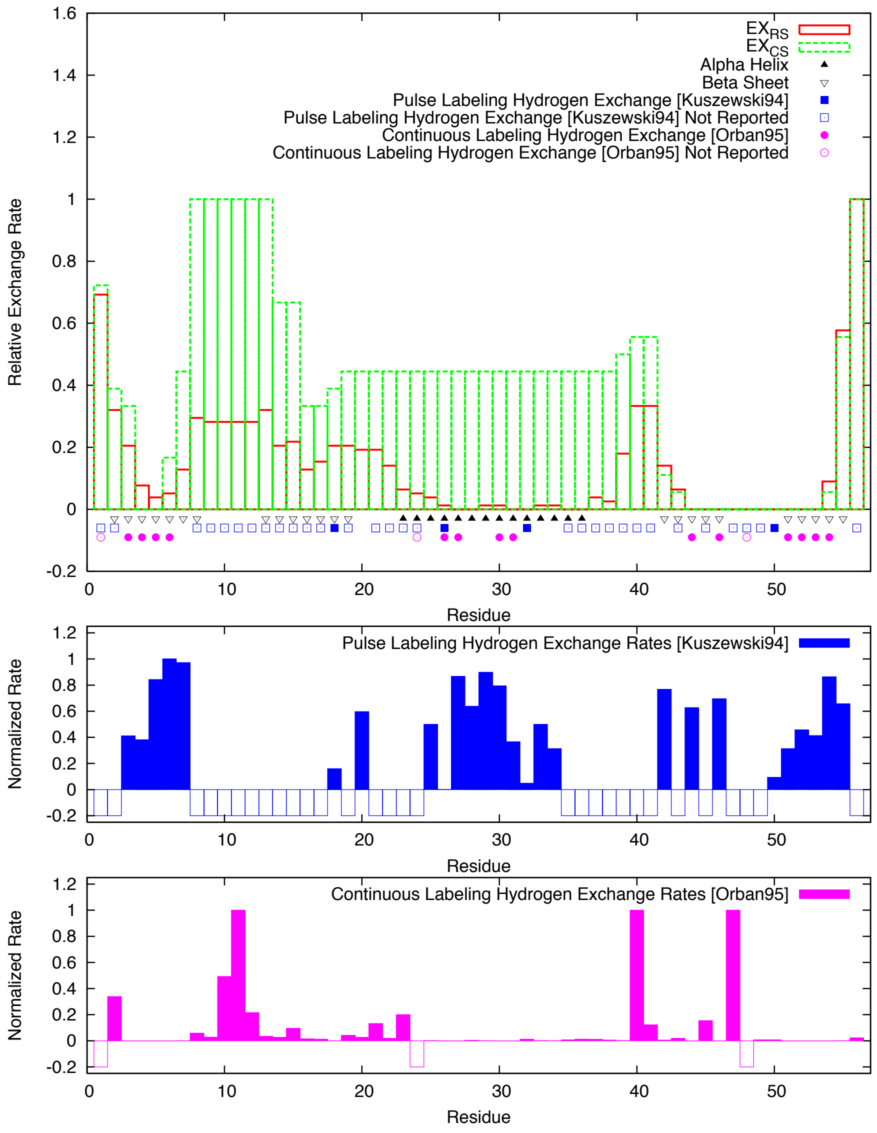
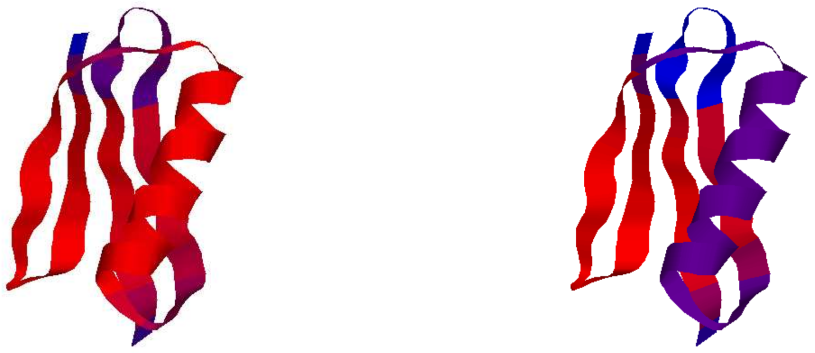


Fig. 47.: Corresponding relative exchange rates from the rigidity scores (red) and the cluster scores (green) for the example unfolding pathway for protein G. Normalized experimental data shown below for reference. In all plots, α-helices (filled triangles) and β-sheets (empty triangles) are indicated along the bottom for reference.



(a) (b)

Fig. 48.: Protein coloring based on relative exchange rates for the example unfolding pathway for protein G. Residues are shaded by relative exchange rate from fastest (blue) to slowest (red) based on (a) rigidity scores and (b) cluster scores.

them, we label them as *contact subsequences.* We then give all residues at that path-step belonging to a rigid cluster present in one of the contact subsequences a score of 1 and all other residues a score of 0. For a residue i, we define its cluster score, $CS(p, i)$, for a pathway p as the average of its cluster scores at each path-step:

$$CS(p,i) = \frac{1}{|p|} \sum_{c \in p} \begin{cases} 1 & \text{if residue } i \text{ at conformation } c \in \text{a rigid cluster from a} \\ & \text{contact subsequence} \\ 0 & \text{otherwise} \end{cases} \tag{5.4}$$

As in the definition of the relative exchange rate from the rigidity score, we can define a relative exchange rate $ex_{\text{CS}}(p, i)$ from the cluster score for a pathway p as

$$ex_{\text{CS}}(p,i) = 1 - \frac{CS(p,i) - CS_{min}(p)}{CS_{max}(p) - CS_{min}(p)} \tag{5.5}$$

where $CS_{min}(p)$ is the smallest $CS(p, i)$ obtained over p for all residues i and $CS_{max}(p)$ is the largest. With this definition, residues with large cluster scores will have low relative exchange rate cluster scores (likely to be in the core), and residues with small cluster scores will have high relative exchange rate cluster scores (unlikely to be in the core).

With the set of stochastic pathways provided by MMC, we can average cluster scores for an overall view across all pathways. For a set of MMC pathways, we define the average relative exchange rate, $EX_{\text{CS}}(i)$, for residue i as the average of the relative exchange rates, $ex_{\text{CS}}(p, i)$, for all pathways $p \in P$:

$$EX_{\text{CS}}(i) = \frac{1}{|P|} \sum_{p \in P} ex_{\text{CS}}(p,i) \tag{5.6}$$

Fig. 46(b) shows the cluster score along the same example pathway for protein G, and Fig. 47 and Fig. 48(b) shows the corresponding relative exchange rates from cluster scores. Again, β-hairpin 2 has a much lower relative exchange rate than

β-hairpin 1.

3. Related Computational Methods

A related computational approach to relative hydrogen exchange is the COREX/BEST algorithm [28, 29]. It is a statistical thermodynamic approach to studying the protein's equilibrium folding pathway. It partitions a protein conformation into a sequence of "folding units", typically 6–8 residues long. Each folding unit may be completely folded/native-like or unfolded/non-native-like. It then enumerates all such protein conformations and computes their relative free energy. With this relative free energy, they can assign probabilities that individual residues will be folded or unfolded for any degree of folding along the equilibrium folding pathway. These probabilities are used to calculate residue stabilities and can be related to hydrogen exchange protection factors.

4. Results

We studied several different proteins of varying size and structure, see Table XII. Fig. 49 compares the simulated exchange rates, using both rigidity scores and cluster scores, to available experimental data (see references in Table XII). The plots also label which residues for which experiment was unable to obtain exchange rates. For the normalized experimental rate data, unmeasured residues are indicated with empty bars at -0.2. While experiment is more limited to the residues it can study, our simulation can study all residues with known atom coordinates. Note that OMTKY3 and Protein A, RNase T1, and Barnase in Fig. 49 do not have values for simulated rates from cluster scores. This is because the native state for these proteins is not sufficiently rigid to obtain a cluster score of 1 for any residue. Of the 21 proteins studied, 4 fell into this category.

Table XII.: Proteins studied for relative hydrogen exchange and folding core identification. The % of a protein measured experimentally has an average of 46.34 and a standard deviation of 22.44. Experiments measuring less than one standard deviation below the average are indicated in boldface.

Protein	**PDB**	**Len.**	**Sec. Struct.**	**Experimental Data (% Measured)**
Ovomucoid third domain (OMTKY3)	1IY5	54	$1\alpha + 4\beta$	[150] (**22.22%**)
B1 domain of protein G (Protein G)	1PGA	56	$1\alpha + 4\beta$	[122] (46.43%), [123] (91.07%)
Chicken src SH3 domain (cSH3)	1SRM	56	$1\alpha + 3\beta$	[151] (42.86%)
Bovine pancreatic trypsin inhibitor (BPTI)	1BPI	58	$2\alpha + 2\beta$	[127] (75.86%), [126] (**15.52%**), [128] (**13.79%**)
Protein A, B domain (Protein A)	1BDD	60	3α	[131] (46.67%, 33.33%)
Cardiotoxin analogue III (CTXIII)	2CRS	60	6β	[152] (50.00%)
B1 domain of protein L (Protein L)	2PTL	62	$1\alpha + 4\beta$	[134] (53.23%), [135] (85.48%)
Chymotrypsin inhibitor 2 (CI2)	2CI2	65	$2\alpha + 5\beta$	[138], (47.69%), [139] (41.54%)
Tendamistat	2AIT	74	7β	[153] (54.05%),

Table XII Continued

Protein	**PDB**	**Len.**	**Sec. Struct.**	**Experimental Data (% Measured)**
				[154] (67.57%)
Ubiquitin	1UBI	76	$3\alpha + 5\beta$	[141] (53.95%)
Pseudomonas aeruginosa cytochrome c_{551} (Pa cyt c_{551})	351C	82	5α	[155] (40.24%)
Ribonuclease T1 (RNase T1)	1BU4	104	$1\alpha + 7\beta$	[156] (**23.08%**), [157] (34.62%)
Barnase	1A2P	108	$4\alpha + 6\beta$	[158] (**23.15%**), [159] (35.19%)
Saccharomyces cerevisiae iso-1-cytochrome c (y-cyt c)	2YCC	108	5α	[160] (83.33%), [161] (80.56%)
α-Lactalbumin	1HML	123	$9\alpha + 3\beta$	[162] (36.59%)
Ribonuclease A (RNase A)	1RBX	124	$4\alpha + 7\beta$	[163] (**21.77%**), [164] (37.10%)
CheY	3CHY	128	$5\alpha + 5\beta$	[165] (28.91%)
Equine lysozyme (Lysozyme)	2EQL	129	$8\alpha + 5\beta$	[166] (51.94%), [167] (51.94%)
Human acidic fibroblast growth factor-1 (hFGF-1)	2AFG	129	$4\alpha + 10\beta$	[168] (72.09%)
Apo-Myoglobin (ApoMb)	1A6M	151	10α	[169] (25.17%), [170] (25.17%)
Ribonuclease H (RNase H)	2RN2	155	$5\alpha + 5\beta$	[171] (90.32%),

Table XII Continued

Protein	**PDB**	**Len.**	**Sec. Struct.**	**Experimental Data (% Measured)**
				[172] (**19.35%**)

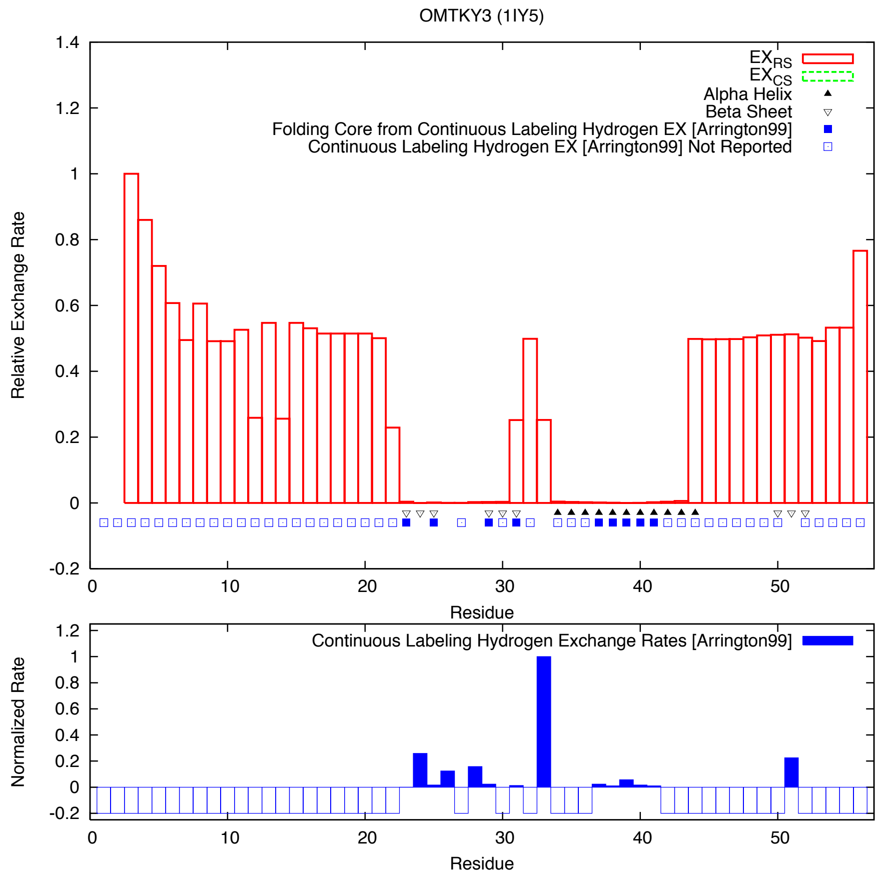


Fig. 49.: Comparison of simulated exchange rates from rigidity scores (red) and cluster scores (green) to experimental data (open/filled squares and circles). α-helices (filled triangles) and β-sheets (empty triangles) are indicated along the bottom for reference. When available, the normalized experimental data is plotted below the simulated exchange rates plot. Unmeasured residues are indicated with empty bars at -0.2. Astricks in the experimental data plots indicate that the authors reported these residues as “fast exchanging” or “slow exchanging” instead of providing a rate.

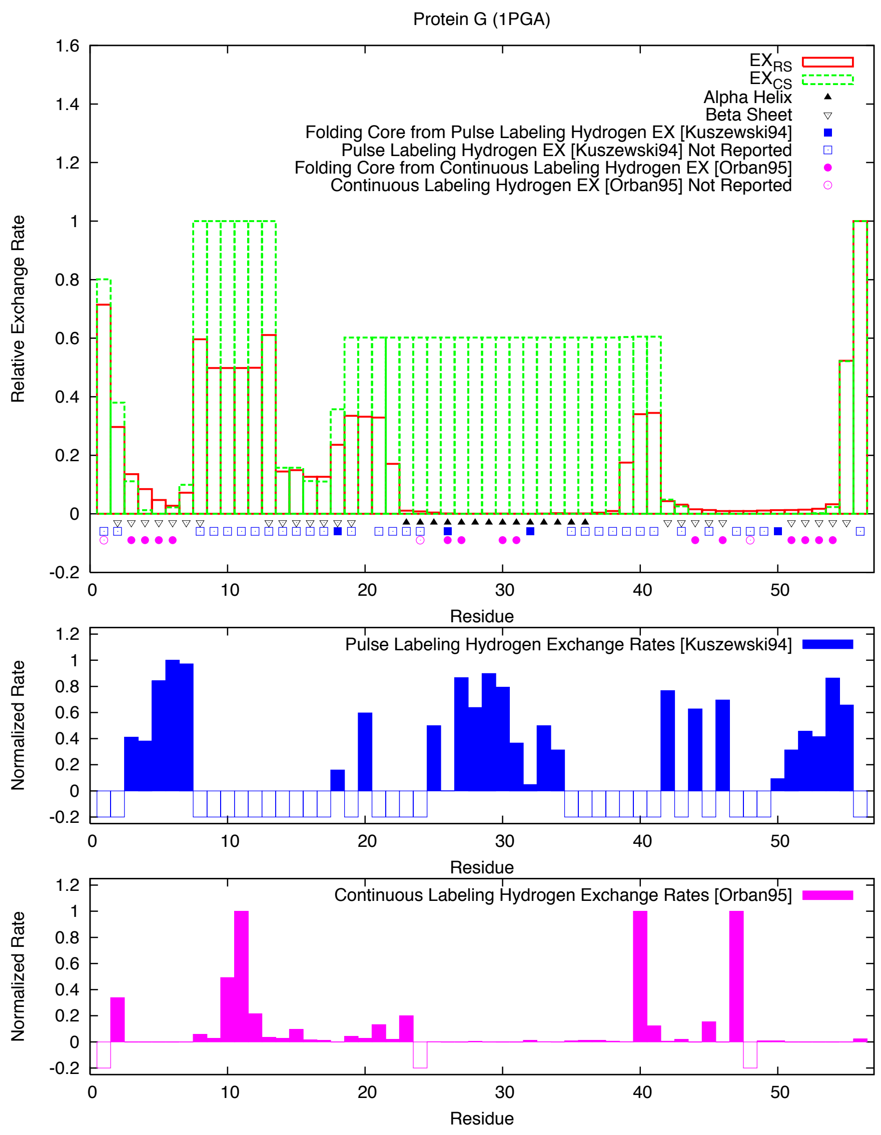


Fig. 49 Continued

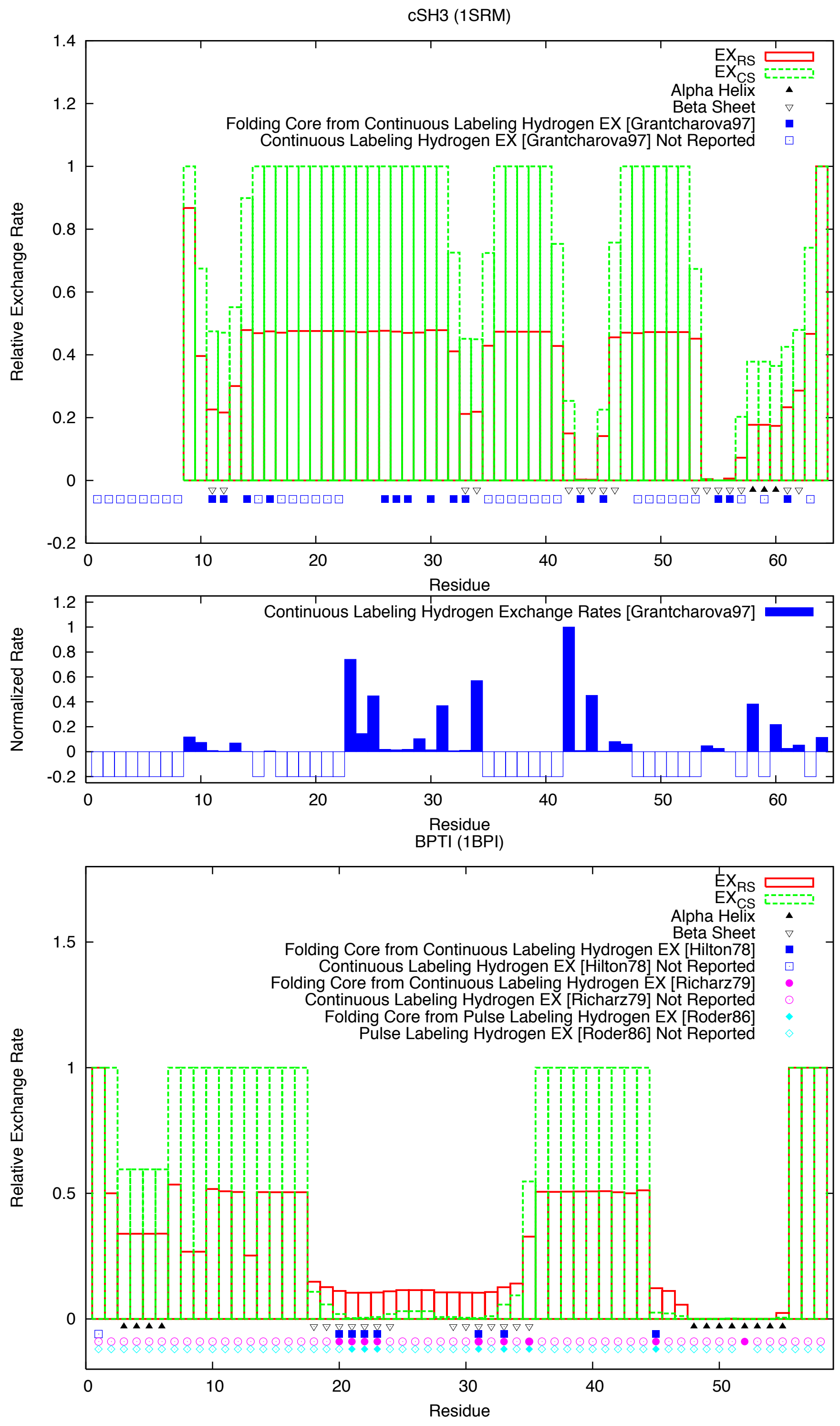


Fig. 49 Continued

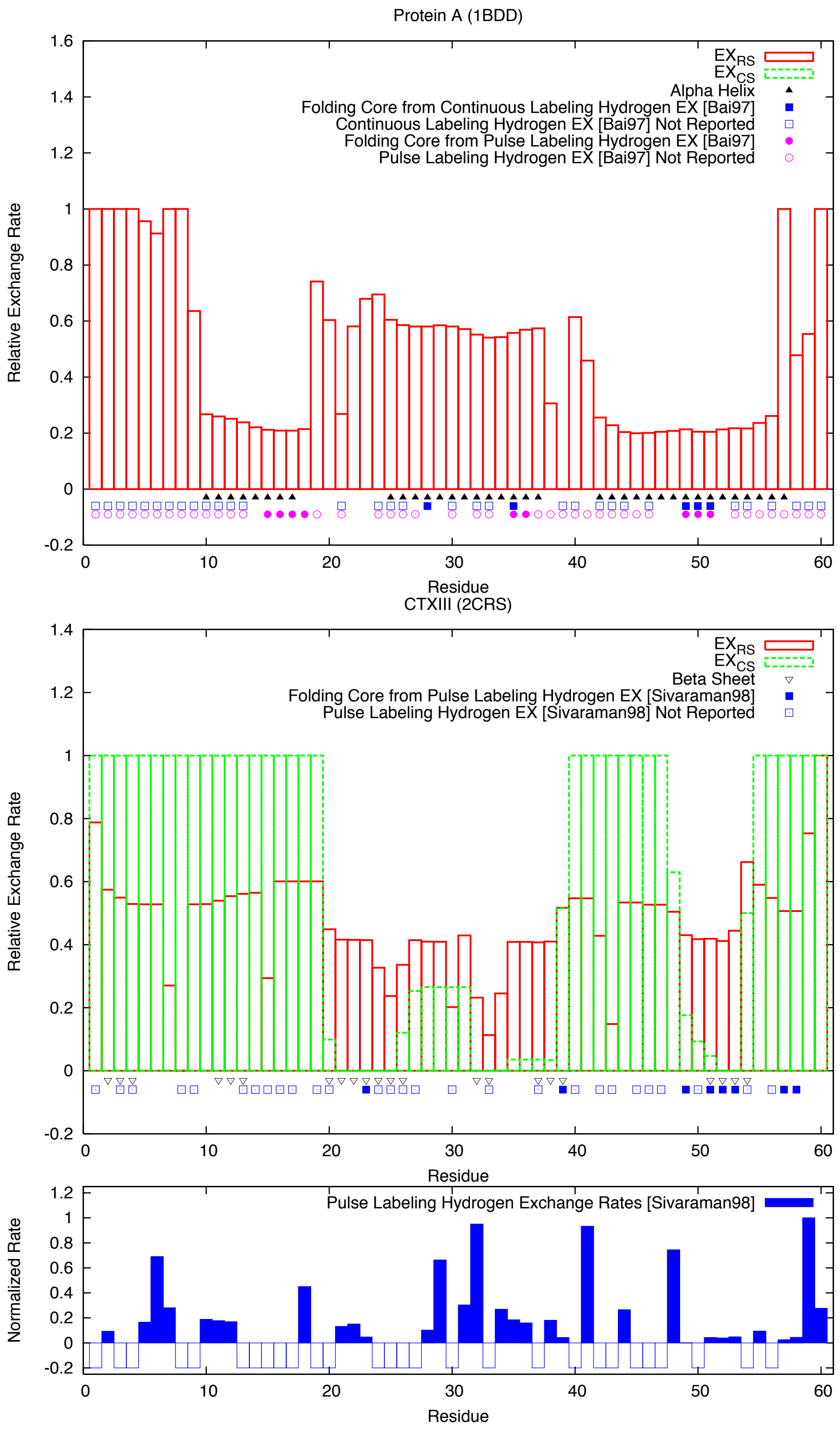


Fig. 49 Continued

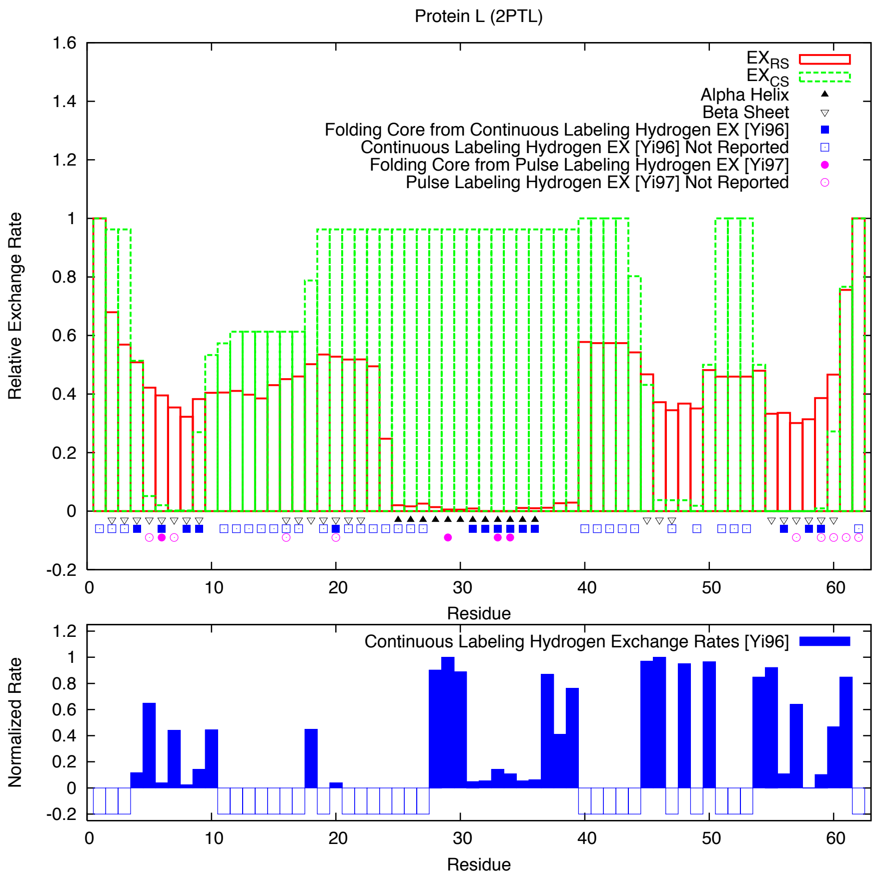


Fig. 49 Continued

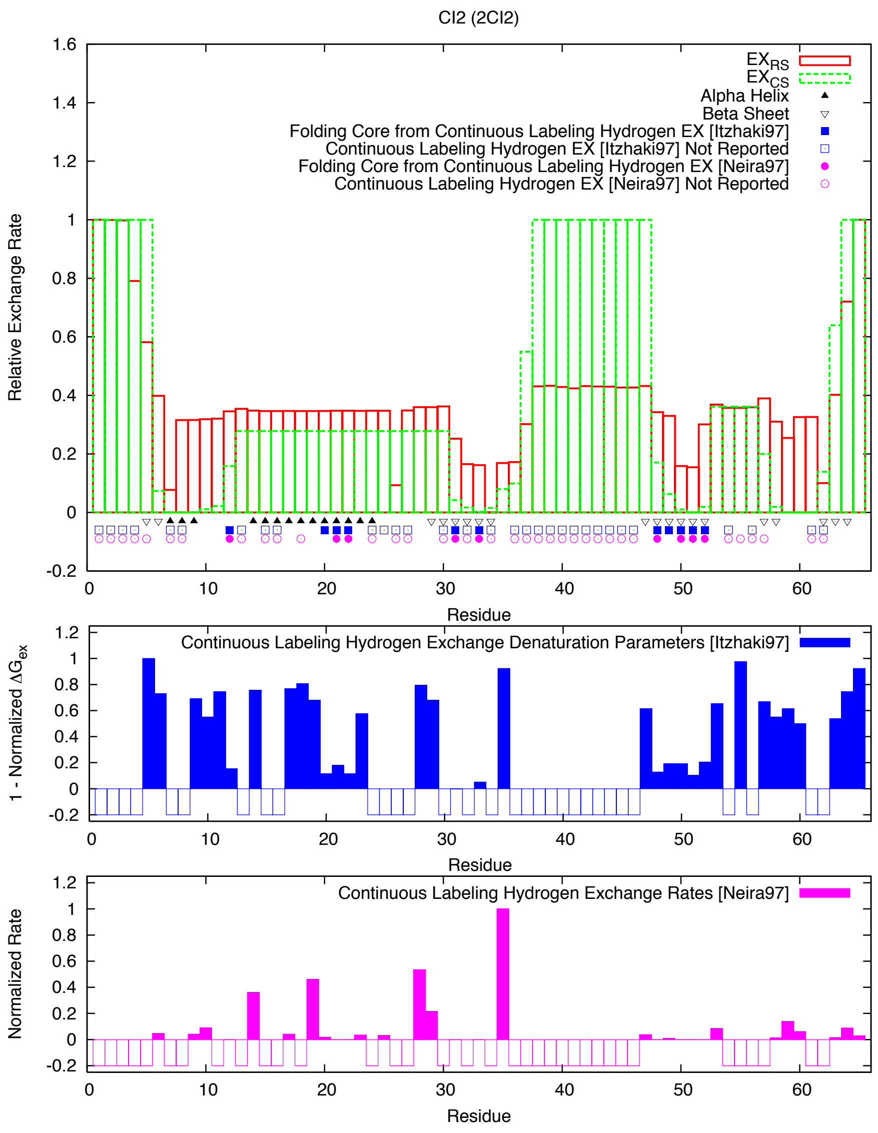


Fig. 49 Continued

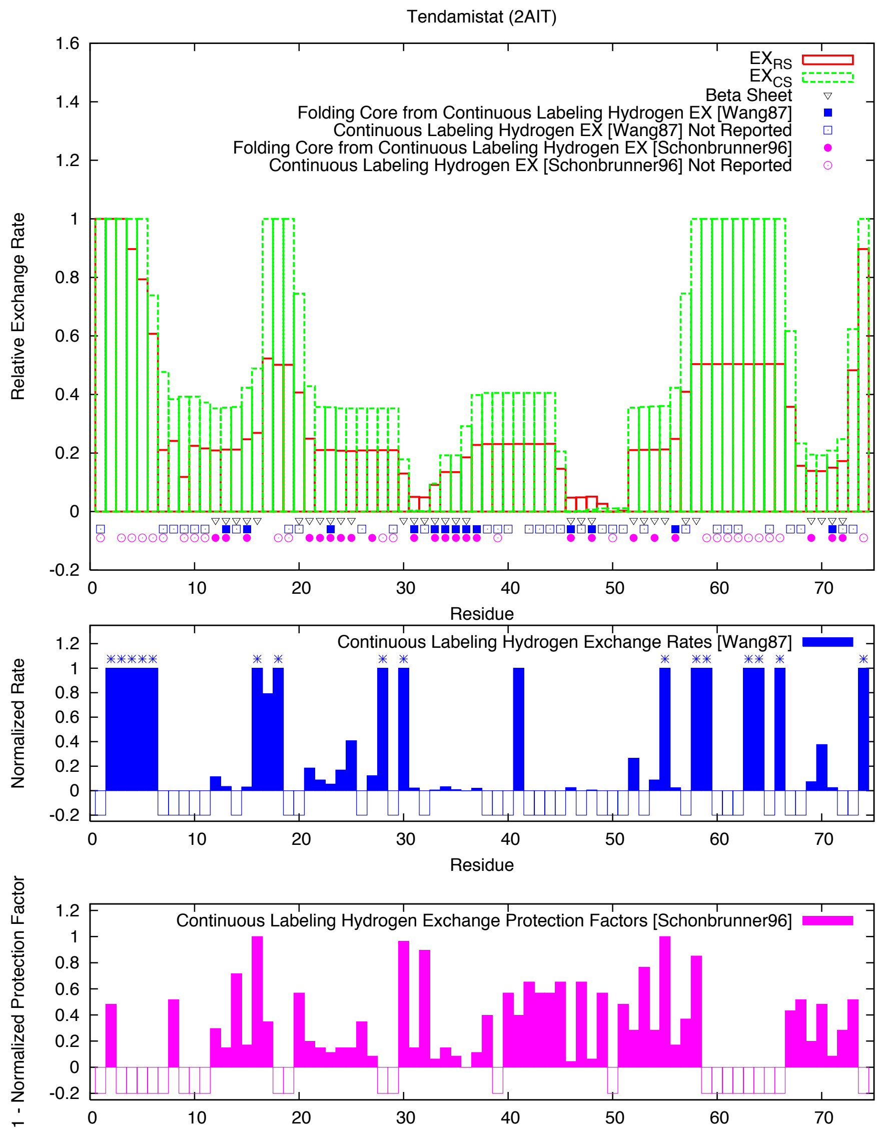


Fig. 49 Continued

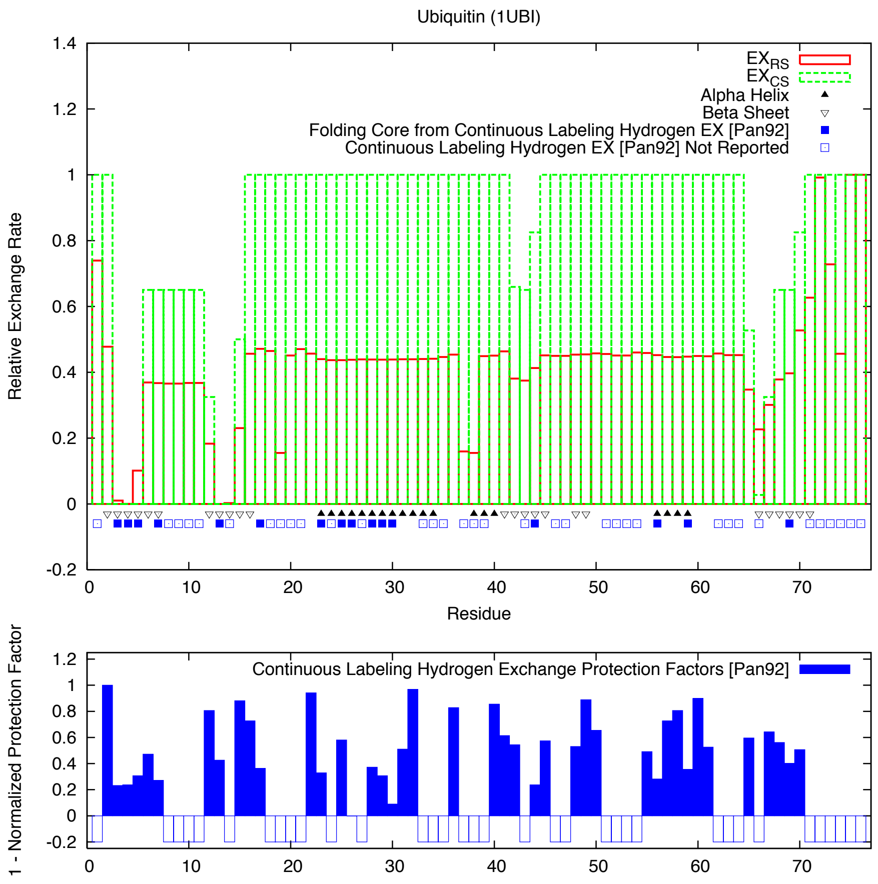


Fig. 49 Continued

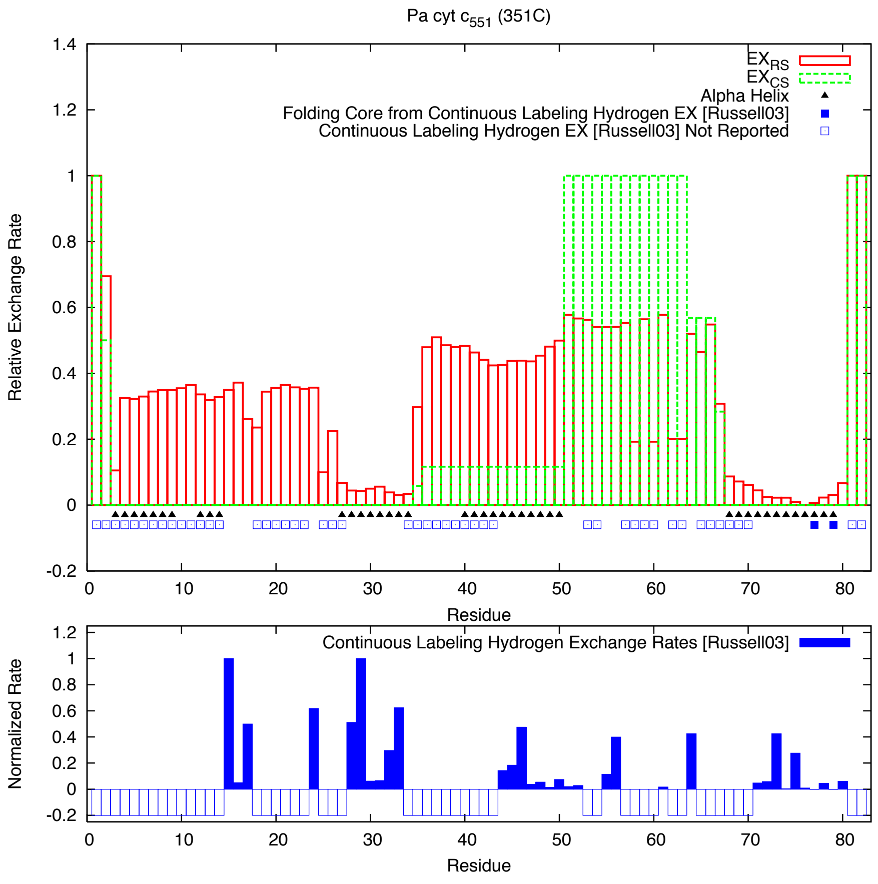


Fig. 49 Continued

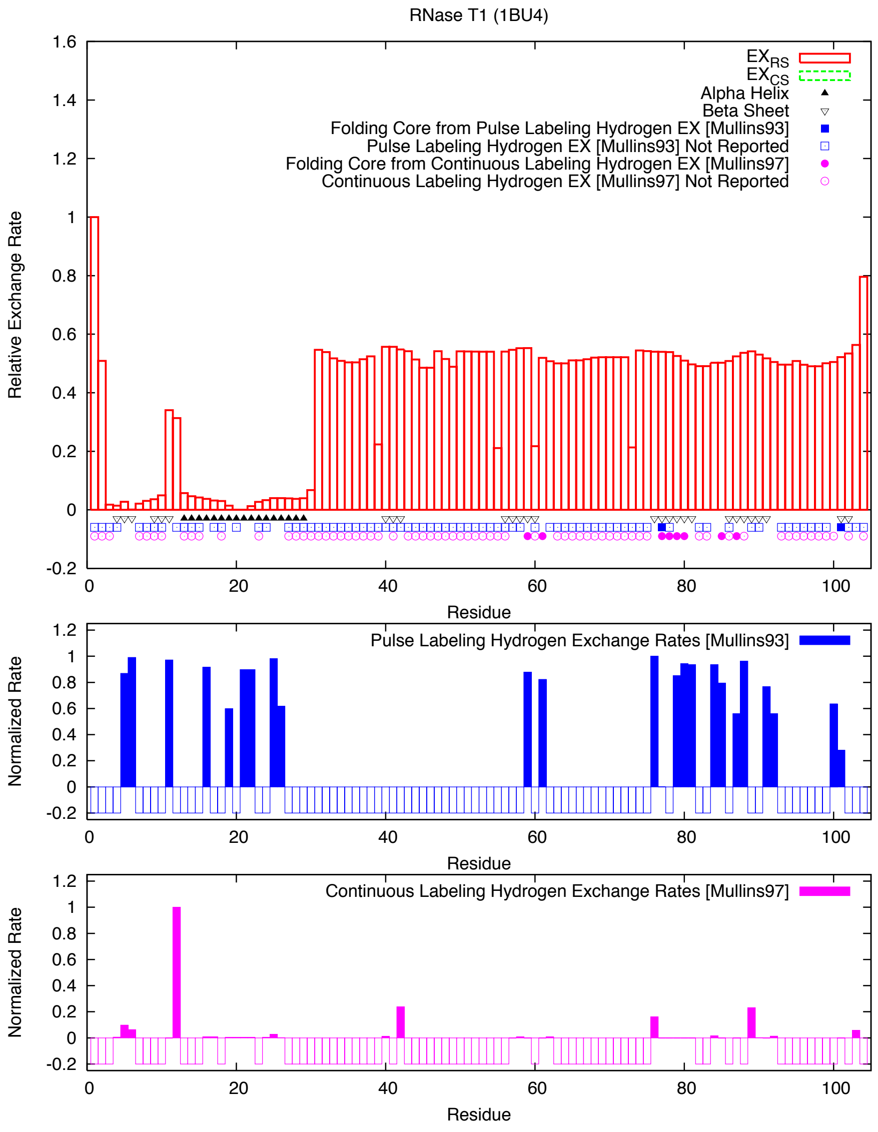


Fig. 49 Continued

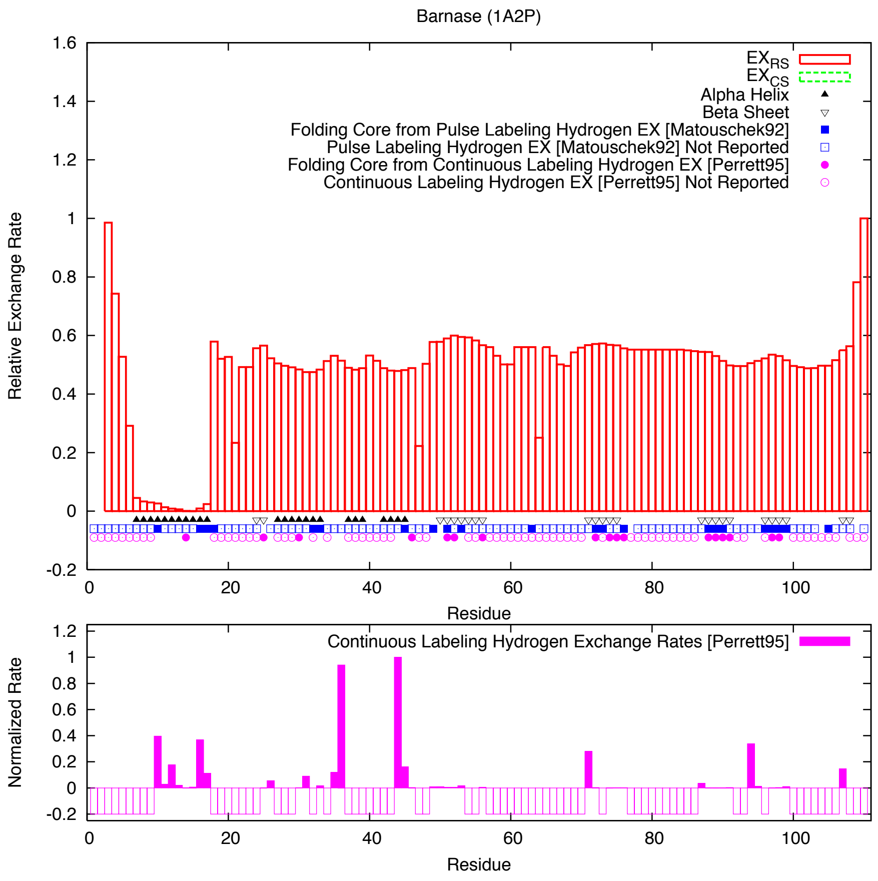


Fig. 49 Continued

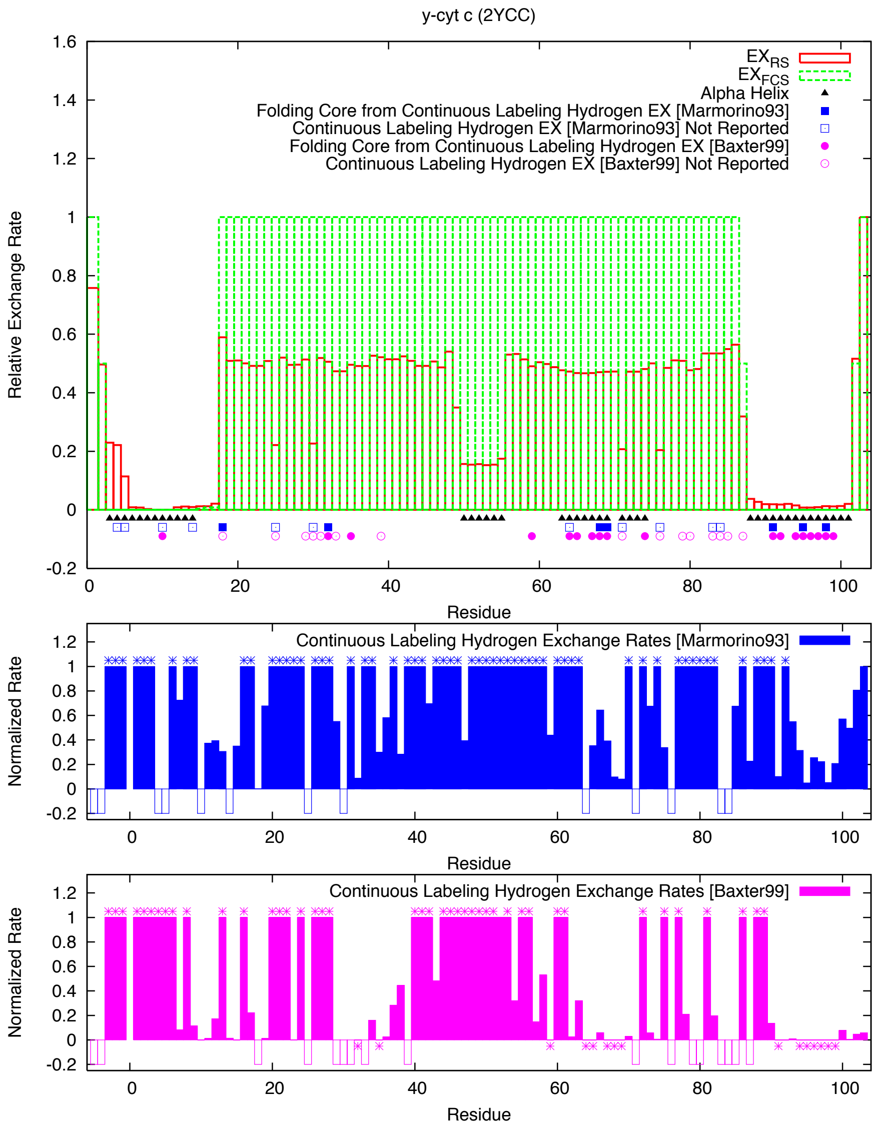


Fig. 49 Continued

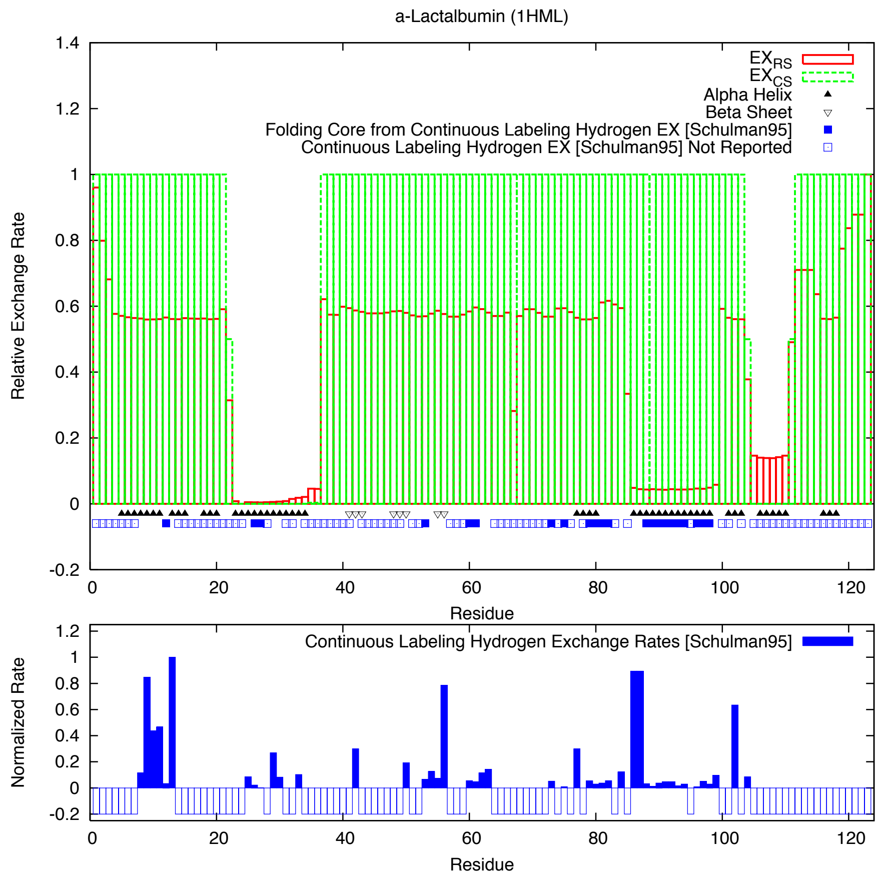


Fig. 49 Continued

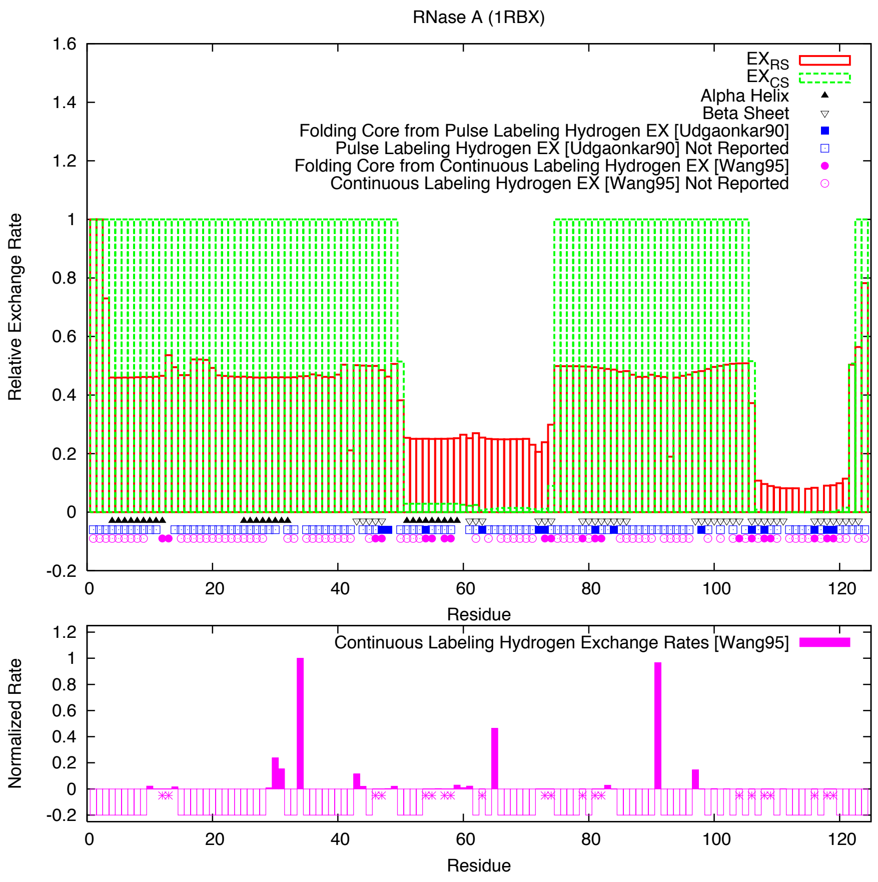


Fig. 49 Continued

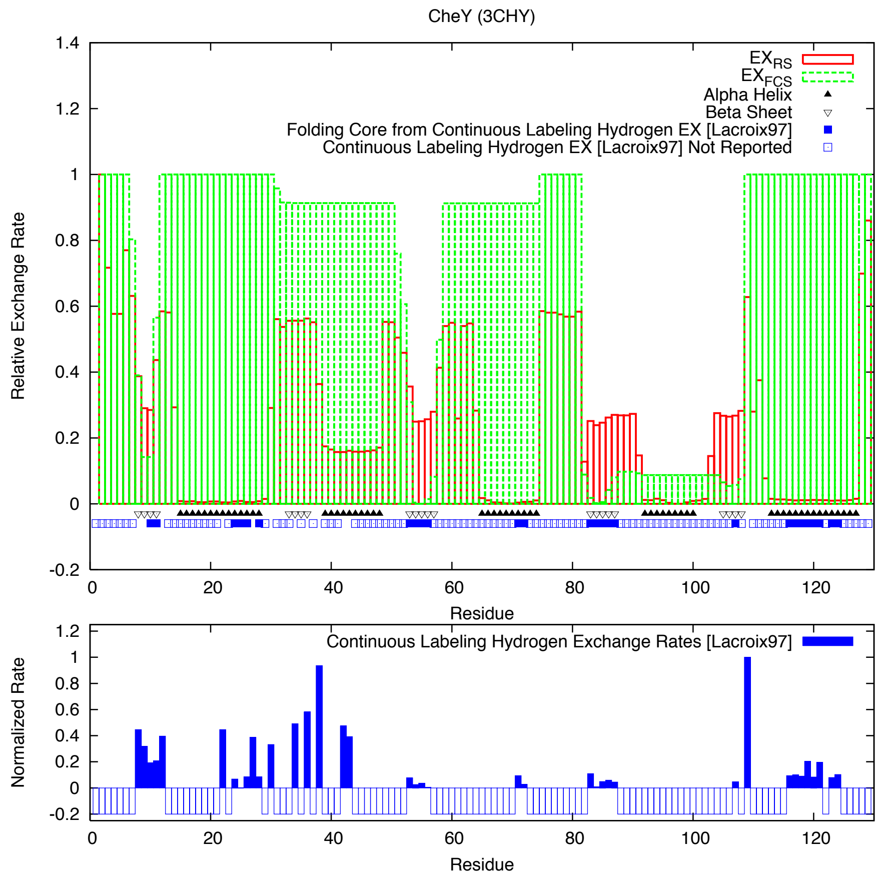
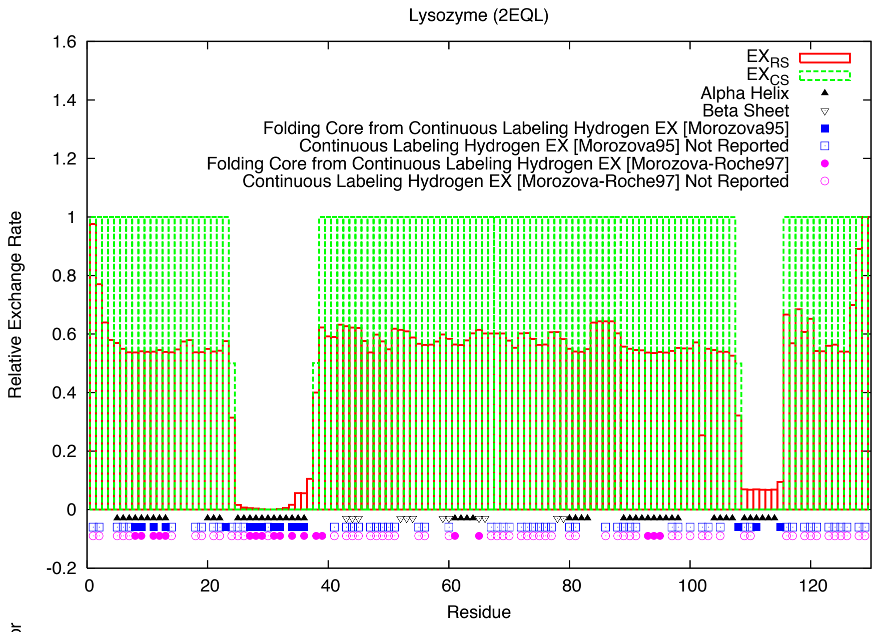
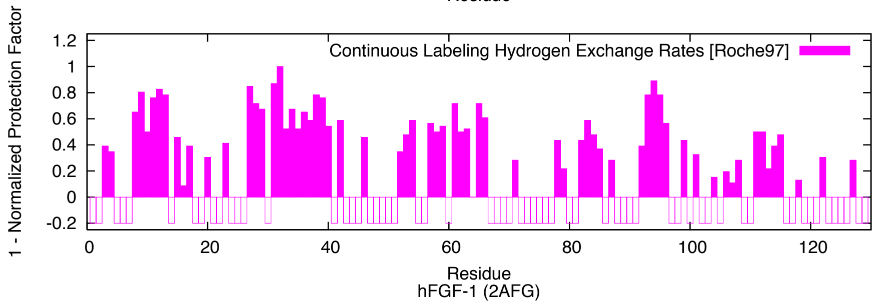


Fig. 49 Continued





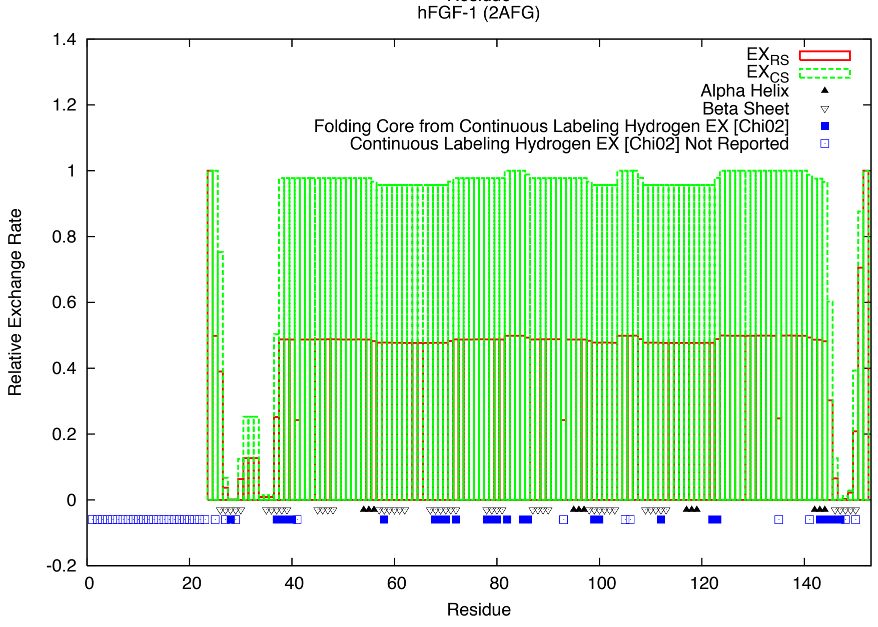
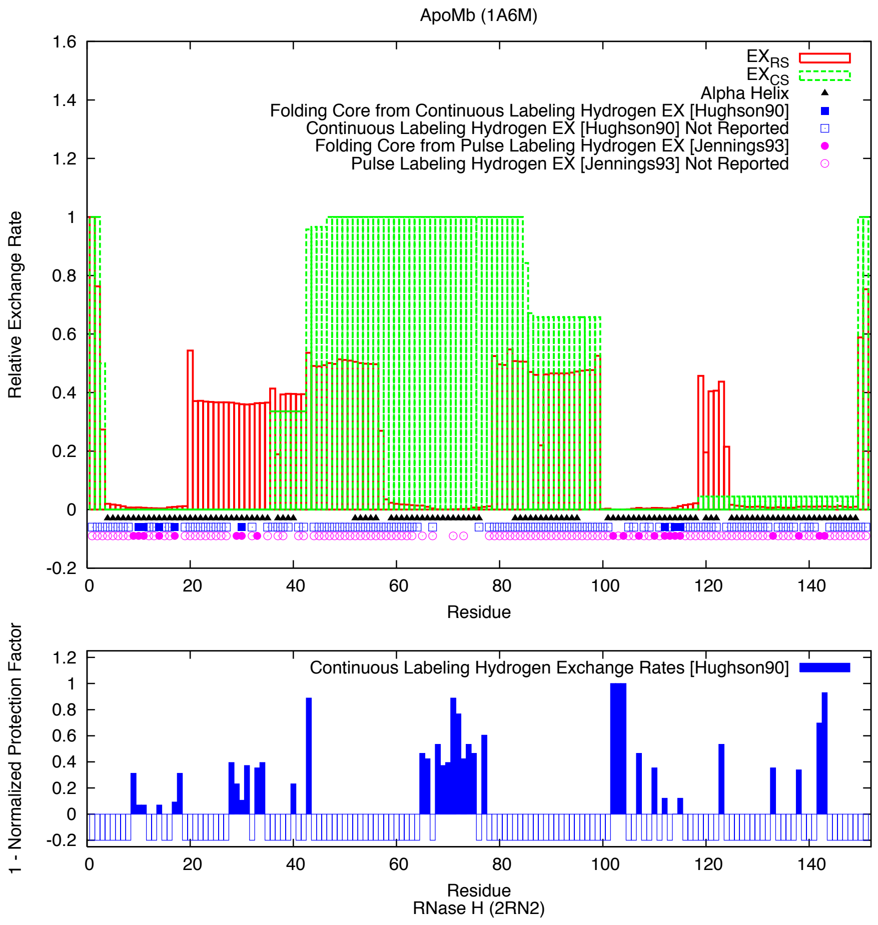


Fig. 49 Continued



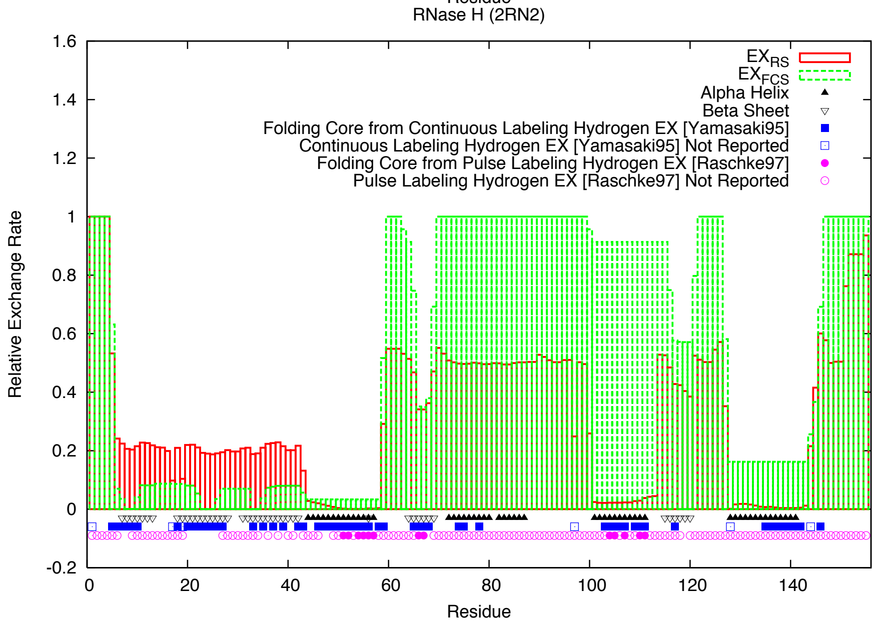


Fig. 49 Continued

The rigidity score exchange rates are more sensitive than the cluster score exchange rates to changes in rigidity. This is due to the nature of the scores: a rigidity score may be one of 3 values (i.e., 0, 0.5, or 1) while a cluster score is limited to 2 (i.e., 0 or 1). However, both simulated exchange rate curves show similar overall shapes with peaks and valleys aligned over many of the residues. In addition, there is good correlation between experimental rates and simulated exchange rates. For example. the peaks and valleys of OMTKY3 and Tendamistat simulated rates match the peaks and valleys of the experimental data.

While there is some correspondence between simulated exchange rate and secondary structure membership, there is not a direct mapping. First, not every secondary structure element has a low simulated exchange rate, e.g., $\beta3$ in OMTKY3, $\alpha1$ in BPTI, $\alpha2$ in Protein A, $\beta1$ and $\beta2$ in CTXIII, and $\beta6$ in Tendamistat. Secondly, low simulated exchange rates are not limited to secondary structure elements. Residues 26–28 in OMTKY3, 25–28 in BPTI, 39 in Protein A, 47–50 in protein G, and 49–51 in Tendamistat all have low simulated exchange rates but are not part of an α-helix or β-sheet. Finally, simulated exchange rates are not always constant across a secondary structure element, but may vary as in $\beta1$ from protein G and $\beta3$ from Tendamistat.

Fig. 50 provides a visual comparison of simulated exchange rates to available experimental data on the 3D structure. Note that the gray residues in the experimental data are not necessarily outside the folding core but may not have been measured. A strength of our simulation is that we can compute relative exchange rates for every residue in the structure while experiments are limited to which residues they can accurately probe.

There is a strong correlation between residues labeled as part of the folding core by experimental data and the slowest exchange residues from simulation (red).

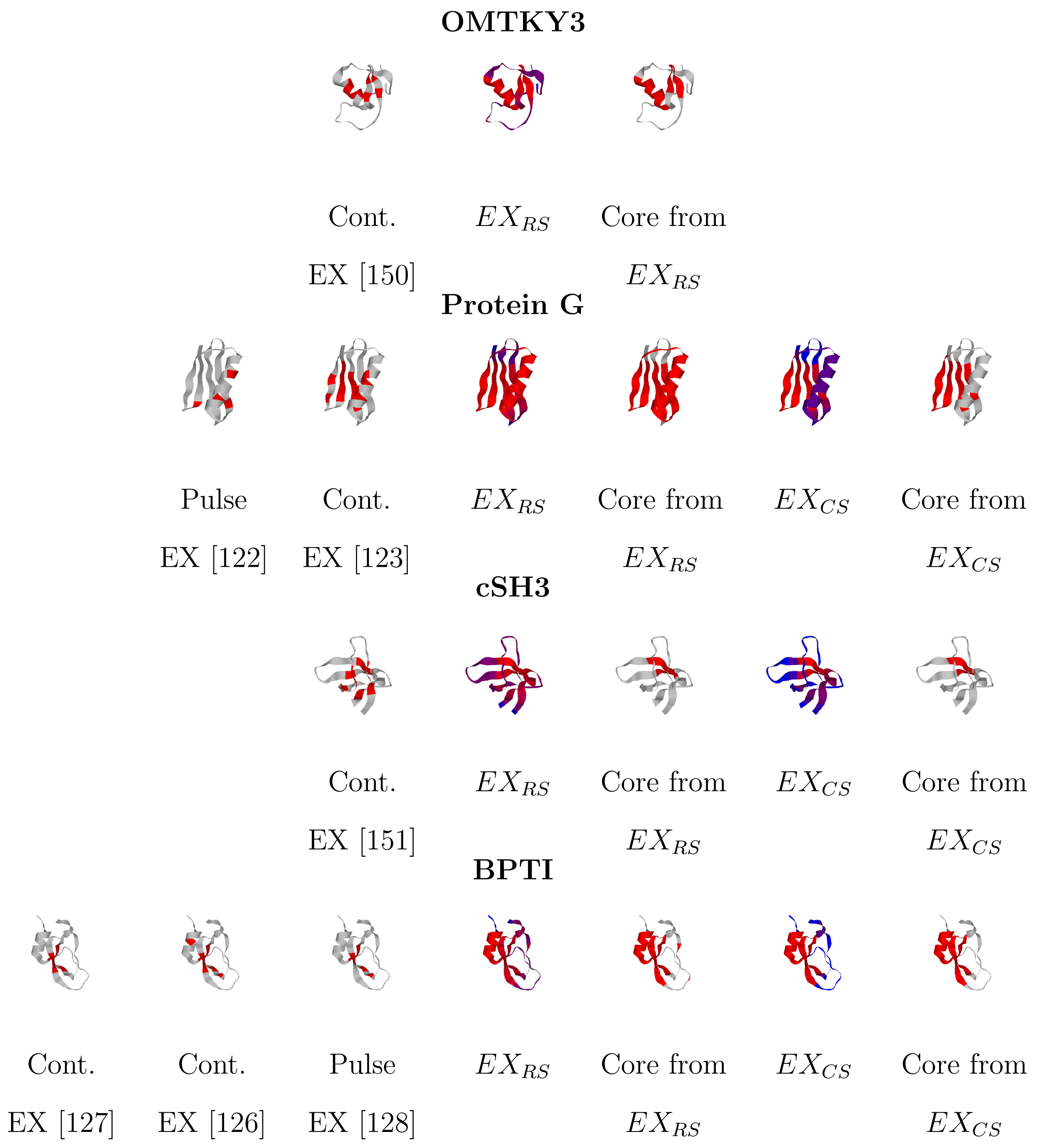
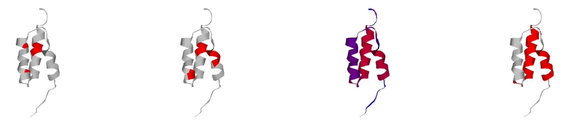


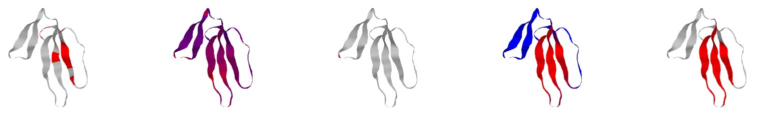
Fig. 50.: Visual comparison of simulated exchange rates to experimental data. Residues identified in the folding core are shaded red. For experimental data, grey residues were either not identified as the folding core or not measured. For simulated exchange rate data, residues are shaded by exchange rate from fastest/blue to slowest/red.

Protein A



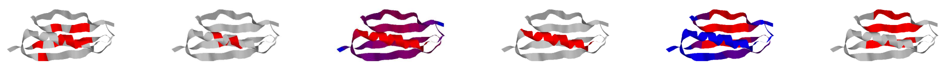
Cont. EX [131]	Pulse EX [131]	EX_{RS}	Core from EX_{RS}

CTXIII



Pulse EX [152]	EX_{RS}	Core from EX_{RS}	EX_{CS}	Core from EX_{CS}

Protein L

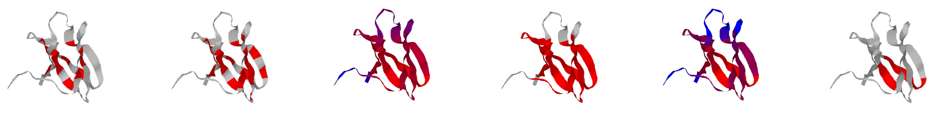


Cont. EX [134]	Pulse EX [135]	EX_{RS}	Core from EX_{RS}	EX_{CS}	Core from EX_{CS}

CI2

Cont. EX [138]	Cont. EX [139]	EX_{RS}	Core from EX_{RS}	EX_{CS}	Core from EX_{CS}

Tendamistat



Cont. EX [153]	Cont. EX [154]	EX_{RS}	Core from EX_{RS}	EX_{CS}	Core from EX_{CS}

Fig. 50 Continued

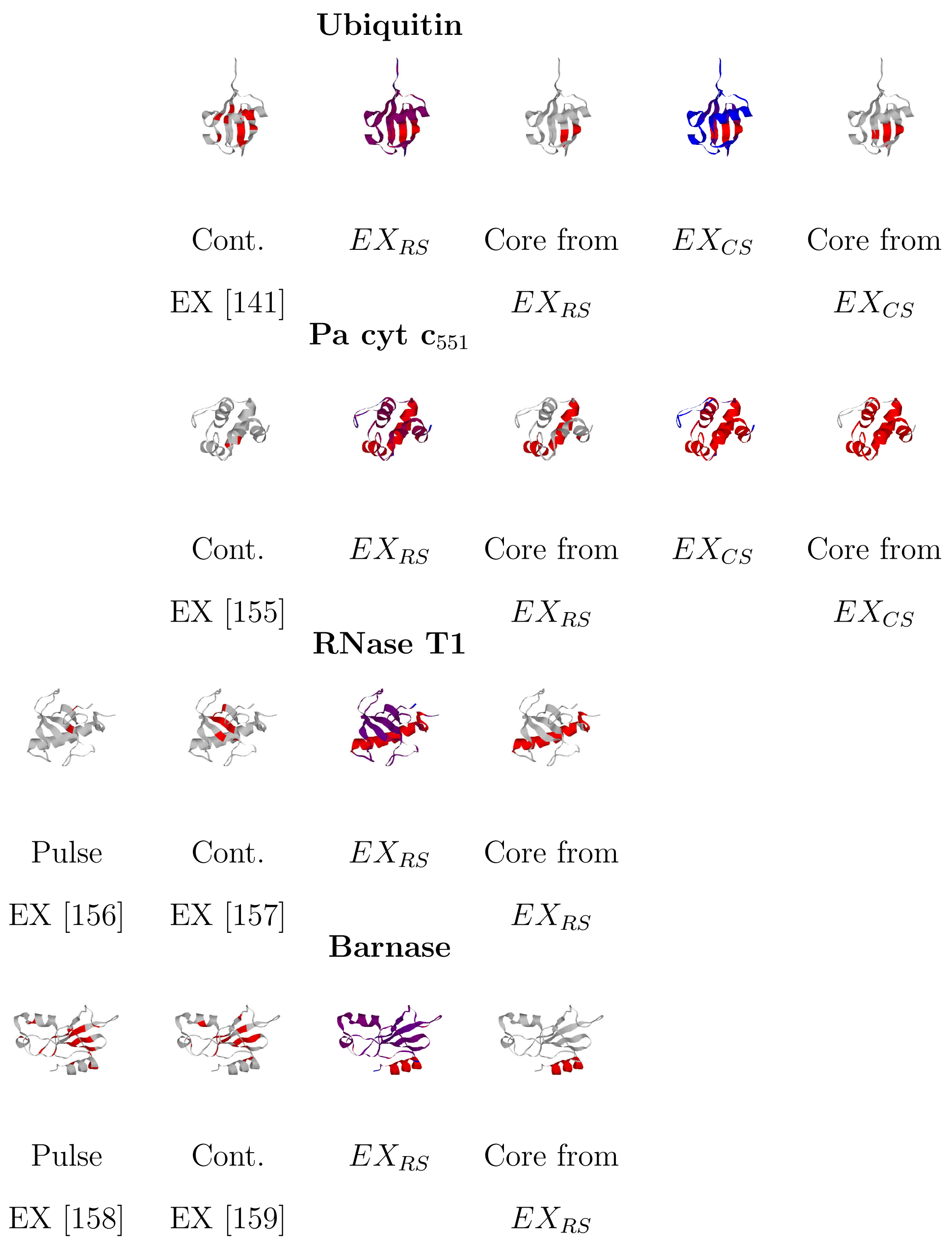


Fig. 50 Continued

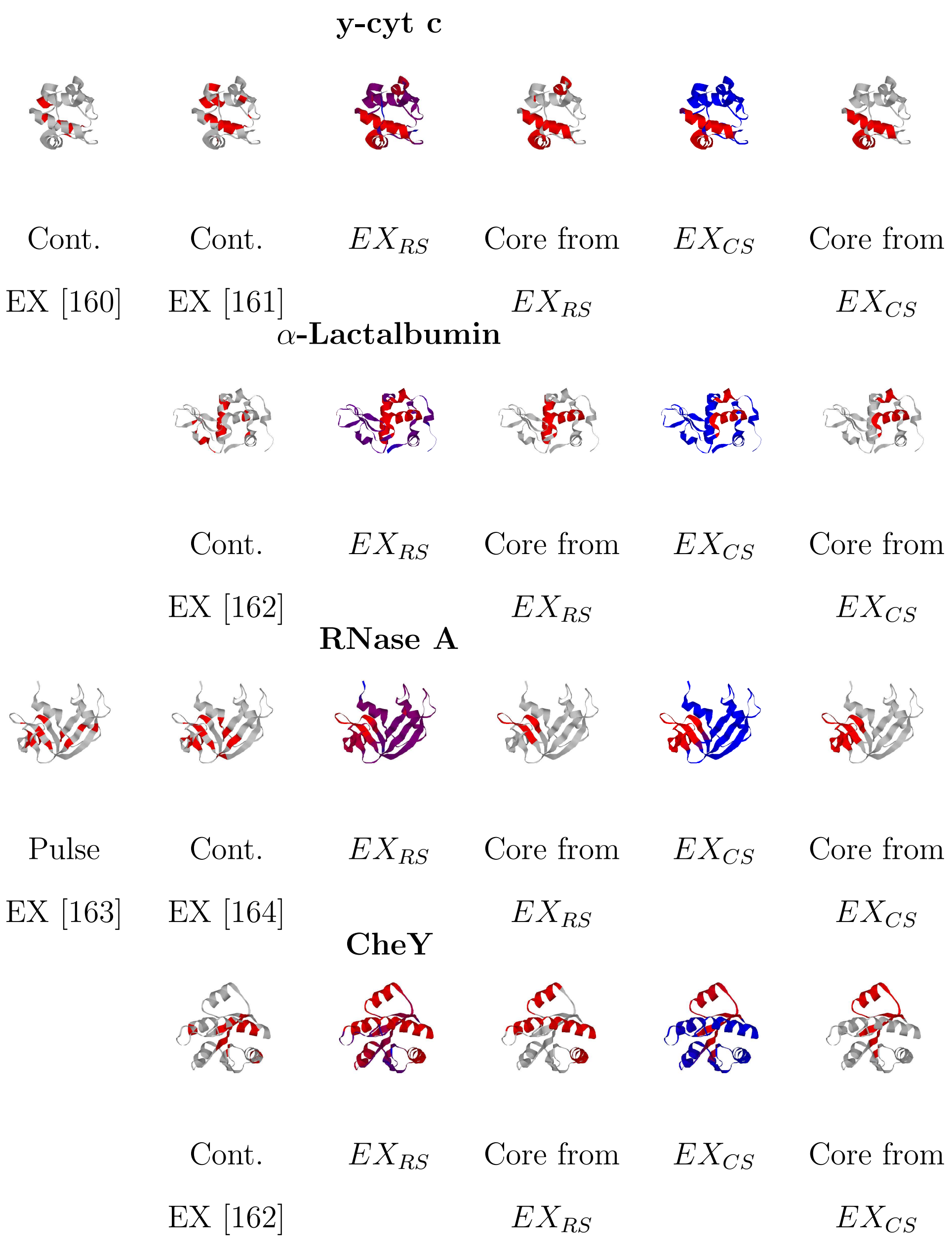
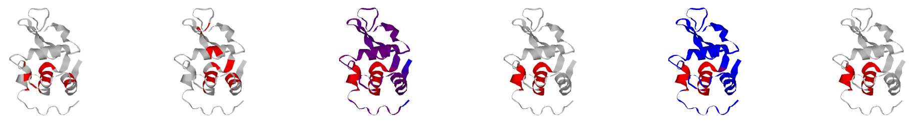


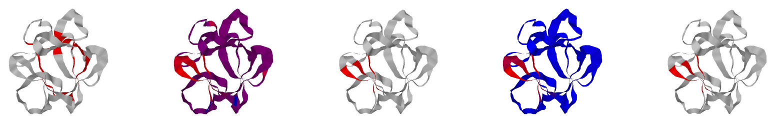
Fig. 50 Continued

Lysozyme



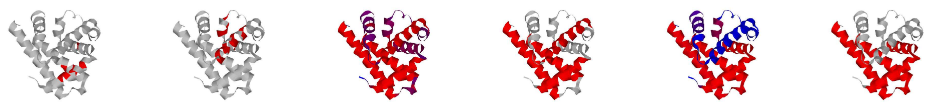
Cont. EX [166]
Cont. EX [167]
EX_{RS}
Core from EX_{RS}
EX_{CS}
Core from EX_{CS}

hFGF-1



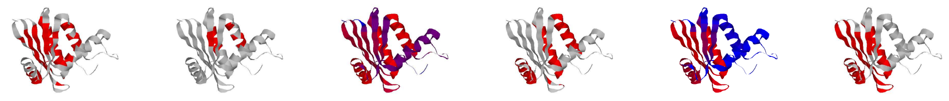
Cont. EX [168]
EX_{RS}
Core from EX_{RS}
EX_{CS}
Core from EX_{CS}

ApoMb



Cont. EX [169]
Pulse EX [170]
EX_{RS}
Core from EX_{RS}
EX_{CS}
Core from EX_{CS}

RNase H



Cont. EX [171]
Pulse EX [172]
EX_{RS}
Core from EX_{RS}
EX_{CS}
Core from EX_{CS}

Fig. 50 Continued

These slowest exchanging residues tend to cluster together into one group on the 3D structure which is not apparent by looking at their placement along the sequence alone. Conversely, the fastest exchanging residues (blue) tend to fall on the outer surface of the 3D structure. Recall that residue placement on the 3D structure (i.e., buried or exposed) is not part of either simulated exchange rate model, yet we see a correlation between tertiary placement and simulated exchange rate as is also observed in experimental data. Thus, rigidity analysis, either through the rigidity score or the cluster score, can approximate the likelihood of exchange without an explicit solvent model.

Table XIII reports the average relative exchange rate differences between the overall protein and the experimentally defined folding core (i.e., slowest exchanging residues) for both rigidity scores and cluster scores. For both scores, there is a dramatic decrease in simulated exchange rate when looking at the folding core versus the entire protein. On average, there is a 36.7% drop in EX_{RS} and a 43.3% drop in EX_{CS}.

Table XIII.: Comparison of average relative exchange rate for both EX_{RS} and EX_{CS} between the entire protein and the experimentally defined folding core (i.e., the slowest exchanging residues). Protein name abbreviations are given in Table XII.

		Average EX_{RS}			**Average EX_{CS}**		
Protein	**Exp. Data**	**All**	**Folding Core**	**% Diff.**	**All**	**Folding Core**	**% Diff.**
OMTKY3	[150]	0.3448	0.0295	-91.4%	n/a	n/a	n/a
Protein G	[122]	0.1641	0.0625	-61.9%	0.4250	0.3904	-8.1%

Table XIII Continued

Protein	**Exp. Data**	**Average EX_{RS}** All	**Average EX_{RS}** Folding Core	**Average EX_{RS}** % Diff.	**Average EX_{CS}** All	**Average EX_{CS}** Folding Core	**Average EX_{CS}** % Diff.
	[123]	0.1641	0.0286	-82.6%	0.4250	0.1843	-56.6%
cSH3	[151]	0.3793	0.2867	-24.4%	0.7332	0.5782	-21.1%
BPTI	[127]	0.3082	0.1112	-63.9%	0.4912	0.0173	-96.5%
	[126]	0.3082	0.1229	-60.1%	0.4912	0.0742	-84.9%
	[128]	0.3082	0.1421	-53.9%	0.4912	0.0926	-81.1%
Protein A	[131]	0.4876	0.3594	-26.3%	n/a	n/a	n/a
	[131]	0.4876	0.4603	-5.6%	n/a	n/a	n/a
CTXIII	[152]	0.4654	0.4562	-2.0%	0.6106	0.3424	-43.9%
Protein L	[134]	0.3626	0.2288	-36.9%	0.6609	0.5394	-18.4%
	[135]	0.3626	0.1003	-72.3%	0.6609	0.7269	10.0%
CI2	[138]	0.3782	0.3412	-9.8%	0.3967	0.1960	-50.6%
	[139]	0.3782	0.3408	-9.9%	0.3967	0.1918	-51.7%
Tendamistat	[153]	0.3079	0.1531	-50.3%	0.4878	0.2267	-53.5%
	[154]	0.3079	0.1746	-43.3%	0.4878	0.2739	-43.8%
Ubiquitin	[141]	0.4199	0.3237	-22.9%	0.8403	0.6750	-19.7%
Pa cyt c_{551}	[155]	0.3282	0.0185	-94.4%	0.2475	0.0000	-100.0%
RNase T1	[156]	0.3900	0.5308	36.1%	n/a	n/a	n/a
	[157]	0.3900	0.5266	35.0%	n/a	n/a	n/a
Barnase	[158]	0.4793	0.4675	-2.5%	n/a	n/a	n/a
	[159]	0.4793	0.5121	6.8%	n/a	n/a	n/a
y-cyt c	[160]	0.3717	0.2966	-20.2%	0.7178	0.5714	-20.4%

Table XIII Continued

Protein	**Exp. Data**	**Average EX_{RS}** All	**Average EX_{RS}** Folding Core	**Average EX_{RS}** % Diff.	**Average EX_{CS}** All	**Average EX_{CS}** Folding Core	**Average EX_{CS}** % Diff.
	[161]	0.3717	0.2457	-33.9%	0.7178	0.5000	-30.3%
α-Lactalbumin	[162]	0.4435	0.2857	-35.6%	0.8253	0.9091	10.2%
RNase A	[163]	0.3945	0.2926	-25.8%	0.6784	0.3976	-41.4%
	[164]	0.3945	0.3142	-20.4%	0.6784	0.4162	-38.6%
CheY	[165]	0.2433	0.1345	-44.7%	0.7128	0.5794	-18.7%
Lysozyme	[166]	0.4950	0.2099	-57.6%	0.8333	0.3437	-58.8%
	[167]	0.4950	0.3472	-29.9%	0.8333	0.6053	-27.4%
hFGF-1	[168]	0.4366	0.4155	-4.8%	0.8661	0.8322	-3.9%
ApoMb	[169]	0.2124	0.0511	-75.9%	0.3983	0.0000	-100.0%
	[170]	0.2124	0.0598	-71.8%	0.3983	0.0089	-97.8%
RNase H	[171]	0.3049	0.1572	-48.4%	0.5307	0.2905	-45.3%
	[172]	0.3049	0.0635	-79.2%	0.5307	0.4207	-20.7%
Average		*0.3623*	*0.2472*	*-36.7%*	*0.5917*	*0.3709*	*-43.3%*

Except for the central α-helix in protein G, the fastest and slowest exchanging residues match between the simulated exchange rates from rigidity scores and from cluster scores. In protein G, the high rigidity scores indicate that this central helix forms rigid structure early and maintains it for a large part of the folding process.

However, recall from Chapter V, Section B.2 that in order to have a high cluster score, the helix must form and maintain contacts to other parts of the structure outside its subsequence. Thus, the cluster score for the helix is low in these two proteins yielding higher simulated exchange rates. The rigidity score model may be more generally applicable by recognizing that exchange rates in some proteins may be more influenced by local rigidity/flexibility than by specific tertiary contact placement.

Finally, we also compared the ability of our method and COREX/BEST to identify the top 5 most stable residues in a protein. A protein's most stable residues are important; their stabilities can be used to determine the stability of the entire protein at equilibrium [173]. For each protein from Table XII with experimental data performed at equilibrium, we compared the number of residues accurately predicted in the top 5 most stable from our method and COREX/BEST. All COREX/BEST results were obtained with their webserver `http://www.best.utmb.edu/BEST` using a full enumeration with a folding unit size of 6 residues and the default values for temperature and overall protein stability [174].

Table XIV summarizes the results. For each set of experimental data, we compared the number of matches in the top 5 most stable residues. However, some data sets had multiple stable residues with the same rate. For these data sets, the set size was increased from 5 to include such residues. Residues for which no experimental data was provided were ignored. On average, all methods predicted a third of the residues correctly with the cluster score performing slightly better. When counting the number of "winning" predictors for each data set, the rigidity score performs the best. Note that for two proteins, RNase T1 and Barnase, the cluster score was unable to produce results because the native state for these proteins is not sufficiently rigid to obtain a cluster score of 1 for any residue. Also, COREX/BEST was the only method to completely mispredict all of the most stable residues for two proteins, α-Lactalbumin and Lysozyme.

Table XIV.: Comparison between rigidity scores (RS), cluster scores (CS), and COREX/BEST of their ability to predict the most stable residues. The set size is the number of most stable (slowest exchanging) residues examined. It was chosen to be 5 unless the data set had multiple residues with the same rate in which it was increased to include such residues. The best predictor for each data set is displayed in boldface. Protein name abbreviations are given in Table XII.

Protein	**Experimental Data**	**Set Size**	**% Top Predicted**		
			RS	**CS**	**COREX/BEST [174]**
Protein G	[123]	9	22	11	**33**
cSH3	[151]	5	20	20	**40**
Protein L	[135]	7	**29**	**29**	14
CI2	[138]	5	**60**	40	40
	[139]	9	**56**	44	22
Tendamistat	[153]	7	**57**	**57**	29
	[154]	7	**29**	**29**	**29**
Ubiquitin	[141]	5	**40**	**40**	**40**
Pa cyt c_{551}	[155]	14	55	55	**70**
RNase T1	[157]	5	**20**	n/a	**20**
Barnase	[159]	5	20	n/a	**40**
y-cyt c	[160]	14	**50**	43	36

Table XIV Continued

Protein	**Experimental Data**	**Set Size**	**% Top Predicted RS**	**% Top Predicted CS**	**% Top Predicted COREX/BEST [174]**
	[161]	11	**55**	**55**	18
α-Lactalbumin	[162]	6	**33**	**33**	0
RNase A	[164]	5	**20**	**20**	**20**
CheY	[165]	5	20	**60**	20
Lysozyme	[167]	14	**7**	**7**	0
ApoMb	[169]	26	65	77	**85**
Average			37	**39**	32
# of 'Wins'			**12**	8	9

C. Folding Core Identification

The protein folding core is the subset of the protein's structure that is the first to form during folding and the last to break during denaturation [175]. Although the idea may seem simple, both experimental and simulation techniques to capture folding core membership have found it challenging and have often been unsuccessful. Folding core identification is critical in tasks such as protein design where the fabrication of proteins with predictable structure is paramount [175].

Various experimental methods have been used to identify the folding core with hydrogen exchange being one of the most common. The folding core is typically identified by slowly exchanging/highly protected regions [175, 176]. Although methods such as hydrogen exchange have taken center stage in identifying folding intermediates and describing the folding core, these methods cannot be easily applied to all proteins. Hydrogen exchange often uses high-resolution NMR, so it requires a prior assignment of the spectrum for the studied protein. Also, information about the hydrogen exchange in local segments is dictated by the number of allowable cleavage points [40].

We present a new computational method for predicting folding core membership based on relative hydrogen exchange rates extracted from our approximate landscape models (see Chapter V, Section B). In contrast to previous simulation methods for identifying the folding core [30, 31], we do not keep the protein structure fixed during the denaturation simulation and we do not restrict the folding identification to residues involved in secondary structure. We compare our predictions to those determined experimentally and to other computational approaches [30, 31]. Our results show good correlation to experiment and also indicate that our technique may be useful in suggesting other components of rigid structure for further study that have not yet been identified by experiment as part of the core.

1. Related Computational Methods

Several simulation methods have also been used to identify the folding core.

Solvent Accessible Surface Area. Solvent accessible surface area (SASA) [177] has been a popular method to characterize residues in proteins. It is typically computed by rolling a sphere/probe over the surface of a protein conformation [178]. It has been conjectured that SASA would be a good indicator of hydrogen exchange rates because residues near the surface (i.e., with high SASA values) should have a higher rate of exchange because they are more exposed to D_2O than residues buried deep inside the protein (i.e., with low SASA values). However, statistical studies of experimental hydrogen exchange rates show poor correlation to SASA [179]. This is due to the inability of SASA to distinguish between deeply buried residues and those just below the surface.

Floppy Inclusions and Rigid Substructure Topography. Floppy Inclusions and Rigid Substructure Topography (FIRST) [30] uses a full atomic description to identify rigid clusters of residues for a fixed protein conformation. They simulate denaturation by iteratively breaking the weakest hydrogen bond and recomputing the resulting rigid residue clusters. They do not, however, change atom placements in response to bond breaking. They then identify the folding core as the set of mutually rigid residues (i.e., in the same rigid cluster) belonging to at least two different secondary structure elements that remain rigid longest in the denaturation simulation.

Gaussian Network Models. Gaussian Network Model (GNM) [31] represents residues as beads connected by elastic springs representing chain connectivity and intermolecular bonding. These are subject to Gaussian fluctuations. GNM simulations provide slow mode minima and fast mode peaks which are used to identify folding cores. These fluctuations and modes of motion are limited to the immediate vicinity of the native state.

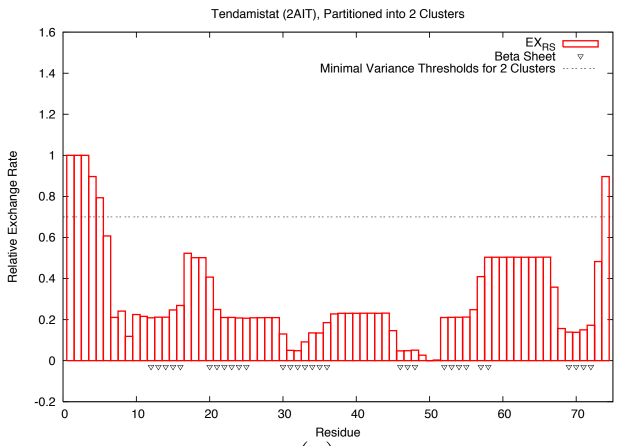
While FIRST and GNM have been able to predict the folding core with higher probability than random selection, they have struggled in identifying some folding cores. For example, α-lactalbumin [162] and T4 lysozyme [169] were mispredicted by FIRST [30]. Another drawback of these methods is that they keep the structure of the protein fixed throughout the denaturation simulation and thus are unable to study the effect on the conformation due to bond-breaking.

2. From Relative Hydrogen Exchange Rates

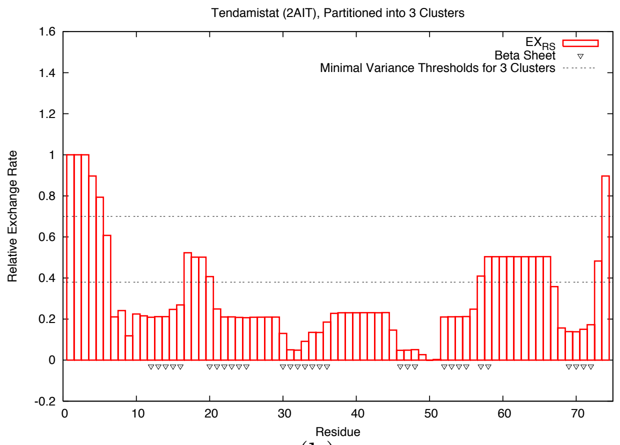
Folding core membership can be inferred from relative exchange rates, whether they are calculated from residue rigidity or from rigid cluster decomposition. Commonly, given a set of relative exchange rates from a set of residues, there exists some threshold t that defines the residues that are most stable (i.e., the folding core). This threshold definition is frequently applied to experimental exchange rates in order to define the folding core [95].

There are many ways to define a threshold t. For experimental data, this is at the discretion of the authors and varies widely in the literature. Instead, we determine the threshold automatically based on the distribution of the data. To determine t, we need to partition the data into two sets: in core and out of core. One method for partitioning data is k-means clustering [180] where the input data is divided into k clusters of minimal variance. This unsupervised learning technique has gained popularity because it requires no user intervention and is simple to apply. We could directly apply k-means clustering with $k = 2$ to determine t. However, the experimental data typically has many large gaps and/or outliers that could lead to inappropriate groupings. For example, consider the experimental rates for Tendamistat in Fig. 51(a). There are several residues at each terminus that have rates significantly higher than the rest. With $k = 2$, the value of t computed from the clustering (dashed line) is too large and would label over 90% of the protein as in the folding core. However, with larger values of k (Fig. 51(b,c)), the percentage of the protein identified by the minimum threshold is much more reasonable. The difficulty then is in automatically determining k.

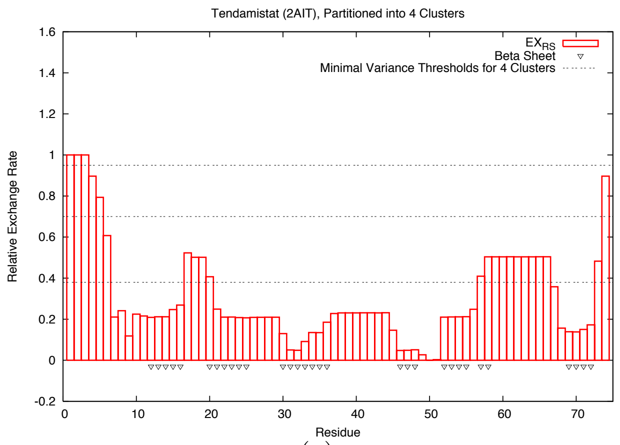
To determine an appropriate value for k, we examine the percentage of the variance explained, $(\sum_{i=1}^{k} \sigma_i^2)/\sigma^2$ where σ_i^2 is the variance of the ith cluster and σ^2 is



(a)



(b)



(c)

Fig. 51.: Minimal variance thresholds (dashed lines) to cluster EX_{RS} for Tendamistat using (a) $k = 2$, (b) 3, and (c) 4 clusters.

the variance of the original data set, for each k. We select the k that maximizes the second derivative of this function. This is commonly known as the elbow criterion [181, 182]. Intuitively, this criterion selects the k such that adding additional clusters does not add sufficient information. Fig. 52 gives an example using Tendamistat: it shows the percentage of variance explained as a function of k and the inset plots the second derivative. In both plots, the elbow is indicated with a black circle.

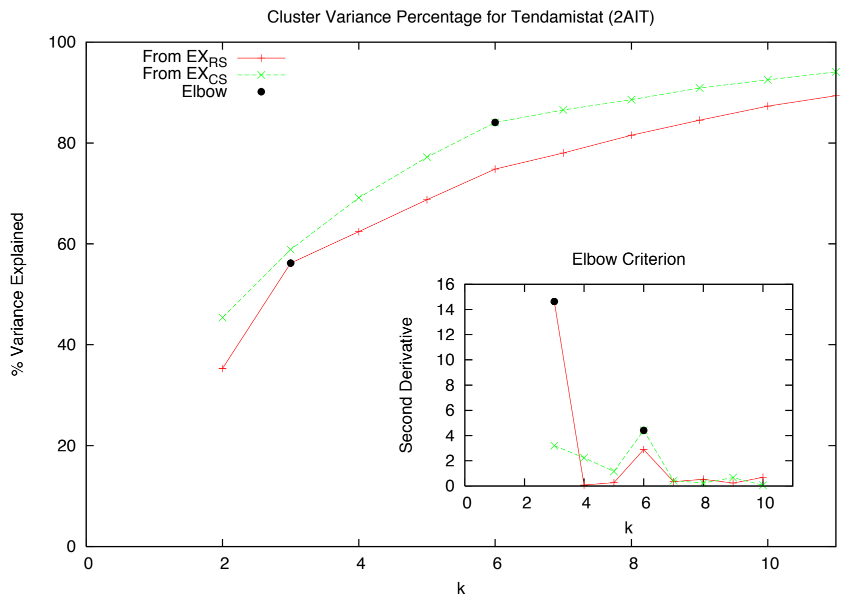


Fig. 52.: Example of the elbow criterion for selecting an appropriate number of clusters k. The "elbow" (black circle) is defined at the point where the second derivative of the percentage variance explained (inset) is maximal. In this example, the "elbow" occurs at $k = 3$ for EX_{RS} and at $k = 6$ for EX_{CS}.

Note that unlike the folding core definition in [30], ours does not require the core to be restricted to residues in secondary structure components. This is useful for proteins like *Bacillus amyloliquefaciens* ribonuclease (Barnase), cardiotoxin analogue III (CTX III), and human α-lactalbumin (α-LA), where the folding core also involves loops [95].

3. Results

We studied the same set of proteins as for the relative hydrogen exchange rate study, see Table XII. Because a folding core threshold in terms of experimental data is not universally agreed upon, we calculate an appropriate threshold instead. We use the same elbow partitioning scheme to label the residues as in or out of the folding core when data is provided. If numerical data is not provided for the experiment, we use the labeling scheme suggested by the authors. Fig. 49 shows the normalized experimental data below the simulated exchange rates plot. Unmeasured residues are indicated with empty bars at -0.2.

We examine the ability of our method to identify the folding core. We also compare our method to 4 other computational techniques: slow mode minimas (GNM-G), fast mode peaks (GNM-H) [31], and FIRST [30] with two different hydrophobic tether definitions: H3 (the default and most restrictive) and H1 (the least restrictive). We used data provided by the iGNM webserver at `http://ignm.ccbb.pitt.edu/` and identified the folding core from slow modes and fast modes as described in [31]. For FIRST, we added missing hydrogens using the WHATIF [183] webserver at `http://swift.cmbi.ru.nl/servers/html/prepdock.html` and computed folding cores using the FIRST 6.2 binary provided at `http://flexweb.asu.edu/software/first/` as described in [30]. Note that some results may differ to those previously reported; disussions with the authors of GNM and FIRST suggest this could be due to webserver and default setting changes [184].

For each set of experimental data, we compare the *sensitivity* and *specificity* of the different folding core identification techniques. We measure *sensitivity* as the ratio of the number of residues accurately labeled as in the folding core by simulation to the number of residues labeled as in the folding core by experimental data:

$$sensitivity(p) = \frac{|FC_S(p) \cap FC_E(p)|}{|FC_E(p)|} \tag{5.7}$$

where $FC_E(p)$ is the set of residues labeled as in the folding core by experiment and $FC_S(p)$ is the set of residues labeled as in the folding core by simulation for the protein p. We measure *specificity* as the ratio of the number of residues accurately labeled as out of the folding core by simulation to the number of residues labeled as out of the folding core by experimental data:

$$specificity(p) = \frac{|NFC_S(p) \cap NFC_E(p)|}{|NFC_E(p)|} \tag{5.8}$$

where $NFC_E(p)$ is the set of residues labeled as out of the folding core by experiment and $NFC_S(p)$ is the set of residues labeled as out of the folding core by simulation for the protein p. We only examine residues that were measured by experimental data which could be as little as 14% of the protein (even though simulations can label all residues with known structure). In Fig. 50, the folding core residues identified by simulation in red.

Figs. 53 and 54 compare the sensitivities and specificities of the various methods for each set of experimental data. Due in part to the large noise present in the experimental data (e.g., reported experimental rates for protein G do not agree, see Fig. 47), missing measurements (i.e., on average less than half of the protein is measured and several data sets measure less than 25%), and labeling convention inconsistencies between data sets, all computational methods exhibit large variances in sensitivity and specificity. For reference, perfect sensitivity and specificity lies at coordinates (0,1) and random guessing performance is indicated by the diagonal black line. It is interesting to note that while all the methods perform worse than random guessing on some of the data, the other methods have more points below the random guessing line (23.53%, 37.50%, 48.00%, and 38.24% for GNM-G, GNM-H, and FIRST-H3, and FIRST-H1) compared to our methods (14.71% and 18.52% for EX_{RS} and EX_{CS}).

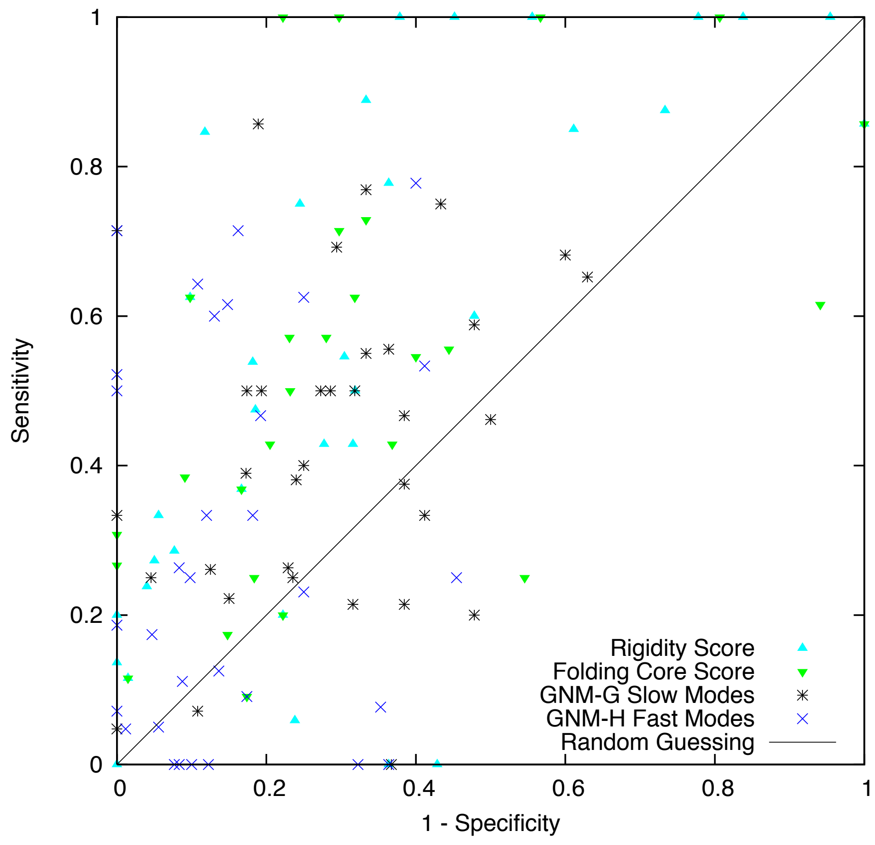


Fig. 53.: Folding core identification sensitivity and specificity by EX_{RS} (triangles) and EX_{CS} (inverted triangles) compared to GNM-G slow modes [31] (stars) and GNM-H fast modes [31] (X's).

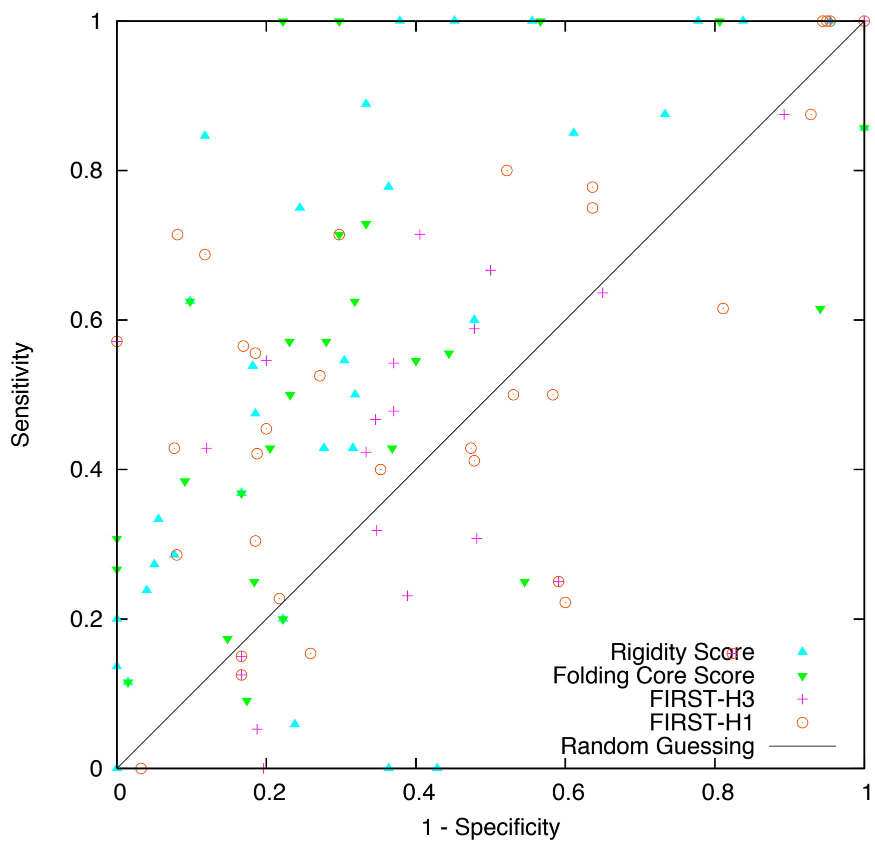


Fig. 54.: Folding core identification sensitivity and specificity by EX_{RS} (triangles) and EX_{CS} (inverted triangles) compared to FIRST-H3 (+'s) [30] and FIRST-H1 (circles) [30].

Table XV summarizes the overall statistics for each method. The error is calculated as the normalized distance from perfect sensitivity and specificity. The best performing method in each category is indicated in boldface. Recall that FIRST and EX_{CS} require at least two mutually rigid subsequences to declare a folding core. For the 21 proteins studied, FIRST-H3 and EX_{CS} did not identify a folding core for 7 and 4 proteins, respectively. These proteins were excluded from the data reported in

Table XV.: Summary of folding core identification performance. Error is the normalized distance to perfect sensitivity, specificity on Figs. 53 and 54. The best performance in each category is in boldface.

		Sensitivity		**Specificity**		**Error**		**% in**
	Method	**Avg.**	**Std.**	**Avg.**	**Std.**	**Avg.**	**Std.**	**Core**
Over All Data Sets	EX_{RS}	**0.545**	0.345	0.671	0.279	0.482	0.160	29.8
	EX_{CS}	0.538	0.283	0.682	0.261	**0.461**	0.150	29.8
	GNM-G	0.432	0.217	0.715	0.161	0.474	0.129	28.5
	GNM-H	0.302	0.263	**0.846**	0.132	0.525	0.173	15.4
	FIRST-H3	0.501	0.310	0.543	0.293	0.551	0.142	38.6
	FIRST-H1	0.518	0.277	0.573	0.313	0.521	0.153	40.7
Over 5 Most Complete Data Sets	EX_{RS}	**0.631**	0.241	0.627	0.279	0.420	0.124	43.5
	EX_{CS}	0.524	0.202	0.750	0.063	0.392	0.104	34.5
	GNM-G	0.189	0.155	0.815	0.105	0.591	0.118	21.1
	GNM-H	0.223	0.246	**0.927**	0.050	0.554	0.171	15.3
	FIRST-H3	0.628	0.122	0.612	0.025	0.384	0.046	30.0
	FIRST-H1	0.572	0.084	0.709	0.145	**0.373**	0.087	39.0
Over 5 Least Complete Data Sets	EX_{RS}	**0.753**	0.262	0.610	0.427	0.402	0.260	32.1
	EX_{CS}	0.733	0.163	0.276	0.428	0.592	0.215	42.5
	GNM-G	0.524	0.233	**0.830**	0.199	0.398	0.184	24.4
	GNM-H	0.547	0.282	0.812	0.218	**0.394**	0.213	16.2
	FIRST-H3	0.681	0.406	0.392	0.534	0.615	0.278	53.1
	FIRST-H1	0.386	0.191	0.625	0.401	0.560	0.247	28.8

the table.

Both our methods perform better than FIRST and GNM-based methods in sensitivity, between GNM and FIRST in specificity, and better than FIRST and similar to GNM-G in error. Table XV also shows how the size of the labeled folding core correlates to sensitivity and specificity. Methods that "under-guess" will tend to have lower sensitivities and higher specificities (i.e., towards the lower-left quadrant of Figs. 53 and 54), and methods that "over-guess" will tend to have higher sensitivities and lower specificities (i.e., towards the upper-right quadrant). GNM-H predicts the smallest folding cores (at 15.4%) and also has the lowest (highest) sensitivity (specificity). GNM-H is overly conservative in its identification of the folding core and thus greatly sacrifices sensitivity by labeling the largest portions of the protein as outside the core. FIRST with the largest folding cores (at 38.6% and 40.7%), has moderate sensitivity and the lowest specificity.

We believe the low sensitivities and specificities for all methods are caused largely by varying experimental conditions and missing experimental data. On average, less than half of the protein was measured experimentally with many below 25%. This imposes a bias to the labeling given as in many cases these missing measurements are caused by residues exchanging outside the experiment's scope. We found that by considering only experiments with greater than 80% measured (the five most complete) significantly reduced the standard deviation for all methods across all metrics indicating greater confidence in the predicted labeling (see Table XV). This bias is also apparent when looking at where experimentally-labeled folding core residues lie on the 3D structure (see Fig. 50). Consider OMTKY3 containing an 11-residue α-helix and 3 β-strands, each 3 residues long. Experimental data indicates that portions of the α-helix and β-strands 1 and 2 are part of the folding core. Notice that for the two β-strands, it does not label the central residue as part of the folding core but does label the two neighboring resides as folding core residues. It is unlikely that the folding core is made up of isolated residues as the labeling would indicate. Conversely, our simulation labels the entire two β-strands and the connecting turn as part of the folding core. Similar trends may be seen for many of the proteins. In general, our technique does not label isolated residues but instead labels collections of residues nearby on the 3D structure. This agrees with the expectation that the folding core consists of residues that neighbor each other on the tertiary structure (but may or may not be distant in sequence).

CHAPTER VI

CONCLUSION AND FUTURE WORK

In this dissertation, we have built upon the existing protein folding framework to extend its applicability to larger proteins and to a wider range of motions. We presented a methodology for incrementally building roadmaps until they satisfy a set of evaluation criteria specifically directed at the motion to be modeled, either folding to a known, native state or transitions between given structures. We generalized the protein folding framework to be biased towards multiple conformations of interest instead of only towards a single conformation (e.g., the native state). This broadens the motions the framework can model from folding to general transitions between multiple conformations. This work also relies heavily on rigidity theory to more efficiently sample the energy landscape, characterize similar conformations in terms of their rigidity composition, and study properties of the energy landscape including hydrogen exchange and folding core identification.

Incremental Map Generation (IMG) relieves the user of the burden of specifying a sampling density (or roadmap size). Selecting this parameter has been problematic since it is highly sensitive to protein size, the ruggedness of the energy landscape, and the the types of motions to be captured. IMG instead builds the roadmap until it meets the set of user-specified evaluation criteria. These evaluation criteria are more straightforward for the user to set because they can be customized for the particular problem at hand. We demonstrated two such criteria for two different problems: secondary structure formation order stabilization for protein folding and max-flow thresholds for protein transitions (Chapter III, Section A.2). We described and evaluated two versions of secondary structure formation order stabilization, one which based formation on native contacts present (Chapter III, Section A.2) and one which based formation on rigidity analysis (Chapter V, Section A). We used roadmaps for protein G as an example to illustrate how the increment size and the stabilization threshold affects roadmap size (Chapter III, Section A.3). We also confirmed the effectiveness of this approach by comparing it previous methods [18, 21, 22, 23] over a set of 26 proteins. IMG, coupled with rigidity-based sampling and distance metrics, produced significantly smaller roadmaps (and higher connectivity) with the same secondary structure formation order distribution as before (Chapter V, Section A.1). The pathway distributions also agreed with experimental data when available for all proteins except 1 (wild-type chyotrypsin inhibitor 2).

Generalizing our protein folding framework to handle multiple conformations of interest expands the types of protein motion we can model. Instead of being limited to protein folding, we can study other problems relating to transitions between bound and unbound conformations, folding intermediates, misfolding, and allosteric interactions. We described this generalized methodology (Chapter III, Section B) and presented results for several different transitions, some involving simple secondary structure rearrangement and some requiring more large-scale conformational change as in the case of calmodulin which entails the unraveling of the central α helix. We provided evidence that the transitions mapped by our approach are more realistic than those given by the computationally less expensive Morph Server [25], especially for transitions requiring large conformational changes. In future research, we would like to remove our fixed bond length and bond angle assumption. Instead of modeling these small degrees of freedom explicitly as many simulations do, we intend to interpolate these values from the input structures for each residue based on the relationship between its ϕ and ψ angles and the ϕ and ψ angles of the input structures.

Rigidity theory has been an invaluable tool in this work. We developed two simplified rigidity models, the single-body, Cα model and the two-body, ϕ-ψ model, for representing protein conformations (Chapter IV, Sections A.1-2). We also described an uncomplicated optimization to the rigidity analysis computation based on the assumption that all motions to be modeled do not involve the severing of any backbone peptide bonds (Chapter IV, Section A.3). Our comparison of the two models on a set of 18 proteins of varying secondary structure makeup and size show that the single-body, Cα model is faster, largely due to the smaller model size and fewer constraints present (Chapter IV, Section A.4). Our results also show the impact of the backbone optimization, sometimes producing a savings of over 50%. In the future, we plan to expand the comparison the two rigidity models presented here from an empirical treatment to a theoretical study by defining the conditions where these models differ with respect to the rigid and flexible regions they report.

We used our rigidity models during roadmap construction to more efficiently model the energy landscape, both through rigidity-based sampling (Chapter IV, Section B) and new rigidity-based distance metrics (Chapter IV, Section C). This efficiency was seen in the dramatic reduction of roadmap sizes for the set of 26 proteins studied for stable secondary structure formation order (Chapter V, Section A.1). The increased roadmap connectivity also speaks to the ability of the rigidity-based distance metrics to identify connectable conformations. It was only with these rigidity-based techniques that we were able to detect the subtle folding differences of structurally similar proteins G, L, and their mutants (Chapter V, Section A.2).

Finally, we used rigidity theory to study two new protein folding properties: relative hydrogen exchange (Chapter V, Section B) and folding core identification (Chapter V, Section C). We compared our simulated exchange rates and ability to identify the folding core to available experimental data and other computational methods over a set of 21 different proteins of varying size and structure. Our method was more successful in identifying the slowest exchanging (and therefore most stable) residues than COREX/BEST [28, 29] (Chapter V, Section B.4). It also performed better than the other computational approaches [30, 31] for folding core identification in terms of sensitivity and better than all but one class in terms of specificity (Chapter V, Section C.3). We believe the real use of our technique will be to aid researchers by providing an indication of fast and slowly exchanging residues to target for protein design for proteins that have not yet been studied experimentally.

In the future, we would like to explore other applications of motion planning methods and rigidity analysis to modeling protein motion. We believe these techniques can be used to study other properties of protein motion such as identifying folding intermediates, explaining possible mechanisms for allosteric interactions, and providing a model of the energy landscape separating misfolded proteins from their functional, native state. In addition, we want our methods to be used as a tool by other researchers for modeling properties of protein folding and motion where experimental data is not available or difficult to obtain.

REFERENCES

[1] F. Chiti and C. Dobson, "Protein misfolding, functional amyloid, and human disease," *Annu. Rev. Biochem.*, vol. 75, pp. 333–366, 2006.

[2] C.-S. Goh, D. Milburn, and M. Gerstein, "Conformational changes associated with protein-protein interactions," *Curr. Op. Str. Biol.*, vol. 14, pp. 1–6, 2004.

[3] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne, "The protein data bank," *Nucleic Acids Res.*, vol. 28, no. 1, pp. 235–242, 2000.

[4] G. Rhodes, *Crystallography Made Crystal Clear.* London: Academic Press, 1993.

[5] K. Wuthrich, *NMR of Proteins and Nucleic Acids.* New York: J. Wiley & Sons, 1986.

[6] M. Levitt, "Protein folding by restrained energy minimization and molecular dynamics," *J. Mol. Biol.*, vol. 170, pp. 723–764, 1983.

[7] J. M. Haile, *Molecular Dynamics Simulation: Elementary Methods.* New York: Wiley, 1992.

[8] V. Daggett and M. Levitt, "Realistic simulation of naive-protein dynamics in solution and beyond," *Annu. Rev. Biophys. Biomol. Struct.*, vol. 22, pp. 353–380, 1993.

[9] Y. Duan and P. A. Kollman, "Pathways to a protein folding intermediate observed in a 1-microsecond simulation in aqueous solution," *Science*, vol. 282, pp. 740–744, 1998.

[10] D. G. Covell, “Folding protein α-carbon chains into compact forms by Monte Carlo methods,” *Proteins: Struct. Funct. Genet.*, vol. 14, no. 4, pp. 409–420, 1992.

[11] A. Kolinski and J. Skolnick, “Monte Carlo simulations of protein folding,” *Proteins: Struct. Funct. Genet.*, vol. 18, no. 3, pp. 338–352, 1994.

[12] Y. Koga, K. Kondo, J. Kuffner, and J.-C. Latombe, “Planning motions with intentions,” in *Proc. ACM SIGGRAPH*, 1995, pp. 395–408.

[13] O. B. Bayazit, J.-M. Lien, and N. M. Amato, “Better flocking behaviors using rule-based roadmaps,” in *Proc. 5th Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, Nice, France, Dec 2002, pp. 95–111.

[14] H. Chang and T. Y. Li, “Assembly maintainability study with motion planning,” in *Proc. 1995 IEEE Int. Conf. Robot. Autom. (ICRA)*, Nagoya, Aichi, Japan, 1995, pp. 1012–1019.

[15] S. Sundaram, I. Remmler, and N. M. Amato, “Disassembly sequencing using a motion planning approach,” in *Proc. 2001 IEEE Int. Conf. Robot. Autom. (ICRA)*, Seoul, Korea, 2001, pp. 1475–1480.

[16] A. P. Singh, J.-C. Latombe, and D. L. Brutlag, “A motion planning approach to flexible ligand binding,” in *Proc. 7th Int. Conf. on Intelligent Systems for Molecular Biology (ISMB)*, Heidelberg, Germany, 1999, pp. 252–261.

[17] M. S. Apaydin, A. P. Singh, D. L. Brutlag, and J.-C. Latombe, “Capturing molecular energy landscapes with probabilistic conformational roadmaps,” in *Proc. 2001 IEEE Int. Conf. Robot. Autom. (ICRA)*, Seoul, Korea, 2001, pp. 932–939.

[18] N. M. Amato and G. Song, "Using motion planning to study protein folding pathways," *J. Comput. Biol.*, vol. 9, no. 2, pp. 149–168, 2002, Special issue of Int. Conf. Comput. Molecular Biology (RECOMB), 2001.

[19] X. Tang, B. Kirkpatrick, S. Thomas, G. Song, and N. M. Amato, "Using motion planning to study RNA folding kinetics," in *Proc. 8th Int. Conf. Comput. Molecular Biology (RECOMB)*, San Diego, CA, 2004, pp. 252–261.

[20] L. E. Kavraki, P. Švestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Automat.*, vol. 12, no. 4, pp. 566–580, August 1996.

[21] N. M. Amato, K. A. Dill, and G. Song, "Using motion planning to map protein folding landscapes and analyze folding kinetics of known native structures," *J. Comput. Biol.*, vol. 10, no. 3-4, pp. 239–255, 2003, Special issue of Int. Conf. Comput. Molecular Biology (RECOMB), 2002.

[22] G. Song, S. L. Thomas, K. A. Dill, J. M. Scholtz, and N. M. Amato, "A path planning-based study of protein folding with a case study of hairpin formation in protein G and L," in *Proc. 2003 Pacific Symposium of Biocomputing (PSB)*, Lihue, HI, 2003, pp. 240–251.

[23] G. Song, "A motion planning approach to protein folding," Ph.D. dissertation, Dept. of Computer Science, Texas A&M University, December 2004.

[24] S. Thomas, X. Tang, L. Tapia, and N. M. Amato, "Simulating protein motions with rigidity analysis," *J. Comput. Biol.*, vol. 14, no. 6, pp. 839–855, 2007, Special issue of Int. Conf. Comput. Molecular Biology (RECOMB), 2006.

[25] N. Echols, D. Milburn, and M. Gerstein, “Molmovdb: Analysis and visualization of conformational change and structural flexibility,” *Nucleic Acids Res.*, vol. 31, pp. 478–482, 2003.

[26] D. Xie, M. Morales, R. Pearce, S. Thomas, J.-M. Lien, and N. M. Amato, “Incremental map generation (IMG),” in *Algorithmic Foundation of Robotics VII* (Editors: S. Akella, N. M. Amato, W. H. Huang, and B. Mishra). Berlin: Springer, 2008, pp. 53–68.

[27] S. Thomas, L. Tapia, and N. M. Amato, Protein Folding Core Identification from Rigidity Analysis and Motion Planning Parasol Lab, Dept. of Comp. Sci., Texas A&M University, College Station, TX, Tech. Rep. TR08-001, Oct 2008 [Online]. Available: http://parasol.tamu.edu/groups/amatogroup/publications.php#bio

[28] V. J. Hilser and E. Freire, “Structure based calculation of the equilibrium folding pathway of proteins. correlation with hydrogen exchange protection factors,” *J. Mol. Biol.*, vol. 262, pp. 756–772, 1996.

[29] V. J. Hilser, E. B. Garcia-Moreno, T. G. Oas, G. Kapp, and S. T. Whitten, “A statistical thermodynamic model of the protein ensemble,” *Chem. Rev.*, vol. 106, no. 5, pp. 1545–1558, 2006.

[30] B. M. Hespenheide, A. J. Rader, M. F. Thorpe, and L. A. Kuhn, “Identifying protein folding cores from the evolution of flexible regious during unfolding,” *J. Mol. Gra. Model.*, vol. 21, pp. 195–207, 2002.

[31] A. J. Rader and I. Bahar, “Folding core predictions from network models of proteins,” *Polymer*, vol. 45, pp. 659–668, 2004.

[32] C. Branden and J. Tooze, *Introduction to Protein Structure*, 2nd ed. New York: Garland Pub., 1999.

[33] C. B. Anfinsen, "Principles that govern the folding of protein chains," *Science*, vol. 181, pp. 223–230, 1973.

[34] M. J. Sternberg, *Protein Structure Prediction.* Oxford: Oxford University Press, 1996.

[35] R. L. Baldwin, "Protein folding: Matching speed and stability," *Nature*, vol. 369, pp. 183–184, 1994.

[36] J. D. Bryngelson, J. N. Onuchic, N. D. Socci, and P. G. Wolynes, "Funnels, pathways, and the energy landscape of protein folding: A synthesis," *Proteins: Struct. Funct. Genet.*, vol. 21, pp. 167–195, 1995.

[37] K. A. Dill and H. S. Chan, "From Levinthal to pathways to funnels: The new view of protein folding kinetics," *Nat. Struct. Biol.*, vol. 4, pp. 10–19, 1997.

[38] H. S. Chan and K. A. Dill, "Protein folding in the landscape perspective: Chevron plots and non-arrhenius kinetics," *Proteins: Struct. Funct. Genet.*, vol. 30, no. 1, pp. 2–33, 1998.

[39] P. A. Evans and S. E. Radford, "Probing the structure of folding intermediates," *Curr. Op. Str. Biol.*, vol. 4, no. 1, pp. 100–106, 1994.

[40] H. Roder, K. Maki, and H. Cheng, "Early events in protein folding explored by rapid mixing methods," *Chem. Rev.*, vol. 106, pp. 1836–1861, 2006.

[41] S. Akiyama, S. Takahashi, K. Ishimori, and I. Morishima, "Stepwise formation of alpha-helices during cytochrome c folding," *Nat. Struct. Biol.*, vol. 7, no. 6, pp. 443–445, 2000.

[42] T. E. Wales and J. R. Engen, “Hydrogen exchange mass spectrometry for the analysis of protein dynamics,” *Mass Spec. Rev.*, vol. 25, no. 1, pp. 158–170, 2006.

[43] L. E. Kay, D. A. Torchia, and A. Bax, “Backbone dynamics of proteins as studied by nitrogen-15 inverse detected heteronuclear NMR spectroscopy: Application to staphylococcal nuclease,” *Biochemistry*, vol. 28, no. 23, pp. 8972–8979, 1989.

[44] A. G. Palmer, C. D. Kroenke, and J. P. Loria, “Nuclear magnetic resonance methods for quantifying microsecond-to-millisecond motions in biological macromolecules,” *Methods Enzymol.*, vol. 339, pp. 204–238, 2001.

[45] A. Mittermaier and L. E. Kay, “New tools provide new insights in NMR studies of protein dynamics,” *Science*, vol. 312, no. 5771, pp. 224–228, 2006.

[46] Z. Ren, B. Perman, V. Srajer, T.-Y. Teng, C. Pravervand, D. Bourgeois, F. Schotte, T. Ursby, R. Kort, M. Wulff, and K. Moffat, “A molecular movie at 1.8 A resolution displays the photocycle of photoactive yellow protein, a eubacterial blue-light receptor, from nanoseconds to seconds,” *Biochemistry*, vol. 40, no. 46, pp. 13788–13801, 2001.

[47] V. Srajer, Z. Ren, T.-Y. Teng, M. Schmidt, T. Ursby, D. Bourgeois, C. Pravervand, W. Schildkamp, M. Wulff, and K. Moffat, “Protein conformational relaxation and ligand migration in myoglobin: a nanosecond to millisecond molecular movie from time-resolved laue x-ray diffraction,” *Biochemistry*, vol. 40, no. 46, pp. 13802–13815, 2001.

[48] N. Go, “Theoretical studies of protein folding,” *Annu. Rev. Biophys. Bioeng.*, vol. 12, pp. 183–210, 1983.

[49] K. A. Dill, S. Bromberg, K. Yue, K. M. Fiebig, D. P. Yee, P. D. Thomas, and H. S. Chan, “Principles of protein folding–a perspective from simple exact models,” *Protein Sci.*, vol. 4, pp. 561–602, 1995.

[50] R. Zhou, M. Eleftheriou, C.-C. Hon, R. S. Germain, A. K. Royyuru, and B. J. Berne, “Massively parallel molecular dynamics simulations of lysozyme unfolding,” *IBM J. Res. & Dev.*, vol. 52, no. 1/2, pp. 19–30, 2008.

[51] S. M. Larson, C. D. Snow, M. Shirts, and V. S. Pande, “Folding@home and genome@home: Using distributed computing to tackle previously intractable problems in computational biology,” *Computational Genomics*, 2003, To appear.

[52] M. Shirts and V. S. Pande, “Screen savers of the world unite,” *Science*, vol. 290, pp. 1903–1904, 2000.

[53] N. L. Fawzi, V. Chubukov, L. A. Clark, S. Brown, and T. Head-Gordon, “Influence of denatured and intermediate states of folding on protein aggregation,” *Protein Sci.*, vol. 14, pp. 993–1003, 2005.

[54] V. Muñoz, E. R. Henry, J. Hoferichter, and W. A. Eaton, “A statistical mechanical model for β-hairpin kinetics,” *Proc. Natl. Acad. Sci. USA*, vol. 95, pp. 5872–5879, 1998.

[55] E. Alm and D. Baker, “Prediction of protein-folding mechanisms from free-energy landscapes derived from native structures,” *Proc. Natl. Acad. Sci. USA*, vol. 96, no. 20, pp. 11305–11310, 1999.

[56] D. Baker, “A surprising simplicity to protein folding,” *Nature*, vol. 405, pp. 39–42, 2000.

[57] A. Matagne, S. E. Radford, and C. M. Dobson, "Fast and slow tracks in lysozyme folding: Insight into the of domains in the folding process," *J. Mol. Biol.*, vol. 267, pp. 1068–1074, 1997.

[58] N. M. Amato, O. B. Bayazit, L. K. Dale, C. V. Jones, and D. Vallejo, "OBPRM: An obstacle-based PRM for 3D workspaces," in *Robotics: The Algorithmic Perspective (Editors: P. K. Agarwal, L. E. Kavraki, and M. T. Mason)*. Natick, MA: A K Peters, Ltd., 1998, pp. 155–168.

[59] O. B. Bayazit, G. Song, and N. M. Amato, "Ligand binding with OBPRM and haptic user input: Enhancing automatic motion planning with virtual touch," in *Proc. 2001 IEEE Int. Conf. Robot. Autom. (ICRA)*, Seoul, Korea, 2001, pp. 954–959.

[60] M. S. Apaydin, D. L. Brutlag, C. Guestrin, D. Hsu, and J.-C. Latombe, "Stochastic roadmap simulation: An efficient representation and algorithm for analyzing molecular motion," in *Proc. 6th Int. Conf. Comput. Molecular Biology (RECOMB)*, Washington, DC, 2002, pp. 12–21.

[61] J. Cortés, T. Siméon, M. Remaud-Siméon, and V. Tran, "Geometric algorithms for the conformational analysis of long protein loops," *J. Computat. Chem.*, vol. 25, no. 7, pp. 956–967, 2004.

[62] J. Cortés and T. Siméon, "Sampling-based motion planning under kinematic loop-closure constraints," in *Algorithmic Foundations of Robotics VI* (Editors: M. Erdmann, D. Hsu, M. Overmars, and A. F. van der Stappen). Berlin: Springer, 2005, pp. 75–90.

[63] M. Gerstein and W. Krebs, "A database of macromolecular motions," *Nucleic Acids Res.*, vol. 26, pp. 4280–4290, 1998.

[64] W. G. Krebs and M. Gerstein, “The morph server: A standardized system for analyzing and visualizing macromolecular motions in a database framework,” *Nucleic Acids Res.*, vol. 28, pp. 1665–1675, 2000.

[65] A. M. Lesk, *Protein Architecture: A Practical Approach.* Oxford: IRL Press, 1991.

[66] M. Gerstein and C. H. Chothia, “Analysis of protein loop closure: Two types of hinges produce one motion in lactate dehydrogenase,” *J. Mol. Biol.*, vol. 220, pp. 133–149, 1991.

[67] C. S. Tung, S. C. Harvey, and J. A. McCammon, “Large-amplitude bending motions in phenylalaine transfer RNA,” *Biopoly.*, vol. 23, pp. 2173–2193, 1984.

[68] S. Wells, S. Menor, B. M. Hespenheide, and M. F. Thorpe, “Constrained geometric simulation of the diffusive motions in proteins,” *Physical Biology*, vol. 2, pp. S127–S136, 2005.

[69] W. L. Nichols, G. D. Rose, L. F. Ten Eyck, and B. H. Zimm, “Rigid domains in proteins: An algorithmic approach to their identification,” *Proteins*, vol. 23, pp. 38–48, 1995.

[70] A. S. Siddiqui and G. J. Barton, “Continuous and discontinuous domains: An algorithm for the automatic generation of reliable protein domain definition,” *Protein Sci.*, vol. 4, pp. 872–884, 1995.

[71] N. Boutonnet, M. Rooman, and S. Wodak, “Automatic analysis of protein conformational changes by multiple linkage clustering,” *J. Mol. Biol.*, vol. 253, pp. 633–647, 1995.

[72] Y. Duan and M. Karplus, "Pathways to a protein folding intermediate observed in a 1-microsecond simulation in aqueous solution," *Science*, vol. 282, pp. 740–744, 1998.

[73] J. Ma and M. Karplus, "The allosteric mechanism of the chaperonin groel: A dynamic analysis," *Proc. Natl. Acad. Sci. USA*, vol. 95, pp. 8502–8507, 1998.

[74] D. A. Case, "Molecular dynamics and normal mode analysis of biomolecular rigidity," in *Rigidity Theory and Applications (Editors: M. F. Thorpe and P. M. Duxbury)*. New York: Kluwer Academic/Plenum Publishers, 1999, pp. 329–344.

[75] O. Keskin, R. L. Jernigan, and I. Bahar, "Proteins with similar architecture exhibit similar large-scale dynamic behavior," *Biophys. J.*, vol. 78, pp. 2093–2106, 2000.

[76] L. Holm and C. Sander, "Parser for protein folding units," *Proteins*, vol. 19, pp. 256–268, 1994.

[77] M. H. Zehfus and G. D. Rose, "Compact units in proteins," *Biochemistry*, vol. 25, pp. 5759–5765, 1986.

[78] P. A. Karplus and G. E. Schulz, "Prediction of chain flexibility in proteins," *Naturwissencschaften*, vol. 72, pp. 212–213, 1985.

[79] J. Janin and S. Wodak, "Structural domains in proteins and their role in the dynamics of protein function," *Prog. Biophys. Mol. Biol.*, vol. 42, pp. 21–78, 1983.

[80] V. Maiorov and R. Abagyan, "A new method for modeling large-scale rearrangements of protein domains," *Proteins*, vol. 27, pp. 410–424, 1997.

[81] D. J. Jacobs and M. F. Thorpe, “Generic rigidity percolation: The pebble game,” *Phys. Rev. Lett.*, vol. 75, no. 22, pp. 4051–4054, 1995.

[82] D. J. Jacobs and M. F. Thorpe, “Generic rigidity percolation in two dimensions,” *Phys. Rev. E*, vol. 53, no. 4, pp. 3682–3693, 1996.

[83] D. J. Jacobs and B. Hendrickson, “An algorithm for two dimensional rigidity percolation: The pebble game,” *J. Comp. Phys*, vol. 137, pp. 346–368, 1997.

[84] D. J. Jacobs, “Generic rigidity in three-dimensional bond-bending networks,” *J. Phys. A: Math. Gen.*, vol. 31, pp. 6653–6668, 1998.

[85] D. J. Jacobs, L. A. Kuhn, and M. F. Thorpe, “Flexible and rigid regions in proteins,” in *Rigidity Theory and Applications (Editors: M. F. Thorpe and P. M. Duxbury)*. New York: Kluwer Academic/Plenum Publishers, 1999, pp. 357–384.

[86] D. J. Jacobs, A. J. Rader, L. A. Kuhn, and M. F. Thorpe, “Protein flexiblility predictions using graph theory,” *Proteins: Struct. Funct. Genet.*, vol. 44, pp. 150–165, 2001.

[87] A. J. Rader, B. M. Hespenheide, L. A. Kuhn, and M. F. Thorpe, “Protein unfolding: Rigidity lost,” *Proc. Natl. Acad. Sci. USA*, vol. 99, no. 6, pp. 3540–3545, 2002.

[88] M. Lei, M. I. Zavodszky, L. A. Kuhn, and M. F. Thorpe, “Sampling protein conformations and pathways,” *J. Comput. Chem.*, vol. 25, pp. 1133–1148, 2004.

[89] T. Lozano-Pérez, “Spatial planning: A configuration space approach,” *IEEE Trans. Comput.*, vol. C-32, pp. 108–120, 1983.

[90] L. Tapia, X. Tang, S. Thomas, and N. M. Amato, "Kinetics analysis methods for approximate folding landscapes," *Bioinformatics*, vol. 23, no. 13, pp. 539–548, 2007, Special issue of Int. Conf. on Intelligent Systems for Molecular Biology (ISMB) & European Conf. on Computational Biology (ECCB) 2007.

[91] A. Lee and I. Streinu, "Pebble game algorithms and sparse graphs," *Discrete Mathematics*, vol. 308, pp. 1425–1437, 2008, Special issue of the Third European Conference on Combinatorics — Graph Theory and Applications.

[92] G. Laman, "On graphs and rigidity of plane skeletal structures," *J. Eng. Math.*, vol. 4, pp. 331–340, 1970.

[93] W. Whiteley, "Some matroids from discrete applied geometry," *Contemp. Math.*, vol. 197, pp. 171–311, 1996.

[94] S. V. Krivov and M. Karplus, "Free energy disconnectivity graphs: Application to peptide models," *J. Chem. Phys*, vol. 114, no. 23, pp. 10894–10903, 2002.

[95] R. Li and C. Woodward, "The hydrogen exchange core and protein folding," *Protein Sci.*, vol. 8, no. 8, pp. 1571–1591, 1999.

[96] E. L. McCallister, E. Alm, and D. Baker, "Critical role of β-hairpin formation in protein G folding," *Nat. Struct. Biol.*, vol. 7, no. 8, pp. 669–673, 2000.

[97] H. J. Vogel, "Calmodulin: A versatile calcium mediator protein," *Biochem. Cell Biol.*, vol. 72, no. 9-10, pp. 357–376, 1994.

[98] A. Crivici and M. Ikura, "Molecular and structural basis of target recognition by calmodulin," *Annu. Rev. Biophys. Biomol. Struct.*, vol. 24, pp. 85–116, 1995.

[99] J. Evenas, A. Malmendal, and S. Forsen, “Calcium,” *Curr. Op. Chem. Biol.*, vol. 2, no. 2, pp. 293–302, 1998.

[100] M. R. Celio, T. Pauls, and B. Schwaller, *Guidebook to the Calcium-Binding Proteins*. Oxford: Oxford University Press, 1996.

[101] S. B. Prusiner, “Prions,” *Proc. Natl. Acad. Sci. USA*, vol. 95, no. 23, pp. 13363–13383, 1998.

[102] J. Collinge, “Prion diseases of humans and animals: Their causes and molecular basis,” *Annu. Rev. Neurosci.*, vol. 24, pp. 519–550, 2001.

[103] K. M. Pand, M. Baldwin, J. Nguyen, M. Gasset, A. Serban, D. Groth, I. Mehlhorn, Z. Huang, R. J. Fletterick, and F. E. Cohen, “Conversion of alpha-helices into beta-sheets features in the formation of the scrapie prion proteins,” *Proc. Natl. Acad. Sci. USA*, vol. 90, no. 23, pp. 10962–10966, 1993.

[104] J. Safar, P. P. Roller, D. C. Gajdusek, and C. J. Gibbs, “Thermal stability and conformational transitions of scrapie amyloid (prion) protein correlate with infectivity,” *Protein Sci.*, vol. 2, no. 12, pp. 2206–2216, 1993.

[105] T. Lazaridis and M. Karplus, “Effective energy function for proteins in solution,” *Proteins*, vol. 35, pp. 133–152, 1999.

[106] R. H. Kretsinger and C. E. Nockolds, “Carp muscle calcium binding protein: Structure determination and general description,” *J. Biol. Chem.*, vol. 248, pp. 3313–3326, 1973.

[107] R. H. Kretsinger, “Calcium coordination and the calmodulin fold: Divergent versus confergent evolution,” *Cold Spring Harbor Symp. Quant. Biol.*, vol. 52, pp. 499–510, 1987.

[108] S. Linse, A. Helmersson, and S. Forsen, “Calcium binding to calmodulin and its globular domains,” *J. Biol. Chem.*, vol. 266, pp. 8050–8054, 1991.

[109] G. Barbato, M. Ikura, L. E. Kay, R. W. Pastor, and A. Bax, “Backbone dynamics of calmodulin studied by N relaxation using inverse detected two-dimensional NMR spectroscopy: The central helix is flexible,” *Biochemistry*, vol. 31, pp. 5269–5278, 1992.

[110] M. Ikura, G. M. Clore, A. M. Gronenborn, G. Zhu, C. B. Klee, and A. Bax, “Solution structure of a calmodulin-target peptide complex by multidimensional NMR,” *Science*, vol. 256, pp. 632–638, 1992.

[111] M. R. Nelson and W. J. Chazin, “An interaction-based analysis of calcium-induced conformational changes in ca2+ sensor proteins,” *Protein Sci.*, vol. 7, pp. 270–282, 1998.

[112] D. M. Zuckerman, “Simulation of and ensemble of conformational transitions in a united-residue model of calmodulin,” *J. Phys. Chem*, vol. 108, pp. 5127–5137, 2004.

[113] B. Kuhlman, J. W. O’Neill, D. E. Kim, K. Y. J. Zhang, and D. Baker, “Accurate computer-based design of a new backbone conformation in the second turn of protein L,” *J. Mol. Biol.*, vol. 315, pp. 471–477, 2002.

[114] S. Nauli, B. Kuhlman, and D. Baker, “Computer-based redesign of a protein folding pathway,” *Nature Struct. Biol.*, vol. 8, no. 7, pp. 602–605, 2001.

[115] S. Forsén and R. A. Hoffman, “Study of moderately rapid chemical exchange reactions by means of nuclear magnetic double resonance,” *J. Chem. Phys*, vol. 39, no. 11, pp. 2892–2901, 1963.

[116] S. Waelder, L. Lee, and A. G. Redfield, “Nuclear magnetic resonance studies of exchangeable protons. I. Fourier transform saturation-recovery and transfer of saturation of the tryptophan indole nitrogen proton,” *J. Am. Chem. Soc.*, vol. 97, no. 10, pp. 2927–2928, 1975.

[117] C. R. Matthews, “Effect of point mutations on the folding of globular proteins,” *Methods Enzymol.*, vol. 154, pp. 498–511, 1987.

[118] A. R. Fersht, A. Matouschek, and L. Serrano, “The folding of an enzyme I. theory of protein engineering analysis of stability and pathway of protein folding,” *J. Mol. Biol.*, vol. 224, pp. 771–782, 1992.

[119] B. A. Krantz and T. R. Sosnick, “Engineered metal binding sites map the heterogeneous folding landscape of a coiled coil,” *Nat. Struct. Biol.*, vol. 8, no. 12, pp. 1042–1047, 2001.

[120] M. U. Johansson, H. Nilsson, J. Evenäs, S. Forsén, T. Drakenberg, L. Björck, and M. Wikström, “Differences in backbone dynamics of two homologous bacterial albumin-binding modules: Implications for binding specificity and bacterial adaptation,” *J. Mol. Biol.*, vol. 316, pp. 1083–1099, 2002.

[121] G. Hernández and D. M. LeMaster, “Reduced temperature dependence of collective conformational opening in a hyperthermophile rubredoxin,” *Biochemistry*, vol. 40, pp. 14384–14391, 2001.

[122] J. Kuszewski, G. M. Clore, and A. M. Gronenborn, “Fast folding of a prototypic polypeptide: The immunoglobulin binding domain of streptococcal protein G,” *Protein Sci.*, vol. 3, pp. 1945–1952, 1994.

[123] J. Orban, P. Alexander, P. Bryan, and D. Khare, “Assessment of stability

differences in the protein G B1 and B2 domains from hydrogen-deuterium exchange: Comparison with calorimetric data," *Biochemistry*, vol. 34, pp. 15291–15300, 1995.

[124] A. R. Viguera, L. Serrano, and M. Wilmanns, "Different folding transitions states may result in the same native structure," *Nat. Struct. Biol.*, vol. 3, no. 10, pp. 874–880, 1996.

[125] J. C. Martínez and L. Serrano, "The folding transition state between SH3 domains is conformationally restricted and evolutionarily conserved," *Nat. Struct. Biol.*, vol. 6, no. 11, pp. 1010–1016, 1999.

[126] R. Richarz, P. Sehr, G. Wagner, and K. Wuthrich, "Kinetics of the exchange of individual amide protons in the basic pancreatic trypsin inhibitor," *J. Mol. Biol.*, vol. 130, pp. 19–30, 1979.

[127] C. K. Woodward and B. D. Hilton, "Hydrogen isotope exchange kinetics of single protons in bovine pancreatic trypsin inhibitor," *Biophys. J.*, vol. 32, pp. 561–575, 1980.

[128] H. Roder and K. Wuthrich, "Protein folding kinetics by combined use of rapid mixing techniques and NMR observation of individual amide protons," *Proteins Struct. Funct. Genet.*, vol. 1, pp. 34–42, 1986.

[129] V. P. Grantcharova, D. S. Riddle, J. V. Santiago, and D. Baker, "Important role of hydrogen bonds in structurally polarized transition state folding of the src SH3 domain," *Nat. Struct. Biol.*, vol. 5, no. 8, pp. 714–720, 1998.

[130] D. S. Riddle, V. P. Grantcharova, J. V. Santiago, E. Alm, I. Ruczinski, and D. Baker, "Experiment and theory highlight role of native state topology in

SH3 folding," *Nat. Struct. Biol.*, vol. 6, no. 11, pp. 1016–1024, 1999.

[131] Y. Bai, A. Karimi, H. J. Dyson, and P. E. Wright, "Absence of a stable intermediate on the folding pathway of protein A," *Protein Sci.*, vol. 6, no. 7, pp. 1449–1457, 1997.

[132] S. Sato, T. L. Religa, and A. R. Fersht, "ϕ-analysis of the folding of the B domain of protein A using multiple optical probes," *J. Mol. Biol.*, vol. 360, pp. 850–864, 2006.

[133] M. C. Baxa, K. F. Freed, and T. R. Sosnick, "Quantifying the structural requirements of the folding transition state of protein A and other systems," *J. Mol. Biol.*, vol. 381, pp. 1362–1381, 2008.

[134] Q. Yi and D. Baker, "Direct evidence for a two-state protein unfolding transition from hydorgen-deuterium exhchange, mass spectrometry, and NMR.," *Protein Sci.*, vol. 5, pp. 1060–1066, 1996.

[135] Q. Yi, M. L. Scalley, K. T. Simons, S. T. Gladwin, and D. Baker, "Characterization of the free energy spectrum of peptostreptococcal protein L," *Folding Design*, vol. 2, pp. 271–280, 1997.

[136] S. E. Jackson, N. elMasry, and A. R. Fersht, "Structure of the hydrophobic core in the transition state for folding of chymotrypsin inhibitor 2: A critical test of the protein engineering method of analysis," *Biochemistry*, vol. 32, pp. 11270–11278, 1993.

[137] L. S. Itzhaki, D. E. Otzen, and A. R. Fersht, "The structure of the transition state for folding of chymotrypsin inhibitor 2 analysed by protein engineering

methods: Evidence for a nucleation-condensation mechanism for protein folding," *J. Mol. Biol.*, vol. 254, pp. 260–288, 1995.

[138] L. S. Itzhaki, J. L. Neira, and A. R. Fersht, "Hydrogen exchange in chymotrypsin inhibitor 2 probed by denaturants and temperature," *J. Mol. Biol.*, vol. 270, pp. 89–98, 1997.

[139] J. L. Neira, L. S. Itzhaki, D. E. Otzen, B. Davis, and A. R. Fersht, "Hydrogen exchange in chymotrypsin inhibitor 2 probed by mutagenesis," *J. Mol. Biol.*, vol. 270, pp. 99–110, 1997.

[140] H. M. Rodriguez, A. D. Robertson, and L. M. Gregoret, "Native state EX2 and EX1 hydrogen exchange of *escherichia coli* CspA, a small β-sheet protein," *Biochemistry*, vol. 41, pp. 2140–2148, 2002.

[141] Y. Pan and M. S. Briggs, "Hydrogen exchange in native and alcohol forms of ubiquitin," *Biochemistry*, vol. 31, pp. 11405–11412, 1992.

[142] B. A. Krantz, R. S. Dothanger, and T. R. Sosnick, "Discerning the structure and energy of multiple transition states in protein folding using ψ-analysis," *J. Mol. Biol.*, vol. 337, pp. 463–475, 2004.

[143] T. R. Sosnick, R. S. Dothanger, and B. A. Krantz, "Differences in the folding transition state of ubiquitin indicated by ϕ and ψ analyses," *Proc. Natl. Acad. Sci. USA*, vol. 101, no. 50, pp. 17377–17382, 2004.

[144] H. M. Went and S. E. Jackson, "Ubiquitin folds through a highly polarized transition state," *Protein Eng. Des. Sel.*, vol. 18, no. 5, pp. 229–237, 2005.

[145] V. Villegas, J. C. Martínez, F. Z. Avilés, and L. Serrano, "Structure of the transition state in the folding process of human procarboxypeptidase A2

activation domain," *J. Mol. Biol.*, vol. 283, pp. 1027–1036, 1998.

[146] K. Teilum, B. B. Kragelund, J. Knudsen, and F. M. Poulsen, "Formation of hydrogen bonds precedes the rate-limiting formation of persistent structure in the folding of ACBP," *J. Mol. Biol.*, vol. 301, pp. 1307–1314, 2000.

[147] D. E. Kim, C. Fisher, and D. Baker, "A breakdown of symmetry in the folding transition state of protein L," *J. Mol. Biol.*, vol. 298, pp. 971–984, 2000.

[148] T. E. Creighton, Ed., *Protein Folding*. New York: W. H. Freeman & Company, 1992.

[149] H. J. Dyson and P. E. Wright, "Unfolded proteins and protein folding studied by NMR," *Chem. Rev.*, vol. 104, pp. 3607–3622, 2004.

[150] C. B. Arrington, L. M. Teesch, and A. D. Robertson, "Defining protein ensembles with native-state NH exchange: Kinetics of interconversion and cooperative units from combined NMR and MS analysis," *J. Mol. Biol.*, vol. 285, pp. 1265–1275, 1999.

[151] V. P. Grantcharova and D. Baker, "Folding dynamics of the src SH3 domain," *Biochemistry*, vol. 36, pp. 15685–15692, 1997.

[152] T. Sivaraman, T. K. S. Kumar, D. K. Chang, W. Y. Lin, and C. Yu, "Events in the kinetic folding pathway of a small, all β-sheet protein," *J. Biol. Chem.*, vol. 273, no. 17, pp. 10181–10189, 1998.

[153] W. Qiwen, A. D. Kline, and K. Wuthrich, "Amide proton exchange in the α-amylase polypeptide inhibitor tendamistat studied by two-dimensional ^{1}H nuclear magnetic resonance," *Biochemistry*, vol. 26, pp. 6488–6493, 1987.

[154] N. Schonbrunner, J. Wey, J. Engles, H. Georg, and T. Kiefhaber, "Native-like β-structure in a trifluoroethanol-induced partially folded state of the all-β-sheet protein tendamistat," *J. Mol. Biol.*, vol. 260, pp. 432–445, 1996.

[155] B. S. Russell, L. Zhong, M. G. Bigotti, F. Cutruzzolà, and K. L. Bren, "Backbone dynamics and hydrogen exchange of Pseudomonas aeruginosa ferricytochrome c_{551}," *J. Biol. Inorg. Chem.*, vol. 8, pp. 156–166, 2003.

[156] L. S. Mullins, C. N. Pace, and F. M. Raushel, "Investigation of ribonuclease T1 folding intermediates by hydrogen-deuterium amide exchange-two-dimensional NMR spectroscopy," *Biochemistry*, vol. 32, pp. 6152–6156, 1993.

[157] L. S. Mullins, C. N. Pace, and F. M. Raushel, "Conformational stability of ribonuclease T1 determined by hydrogen-deuterium exchange," *Protein Sci.*, vol. 6, pp. 1387–1395, 1997.

[158] A. Matouschek, L. Serrano, E. M. Meiering, M. Bycroft, and A. R. Fersht, "The folding of an enzyme v. $H/^2H$ exchange–nuclear magnetic resonance studies on the folding pathway of barnase: Complementarity to and agreement with protein engineering studies," *J. Mol. Biol.*, vol. 224, pp. 837–845, 1992.

[159] S. Perrett, J. Clarke, A. M. Hounslow, and A. R. Fersht, "Relationship between equilibrium amide proton exchange behavior and the folding pathway of barnase," *Biochemistry*, vol. 34, pp. 9288–9298, 1995.

[160] J. L. Marmorino, D. S. Auld, S. F. Betz, D. F. Doyle, G. B. Young, and G. J. Pielak, "Amine proton exchange rates of oxidized and reduced Saccharomyces cerevisiae iso-1-cytochrome c," *Protein Sci.*, vol. 2, pp. 1966–1974, 1993.

[161] S. M. Baxter and J. S. Fetrow, “Hydrogen exchange behavior of [U-^{15}N]-labeled oxidized and reduced iso-1-cytochrome c,” *Biochemistry*, vol. 38, pp. 4493–4503, 1999.

[162] B. A. Schulman, C. Redfield, Z.-Y. Peng, C. M. Dobson, and P. S. Kim, “Different subdomains are most protected from hydrogen exchange in the molten globule and native state of human α-lactalbumin,” *J. Mol. Biol.*, vol. 253, pp. 651–657, 1995.

[163] J. B. Udgaonkar and R. L. Baldwin, “Early folding intermediate of ribonuclease A,” *Proc. Natl. Acad. Sci. USA*, vol. 87, pp. 8197–8201, 1990.

[164] A. Wang, A. D. Robertson, and D. W. Bolen, “Effects of a naturally occurring compatible osmolyte on the internal dynamics of ribonuclease A,” *Biochemistry*, vol. 34, pp. 15096–15104, 1995.

[165] E. Lacroix, M. Bruix, E. López-Hernández, L. Serrano, and M. Rico, “Amide hydrogen exchange and internal dynamics the chemotatic protein CheY from escherichia coli,” *J. Mol. Biol.*, vol. 271, pp. 472–487, 1997.

[166] L. A. Morozova-Roche, D. T. Haynie, C. Arico-Muendel, H. V. Dael, and C. M. Dobson, “Structural basis of the stability of a lysozyme molten globule,” *Nat. Struct. Biol.*, vol. 2, no. 10, pp. 871–875, 1995.

[167] L. A. Morozova-Roche, C. C. Arico-Muendel, D. T. Haynie, V. I. Emelyanenko, H. V. Dael, and C. M. Dobson, “Structural characterisation and comparison of the native and A-states of equine lysozyme,” *J. Mol. Biol.*, vol. 268, pp. 903–921, 1997.

[168] Y.-H. Chi, T. K. S. Kumar, K. M. Kathir, D.-H. Lin, G. Zhu, I.-M. Chiu, and

C. Yu, "Investigation of the structural stability of the human acidic fibroblast growth factor by hydrogen–deuterium exchange," *Biochemistry*, vol. 41, pp. 15350–15359, 2002.

[169] F. M. Hughson, P. E. Wright, and R. L Baldwin, "Structural characterization of a partially folded apo-myoglobin intermediate," *Science*, vol. 249, pp. 1544–1548, 1990.

[170] P. A. Jennings and P. E. Wright, "Formation of a molten globule intermediate early in the kinetic folding pathway of apomyoglobin," *Science*, vol. 262, pp. 892–896, 1993.

[171] K. Yamasaki, K. Ogasahara, K. Yutani, M. Oobatake, and S. Kanaya, "Folding pathway of escherichia coli ribonuclease HI: A circular dichroism, fluorescence, and NMR study," *Biochemistry*, vol. 34, pp. 16552–16562, 1995.

[172] T. M. Raschke and S. Marqusee, "The kinetic folding intermediate of ribonuclease H resembles the acid molten globule and partially unfolded molecules detected under native conditions," *Nat. Struct. Biol.*, vol. 4, pp. 298–304, 1997.

[173] B. M. P. Huyghues-Despointes, J. M. Scholtz, and C. N. Pace, "Protein conformational stabilities can be determined from hydrogen exchange rates," *Nat. Struct. Biol.*, vol. 6, no. 10, pp. 910–912, 1999.

[174] J. Vertrees, P. Barritt, S. Whitten, and V. J. Hilser, "COREX/BEST server: a web browser-based program that calculates regional stability variations within protein structures," *Bioinformatics*, vol. 21, pp. 2218–3319, 2005.

[175] C. Woodward, "Is the slow exchange core the protein folding core?," *Trends Biochem. Sci.*, vol. 18, pp. 359–360, 1993.

[176] K. S. Kim, J. A. Fuchs, and C. K. Woodward, "Hydrogen exchange identifies native-state motional domains important in protein folding," *Biochemistry*, vol. 32, pp. 9600–9608, 1993.

[177] B. Lee and F. M. Richards, "The interpretation of protein structures: estimation of static accessibility," *J. Mol. Biol.*, vol. 55, no. 3, pp. 379–400, 1971.

[178] A. Shrake and J. A. Rupley, "Environment and exposure to solvent of protein atoms. lysozyme and insulin.," *J. Mol. Biol.*, vol. 79, no. 2, pp. 351–371, 1973.

[179] B. C. Bennett, A. S. Gardberg, M. D. Blair, and C. G. Dealwis, "On the determinants of amide backbone exchange in proteins: A neutron crystallographic comparative study," *Acta Crystallogr. D Biol. Crystallogr.*, vol. 64, pp. 764–783, 2008.

[180] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, NJ: Prentice Hall, 1988.

[181] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer, 2001.

[182] L. Lieu and N. Saito, "Automated discrimination of shapes in high dimensions," in *Proc. of SPIE*, 2007, vol. 6701, paper 67011V.

[183] G. Vriend, "What if: A molecular modeling and drug design program," *J. Mol. Gra. Model.*, vol. 8, no. 1, pp. 52–56, 1990.

[184] A. J. Rader, 2008, Private correspondance.

VITA

Shawna Lynn Thomas received her B.S. in computer engineering from Texas A&M University in December 2001. She graduated Summa Cum Laude and was a University Undergraduate Research Fellow. Her Senior Honors Thesis was titled "Using Spheres to Improve Motion Planning Algorithms." As an undergraduate, she was part of the Undergraduate Summer Research Program with Dr. Nancy Amato. She also participated in the Computing Research Association's Distributed Mentor Program under the direction of Dr. Lydia Kavraki at Rice University on a project titled "Modeling Protein-Protein Interactions with the Aid of Motion Planning Algorithms." During her graduate career, she received an NSF Graduate Research Fellowship (2002–2005), a Philanthropic Educational Organization (P.E.O.) Scholar Award (2005–2006), a Department of Education Graduate Assistance in Areas of National Need (GAANN) Fellowship (2006–2007), an IBM Fran Allen Ph.D. Fellowship (2007–2008), and an IBM Ph.D. Fellowship (2008–2009). Her research focus is on randomized motion planning algorithms and their application to problems in computational biology. She is also interested in the supporting areas of scientific visualization, physically-based modeling, and parallel computing. She received her Ph.D. in Computer Science from Texas A&M University in May 2010.

More information about Shawna Lynn Thomas' research and publications may be found at http://parasol.tamu.edu/~sthomas. She may be reached at: Parasol Lab, 301 Harvey R. Bright Bldg, 3112 TAMU, College Station, TX 77843-3112.

The typist for this dissertation was Shawna Lynn Thomas.