

A VIRTUAL PEBBLE GAME TO ENSEMBLE AVERAGE GRAPH RIGIDITY

by

Luis Carlos González Gurrola

A dissertation submitted to the faculty of
the University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Information Technology

Charlotte

2011

Approved by:

Dr. Dennis R. Livesay

Dr. Donald J. Jacobs

Dr. Jun-tao Guo

Dr. Ronald Sass

ABSTRACT

LUIS CARLOS GONZÁLEZ GURROLA. A virtual pebble game to ensemble average graph rigidity. (Under the direction of DR. DENNIS R. LIVESAY)

Previous works have demonstrated that protein rigidity is related to thermodynamic stability, especially under conditions that favor formation of native structure. Mechanical network rigidity properties of a single conformation are efficiently calculated using the integer Pebble Game (PG) algorithm. However, thermodynamic properties require averaging over many samples from the ensemble of accessible conformations, leading to fluctuations within the network. We have developed a mean field Virtual Pebble Game (VPG) that provides a probabilistic description of the interaction network, meaning that sampling is not required. We extensively test the VPG algorithm over a variety of body-bar networks created on disordered lattices, from these calculations we fully characterize the network conditions under which the performance of the VPG offers the best solution. The VPG provides a satisfactory description of the ensemble averaged PG properties, especially in regions removed from the rigidity transition where ensemble fluctuations are greatest. In further experiments, we characterized the VPG across a structurally nonredundant dataset of 272 proteins. Using quantitative and visual assessments of the rigidity characterizations, the VPG results are shown to accurately reflect the ensemble averaged PG properties. That is, the fluctuating interaction network is well represented by a single calculation that replaces density functions with average values, thus speeding up the desired calculation by several orders of magnitude. Finally, we propose a new algorithm that is based on the combination of PG and VPG to balance the amount of sampling and mean field treatment. While offering interesting results, this approach needs to be further optimized to fully leverage its utility. All these results position the VPG as an efficient alternative to understand the mechanical role that chemical interactions play in maintaining protein stability.

ACKNOWLEDGMENTS

I remember those days in high school when I dreamed about being a doctor, this has been a long way since then. I would like to thank the co-authors of this Ph.D. dissertation, Dr. Dennis R. Livesay and Dr. Donald J. Jacobs. They formed me as a scientist and I will always consider them my mentors.

Dr. Jun-tao Guo and Dr. Ronald Sass, thank you for accepting being part of this team.

I want to acknowledge the lovely presence of three persons in my life, without them everything would have been extremely more difficult. Nancy, “Chaparro” y Nicolás, este logro lo comparto con ustedes, los amo!

To Mexico’s National Council on Science and Technology (CONACYT) and The University of North Carolina at Charlotte for their support.

TABLE OF CONTENTS

LIST OF FIGURES	viii
LIST OF TABLES	x
CHAPTER 1: RIGIDITY ANALYSIS OF PROTEIN NETWORKS USING A MEAN FIELD APPROACH	1
1.1 Introduction	1
1.2 Dissertation outline	3
1.2.1 Chapter 2: Description of the VPG and body-bar networks	4
1.2.2 Chapter 3: VPG and PG comparison over a nonredundant protein dataset	4
1.2.3 Chapter 4: Meand field + sampling = the best of both worlds	4
1.2.4 Chapter 5: VPG and PG comparison using heterogeneous probabilities	4
1.2.5 Chapter 6: Discussion	4
1.2.6 Chapter 6: Conclusions	4
CHAPTER 2: A VIRTUAL PEBBLE GAME TO ENSEMBLE AVERAGE GRAPH RIGIDITY	5
2.1 Introduction	5
2.2 Test Body-Bar Networks	5
2.3 Constraint Counting and Pebble Games	9
2.4 Description of the VPG Algorithm	11
2.5 Results and Discussion	14
2.6 Conclusions	24
2.7 Appendices: VPG Pseudo-Codes	25
CHAPTER 3: CALCULATING ENSEMBLE AVERAGED DESCRIPTIONS OF PROTEIN RIGIDITY WITHOUT SAMPLING IN A PROTEIN DATASET	29
3.1 Introduction	29
3.2 Materials and Methods	30
3.2.1 Protein Structure Description	30

	vi
3.2.2 Rigid Cluster Decomposition	31
3.2.3 Network Similarity Metrics	32
3.2.4 Rigidity Profiles	34
3.3 Results and Discussion	34
3.3.1 Quantifying Rigid Cluster Similarity	34
3.3.2 Over versus Under Prediction of Rigidity	36
3.3.3 Rigidity Profile Similarity	37
3.3.4 The Rigidity Transition	43
3.4 Conclusions	43
3.5 Appendix	47
CHAPTER 4: IMPROVING PROTEIN FLEXIBILITY PREDICTIONS BY COMBINING STATISTICAL SAMPLING WITH A MEAN-FIELD VIRTUAL PEBBLE GAME	48
4.1 Introduction	48
4.2 Materials and methods	48
4.2.1 Protein network descriptions	48
4.2.2 The algorithms	49
4.2.3 Similarity measures	51
4.3 Results and discussion	52
4.4 Conclusions	59
4.5 Future work	60
CHAPTER 5: COMPARISON OF A MEAN FIELD ALGORITHM AGAINST AN ENSEMBLE-BASED ALGORITHM USING PROBABILITIES FOR HYDROGEN BONDS DERIVED FROM THEIR RESPECTIVE ENERGY	62
5.1 Introduction	62
5.2 Materials and Methods	65
5.2.1 Network Similarity Metrics	66
5.3 Results and Discussion	67

	vii
5.3.1 Physicality of three bars per hydrogen bond	72
5.4 Future work	73
5.5 Conclusions	75
CHAPTER 6: DISCUSSION	77
6.1 Heuristics for the VPG	77
6.2 On the various ways to improve the accuracy of the VPG	80
6.2.1 Results for the comparison of the VPG methods	81
6.3 Future work	83
6.4 Products of the dissertation	85
CHAPTER 7: CONCLUSIONS	86
REFERENCES	89

LIST OF FIGURES

FIGURE 2.1: Creation of test body-bar network topologies.	6
FIGURE 2.2: A representative VPG network, and, an exhaustive list of possible PG network topologies.	8
FIGURE 2.3: Application of the VPG on a 3 vertex network.	13
FIGURE 2.4: Comparison of the number of internal DOF within the network as calculated by the algorithms.	16
FIGURE 2.5: Contour plots showing the maximum difference in DOF between PG and VPG for different system sizes.	18
FIGURE 2.6: Maximum difference in internal DOF versus the heterogeneity index.	19
FIGURE 2.7: Rigid Cluster Susceptibility curves for four characteristic cases.	21
FIGURE 2.8: Average number of internal DOF within the network for four proteins.	22
FIGURE 2.9: VPG execution time.	23
FIGURE 3.1: Comparison of network descriptions for PG and VPG.	31
FIGURE 3.2: Comparison of two cases of rigid cluster decomposition.	32
FIGURE 3.3: Rand measures (RM), agreement measure (AM) and Pearson correlation coefficient for all the MCMs for PG and VPG.	37
FIGURE 3.4 Agreement measure (AM) for three exemplar proteins.	39
FIGURE 3.5: Rigid cluster maps (RCM) of chemotaxis receptor methyltransferase CheR.	40
FIGURE 3.6: Rigid cluster maps for four exemplar protein near their respective P_{RM} values.	41
FIGURE 3.7: Mechanical coupling maps for four exemplar proteins near their respective P_{RM} values.	42
FIGURE 3.8: Boxplots describing the ensemble of Pearson correlations coefficients comparing each PG realization to the \overline{PG} behavior.	44
FIGURE 3.9: Rigid cluster susceptibility (RCS) for 12 typical protein examples.	45
FIGURE 3.10: Rigidity transition effects.	46
FIGURE 4.1: Comparison of network descriptions for all three PG variants.	50
FIGURE 4.2: The Rand measure (RM) is plotted versus P_{nat} for a typical case.	52
FIGURE 4.3: The number of floppy modes (FM) is plotted versus P_{nat} .	54

FIGURE 4.4: Rigid clusters maps (RCM) identify all co-rigid (red) residue pairs in four example protein structures.	55
FIGURE 4.5: Mechanical coupling maps (MCM) provide a more nuanced view of co-rigidity.	57
FIGURE 4.6 Rigid Cluster decomposition for VPG (a) and VPG- x (b) for a large methyltransferase structure (1AF7). The black circles in the bottom panel highlight regions of disagreement between the algorithms.	58
FIGURE 4.7: Accuracy versus computational cost.	59
FIGURE 5.1 Potential energy of each H-bond versus its corresponding occupational probability (a) and the RCS curve (b) for a neurotoxin structure.	63
FIGURE 5.2 The probability of existence of a fluctuating interaction is based on its respective energy. Based on this model, for PG several network topologies are created whereas for the VPG just one topology is needed.	64
FIGURE 5.3: Comparison of the calculation of the total number of available DOF for eight proteins.	68
FIGURE 5.4: Rigid cluster maps (RCM) for eight proteins.	70
FIGURE 5.5: Mechanical coupling maps (MCM) for eight proteins.	71
FIGURE 5.6: Calculation of DOF by PG when using three versus five distance constraints per hydrogen bond.	74
FIGURE 5.7 Comparison of the calculation of DOF by PG and VPG using the five-bar model. The protein networks are: a) oncogene, b) oxidoreductase, c) apolipoprotein, d) neurotoxin, e) spasmolytic polypeptide and f) subtilisin BPN'	75
FIGURE 6.1: The free DOF in the system for $L = 5$ as a function of the probability of the fluctuating edges for the algorithms.	83
FIGURE 6.2: The free DOF in the system for $L = 10$ as a function of the probability of the fluctuating edges for the algorithms.	84

LIST OF TABLES

TABLE 3.1: Rigid cluster visualizations for four exemplar proteins.	38
---------------------------------------------------------------------	----

CHAPTER 1: RIGIDITY ANALYSIS OF PROTEIN NETWORKS USING A MEAN FIELD APPROACH

1.1 Introduction

Conformational flexibility links protein structure and function. For example, an enzyme must be rigid enough for reproducibility in molecular recognition, yet flexible enough to mediate the catalytic mechanism [1]. This inseparability between flexibility and function makes computational descriptions of protein structure difficult. Nevertheless, high accuracy in computational modeling of proteins can only be achieved by accounting for the fact that structure is dynamically changing, thus making the problem computationally expensive. Therefore, developing new and improved methods that better optimize the balance between computational cost and model accuracy is necessary to fully understand protein structure and dynamics.

An important characteristic of a physical system is the number of degrees of freedom (DOF) that describes its accessible motion. This is particularly true in molecular networks where structure greatly depends on the degree of cross-linking due to chemical interactions. Therein, covalent bonds are modeled as a set of distance constraints between atoms. As more covalent bonds form, more distance constraints are added to the network. If a new distance constraint is placed in a flexible region, a DOF available to the network will be removed because motion is reduced, and the distance constraint is said to be independent. If a distance constraint is placed in a rigid region, there is no change in the number of DOF, and the distance constraint is said to be redundant. Compared to generic networks, the DOF calculation in protein structures presents an additional complicating factor because they include many noncovalent interactions that repetitively break and re-form based on ambient thermal motion. These noncovalent interactions include: hydrogen bonds (H-bonds), salt bridges and van der Waals interactions. Consequently, rigidity analyses used to quantify the number of DOF within a protein structure require averages over an ensemble of structural networks, each one having a different set of noncovalent interactions.

From the discussion above, it follows that discrimination between independent and redundant edges over a set of vertices is a critical aspect of rigidity analyses. The first contributions in this area were given for networks in two dimensions where the first polynomial algorithm to identify an independent set of edges was proposed by Sugihara [2]. An $O(N^2)$ algorithm was later developed by Imai [3] based on a network flow approach, which was later matched using matroid sums by Gabow and Westermann [4,5] and by Hendrickson [6] using a bipartite matching algorithm. In the same direction, a new method based on a hierarchical decomposition, called red-black hierarchy, that takes quadratic time to identify Laman graphs (minimally rigid graphs) [7] was developed by Bereg [8].

The algorithm proposed by Hendrickson (bipartite matching) turned out to be a fertile ground for the appearance of an elegant algorithm called the pebble game, developed by Jacobs and Hendrickson [9]. The pebble game algorithm was developed to calculate exact properties of graph rigidity for generic two-dimensional networks using a combinatorial characterization [9,10] based on Laman’s theorem [7]. This means that only constraint topology is necessary to determine rigidity properties, which is completely specified by a graph. The network rigidity properties include identifying: (1) the number of independent distance constraints; (2) the rigid and flexible regions within the network; and (3) over-constrained regions that have more distance constraints than needed for the region to be rigid. Within the PG algorithm pebbles represent degrees of freedom. Extensions to three-dimensions using similar pebble game algorithms have been made for a limited number of network types [11–17], and further generalizations have been made to an entire class of graph rigidity problems that define a matroid [18,19]. Interestingly, much of these developments were motivated by applications to predict flexible and rigid regions within biomolecular structures, such as proteins [12,20,21].

Developed by the BioMolecular Physics Group at UNC Charlotte, which I am a member of, thermodynamic properties of protein-like systems have been accurately predicted using a distance constraint model (DCM) that regards network rigidity as an underlying mechanical interaction, and considers an ensemble of all possible constraint topologies [22–24]. Put simply, the DCM is a statistical mechanical model that averages over an ensemble of PG calculations, thus appropriately describing fluctuations within the noncovalent bond net-

work. Within the DCM, the free energy of the protein is calculated over a two-dimensional grid specified by number of H-bonds and native torsion interactions. Within each grid point (called a macrostate), the PG is applied hundreds of times to determine the average probability for distance constraints to be independent or redundant. From this information, the average number of DOF remaining in the system can be readily calculated. The free energy of the protein is subsequently expressed as a function of its global flexibility [23, 24].

It was recently demonstrated that the computationally expensive PG calculation could be replaced by a fast mean-field approach called Maxwell Constraint Counting (MCC), which is based on the pioneering work of the famous physicist James Clark Maxwell [25]. Therein, only global network properties are considered. Nevertheless, the thermodynamic properties of structural transitions within polypeptide and protein folding are appropriately described, including: the β -hairpin to coil transition [26], the α -helix to coil transition [27] and protein folding [28]. Unfortunately, all details regarding constraint density fluctuations are lost within MCC.

In this dissertation we present a new mean field approach, called the Virtual Pebble Game (VPG). The underlying motivation of the VPG is to optimally balance computational efficiency and descriptions of network rigidity. The VPG calculates ensemble average rigidity properties using a single network by tracing pebble placement probabilities rather than the pebbles themselves. In this context, distance constraints associated with fluctuating noncovalent interactions are assigned a probability as well. Constraint probabilities are translated into pebble capacities, where the greater capacities correspond to increased probabilities. As such, pebble rearrangements in the VPG correspond to pebble probability flow through the network. Our objective is to calculate the average number of DOF more accurately than MCC and to retain localized network rigidity information. Additionally, the VPG calculation is designed to be much faster than the time it takes to perform repetitive application of the PG during the process of ensemble averaging.

1.2 Dissertation outline

The organization of this dissertation is as follows.

1.2.1 Chapter 2: Description of the VPG and body-bar networks

The Virtual Pebble algorithm is described. We compare the PG and VPG over several networks created on cubic lattices.

1.2.2 Chapter 3: VPG and PG comparison over a nonredundant protein dataset

An extensive comparison of PG and VPG algorithms across a data set of 272 protein structures is made. The comparisons are based on the rigid clusters identified by both algorithms, which represent groups of atoms that behave as a single body.

1.2.3 Chapter 4: Meand field + sampling = the best of both worlds

A new hybrid algorithm called VPG- x that integrates the PG and VPG algorithms is presented. The VPG- x approach bridges the divide between the PG and VPG algorithms, allowing one to continuously vary from one to the other as a function of x .

1.2.4 Chapter 5: VPG and PG comparison using heterogeneous probabilities

A comparison of PG and VPG is performed over some exemplar protein networks. The probability of existence of fluctuating interactions (noncovalent bonds) is determined based on their respective energy. We also present some discussion about using different numbers of distance constraints to represent a fluctuating interaction.

1.2.5 Chapter 6: Discussion

Several topics are presented, first we elaborate in more detail about similarities and differences of PG and VPG. Afterwards, we present a number of algorithms that were tested to try to improve the VPG mechanical predictions. Finally, future work motivated from this dissertation is suggested.

1.2.6 Chapter 6: Conclusions

Finally, we summarize the conclusions from this dissertation.

CHAPTER 2: A VIRTUAL PEBBLE GAME TO ENSEMBLE AVERAGE GRAPH RIGIDITY

2.1 Introduction

In this chapter, we algorithmically define for the very first time the Virtual Pebble Game algorithm (VPG). This definition is given in the context of the body-bar PG that has been used for determining rigidity properties in proteins. We compare PG and VPG side by side and schematically explain the differences. We formally introduce the Maxwell Constraint Counting (MCC) [25] and compare it to both PG and VPG algorithms on the calculation of degrees of freedom (DOF). For the experimental section, we create an extensive set of body-bar networks on cubic lattices that will allow us to fully characterize the performance of the VPG on specific network conditions. We go beyond the calculation of DOF to identify rigid clusters of atoms and consequently calculate the Rigid Cluster Susceptibility Curve (RCS), that determines the point where the rigidity transition occur. To complement the comparison of both algorithms beyond disordered lattices, we introduce a protein dataset where we compare the number of internal DOF as calculated by PG and VPG.

2.2 Test Body-Bar Networks

Body-bar networks are represented by a set of rigid bodies connected by one or more fixed-length bars, where universal joints keep attached the bars to the bodies. This model allows us to represent a molecular network, where atoms are rigid bodies in space and the chemical interactions are represented by bars (edges). In order to characterize the VPG rigidity estimations we created a comprehensive number of body-bar networks. These networks vary from protein-like networks to topologies with maximal fluctuating edges. Particular emphasis is given to the latter cases to benchmark the worst case scenario for the VPG. To test the performance characteristics of the VPG, we consider square ($d = 2$) and cubic ($d = 3$) lattices with L vertices in each dimension, with a total of $N = L^d$

vertices. Periodic boundary conditions are used in each dimension. Each vertex models a rigid body, having 6 DOF. One of two types of edges can be placed between any pair of vertices: quenched or fluctuating. Quenched and fluctuating edges are used to model different types of physical/chemical interactions in physical systems such as proteins. For example, covalent interactions are typically modeled by quenched edges, which are present in the network with probability one; noncovalent interactions such as hydrogen bonds or torsion interactions break and re-form with thermal fluctuations, so they are modeled by fluctuating edges which are present in the network only p -fraction of the time. In the model lattice, a vertex can be connected to any one of its $2d$ nearest neighbors either by a quenched edge with probability q_{fix} or fluctuating edge with probability q_{fluct} . Thus, any two neighboring vertices are disconnected with probability $(1 - q_{\text{fix}} - q_{\text{fluct}})$. Based on earlier works [13, 14] that model covalent bonding and hydrogen bonding using 5 distance constraints, we designate the pebble capacity of an edge to be 5. In general, the pebble capacity of an edge will equal the number of distance constraints connecting neighboring vertices.

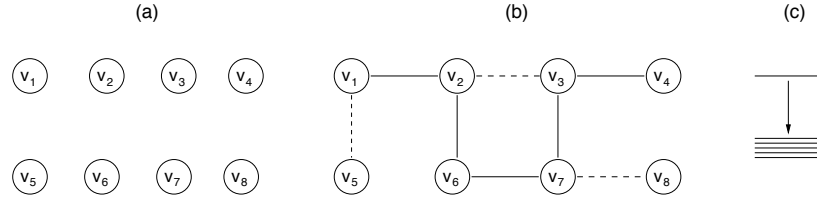


Figure 2.1: Creation of test body-bar network topologies. At the beginning of the process, (a) the network is a set of disconnected rigid bodies. Then (b), edges between nearest neighboring vertices are added to the lattice, where some are quenched (solid lines) and others fluctuate between being and not being present (dashed lines). Missing edges are not shown. (c) When an edge is present, it represents 5 bars (or distance constraints).

The process of creating test body-bar networks is shown in Fig. 2.1 for a simple case in two dimensions on a square lattice. Initially, we begin with a set of unconnected vertices, each representing a rigid body (Fig. 2.1a). Next, quenched and fluctuating edges are randomly placed between neighboring vertices in the lattice based on probabilities q_{fix} and q_{fluct} , respectively. Specifically, a uniform random number, $0 \leq x \leq 1$, is generated for each pair of neighboring vertices. When $x \leq q_{\text{fix}}$ a quenched edge is placed between the pair of vertices; when $q_{\text{fix}} < x \leq (q_{\text{fix}} + q_{\text{fluct}})$ a fluctuating edge is placed; otherwise

no edge is placed. The probability q_{fluct} determines if an edge is going to be part of the set of fluctuating edges (shown as dashed lines in Fig. 2.1b). Once an edge is known to be fluctuating, another probability, p , is introduced to determine if a particular fluctuating edge is actually present in the network or missing. As $p \rightarrow 0$, the fluctuating edge will be essentially missing. As $p \rightarrow 1$ the fluctuating edge will essentially be present. Thus, once an edge is classified as being fluctuating, the probability p controls the level of fluctuation of being present or empty. These possibilities dictate random events that can occur, for each edge. Over the entire network, we have the identical independent probability, p , assigned to each fluctuating edge. As such, an ensemble of possible networks characterized by q_{fix} , q_{fluct} and p can be generated, through a Monte Carlo process.

We apply the PG and the VPG to an ensemble of networks and to a single representative of that ensemble, respectively. All members of the ensemble and the representative network are identical with respect to missing and quenched edges. Therefore, we first determine the edges that are fixed, fluctuating and missing. This is a single network that acts as a common template between the PG and the VPG. Within the PG, an entire ensemble of networks is generated based on this template, by randomly determining which fluctuating edge is present or missing. In the VPG, this is quantified by the probability, p . In the example shown in Fig. 2.1b the edges $v_1 - v_2, v_2 - v_6, v_6 - v_7, v_3 - v_7$ and $v_3 - v_4$ are quenched describing covalent bonding, while edges $v_1 - v_5, v_2 - v_3$ and $v_7 - v_8$ fluctuate, describing noncovalent bonding (such as a hydrogen bond). Whenever an edge is present, it represents 5 distance constraints (referred to as bars) that are generically placed between two vertices (rigid-bodies) which are bundled together (see Fig. 2.1c). The remaining adjacent vertex pairs in Fig. 2.1b represent “missing” interactions, e.g., $v_5 - v_6$.

Comparisons between the PG and VPG are done on networks with an identical set of quenched and fluctuating edges, but the fluctuating edges are treated differently. For a network with N_f fluctuating edges, an ensemble consisting of 2^{N_f} different realizations must be generated to exactly capture network fluctuations within the PG approach. The word *realization* in this context means the observed network of constraints after the presence or absence of each fluctuating edge is determined as one possible outcome (or random event). In other words, a single realization refers to a particular arrangement of edges in the network,

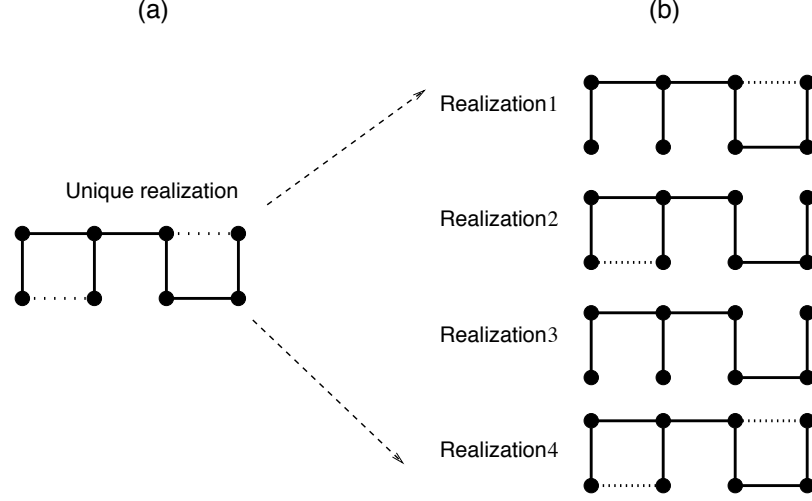


Figure 2.2: (a) A representative VPG network, and, (b) an exhaustive list of possible PG network topologies. The missing edges and the quenched edges (thick lines) are the same in both the PG and VPG networks. Multiple realizations are generated for PG analysis to account for fluctuating edges (dashed lines). The VPG analysis requires only one realization, in which the fluctuating edges (dotted lines) are assigned an average capacity.

where in one realization a fluctuating edge is present, whereas in another realization the same fluctuating edge is not present. Furthermore, the only differences between different realizations occur within the fluctuating edges. The randomness we are dealing with involves consideration of different body-bar topologies, where each realization in topology represents a distinct generic body-bar network. It is worth noting that the term realization should not be confused with the common use in the study of rigidity to make a distinction between the topology of a network, and the coordinates of the vertices of that network. Because we are only interested in generic rigidity, this latter sense of the word is not used here.

In practice, only a relatively small number of samples (typically ranging from a few hundred to hundreds of thousands) can be considered. For each fluctuating edge associated with probability p , a uniform random number x is generated such that $0 \leq x \leq 1$, and then an edge is placed in the PG network only when $x < p$. Within the VPG, fluctuating edges will have 5 bars with probability p , and 0 bars with probability $(1 - p)$. If all bars are independent, then on average a fluctuating edge can remove up to $5p$ pebbles. As such, we assign a pebble capacity of $5p$ to fluctuating edges, while quenched edges retain a pebble capacity of 5. As $p \rightarrow 1$, the pebble capacity of a fluctuating edge approaches that of a quenched edge, and as $p \rightarrow 0$, the fluctuating edge approaches that of a missing edge. An

example of how an ensemble of PG networks is represented by a single VPG network is shown in Fig. 2.2. With two fluctuating edges, the ensemble of the PG networks consist of 4 distinct constraint topologies, which are compactly represented as a single VPG network where the fluctuating edges are assigned average pebble capacities.

2.3 Constraint Counting and Pebble Games

The simplest estimate for the number of internal DOF remaining in a network is given by Maxwell Constraint Counting (MCC), which assumes that as constraints are added to the network they will be independent until the network becomes rigid [25]. This assumption goes wrong when a constraint is added to a rigid region which is nucleated by a localized, higher than average density of constraints. By suppressing fluctuations in constraint density throughout the network, MCC provides a rigorous lower bound on the number of DOF available to the network as constraints are added. It is an easy task to find the average number of constraints within a network, given the probabilities for the constraints to form. For the body-bar networks considered here, Eq. 2.1 gives the Maxwell lower bound estimate for the mean number of independent internal DOF over an ensemble of networks that have different placements of quenched and fluctuating edges.

$$F_M = L^d \max(6 - d(q_{\text{fix}} + pq_{\text{fluct}})c, 0) \quad (2.1)$$

In Eq. 2.1, the capacity of an edge is given by c , which is 5 in our study, p is the probability for a fluctuating edge to be present within the network. If there were no distance constraints at all ($q_{\text{fix}} = q_{\text{fluct}} = 0$), the total number of DOF calculated by MCC would be $6L^d$, given that 6 trivial DOF are assigned to each vertex. As q_{fix} and/or the product of probabilities, pq_{fluct} increases, more distance constraints are added to the network, which reduces F_M . Since we employ periodic boundary conditions, the total number of possible edges within the network is dL^d because there are d edges per vertex. This local connectivity count is specific to hyper-dimensional cubic lattices. In the work described here, we consider either a square lattice representing a sheet ($d = 2$) embedded in three-dimensional space, or a cubic lattice ($d = 3$). Given that every edge consumes c DOF, the maximum amount of DOF that can be consumed by all the edges associated with a given vertex is dc . However,

the actual consumption of DOF will depend upon all of the edges found throughout the network based on the probabilities mentioned above. The $\max()$ function must be used because once the network becomes rigid — the number of independent internal DOF will be zero. Once the network becomes completely rigid, no additional DOF can be removed by adding constraints, and consequently, this prevents F_M from becoming negative. In other words, once the network reaches the rigidity transition, no internal DOF are available, and the remaining edges to be added will all be redundant. An important characteristic of Eq. 2.1 is that the number of available DOF within the network is an extensive quantity, being proportional to the number of vertices within the network.

For generic body-bar networks, exact constraint counting using an integer algorithm can be implemented as a pebble game (PG) [13]. Each DOF is represented by one pebble and each distance constraint is represented by one bar. A vertex representing a 3D object has 6 pebbles to account for its 3 translational and 3 rotational DOF. The PG builds up the network from a set of isolated vertices by placing one edge at a time in a recursive fashion. When an edge placed in the network is found to be independent, it is covered by consuming a free pebble from either of its two incident vertices. During this process, the number of available free pebbles on vertices monotonically decreases. Operationally, pebbles are rearranged within the network to cover the added edges. A redundant edge is identified when a free pebble cannot be found. Although the order of edge placements affects which edge is identified as redundant or independent, the constraint counting (number of independent constraints or DOF) is independent of the ordering of the edges. Note that the correctness of this procedure has been proved based on the matroidal property of sparse graphs [19, 29].

The VPG operationally parallels the PG, except pebbles and edge capacities are described by real numbers. The term “virtual” is used because pebbles and constraints are not discrete counting entities. The capacity of an edge represents average numbers of constraints and need not be an integer. The VPG network is described by an edge-list representation, where each edge is associated with a capacity, c . The VPG network is built up from a set of isolated vertices (denoted as $\{v_i\}$) by placing one edge at a time into the network. A pebble is *free* if it is on a vertex, and *consumed* if it is used to cover an edge. An edge can

be covered by pebbles from either or both of its two incident vertices, and is *fully covered* if the sum of the consumed pebbles on both sides of the edge equals its pebble capacity. A *partially covered* edge occurs when the sum of consumed pebbles is less than the edge capacity. An edge is redundant whenever an insufficient number of pebbles is found within the network for the edge to be fully covered. An edge can never be covered in excess of its capacity. As in the PG, the directional nature of a covered edge provides a *viable path* from vertex v_i to v_j when the edge is covered by pebbles from vertex v_i , and vice versa. As a consequence of the non-integer generalization, the VPG is representative of a flow problem. For instance, it is possible to cover an edge from either of its two incident vertices resulting in a bidirectional edge. This fact guaranties an exact mapping between the directionality of individual distance constraints and the bundle of constraints represented by a current edge. The search for pebbles in this direct graph resembles a network flow problem [30,31], given that the covered capacity of any edge will determine the maximal flow of pebbles through that edge. The VPG has the additional important feature that the capacities of the directed-edges can change over time caused by the rearrangement of pebbles.

2.4 Description of the VPG Algorithm

Consider a network consisting of vertices $\{v_n\}, n = 1, 2, \dots, N$, with a list of edges $\{e_m\}, m = 1, 2, \dots, M$. The capacity for the m -th edge is denoted by c_m . The VPG follows the following procedures and operations:

1. Initialize the graph with a set of isolated vertices $\{v_n\}$, with the free DOF of each vertex v_n being 6.
2. From the list of edges $\{e_m\}$, insert edge e_k with capacity c_k into the graph. Let v_i and v_j be the two incident vertices for edge e_k .
3. Collect 6 pebbles for vertex v_i by doing a breadth first search.
4. Flag vertex v_i as visited, try to collect c_k pebbles for vertex v_j by doing a breadth first search while holding the 6 pebbles on v_i in place. If not all c_k pebbles can be found in one trial, continue to collect more pebbles by carrying out the search repetitively

until there are enough free pebbles on v_j to cover edge e_k , or if no new pebbles are found (a failed search).

5. If c_k or more pebbles are collected on vertex v_j , cover edge e_k with c_k pebbles. Otherwise, all the visited vertices within the failed search are condensed into a single vertex. If $\{e_m\}$ is not empty, go to step 2.
6. End of VPG.

A simple example shown in Fig. 2.3 illustrates VPG operations. The edge list $v_1 - 2.5 - v_2, v_2 - 5.0 - v_3, v_1 - 2.0 - v_3, v_2 - 1.5 - v_3$ is used to construct the graph. The operations described in Fig. 2.3 include: (1) pebble rearrangements between vertex and edge (Fig. 2.3a); (2) pebble backtracking (Fig. 2.3c); and (3) condensation (Fig. 2.3g). Detailed VPG pseudo-codes are provided in the appendix of the chapter.

The essential step in the VPG is to collect c_k pebbles on vertex v_j by employing a pebble search multiple times. How the pebble search is implemented does not matter provided it is exhaustive. We implemented the pebble search as a Breadth-First Search (BFS), although a Depth-First Search (DFS) is an obvious alternative. The BFS is employed only because it is physically intuitive to look for free pebbles on vertices near the vertex they are needed, rather than potentially searching over long paths that traverse to remote vertices to obtain a free pebble, despite having access to a free pebble that may only be a few neighbors away in a different direction. However, we are not aware of any change in complexity within the pebble game by using BFS over DFS, since the main controlling factor is when pebble searches fail. When a free pebble is found on a remote vertex, it is brought back to vertex v_j along the shortest route through viable edges in a backtracking process.

When a failed search occurs, all the vertices visited become part of a minimally rigid object in the body-bar representation [32]. We loosely call these failed searches “Laman subgraphs” because of the analogous correspondence with the original 2D pebble game [9]. A property of a failed search is that no directed edge can point from vertices within the Laman subgraph to vertices outside the Laman subgraph. This condition is guaranteed by the fact that the pebble search itself failed to find any excess free pebble. Once a failed search occurs, the vertices that comprise the Laman subgraph are subsequently condensed

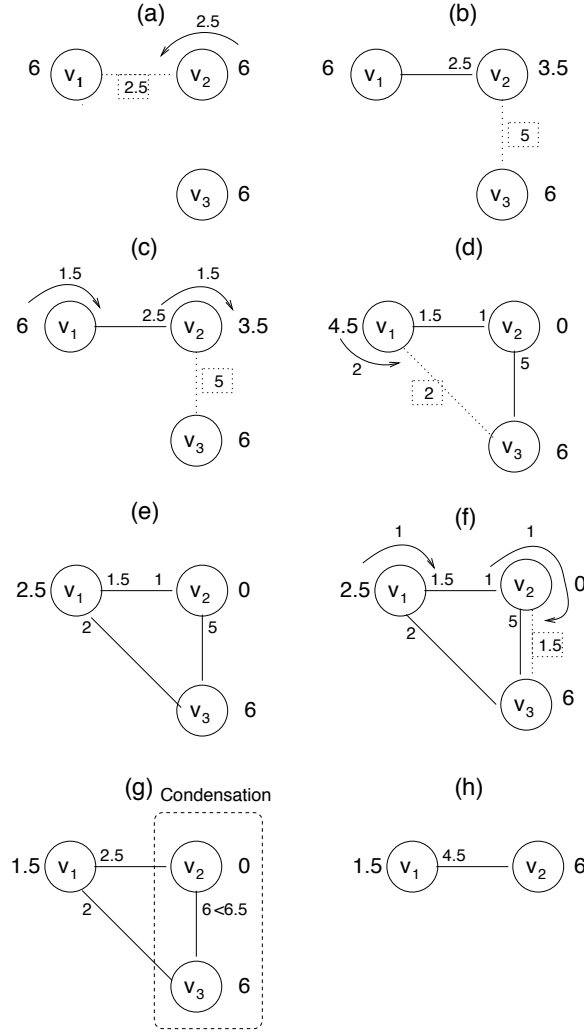


Figure 2.3: Application of the VPG on a 3 vertex network. A dashed line denotes the edge that is being added at the current step, whose capacity is indicated in a dashed box. Appearing at the ends of an edge are the number of pebbles consumed from the corresponding vertex used to cover the edge. (a-b) Each vertex has 6 pebbles after initialization, and an edge of capacity 2.5 is added to the graph. (c) Vertex v_2 has 3.5 free pebbles and cannot fully cover the new edge between v_2 and v_3 (which requires 5 pebbles). A pebble search is carried out and 1.5 pebbles are backtracked through the edge between v_1 and v_2 . (d-e) Vertex v_1 has enough free pebbles to cover the newly added edge. (f) Adding an additional edge between v_2 and v_3 , the two edges (of capacity 5 and 1.5) combine, yielding a partially covered edge with capacity 6.5. (g) Because the edge between v_2 and v_3 cannot be fully covered, the pebble search has failed and vertices v_2 and v_3 condensate into a single vertex denoted as v_2 . (h) Edges v_1-v_2 and v_1-v_3 in step (g) are combined into one edge v_1-v_2 .

into a representative vertex having 6 pebbles. The process used in the VPG is identical to that used in the corresponding body-bar 3D PG, which is nearly the same as that used in the 2D Pebble Game [9]. Specifically, no vertices within a Laman subgraph are removed. Rather, the connectivity of the underlying graph is modified, such that the Laman subgraph is represented as a star graph, where the sole vertex having 6 pebbles is selected as the representative condensed vertex, and all other vertices that are depleted of pebbles are directly connected to this center vertex with 5 bars. In this way, future pebble searches are dramatically shortened, and the effective size of the network continues to decrease as more Laman subgraphs are detected. This implementation is both simple and very effective, transforming the typical performance from $O(N^2)$ to $O(N)$.

2.5 Results and Discussion

The average number of internal DOF per vertex (F/N) as a function of the fluctuating edge probability, p , is shown in Fig. 2.4 at four exemplar values of q_{fix} and q_{fluct} for four algorithms: MCC, VPG, ensemble averaged PG and ensemble averaged bar-PG. The bar-PG denotes another type of body-bar network that generally allows $\{0, 1, 2, 3, 4, 5, 6\}$ generically placed distance constraints between rigid bodies. In the context of the work presented here, the bar-PG assigns 5 and 0 distance constraints for quenched and missing edges respectively, and based on a random mechanism there can be 0 to 5 distance constraints within a fluctuating edge. The method for how distance constraints are distributed within fluctuating edges is described in detail below. In the latter three cases, the results are for the same set of quenched, fluctuating and missing edges placed within a network of 8000 vertices ($L = 20$, $d = 3$). For the two PG cases, ensemble averaging is performed over 100 randomly generated realizations that determine for each fluctuating edge whether it is present or missing. Like the PG, the VPG results for F are independent of the order that edges are placed in the network, which was extensively verified.

As expected, the results show that MCC overestimates the minimum number of constraints needed for the network to become rigid. When invoking MCC, the rigidity threshold is defined as the lowest value of p for which $F = 0$, because when $F > 0$ the network is flexible, and when $F = 0$ the network is rigid. However, in the PG, a network will generally have

many localized rigid and flexible regions. This inhomogeneity is caused by fluctuations of constraint density due to statistical variations of fluctuating constraints in different network realizations. As such, the rigidity threshold for the PG is below the MCC prediction.

The VPG results are much closer to PG calculations than MCC, especially on the rigid side of the rigidity transition, while F is also underestimated compared to the PG results across intermediate values of p . Relative to MCC, the improved accuracy of the VPG occurs because it applies a mean-field approximation locally at the edge level. In contrast to a global mean-field approximation that enforces an average constraint density across the entire network in MCC, the VPG averages the constraint density at the edge level, and distinguishes between quenched, missing and fluctuating constraints. At the edge level, the VPG replaces the random sampling of realizations of placing distance constraints with its average value without regard to the underlying random process. For example, for the test body-bar networks described above, a fluctuating edge is associated with either a bundle of 5 distance constraints present with probability, p , or no distance constraints with probability, $(1 - p)$. On average, this random process gives $5p$ distance constraints, per fluctuating edge.

It is interesting to note that due to the cooperative nature of the distance constraints that are used to represent a fluctuating edge (i.e., either all 5 distance constraints or nothing) in the PG network, the fluctuations in the number of distance constraints for a fluctuating edge is maximal in the PG network. Because the VPG is a mean-field approximation, VPG and ensemble averaged PG results could be in better agreement if the constraint density fluctuations at the edge level were less. Although the nature of chemical bonding imposes this cooperativeness when studying molecular networks, applications of the VPG can extend beyond molecular networks. Therefore, a different type of body-bar network that minimizes the constraint density fluctuations within edges is considered using a “bar-PG”.

In the bar-PG, quenched, missing and fluctuating edges are placed in the same way as described above. However, each of the five distance constraints representing a fluctuating edge are placed independent of each other, each with probability p . Thus a fluctuating edge is now allowed to have $\{0, 1, 2, 3, 4, 5\}$ distance constraints, instead of either 0 or 5. Here, two possible random events defined by whether a distance constraint is present

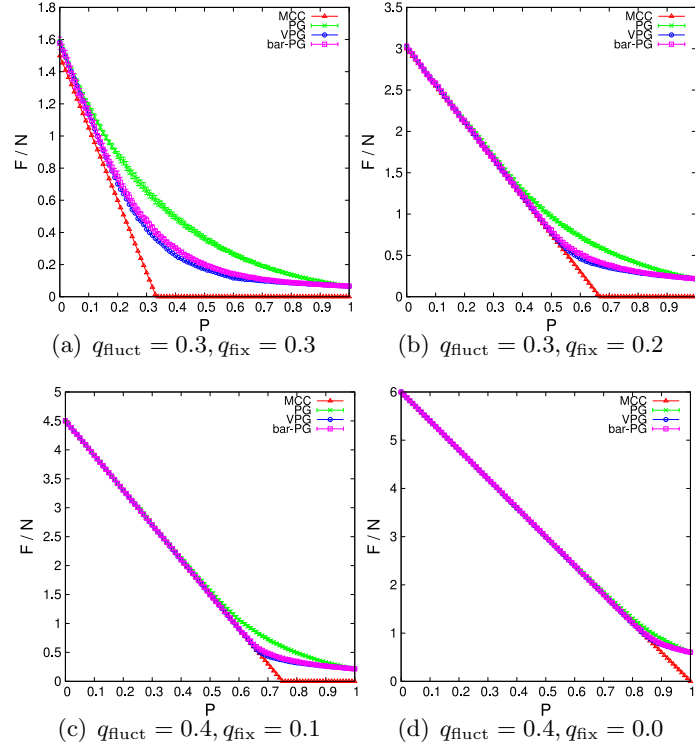


Figure 2.4: The average number of internal DOF within the network, F , for $L = 20$ are plotted as a function of fluctuating edge probability, p , based on Maxwell counting (red triangle), ensemble averaged PG (green cross), VPG (blue circle), and ensemble averaged bar-PG (purple square). The straight line is the lower bound estimate of F characteristic of MCC. The VPG always falls between the results for the PG and MCC, being a better approximation to the PG than MCC. The maximum difference in F between the PG and VPG occurs at values of p close to the Maxwell rigidity transition.

or absent is considered 5 independent times drawn from a binomial distribution for each of the 5 distance constraints that comprise a fluctuating edge. In other words, the single binomial distribution that was controlling whether a fluctuating edge is present (with 5 distance constraints) or absent (no distance constraints), is replaced with 5 identical binomial distributions, one for each distance constraint. Since the average frequency for the presence of a particular distance constraint is p , the edge maintains the same overall average of $5p$ distance constraints. Therefore, the two models share the same average property, while the fluctuation within the edge is greatly reduced in the bar-PG, compared to the initial body-bar PG. The VPG is based on the average pebble capacity of an edge, which is identical for the PG and bar-PG. The average number of constraints for a fluctuating edge remains $5p$, and the VPG results as well as the MCC results remain unchanged. However, the ensemble average results of the PG (see Fig. 2.4) for the bar-PG networks yields results that are markedly close to the VPG.

In summary, MCC suppresses constraint density fluctuations within the network, while the VPG suppresses constraint density fluctuations within individual edges. Therefore, there is yet another important limit to check when applying the VPG. If there are no constraint density fluctuations within any edge throughout the entire network, then the VPG results must be identical to the PG. Indeed, this has been verified by applying the VPG on a single realization that is used for a PG, and we find that the results are identical. This result is a verification of the fact that the VPG and the PG are identical for the cases when $p = 0$ and $p = 1$, as discussed above. In view of this result, it becomes clear that the VPG results will compare best to the ensemble averaged PG results in the limit that $q_{\text{fluct}}/q_{\text{fix}}$ is well below 1. One application that motivated this work is to study protein structures where this ratio is about 5%. In general, it is expected that $q_{\text{fluct}} = 1$ will yield the greatest difference in results between the VPG and the ensemble averaged PG.

For a comprehensive comparison between the VPG and PG results, contour plots given in Fig. 2.5 show the maximum deviation of the internal DOF per vertex between the two

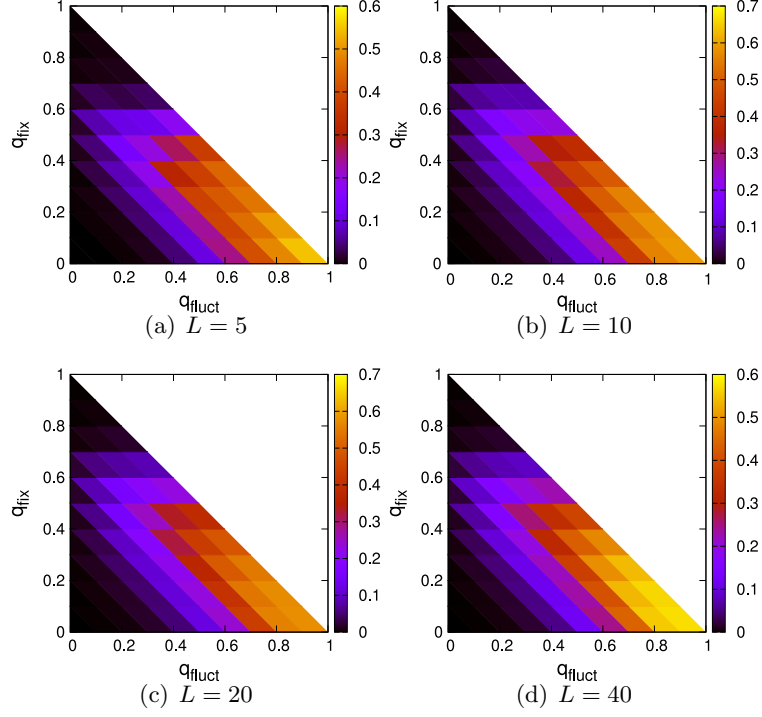


Figure 2.5: Contour plots showing the maximum difference between F_{pg} and F_{vpg} per vertex across four different system sizes.

methods, which is given by

$$\frac{\Delta F}{N} = \frac{\max[F_{\text{pg}}(p) - F_{\text{vpg}}(p)]}{N}. \quad (2.2)$$

Note that the maximum deviation $\Delta F/N$ in Eq. 2.2 is determined by scanning p in the range $(0, 1)$, where the values of p at which the maximum occurs vary for different q_{fix} and q_{fluct} values. As expected, the maximum difference occurs when $q_{\text{fluct}} = 1$, where $\Delta F/N \sim 0.6$, which is about 10% error considering there are 6 DOF per vertex. However, the maximum deviation depends on the details of the network topology. Based on the discussions above, both MCC and VPG are more accurate when the density of constraints in the network is more uniform.

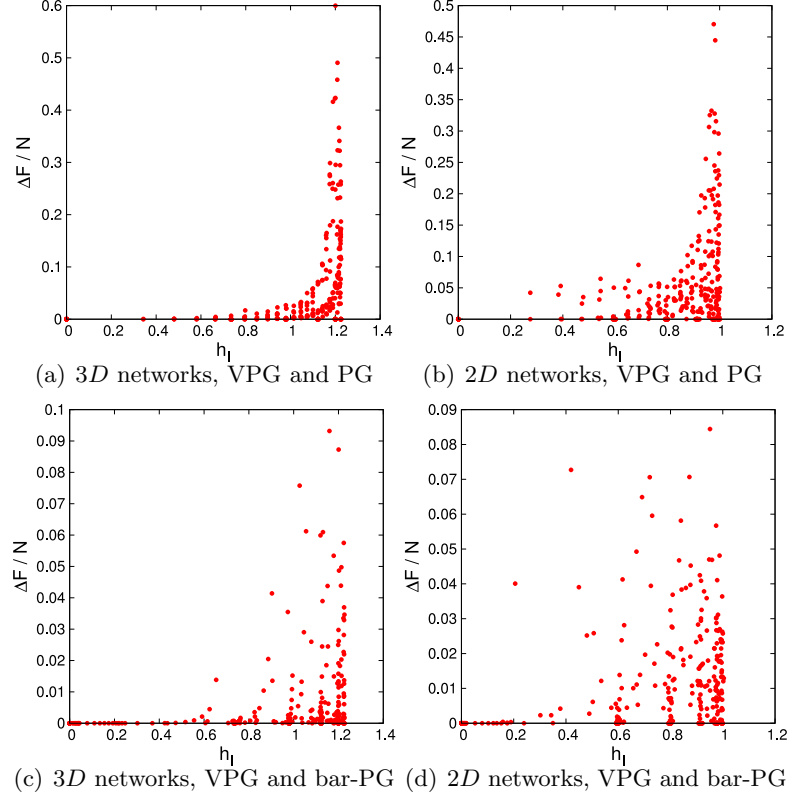


Figure 2.6: The maximum difference of internal DOF $\Delta F/N$ between the PG and VPG is plotted versus the heterogeneity index. The data are collected for different combinations of q_{fix} and q_{fluct} . Note the sharp increase in $\Delta F/N$ at a threshold value h_I^c , which is ~ 1.0 for 2D systems, and ~ 1.2 for 3D systems. Below this threshold value, the difference is negligibly small. Panels (a) and (b) correspond, respectively, to three and two dimensional lattices with $L = 20$. Similarly, panels (c) and (d) show that the differences are much smaller between the VPG and the bar-PG.

To better understand the computational accuracy of the VPG in relation to the heterogeneous character of the network topology, a quantitative measure, the *heterogeneity index*, is introduced. We define the heterogeneity index, h_I , as the standard deviation of the coordination number (degree of a vertex) across all vertices throughout the network. This index is given by

$$h_I = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (C_i - \bar{C})^2} \quad (2.3)$$

where C_i is the coordination number of vertex v_i , and \bar{C}_i is the average coordination of the network. The heterogeneity index was calculated based on simulation. Operationally, once a network is generated as described above, where fluctuating edges have been realized as either being present or not, then, a simple count of edges connecting to each vertex is averaged. Similarly, the standard deviation, given in Eq. 2.3 is applied. The heterogeneity index quantifies the amount of variation of vertex connectivities throughout the network.

In agreement with our intuition, the difference in the internal DOF between the PG and VPG is negligibly small when h_I is small. The difference exhibits a sharp increase at some characteristic value h_I^c (≈ 1.2 for $d = 3$, and ≈ 1.0 for $d = 2$, see Figs. 2.6a and 2.6b). Figure 2.6 provides a guidance regarding the accuracy of the VPG when applied to real systems. The VPG is an excellent approximation to the PG when the heterogeneity index of the network is below the critical value h_I^c . Interestingly, similar behavior between the 2D and 3D networks is found. The main reason for constructing a 2D network is because fluctuations are generally more important in lower dimensions (mean-field approximations become more accurate in higher dimensions). As seen in these results, the robustness of the VPG holds up in 2D networks markedly well, suggesting the heterogeneity index is an excellent quantity to characterize the accuracy expectation of the VPG by looking at local connectivity, without concern for the global topological character of the network.

The above analysis reveals that the VPG is more accurate for homogeneous networks. However, as noted above, the accuracy of VPG is only affected by the density fluctuations at the edge level, not necessarily at the network level. To test this aspect, the maximum deviation of internal DOF between VPG and bar-PG is plotted against the heterogeneity index for the 3D and 2D cases. In both cases (Figs. 2.6c and 2.6d) a similar qualitative dependence on h_I is found (as compared to the difference between VPG and PG). Most significantly, the scale of the errors is dramatically decreased by almost an order of magnitude, demonstrating that the VPG is capturing information about the detailed local properties of the network topology. In most regimes of the parameter space $\{q_{\text{fix}}, q_{\text{fluct}}, p\}$, the VPG will provide markedly accurate detailed information regarding the rigidity of the network.

After placing all the constraints, the PG and VPG determine the number of DOF left in the network. When the network is globally rigid, just the six trivial DOF are remaining,

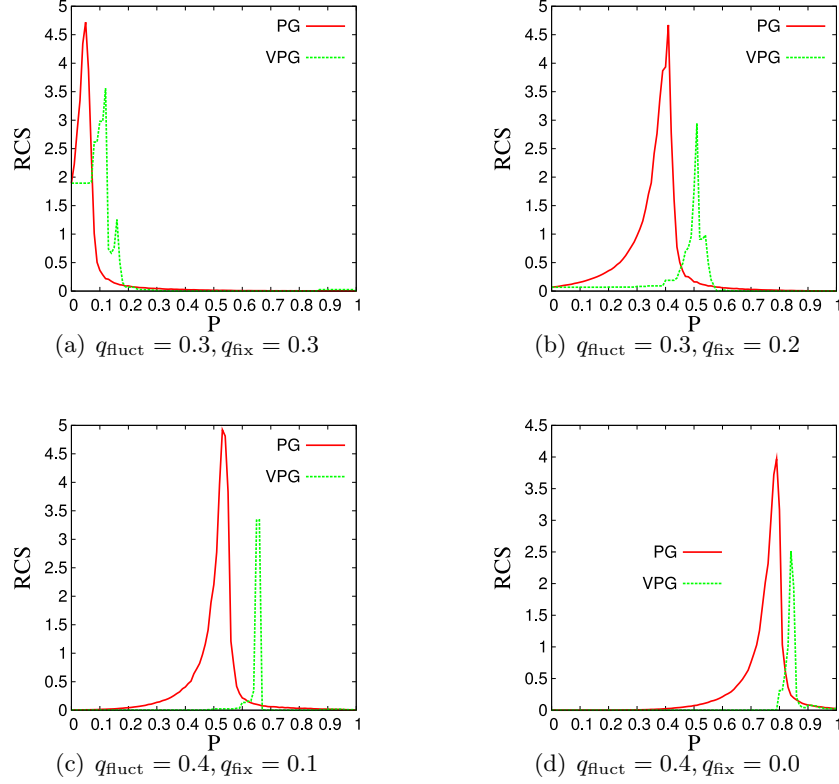


Figure 2.7: Rigid Cluster Susceptibility curves for the four characteristic cases shown in Fig. 2.4.

identifying a single rigid body (rigid cluster). The number of pebbles exceeding the trivial DOF represent the degree of flexibility within the network. The localization of these DOF help identify flexible and rigid regions. For two vertices to belong to the same rigid cluster, there must be a maximum of six DOF between them. In order to identify rigid clusters, the counting of pebbles is carried out for all possible pairs of vertices in the network. Once all the vertices are assigned to a rigid cluster is possible to calculate the Rigid Cluster Susceptibility (RCS) curve. The RCS is defined as the reduced second moment in rigid cluster size. That is, the peak in RCS identifies the point where the rigid cluster sizes are maximally fluctuating, indicating a transition from a globally flexible to globally rigid network (rigidity transition). In figure 2.7 we compared the RCS for the four exemplar cases from figure 2.4. These cases were chosen to analyze the behavior of the VPG algorithm when there is a high percentage of fluctuating and missing edges, i.e., in all four cases this percentage is more than 70%. According to this figure, the RCS curves qualitatively show a very similar behavior for both algorithms, meaning that the rigidity transition is well

detected by the VPG.

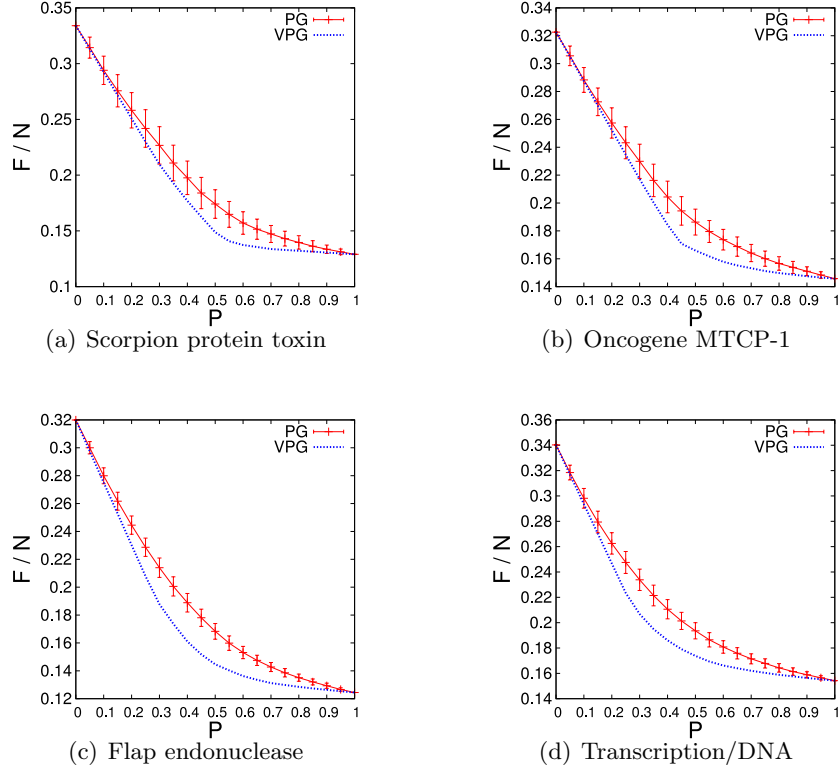


Figure 2.8: The average number of internal DOF within the network, F , for four proteins are plotted as a function of fluctuating edge probability, p , based on ensemble averaged PG and VPG. For PG the standard deviation is shown for comparison purposes. The VPG gives a very good approximation to the results for the PG. The maximum difference in F between the PG and VPG occurs at values of p with maximal fluctuations.

To show the validity of the VPG algorithm we extend the calculation of DOF to proteins. The procedure to model the protein networks is similar to the one we employed to create cubic lattices, all the covalent bonds are represented by quenched edges, while the hydrogen bonds are represented by fluctuating edges. For this analysis we use four proteins that are non redundant at SCOP [33] family level: a scorpion toxin (pdbid = 1AHO) [34], the biomedical relevant oncogene MTCP-1 (pdbid = 1A1X) [35], the FLAP endonuclease from *M. jannashii* (pdbid = 1A76) [36] and a DNA transcription regulator (pdbid = 3COQ) [37]. Figure 2.8 shows the calculation of DOF by both algorithms over this protein dataset, the PG algorithm was averaged over 200 realizations. All four calculations on proteins are consistent with previous results on lattices. The region that shows a larger difference corresponds to the rigidity transition and it is where the PG presents a larger

standard deviation as well. VPG continues being a very good approximation to the actual calculation of the PG ensemble averaged.

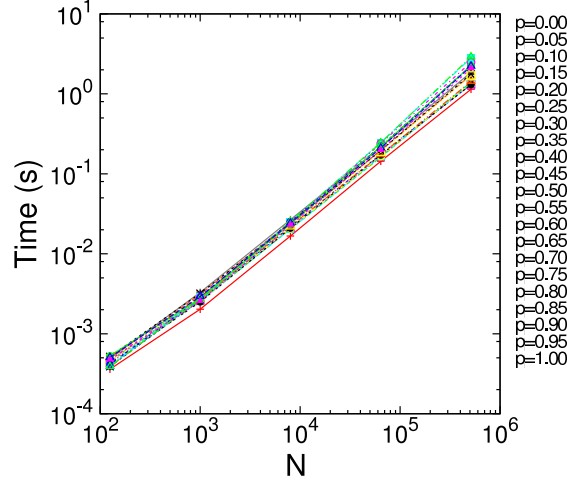


Figure 2.9: VPG execution times are plotted versus the number of vertices N within the network, for 21 uniformly spaced values of p for $q_{\text{fix}} = 0$ and $q_{\text{fluct}} = 1$. In all cases, the execution time is essentially linear with respect to N .

The last aspect of the VPG to characterize here is to show the execution times for the VPG is comparable to a single PG on a fixed network. As demonstrated in Fig. 2.9, the execution time of the VPG (and PG) scales approximately linearly with the number of vertices, N , in the network. This time benchmarking holds up for all regions of parameter space $\{q_{\text{fix}}, q_{\text{fluct}}, p\}$, except exceedingly close to the rigidity transition, where the VPG scales as $O(N^2)$. It is generally found from simulation on networks that model molecular structure that condensation improves the scaling of the VPG from $O(N^2)$ to almost always $O(N)$ above the rigidity transition. This dramatic increase in performance characteristic follows the same response as that of the 2D and 3D PGs. Here, “almost always” means that as long as the network is not very near the rigidity transition, we empirically find a linear dependence on the number of vertices in the network. Interestingly, for the case of a complete lattice with all fluctuating edges ($q_{\text{fix}} = 0$, $q_{\text{fluct}} = 1$, $0 < p < 1$), the rigidity transition is in perfect agreement with the Maxwell prediction (where the critical transition probability p_c is given by the p in Eq. 1 that sets $F_m = 0$). Note that the VPG recovers MCC when no constraint density fluctuations are present. In this atypical case, the scaling of the VPG is found to be $O(N^2)$ at very near the rigidity transition, where p must be well

within a 1% deviation from p_c . The worse case situation is very difficult to find, and almost will never happen in practice. However, because the accuracy of the VPG algorithm is comparable to the ensemble average over hundreds of PG network realizations for relatively homogeneous systems, the VPG provides a tremendous increase in computational speed of rigidity analyses in the desired applications.

2.6 Conclusions

In this chapter, a Virtual Pebble Game (VPG) algorithm is given that estimates the average number of internal degrees of freedom (DOF) in a generic body-bar network that has fluctuating distance constraints. The accuracy of the VPG is compared directly with ensemble averaged estimates by repetitively employing an exact Pebble Game (PG) a hundred times to perform random sampling. In the network representation for the PG, each edge corresponds to an integer number of distance constraints, and vertices have integer number of free pebbles. In the VPG, an ensemble of networks is represented as a single effective network, where each edge represents an average number of distance constraints. Consequently, fractional number of pebbles are associated with edges and vertices, which flow through the network. The VPG has comparable execution time to one run of the PG, where both algorithms almost always scale linearly with the number of vertices in the network.

The VPG provides an accurate mean-field approximation to the PG algorithm at the level of individual edges, rather than across the entire network. Because of its mean-field nature, the VPG calculation deviates from the ensemble averaged PG when the network becomes more heterogeneous in distance constraint density. A heterogeneity index measure (h_I) that quantifies the standard deviation in the coordination number of vertices in a network provides a good indicator to the degree of accuracy of the VPG. The average number of internal DOF estimated by the VPG is less than 5% error when h_I is below a critical value, which is $h_I^c \approx 0.7$ for the 2D square lattice, and $h_I^c \approx 1.1$ for the 3D cubic lattice. Beyond these critical values the VPG errors increase, but are always less than 10%, and are rarely that high.

The comparison of both algorithms at their calculation of Rigid Cluster Susceptibility

(RCS) curves on lattices holds optimistic expectations to apply the rigid cluster decomposition on protein networks. This fact is possible since VPG facilitates detailed analyses of network rigidity properties that is impossible under the global mean-field approximation employed by MCC. To show the validity of the VPG, we extended the calculation of DOF to proteins. The results are promising and consistent with our previous calculations on lattices where the accuracy was well characterized.

2.7 Appendices: VPG Pseudo-Codes

In this section, we present the pseudo-codes for the virtual pebble game algorithm. Our pseudo-codes are written such that they are independent of the data structures representing the network.

Description of the algorithms

1. The *virtual_pebble_game*(*bars*, *N*) algorithm considers a list of bars and tries to cover them in full one at a time. When covering a bar, we first pull back the 6 pebbles associated with one of the vertices corresponding to the bar. Rigidity theory guarantees that 6 pebbles can be pulled back for any vertex. Then a second search is carried out for the other vertex corresponding to the bar, while blocking access to the first vertex. If enough pebbles can be found to cover the bar, the search is successful; otherwise the search fails and all the vertices that are visited during the search condensate to one representative vertex. The same procedure is repeated for all the bars.

2. The *try_to_find_pebbles*(*u*, *pebbles_required*, *blocked_vertex*) algorithm is applied to pull back pebbles to a vertex. This algorithm stops when either one of two conditions is reached: the total pebbles found is equal to the pebbles required or there are no more free pebbles to pull back. The visited vertices are added to the *visited_vertices* list to block further visits in the current search.

3. The *collect_pebbles*(*vertex_queue*, *visited_vertices*, *pebbles_required*) algorithm is a BFS algorithm that is applied when searching the network for free pebbles. It continues searching for free pebbles until either one of two conditions is satisfied: (1) there are no more vertices to visit in the *vertex_queue* or (2) the amount of pebbles found is equal to the amount of pebbles required. When backtracking the pebbles the algorithm needs to make

sure that vertex containing free pebbles are accessible through a viable path from the root vertex.

4. The *back_track(pebbles_found, v, route)* algorithm follows the route dictated by the visited vertices list to pull back the pebbles found in the search up to the root vertex.

5. The *collapse_vertices(visited_vertices)* algorithm is employed after a pebble search fails. This algorithm chooses the vertex with the lowest index as the root, and the vertices in the visited_vertices list all assume the same index as the root index. Hence all the vertices in the visited_vertices list condensate to one representative vertex – the root.

Algorithm 1 virtual_pebble_game(bars, M)

```

1: edge_capacity = {0, 0 ... 0}
2: vertex_capacity = {6, 6 ... 6}
3: for  $i = 1$  to  $M$  do
4:    $b \leftarrow \text{bars}[i]$ 
5:    $u \leftarrow \text{first vertex of } b$ 
6:    $v \leftarrow \text{second vertex of } b$ 
7:    $c \leftarrow \text{capacity of } b$ 
8:   blocked_vertex  $\leftarrow$  none
9:   [visited_vertices, collected_pebbles]  $\leftarrow$  try_to_find_pebbles( $u$ , 6, blocked_vertex)
10:  blocked_vertex  $\leftarrow$   $u$ 
11:  [visited_vertices, collected_pebbles]  $\leftarrow$  try_to_find_pebbles( $v$ ,  $c$ , blocked_vertex)
12:   $e \leftarrow \text{edges}(u, v)$ 
13:  edge_capacity[ $e$ ]  $\leftarrow$  edge_capacity[ $e$ ] + collected_pebbles
14:  vertex_capacity[ $v$ ]  $\leftarrow$  vertex_capacity[ $v$ ] - collected_pebbles
15:  if collected_pebbles <  $c$  then
16:    collapse_vertices(visited_vertices)
17:  end if
18: end for

```

Algorithm 2 try_to_find_pebbles(u , pebbles_required, blocked_vertex)

```

1: total_pebbles_found  $\leftarrow$  0
2: fail_flag  $\leftarrow$  false
3: while (total_pebbles_found < pebbles_required) and (fail_flag is false) do
4:   visited_vertices  $\leftarrow$  empty
5:   add blocked_vertex to visited_vertices
6:   vertex_queue  $\leftarrow$   $u$ 
7:   total_pebbles_found  $\leftarrow$  collect_pebbles(vertex_queue, visited_vertices, pebbles_required)
8:   capacity[ $u$ ]  $\leftarrow$  total_pebbles_found
9: end while
10: return (visited_vertices, total_pebbles_found)

```

Algorithm 3 collect_pebbles(vertex_queue, visited_vertices, pebbles_required)

```

1: route  $\leftarrow$  empty
2: filled_flag  $\leftarrow$  false
3: total_pebbles_found  $\leftarrow$  0
4: while (vertex_queue is not empty) and (filled_flag is false) do
5:     w  $\leftarrow$  front of vertex_queue
6:     add w to visited_vertices
7:     p_deficit  $\leftarrow$  pebbles_required - total_pebbles_found
8:     pebbles_found  $\leftarrow$  min(p_deficit, vertex_capacity[w])
9:     if pebbles_found > 0 then
10:         pebbles_back_tracked  $\leftarrow$  back_track(pebbles_found, w, route)
11:         if pebbles_back_tracked < pebbles_found then
12:             limiting_edge_found  $\leftarrow$  true
13:         end if
14:         pebbles_found  $\leftarrow$  pebbles_back_tracked
15:     end if
16:     total_pebbles_found  $\leftarrow$  total_pebbles_found + pebbles_found
17:     if total_pebbles_found  $\geq$  pebbles_required then
18:         filled_flag  $\leftarrow$  true
19:     else if limiting_edge_found is false then
20:         for each edge e associated with vertex w do
21:             t  $\leftarrow$  second vertex of e
22:             if e is covered by pebbles and vertex t has not been visited then
23:                 t  $\leftarrow$  second vertex of e
24:                 route[t]  $\leftarrow$  e
25:             end if
26:         end for
27:     end if
28:     pop out front vertex from vertex_queue
29: end while
30: return total_pebbles_found

```

Algorithm 4 back_track(pebbles_found, v, route)

```

1:  $e \leftarrow \text{route}[v]$ 
2: while  $e$  is not end of route do
3:    $\text{diff} \leftarrow \text{pebbles\_found} - \text{edge\_capacity}[e]$ 
4:   if  $\text{diff} > 0$  then
5:      $\text{pebbles\_found} \leftarrow \text{edge\_capacity}[e]$ 
6:      $\text{edge\_capacity}[e] \leftarrow 0$ 
7:      $t \leftarrow \text{second vertex of } e$ 
8:      $\text{vertex\_capacity}[t] \leftarrow \text{vertex\_capacity}[t] + \text{diff}$ 
9:   else
10:     $\text{edge\_capacity}[e] \leftarrow \text{edge\_capacity}[e] - \text{pebbles\_found}$ 
11:   end if
12:    $e_r \leftarrow \text{reverse of edge } e$ 
13:    $\text{edge\_capacity}[e_r] \leftarrow \text{edge\_capacity}[e_r] + \text{pebbles\_found}$ 
14:    $v \leftarrow \text{first vertex of edge } e$ 
15:    $e \leftarrow \text{route}[v]$ 
16: end while
17: return pebbles_found

```

Algorithm 5 collapse_vertices(visited_vertices)

```

1: root  $\leftarrow$  vertex with lowest index in visited_vertices
2: while visited_vertices is not empty do
3:   current_vertex  $\leftarrow$  front of visited_vertices
4:   for each edge  $e$  associated with current_vertex do
5:      $t \leftarrow \text{second vertex of } e$ 
6:     if  $t$  is not present in visited_vertices then
7:       first vertex of  $e \leftarrow$  root
8:     end if
9:   end for
10:  remove current_vertex from visited_vertices
11: end while

```

CHAPTER 3: CALCULATING ENSEMBLE AVERAGED DESCRIPTIONS OF PROTEIN RIGIDITY WITHOUT SAMPLING IN A PROTEIN DATASET

3.1 Introduction

In chapter two we mainly compared the calculation of DOF by the VPG and PG algorithms over body-bar networks on cubic lattices. The results were positive, supporting optimistic expectations for the application of VPG. Following the experimental design, several questions were raised from those comparisons, for example: *how would the VPG behave on protein networks?, does the comparison of the algorithms on the calculation of DOF is enough to suggest that VPG is a good alternative to PG?, would the mechanical calculations made by VPG be consistent on larger datasets?* This chapter addresses these questions regarding the applicability of the VPG algorithm in a broader context.

In this chapter, we look to assess the suitability of the VPG algorithm on the proteins realm. We make an extensive comparison of both algorithms across a nonredundant data set of 272 protein structures. Our comparisons are based on the rigid clusters identified by both algorithms, which represent groups of atoms that behave as a single body. We show through a number of metrics that the VPG rigidity calculations represent an overwhelming majority of the ones performed by the PG. Varying the number of interactions present in the network allows us to identify the rigidifying effect that they have on protein structure. Through this variation we are able to find the rigidity threshold that identifies the greatest number of fluctuations within the network, which therefore represents the greatest difference between the PG ensemble averaged ($\overline{\text{PG}}$) and VPG. Remarkably, the similarity between the two algorithms at this threshold, previously viewed as the folded/unfolded transition point [38], is surprisingly high. As such, the VPG is ideally positioned as a viable alternative to ensemble averaging in the characterization of protein rigidity.

3.2 Materials and Methods

3.2.1 Protein Structure Description

We consider a dataset composed of 272 protein structures that are nonredundant at the SCOP [33] family level. Our dataset includes one, two and three domain proteins (see appendix of this chapter for PDB codes), that range from dozens to 800 residues. We focus on three types of chemical interactions, which are: *intra-residue*, *linker* and *hydrogen bond*. The intra-residue interaction models the covalent bonds that exist within a residue. The linker interaction represents the peptide bond that connects the C-N terminal atoms in adjacent residues. The reason we make a distinction between these two types of covalent bonds is due to the number of DOF they consume. While an intra-residue bond consumes five (leaving one for the dihedral angle), the linker consumes six (locking any possible rotation) due to the partial double bond character of the amide group. The last interaction is the hydrogen bond (H-bond), which we specifically control whether a H-bond is present or not by the parameter $0.0 \leq P_{nat} \leq 1.0$. In this fashion, all possible H-bonds within the structure are present when $P_{nat} = 1.0$, whereas no cross-linking H-bonds exist when $P_{nat} = 0.0$. An independent H-bond consumes three DOF in order to account for the distance and angular constraints it imposes.

A constraint topology file (CTF) contains a list of all the possible interactions that are to be considered within a specific protein structure. It is constructed from the original PDB file. The CTF defines each interaction type, as well as their probabilities. Quenched covalent interactions never change from one CTF to another, whereas fractional probabilities occur in H-bonds due to thermal fluctuations as described by the variable P_{nat} . Fig. 3.1 compares the PG and VPG descriptions of a toy network with eight nodes, where quenched covalent bonds are solid and H-bonds are dashed. Two possible H-bonds exist in this example, leading to an ensemble of 2^2 PG networks. Within each realization, the H-bond is either fully present ($P_{nat} = 1$) or not ($P_{nat} = 0$). $\overline{\text{PG}}$ properties are determined by averaging over the ensemble. Conversely, the VPG requires only one probabilistic network to describe the ensemble because each H-bond presence is scaled by its fractional P_{nat} value.

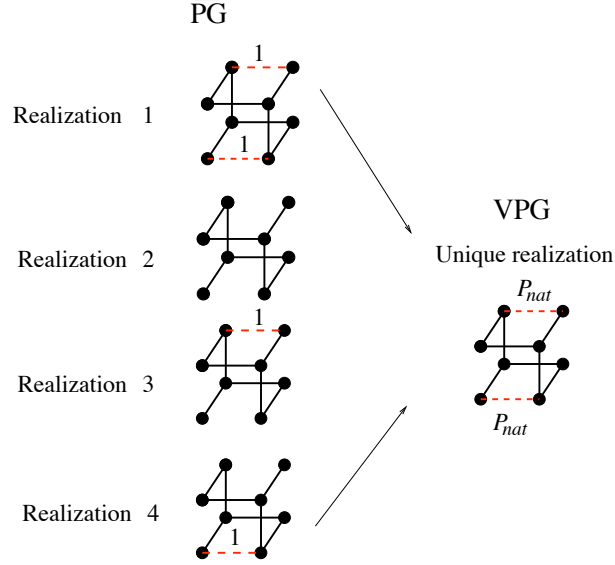


Figure 3.1: The respective network descriptions are compared. Equilibrium rigidity properties (designated as $\overline{\text{PG}}$) are calculated by averaging across an ensemble of binary networks where H-bonds are either present or not. Conversely, the VPG describes the network with fractional H-bond probabilities.

3.2.2 Rigid Cluster Decomposition

After having placed all the constraints, the PG and VPG algorithms determine the number of DOF left in the network. The trivial case is when there are just six DOF remaining, meaning that the network is globally rigid and all atoms are contained in a single rigid cluster. When there are greater than six remaining DOF, pebble location identifies which regions of the protein network are flexible or rigid. Excess pebbles identify flexible regions, whereas rigid regions occur when no free pebbles are accessible. From this information, it is possible to apply a *Rigid Cluster Decomposition* (RCD) to localize groups of atoms that move together as a whole body. A rigid cluster is a subgraph with all of its atoms completely rigid among themselves. The process of finding rigid clusters proceeds as follows: for any pair of vertices add a hypothetical edge, then try to cover it with $\epsilon > 0$ DOF, always fixing the six DOF on one of the incident vertices. If ϵ DOF is found, then both vertices do not belong to the same rigid cluster, otherwise a failed search is declared and all the vertices involved in the search are part of the rigid cluster. Fig. 3.2 presents two example of RCD cases. Notice that all the edges have been covered and they have different

capacities. In the first case (Fig. 3.2a), there is a total of 7.4 available DOF, therefore for any pair of vertices always is going to be possible to gather $6 + \epsilon$ DOF, causing the three vertices to be independent with respect to each other. For the second case (Fig. 3.2b), the number of available DOF is exactly six (on vertex four), therefore no ϵ DOF will be found under any circumstance, thus creating a rigid cluster that includes all five vertices.

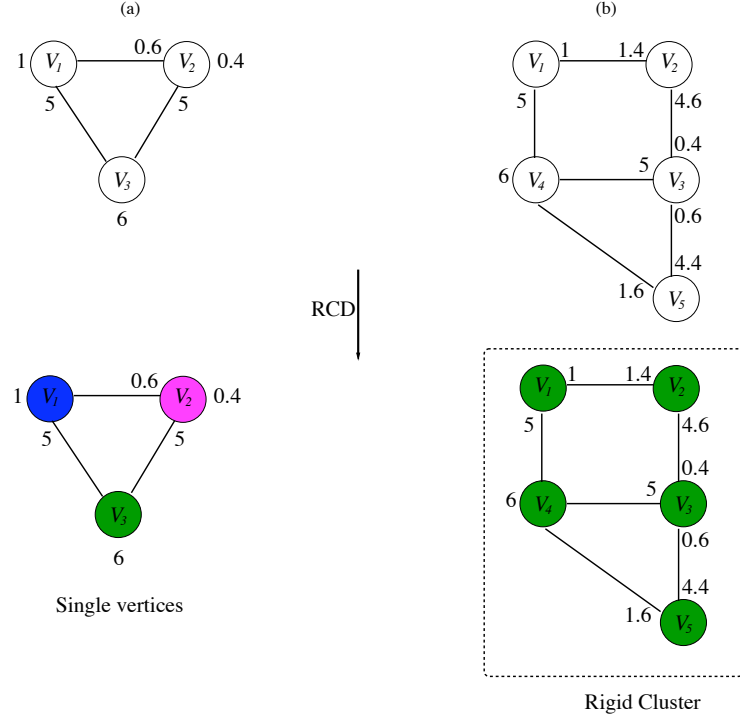


Figure 3.2: As discussed in the text, two different cases of cluster decomposition are compared. Since $\epsilon > 0$ DOF are present in the example on (a), the three vertices result in single bodies. Conversely, only the six trivial DOF can be found in the example on (b), meaning the five vertices belong to the same rigid cluster.

3.2.3 Network Similarity Metrics

We primarily employ two distinct metrics to compare the networks identified by the VPG and PG algorithms. To quantify rigid cluster similarity, we employ the Rand Measure (RM) [39]. The RM is a very well suited metric to compare clusters of elements, if both clusters are identical the RM is equal to 1.0, otherwise it is equal to 0.0. This metric is a combinatory count of all possible pairs of elements in the clusters, meaning it counts the number of cluster composition coincidences between the two approaches. If for a specific pair of vertices, the two approaches conclude that they belong whether to the same cluster

or different cluster, it is count as a match. Therefore, the RM is calculated by the number of matches divided by the total number of possible pairs. A formal definition as given in [39] is: given N points, X_1, X_2, \dots, X_N , and two clusterings of them $Y = \{Y_1, \dots, Y_{k_1}\}$ and $Y' = \{Y'_1, \dots, Y_{k_2}\}$ there is defined

$$c(Y, Y') = \sum_{i < j}^N \lambda_{ij} / \binom{N}{2}$$

where $\lambda_{ij} = 1$ if $k, k' \exists$ such that X_i and X_j are in both Y_k and $Y'_{k'}$ or if X_i is in both Y_k and $Y'_{k'}$ while X_j is in neither Y_k nor $Y'_{k'}$, otherwise $\lambda_{ij} = 0$.

We also compare the rigidity assessment of a dihedral angle at every particular protein in the dataset to analyze the agreement at a more sensitive level. That is, we count the number of times that both approaches agree in their rigid versus flexible assessment, normalized by the total number of comparisons. This calculation leads to an agreement measure (AM) that ranges from -1 to 1. When the VPG matches the majority of PG realization rigidity estimations (i.e., torsion is flexible or locked), the AM equals 0. When the VPG fails to match the majority of PG designations, the AM varies towards ± 1 (-1 = flexible and +1 = rigid). The variance from 0 indicates the proportion of disagreement. To calculate the AM index for the n -th torsion there is defined

$$\begin{aligned} & \text{if } (N_{wrong} > N_{agree}) \\ & \quad \text{if (VPG deemed as rigid the } n\text{-th torsion)} \\ & \quad \quad measure = 50 - 49 \times (N_{wrong} - N_{agree}) / N_{total} \\ & \quad \text{else} \\ & \quad \quad measure = 50 + 49 \times (N_{wrong} - N_{agree}) / N_{total} \\ & \quad \text{else} \\ & \quad \quad measure = 50 \end{aligned}$$

$$AM = (measure - 50) / 49$$

where N_{wrong} is the count of times that PG disagreed with VPG, N_{agree} is the count of times that PG matched the VPG, and $N_{total} = N_{agree} + N_{wrong}$ is the total number of realizations

for the PG. For instance, if a particular angle has a value of $AM = -1$, it indicates that the VPG assesses the angle to be rigid, whereas the PG indicates the opposite (flexible) in all the realizations.

3.2.4 Rigidity Profiles

To complement the quantities above, we also compare two different graphical descriptions of network rigidity. The *Rigid Cluster Map* (RCM) identifies co-rigid α -carbons pairs within the protein structure. By definition the main diagonal is rigid (an α -carbon is rigid with respect to itself). When constructing the RCM, if a pair of α -carbons belong to the same rigid cluster a value of 1 is assigned to the intersection of both elements, else 0 is given, meaning it is a binary plot simply highlighting co-rigid residue pairs. The $\overline{\text{PG}}$ RCM plots are based on a majority rule across the ensemble. That is, if 50% or more of the realizations is rigid a 1 is assigned, otherwise 0 is assigned. To better quantify the number of DOF located in the flexible α -carbons, we also employ the *Mechanical Coupling Map* (MCM). The MCM offers the maximum number of available DOF that a pair of α -carbons have. After fixing the trivial six DOF in one α -carbon a search for pebbles is launched on the other α -carbon to gather the maximum number of DOF. For normalization purposes, the DOF found are divided by six (being the maximum number of DOF that an α -carbon can have). This information is presented using a color code scheme in the MCM that ranges from zero to one, thus providing a more nuanced view of network rigidity. Since they are not binary like the RCM, the reported $\overline{\text{PG}}$ MCM values are simply the arithmetic average across the ensemble.

3.3 Results and Discussion

3.3.1 Quantifying Rigid Cluster Similarity

Characterizing the rigid clusters offers a unique view in terms of the role that chemical interactions play within proteins. In prior work, we have used rigid cluster composition to provide a statistically robust description of thermodynamic coupling within double mutant cycles [40]. Moreover, there have been many investigations characterizing the loss of rigidity that occurs upon protein unfolding using a H-bond dilution model [20, 38, 41–45]. Finally, PG characterizations of rigidity have been used to explain the increased stability

of thermophilic proteins [46], RNA function [47], the effects of ligand binding [12] and the identification of critical interactions [48]. In these works, an energy cutoff is used to identify which H-bonds are present. As the cutoff is relaxed, more H-bonds are included in the structure, thus increasing rigidity. In the same way, we vary P_{nat} to control the number of H-bonds, which we treat uniformly. That is, the probability $3 \times P_{nat}$ defines the number of DOF removed from the VPG when the H-bond is independent (recall each H-bond is described by three distance constraints). Conversely, a random number between zero and one is assigned to each possible H-bond in each PG realization to determine if it is present or not (H-bond is present if $RAND(0,1) < P_{nat}$). This process is repeated 1000 times to generate the $\overline{\text{PG}}$ ensemble, which is much more than necessary to achieve good statistics.

As discussed above, the Rand Measure [39] (RM) compares the rigid cluster compositions from the VPG and $\overline{\text{PG}}$. Fig. 3.3a presents the RM calculation across the range P_{nat} values for four exemplar protein structures. The four example proteins span a range of sizes (from 64 to 315 residues) and topological architectures. Specifically, they are the chemotaxis receptor methyltransferase CheR structure (pdbid = 1AF7) [49], the FLAP endonuclease from *M. jannaschii* (pdbid = 1A76) [36], a small scorpion protein toxin (pdbid = 1AHO) [34] and the disulfide oxidoreductase from *P. furiosus* (pdbid = 1A8L) [50]. In each, there is a region where the RM decreases sharply, which corresponds to the worst-case situation when the fluctuations are maximized, for some networks this point represents the transition from flexible to rigid. A similar pattern was detected across our entire dataset, which appears at values as low as $P_{nat} = 0.60$ and ends at values as high as $P_{nat} = 0.95$. To calculate RM at each P_{nat} , each one of the 1000 PG realizations is compared to the single VPG description. The $\overline{\text{PG}}$ RM value is simply the average of the RM 1000 realizations. The P_{nat} at the minimum in RM, designated P_{RM} , identifies the worst-case scenario.

The high RM values indicate that the rigid cluster composition is very similar across the $\overline{\text{PG}}$ and VPG. This point is shown structurally in Table 3.1, which presents the rigid clusters found by both the $\overline{\text{PG}}$ and VPG algorithms using the same four example proteins. Each rigid cluster is highlighted in color, meaning that a color change indicates a rigid cluster boundary. Grey indicates a flexible region. While the similarity is apparent by just qualitatively comparing the rigid cluster decompositions from each algorithm, the difference

plots provided in the third column are most compelling. Here red coloring is used to identify regions that disagree in rigid cluster composition, whereas grey coloring indicates agreement.

Expanding to our entire dataset, Fig. 3.3b plots a histogram of all RM scores at the respective P_{RM} value for each protein. The worst-case RM scores are encouragingly large ($> 80\%$) for an overwhelming majority of the proteins, thus indicating that the rigid clusters identified by the two algorithms are quite similar. Slight shifts to the considered P_{nat} to just above and below P_{RM} negligibly affects the histograms.

3.3.2 Over versus Under Prediction of Rigidity

The RM used above is based on rigid cluster composition, but it is also important for us to determine if the heuristic VPG tends to over and/or underestimate the amount of rigidity within the structure. To determine how often each happens, we quantify similarity within the rigidity of all backbone (ϕ and ψ) dihedral angles. The agreement measure (AM) ranges from -1 to $+1$. A value of zero is returned when the \overline{PG} and VPG agree, meaning they both identify a particular dihedral as rigid or flexible. AM values of -1 indicate that the VPG identifies a dihedral as rigid, whereas the \overline{PG} says that it is flexible. Conversely, $AM = +1$ when the VPG identifies a dihedral as flexible, but the \overline{PG} says that it is rigid. Fig. 3.4a, b and c present histograms of AM values for three of the proteins from above. In panel (a), it is shown that the VPG slightly overestimates the amount of rigidity within the methyltransferase CheR structure, whereas panel (b) indicates that it slightly underestimates the amount of rigidity within FLAP endonuclease. Panel (c) shows that VPG overestimates the amount of rigidity within the disulfide oxidoreductase. Fig. 3.3c presents a histogram of the entire dataset. Clearly, the overwhelming majority ($> 92\%$) of dihedral angles are in close agreement. Disagreement ($AM \neq 0$) between both algorithms is minimal, especially considering the comparison is at the P_{RM} of each protein that defines the worst case. Nonetheless, it is interesting to identify when discrepancies are most likely to occur. A survey of the differences reveals that they generally occur in loop regions and edges of secondary structures, as typified in Fig. 3.4d.

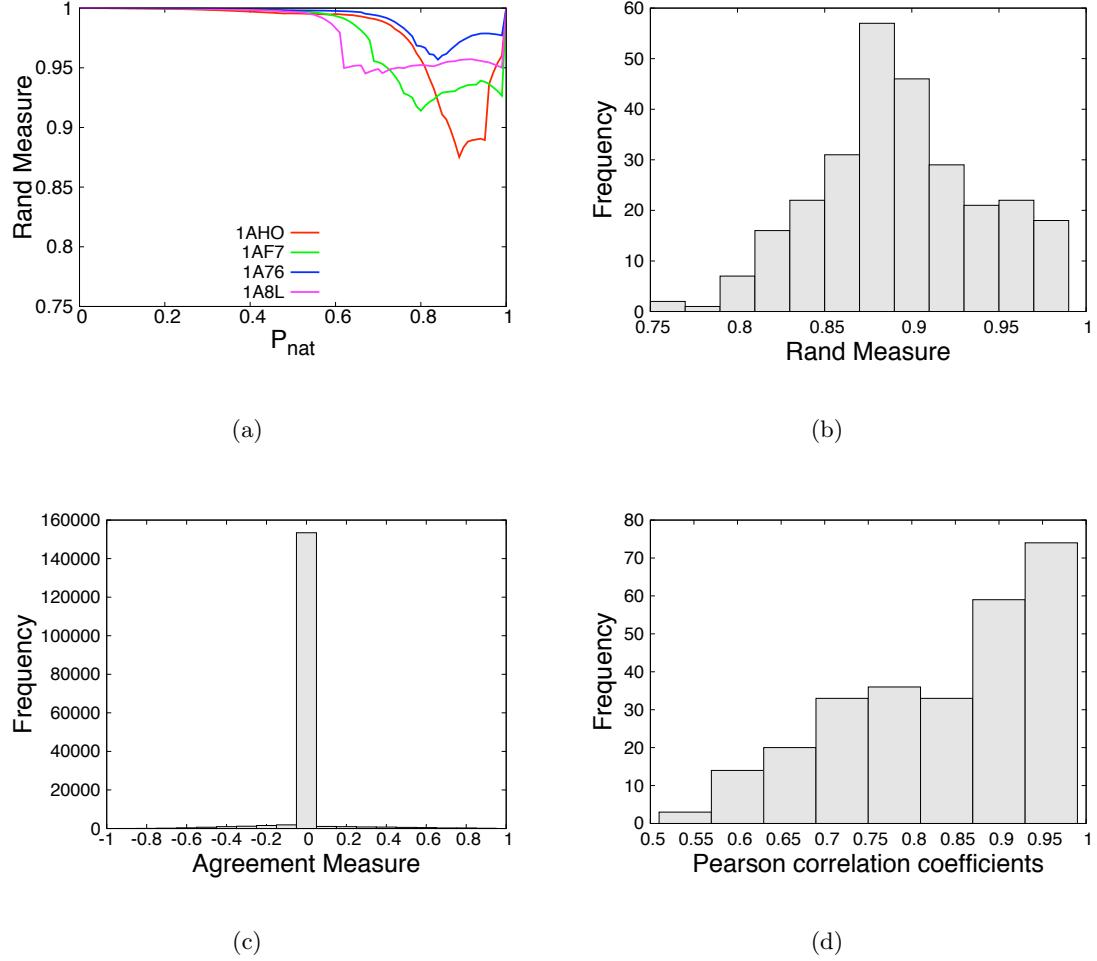

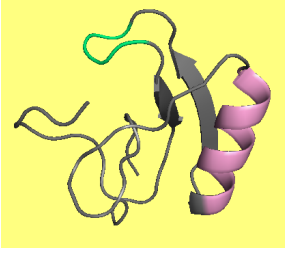

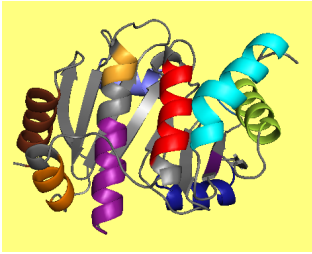
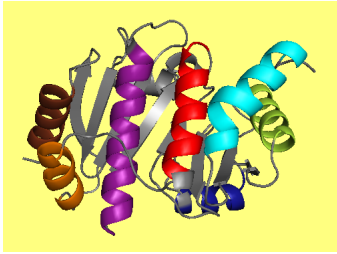
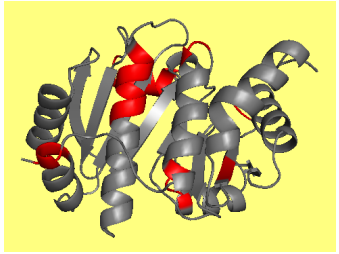
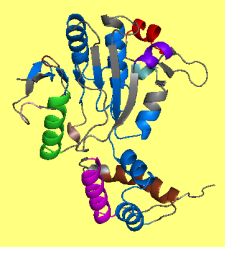
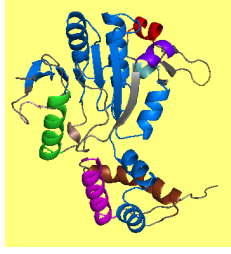
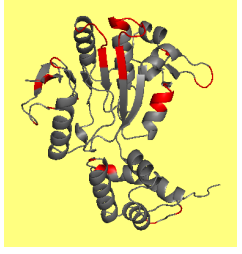





Figure 3.3: Quantifying \overline{PG} and VPG similarity. (a) The Rand Measure (RM) is plotted versus P_{nat} for four exemplar proteins that span a range of sizes (from 64 to 315 residues) and topological architectures. All proteins across the full dataset have the same characteristic shape where the minima in RM is related to the protein structure’s rigidity transition. The P_{nat} value corresponding to the worst RM is defined as P_{RM} . (b) Histogram detailing P_{RM} values for each protein within our dataset. Encouragingly, an overwhelming majority of cases have RMs greater than 80%. (c) Histogram detailing the agreement measure for each backbone torsion within our dataset at each protein’s respective P_{RM} value. (d) Histogram detailing the Pearson correlation coefficient comparing the \overline{PG} and VPG mechanical coupling maps across the dataset at each protein’s respective P_{RM} value.

3.3.3 Rigidity Profile Similarity

We use Rigid Cluster Maps (RCM) to visually highlight pairwise mechanical couplings within structure. That is, analogous to protein contact maps, RCMs highlight α -carbon pairs within the same rigid cluster by a red mark, otherwise no mark is provided. For ease of comparison, the \overline{PG} results are presented in the upper triangle, whereas the VPG results

Table 3.1: Rigid Cluster visualizations for four example proteins using the $\overline{\text{PG}}$ (first column) and VPG (center column). Differences between the two PG variants are highlighted in the third column.

PG	VPG	Disagreement ($ \text{PG}-\text{VPG} $)
Scorpion protein toxin, $P_{\text{nat}} = 0.62$		
		
Disulfide oxidoreductase, $P_{\text{nat}} = 0.57$		
		
Methyltransferase CheR, $P_{\text{nat}} = 0.72$		
		
Flap endonuclease, $P_{\text{nat}} = 0.75$		
		

are presented in the lower triangle. The $\overline{\text{PG}}$ values correspond to *majority* behavior across the 1000 realizations. By construction, the protein backbone corresponding to the RCM diagonal is always rigid in both variants. Using the methyltransferase structure from above

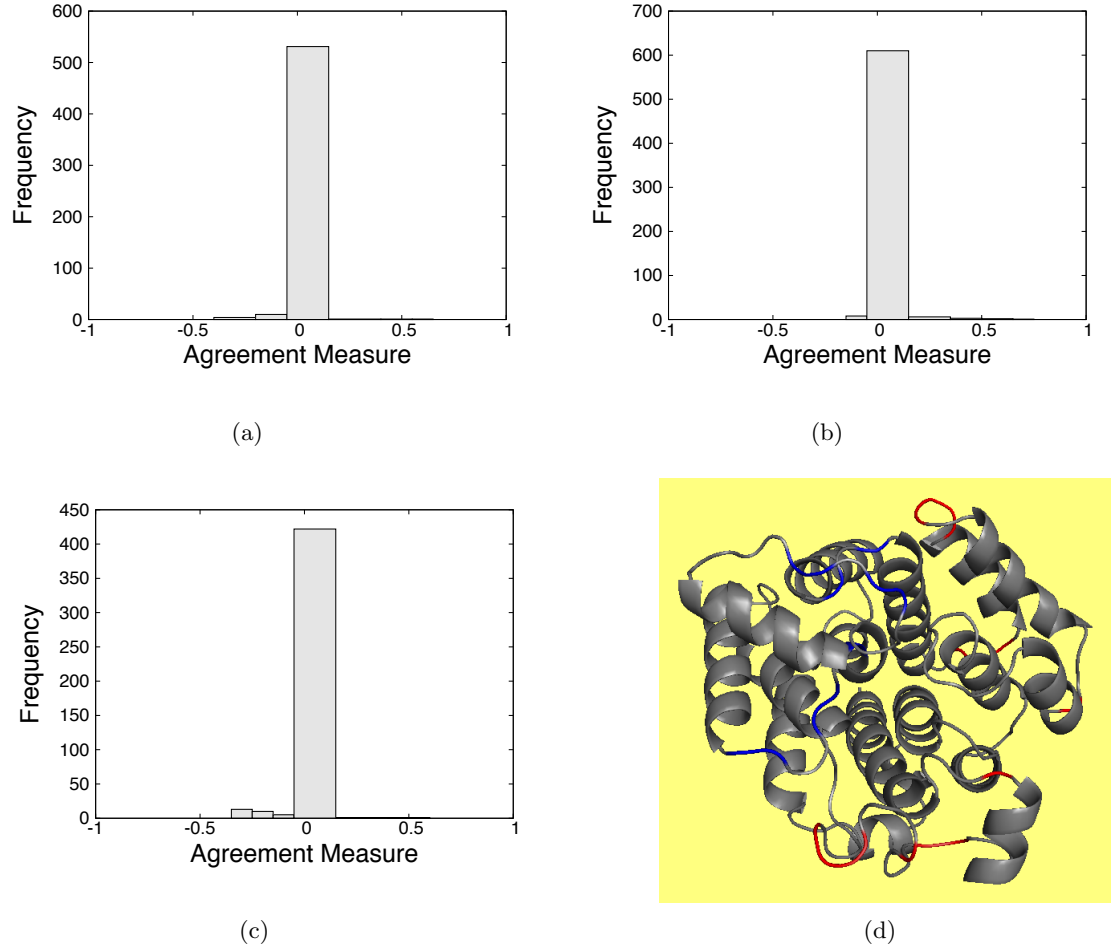


Figure 3.4: Agreement measure (AM) results. AM histograms for the (a) methyltransferase, (b) FLAP endonuclease and (c) disulfide oxidoreductase at their respective P_{RM} = values. (d) Differences between the $\overline{\text{PG}}$ and VPG are mapped to the ribosylglycohydrolase structure from *M. jannaschii*, which is presented as a typical case. Red coloring indicates that the VPG overestimates rigidity relative to $\overline{\text{PG}}$, whereas blue indicates an underestimate. Across our dataset, as shown in this example, differences occur most frequently in loop regions.

as a typical cases, the two panels in Fig. 3.5 correspond to two different values of P_{nat} , ranging from a completely flexible (unfolded) structure with few crosslinking H-bonds to a predominantly rigid structure with many crosslinking H-bonds. As one can see, the VPG and $\overline{\text{PG}}$ algorithms give very similar results.

Going a step further, Fig. 3.6 presents the RCMs of our four example proteins near P_{RM} , thus P_{nat} is corresponding to the critical region. The presented values are slightly shifted from exact P_{RM} values to highlight interesting features. Note that the changes

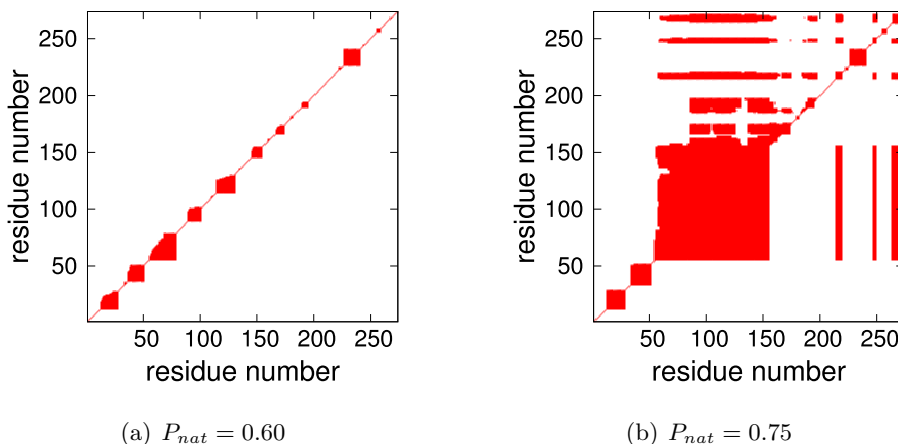


Figure 3.5: Rigid cluster maps (RCM) of chemotaxis receptor methyltransferase CheR structure is plotted at two different P_{nat} values. Red coloring identifies residue pairs that are co-rigid. $\overline{\text{PG}}$ results are presented in the upper triangle, whereas the VPG is presented in the lower. At $P_{nat} = 0.60$, the protein is mostly flexible due to a lack of crosslinking H-bonds. However, the structure becomes increasingly rigid as H-bonds are added to the network. At $P_{nat} = 0.75$, the VPG slightly under-predicts the extent of rigidity. For this protein $P_{RM} = 0.80$.

in P_{nat} actually make the RCM plots appear more dissimilar. The large square region along the backbone corresponds to a rigid α -helix. A similar pattern is observed in the disulfide oxidoreductase, which also has few crosslinking interactions at this value of P_{nat} . Conversely, the off-diagonal features are mostly conserved in the methyltransferase CheR structure, but the VPG slightly overestimates the extent of rigidity within the core region (residues $\sim 75 - 150$). The FLAP endonuclease example provides the most interesting visual differences between the two PG variants, where the VPG underestimates the $\overline{\text{PG}}$ predictions. That is, the VPG fails to identify rigid clusters present within the $\overline{\text{PG}}$. However, the differences are found to be much less severe on closer inspection regarding the number of available DOF. While there are no free pebbles within the $\overline{\text{PG}}$ in these regions, the probabilistic VPG identifies a tiny nonzero fraction (3×10^{-3}). Clearly, this difference is negligible, but the binary RCM makes the difference appear much larger than it actually is.

The Mechanical Coupling Maps (MCM) provide a more nuanced view of rigid cluster decomposition. Unlike the binary RCMs, MCMs are based on a continuous scale that identifies the fractional number of pebbles shared between each α -carbon pair. In this way, the MCMs are quite similar to the cooperativity correlation plots calculated by our

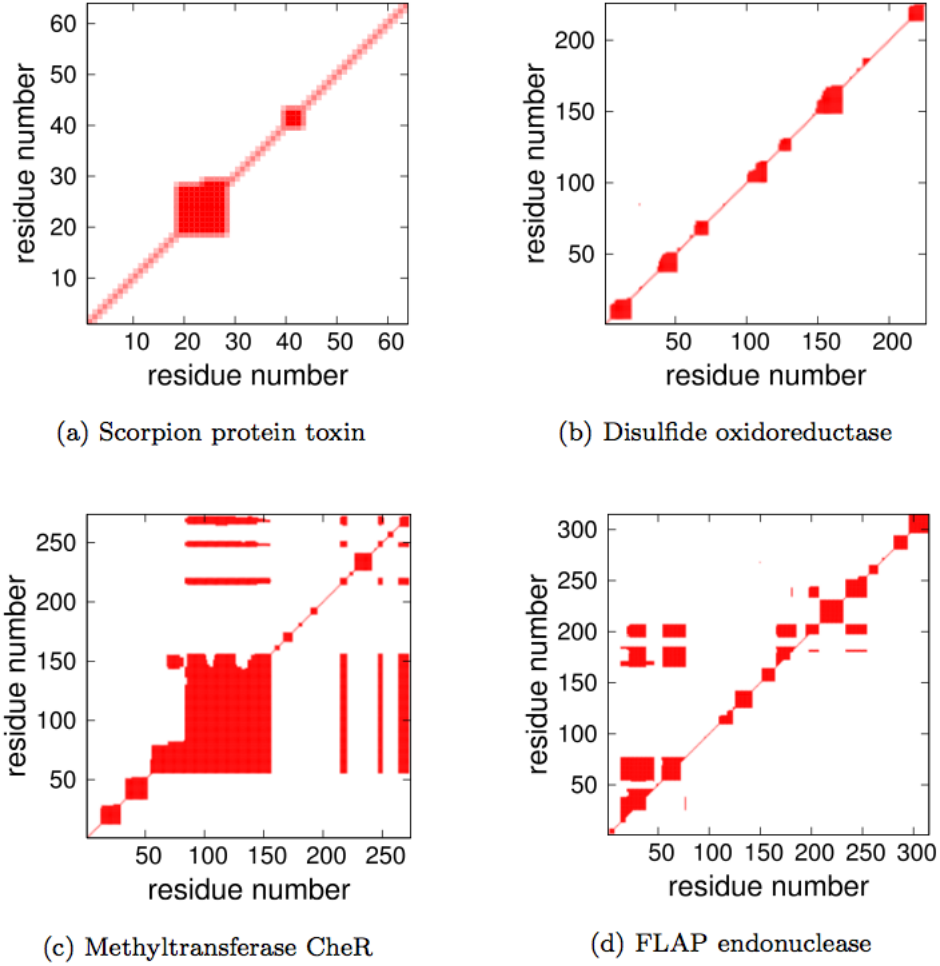


Figure 3.6: Rigid Cluster Maps (RCM) for four different example proteins near their respective P_{RM} values. \overline{PG} results are presented in the upper triangle, whereas the VPG is presented in the lower. Note that the presented proteins are the same from Fig. 3.3a.

statistical mechanical DCM [23, 24, 51–54]. All the values are normalized for practical reasons by the number of trivial DOF. That is, six free pebbles corresponds to a MCM value of one. Fig. 3.7 compares the MCMs for the same four proteins in Fig. 3.6, using the same P_{nat} values. The rigidity over-prediction by the VPG in the methyltransferase example is again clear. However, there is appreciable co-rigidity within the residue pairs contained within the range of residues $\sim 60 - 80$ and residues $\sim 80 - 150$, which was identified as flexible by the the RCM. Additionally, the MCMs reveal a more interesting set of similarities throughout the plots. In the same way, the similarity within FLAP endonuclease is also more pronounced, although the VPG again somewhat overestimates the extent of rigidity.

Conversely, the MCMs actually show more dissimilarity within the two examples without any off-diagonal RCM components. In both, there is marginal co-rigidity identified by the $\overline{\text{PG}}$ (the reddish shadowing) due to some rigidity fluctuations throughout the ensemble that is suppressed by the VPG.

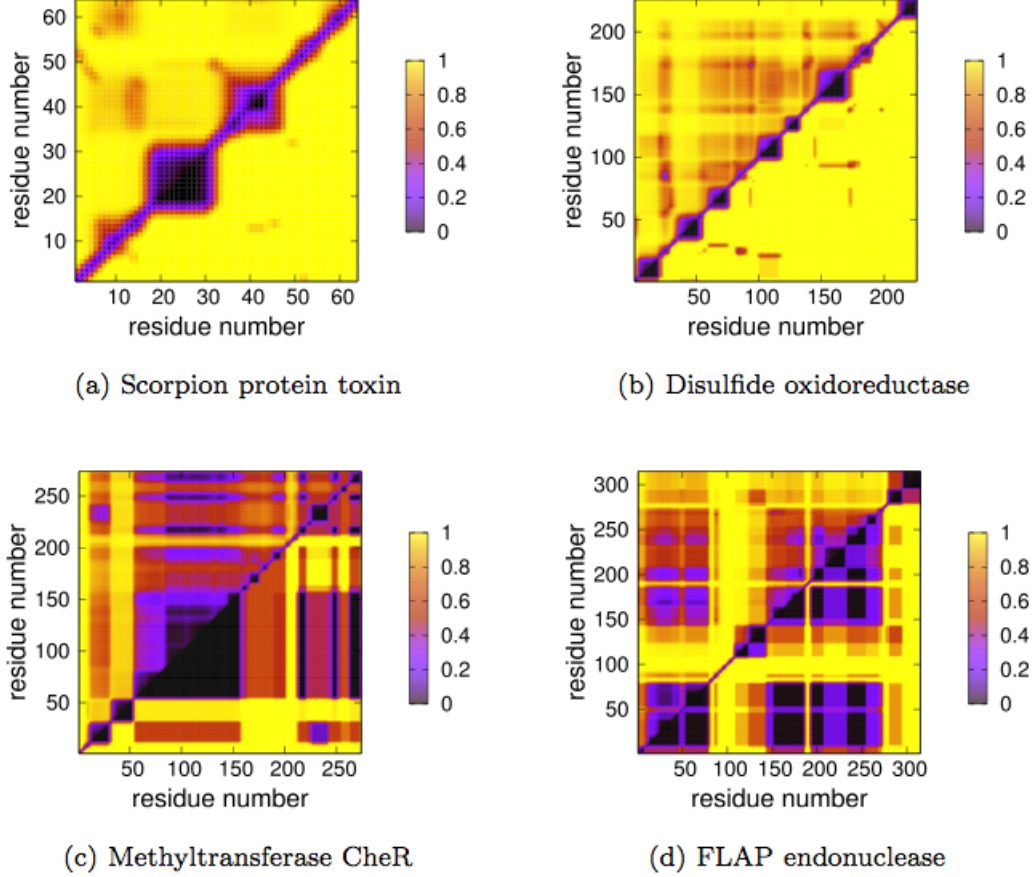


Figure 3.7: Mechanical Coupling Maps (MCM) provide a more nuanced description of co-rigidity. Specifically, the continuous scale provides a normalized description of how many free pebbles (DOF) are shared between each residue pair ($0 = 0$ pebbles, whereas $1 = 6$ pebbles). Again, each MCM is plotted near their respective P_{RM} values for the same four proteins presented Figs. 3.3a and 3.6.

Expanding across the entire dataset, Fig. 3.3d provides a histogram of the Pearson correlation coefficient between the MCM matrices calculated by the $\overline{\text{PG}}$ and the VPG. Clearly, the VPG is consistently a good estimator of the $\overline{\text{PG}}$ behavior. This point is strengthened by Fig. 3.8, which compares the MCM of each unique PG realization to the $\overline{\text{PG}}$ plot for

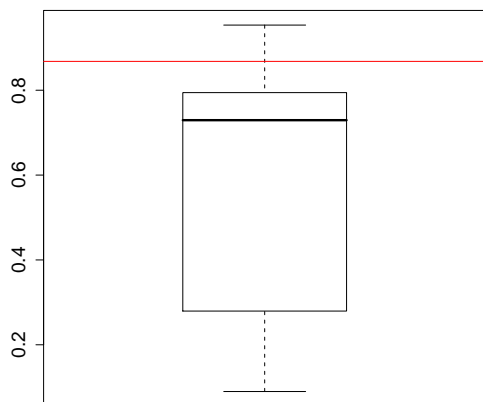
the same four proteins considered above. That is, the boxplots describe the intrinsic variability across the PG ensemble. The red line corresponds to the similarity between the $\overline{\text{PG}}$ and VPG MCMs, which is encouragingly strong. In fact, the VPG similarity to the $\overline{\text{PG}}$ behavior is better than the third quartile, which is indicated by the top of the box. This result clearly indicates that VPG approximates $\overline{\text{PG}}$ behavior better than the vast majority of the single PG realizations. These comparisons are calculated at the same value of P_{RM} as above, meaning they again correspond to the critical region.

3.3.4 The Rigidity Transition

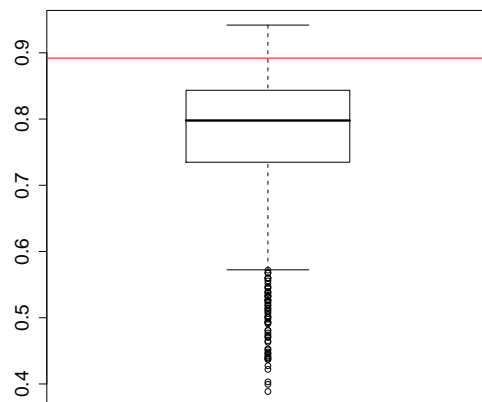
Following earlier works [23, 24, 38, 51, 54], we define P_t (for transition) as the peak in the rigid cluster susceptibility (RCS) curve, which is defined as the reduced second moment in rigid cluster size. That is, the peak in RCS identifies the point in which the rigid cluster sizes are maximally fluctuating, indicating a transition from a globally flexible to globally rigid network. Twelve examples (including the four proteins discussed above) of $\overline{\text{PG}}$ and VPG RCS curves are shown in 3.9, all of which are qualitatively similar. As shown by the scatter plot in Fig. 3.10a, the rigidity transitions identified by the $\overline{\text{PG}}$ and VPG algorithms are highly correlated. In addition, the Average Cluster Size (ACS) at P_t is also highly correlated across the two algorithms (Fig. 3.10b). Since P_t identifies the P_{nat} value with maximal variability within the rigid cluster sizes, it is expected to be related to the P_{RM} because the single VPG mean-field calculations suppresses fluctuations. This is indeed the case as indicated by the strong correlation between P_t and P_{RM} for both the $\overline{\text{PG}}$ and VPG algorithms (Fig. 3.10c-d).

3.4 Conclusions

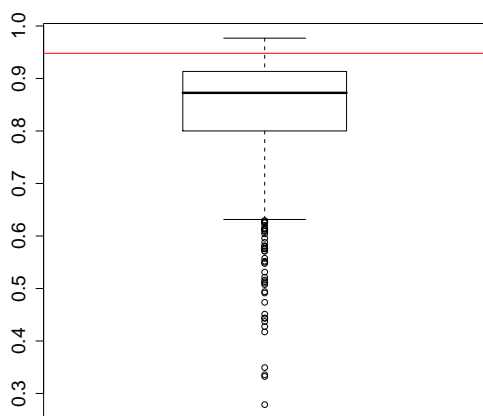
In this chapter we compared the PG versus the VPG algorithms on an extensive set of protein networks. We demonstrated that ensemble averaged $\overline{\text{PG}}$ properties, which requires sampling, is approximated well by a single mean field calculation. That is, the probabilistic VPG accurately reproduces a number of ensemble-averaged network rigidity properties. The high values of the RM clearly indicate that the rigid cluster compositions are very similar, especially at $P_{nat} \neq P_{RM}$. The AM and structural comparisons of the rigid clusters respectively provide further quantitative and qualitative support for this point. Compar-



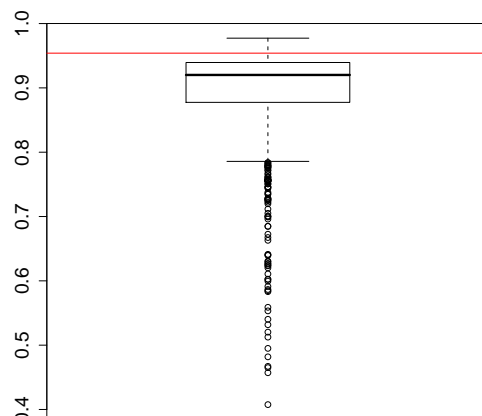
(a) Scorpion protein toxin



(b) Disulfide oxidoreductase



(c) Methyltransferase CheR



(d) FLAP endonuclease

Figure 3.8: Boxplots describing the ensemble of Pearson correlations coefficients comparing each PG realization to the $\overline{\text{PG}}$ behavior. The red line represents the correlation between the $\overline{\text{PG}}$ and the VPG. In all cases, the $\overline{\text{PG}}$ to VPG similarity is greater than the 75th percentile of the intrinsic fluctuations within the PG ensemble.

isons of the RCM and MCM rigidity profiles between the $\overline{\text{PG}}$ and VPG variants also indicate that the calculated rigidity properties are highly similar. In fact, the $\overline{\text{PG}}$ to VPG MCMs are much more similar than the intrinsic variability across the ensemble of PG snapshots. Finally, the mechanical transitions identified by the peak in the rigid cluster susceptibility curves are highly correlated across the two variants. Taken together, these results collectively demonstrate the utility and power of this new virtual pebble game.

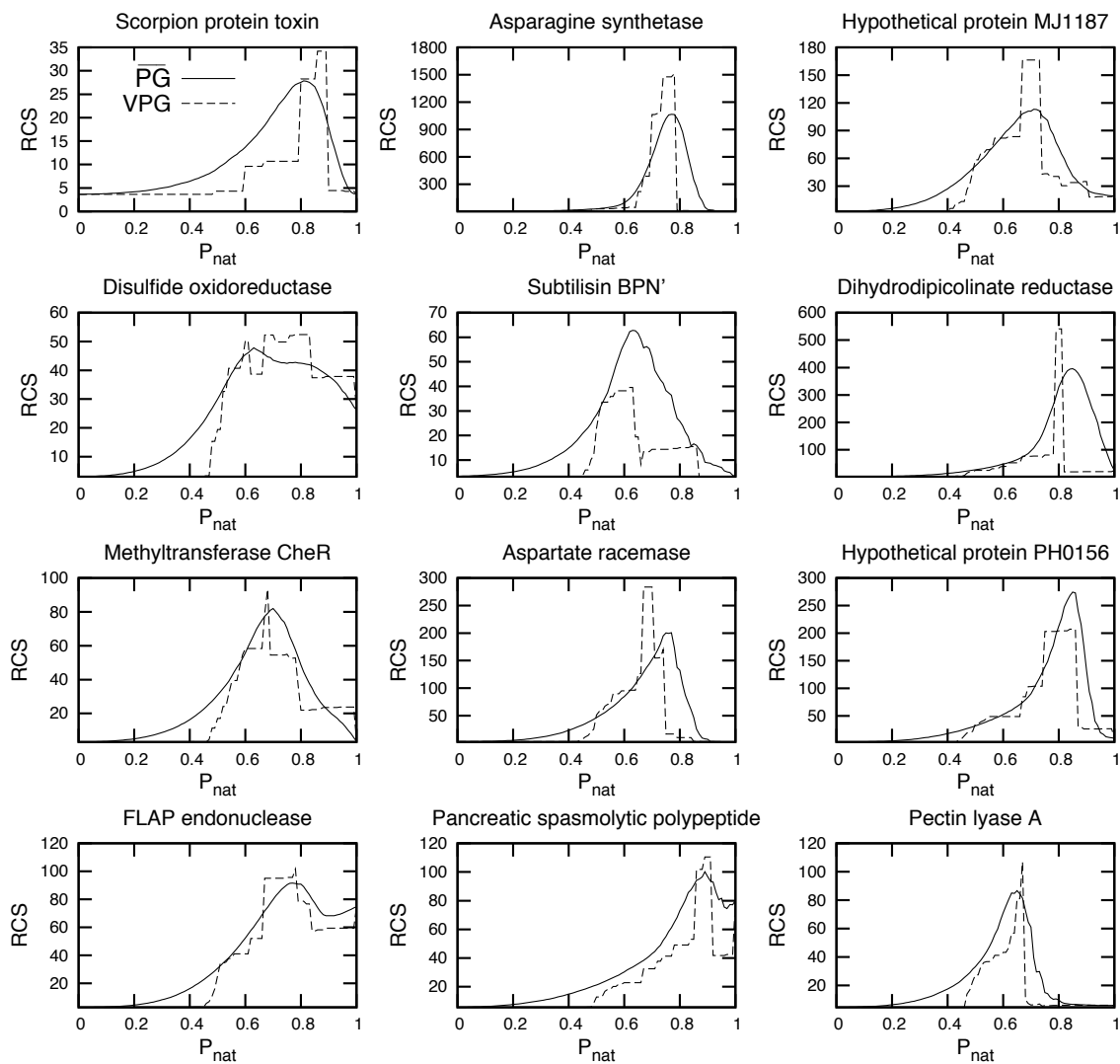
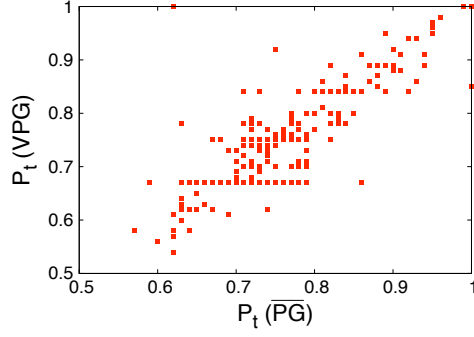
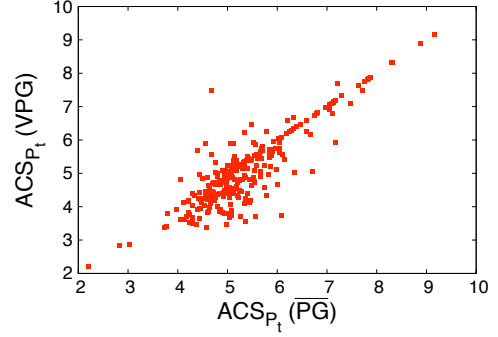


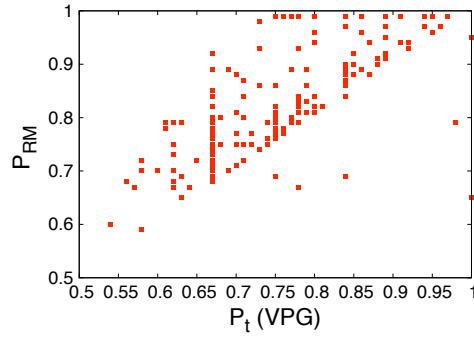
Figure 3.9: Rigid Cluster Susceptibility (RCS) is plotted versus P_{nat} for 12 typical protein examples (\overline{PG} = solid line and \overline{VPG} = dashed line). Note that the proteins presented in the first column are the same from Fig. 3.3a.



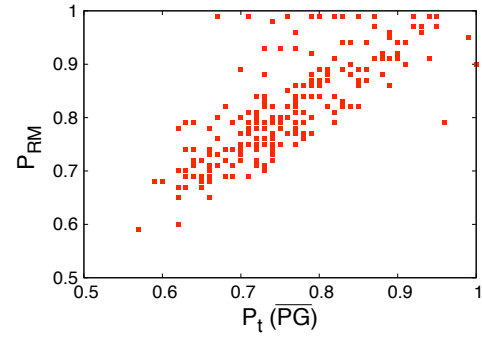
(a) Pearson correlation = 0.92



(b) Pearson correlation = 0.87



(c) Pearson correlation = 0.72



(d) Pearson correlation = 0.77

Figure 3.10: Rigidity transition effects. (a) The rigidity transition (P_t) is compared across the $\overline{\text{PG}}$ and VPG algorithms. (b) Similarly, the average cluster size (ACS) at their respective P_t values are compared across the two algorithms. The value of P_{nat} with the worst RM (called P_{RM}) is compared to P_t calculated using the (c) VPG and (d) $\overline{\text{PG}}$. The linear relationships occur because the mean field approximation is maximally inaccurate in this range. Note, a few proteins do not have completed peaks in their rigid cluster susceptibility curves because the protein never crosses the rigidity transition, which have been excluded from panels (c) and (d).

[illegible]

CHAPTER 4: IMPROVING PROTEIN FLEXIBILITY PREDICTIONS BY COMBINING STATISTICAL SAMPLING WITH A MEAN-FIELD VIRTUAL PEBBLE GAME

4.1 Introduction

In this chapter we present a new hybrid algorithm called VPG- x that integrates the $\overline{\text{PG}}$ and VPG treatments. Therein, some percentage of constraints (controlled by the variable x) are statistically sampled. This approach bridges the divide between the $\overline{\text{PG}}$ and VPG algorithms, allowing one to continuously vary from one to the other as a function of x . As summarized below, the hybrid VPG- x algorithm consistently improves upon the completely mean-field VPG with small additional computational cost. Specifically, the VPG- x algorithm is able to capture some of the marginal rigidity within frustrated regions that is missed by the VPG. However, the VPG- x algorithm is unable to quantitatively reproduce the $\overline{\text{PG}}$ descriptions precisely, and fluctuations are missed for any x that is not essentially 1. In cases where high accuracy is desired regardless of the computational cost involved, the original $\overline{\text{PG}}$ remains optimal. As the system moves away from the rigidity transition, VPG- x provides an alternative method. Far away from the rigidity transition, the VPG is sufficient (i.e. $x = 0$). These general features reflect the general property that $\overline{\text{PG}}$ and VPG can be morphed into one another continuously, and the results agree with intuition that the errors and speed change continuously as a linear function of x as well.

4.2 Materials and methods

4.2.1 Protein network descriptions

The details for the creation of protein networks are the same as the ones explained in chapter three. The protein dataset that we use for testing purposes has ten proteins. The PDB codes for these proteins are: 1AHO [34], 1AF7 [49], 1A76 [36], 1A8L [50], 12AS [55], 1AEP [56], 1A1X [35], 2SIC [57], 3COQ [37] and 2PSP [58]. A schematic explanation of the three algorithms is presented in Fig. 4.1.

Fig. 4.1 illustrates different topologies for PG, whereas VPG has only one topology. In this example, there are 2 possible fluctuating constraints representing a H-bond (red dashed lines), leading to $2^2 = 4$ realizations. The single VPG network is also shown, where the pebble capacity of each constraint has been attenuated by P_{nat} . As discussed further below, the VPG- x algorithm treats some portion $(1 - x)$ of the H-bond network probabilistically, whereas the remainder (x) is sampled with probability P_{nat} . The network for VPG- x with $x = 0.5$ is shown in Fig. 4.1.

4.2.2 The algorithms

In chapter two we saw that despite differences (binary versus probabilistic), the PG and VPG algorithms have identical pebble search rules, and not surprising share similar execution times (linear in practice, although quadratic in worst case [18]). However, the total amount of time needed by $\overline{\text{PG}}$ must be multiplied by the number of realizations sampled. On the other hand, the single VPG calculation suppresses fluctuations within the network, making it susceptible to error in regions that have high variance within the $\overline{\text{PG}}$. The VPG- x algorithm addresses both issues by balancing the amount of sampling and mean field character to obtain more accurate results than VPG, while being faster than $\overline{\text{PG}}$. VPG- x randomly selects x -fraction of the H-bond edges to sample over, and a mean field treatment is made for all remaining H-bond edges. Due to sampling, both execution times and accuracy are increased. When $x = 0$, the calculation is entirely VPG, whereas when $x = 1$, the calculation is entirely $\overline{\text{PG}}$.

A modification in line 3 of the VPG (algorithm 6) is needed to run the VPG- x algorithm. The detailed description of this modification is given by algorithm 7. Note that just the H-bonds are fully evaluated by algorithm 7 (covalent bonds are treated the same in all variants). A double condition assures the right balance between both regular sampling and mean field treatment. Let x be the variable that controls the amount of fluctuating interactions, e_k the current edge that is being evaluated and P_{nat} the probability of existence of a H-bond. The condition at step 3 (algorithm 7) evaluates which treatment is to be used for each H-bond. If this condition is true, regular $\overline{\text{PG}}$ sampling is applied. Otherwise the *else* clause is evaluated and the constraint is treated probabilistically. When a H-bond is

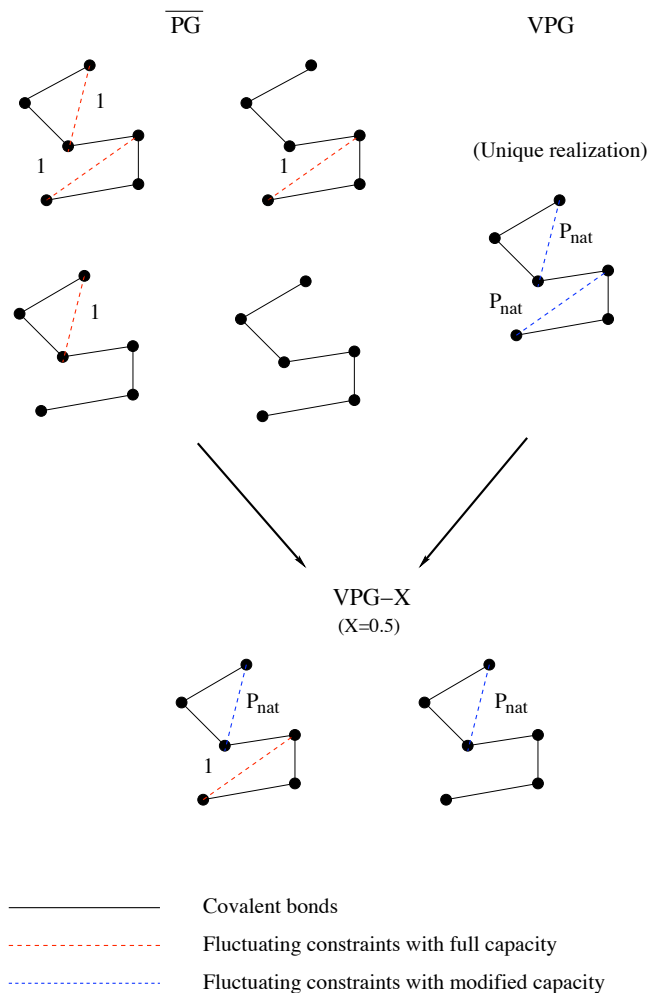


Figure 4.1: Comparison of network descriptions for all three PG variants. The $\overline{\text{PG}}$ is based on an ensemble networks where each constraint is either fully present or not. In this toy example, all members of the ensemble are described, but this is impossible in protein structures. As such, we statistically sample networks where presence vs. absence of each constraint is based on comparing a random number to P_{nat} . The VPG directly calculates average properties because DOF have been fractionalized and each constraint is assigned a pebble capacity defined by P_{nat} . The VPG- x algorithm integrates both approaches. In this example, the top constraint is treated probabilistically, whereas the bottom is sampled. The variable x controls how much of the network is directly sampled.

statistically sampled, another random number is compared against P_{nat} to evaluate the presence of that given H-bond, whose capacity is 3 when present and 0 when absent.

Algorithm 6 VPG

Consider a network consisting of vertices $\{V_n\}, n = 1, 2, \dots, N$, with a list of edges $\{e_m\}, m = 1, 2, \dots, M$.

```

1: Initialize:  $\{V_n\}$  with 6 DOF for each vertex
2: for  $k = 1$  to  $M$  do
3:   insert edge  $e_k$  with capacity  $c_k$ , let  $v_i$  and  $v_j$  be its two incident vertices
4:   collect 6 pebbles on vertex  $v_i$ 
5:   flag vertex  $v_i$ 
6:   launch a search for  $c_k$  pebbles on vertex  $v_j$ 
7:   if  $c_k$  pebbles were collected then
8:     cover edge  $e_k$ 
9:   else
10:    condense all visited vertices in the failed search into a single vertex
11:   end if
12: end for

```

Algorithm 7 VPG- x add-on

```

1: if  $e_k$  is a H-bond then
2:    $r \leftarrow \text{random}(0,1)$ 
3:   if  $r < x$  then
4:      $s \leftarrow \text{random}(0,1)$ 
5:     if  $s \geq P_{nat}$  then
6:        $c_k \leftarrow 0$ 
7:     end if
8:   else
9:      $c_k \leftarrow c_k \times P_{nat}$ 
10:  end if
11: end if

```

4.2.3 Similarity measures

To compare the algorithms in the experimental section below we have employed some of the metrics that we already defined in previous chapters. The simplest of these metrics just compares the number of floppy modes at a given P_{nat} value. Network similarity is also compared using Rigid Cluster Maps (RCM) and Mechanical Coupling Maps (MCM).

We have compared the RCM and MCM plots over the entire range of P_{nat} between 0 to 1 for the VPG against $\overline{\text{PG}}$. We focused only in the worst case scenario where the greatest difference is found between $\overline{\text{PG}}$ and VPG. The Rand Measure (RM) is a common metric to compare the similarity between two sets of clusters [39], which we use to compare the rigid cluster decompositions. That is, the vertices in the network are assigned to clusters

based on rigid cluster decomposition that comes from a PG and the VPG. The RM is then averaged over 1000 distinct networks for each P_{nat} to obtain $\overline{RM}(P_{nat})$. Scanning across the full range of P_{nat} , we determine the minimum value of the \overline{RM} for each respective protein structure. An exemplar RM curve is shown in Fig. 4.2. In all cases, the RM curves have this characteristic dip, which approximately corresponds to the rigidity transition, where maximum fluctuations are found within \overline{PG} as expected.

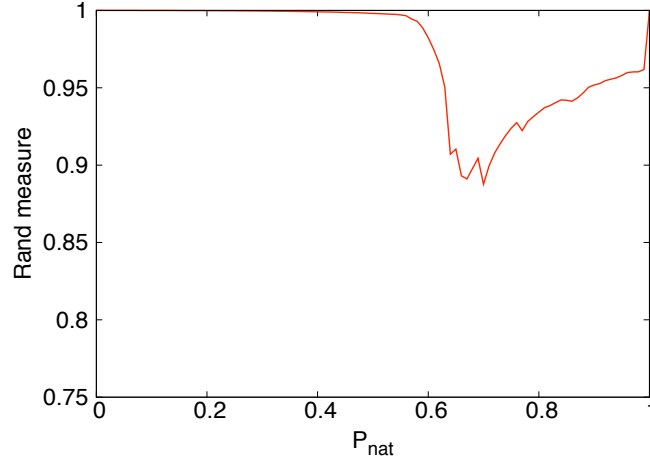


Figure 4.2: The Rand measure (RM) is plotted versus P_{nat} for a typical case. RM values of 1 indicate identical cluster compositions, whereas values of 0 indicate completely different compositions. Due to the extreme nature of the networks at low and high values of P_{nat} (meaning mostly flexible or mostly rigid), the RM is ~ 1 indicating that the \overline{PG} and VPG algorithms provide nearly identical results. Conversely, pronounced differences arise at the rigidity transition, as indicated by the dip in RM. While details vary, all proteins under consideration have the same characteristic shape as shown here.

4.3 Results and discussion

An important quantity of interest that quantifies global network flexibility is the number of floppy modes in the network once the PG has completed. Fig. 4.3 shows the number of floppy modes calculated by the various algorithms, including a series of VPG- x variants corresponding to a range of x values. Corresponding to the dip in RM, the transition from a flexible to rigid protein structure is identified here by the change in slope, which is where most fluctuations across the network occur. As a consequence, the disagreement between the \overline{PG} and VPG algorithms is maximized here. Fig. 4.3a plots the number of floppy

modes within a cubic lattice model with boundary conditions. For this homogenous example, the $\overline{\text{PG}}$ and VPG- x algorithms are averaged over 100 realizations, which is sufficient for good statistics. Due to network uniformity, this example cleanly highlights the effect of different values of x , which systematically morphs from VPG to $\overline{\text{PG}}$ character. Conversely, the transition is compressed in real protein systems. For example, the number of floppy modes in the structure of lipoprotein (pdbid = 1AEP) [56], disulfide oxidoreductase (pdbid = 1A8L) [50] and chemotaxis receptor methyltransferase (pdbid = 1AF7) [49] are also presented (cf. Fig.4.3b-d). The calculation of PG is averaged over 200 realization, whereas VPG- x is averaged over 50. Note that fewer VPG- x realizations are needed in practice to achieve good statistics due to an overall fewer number of fluctuating interactions. Zooming in, Fig. 4.3e-f focuses on the rigidity transition region of interest, which also highlights the systematic morphing from VPG to $\overline{\text{PG}}$ behavior.

RCM's are used to visualize rigid cluster composition (cf. Fig. 4.4). Plotting residue number versus residue number, red coloring identifies a co-rigid pair, whereas white indicates some degree of flexibility present. Because all possible residue pairs within a protein structure are contained above or below the diagonal, each RCM compares two PG variants. In all cases, the most accurate $\overline{\text{PG}}$ results are presented in the upper triangle. To ensure more than sufficient sampling, these results are averaged over 1000 PG realizations. The VPG- x results are presented in the lower triangle in the left column, whereas the VPG results are shown in the lower triangle in the right column. The number of VPG- x samples, R , and x values are tailored to each protein with the goal of only using a minimal number of needed samples, which is controlled by protein size and architecture.

Starting with a small neurotoxin structure (pdbid = 1AHO) [34] that lacks crosslinking H-bonds at this value of P_{nat} , the RCM's of both the VPG- x and VPG algorithms reproduce the $\overline{\text{PG}}$ description (cf. Fig. 4.4a,b). Therein, all PG variants identify two main rigid clusters, the larger of which corresponds to an α -helix. Due to the lack of complexity within this network, we simply chose the intermediate value of $x = 0.5$ and a minimal number of samples ($R = 30$). Conversely, the methyltransferase structure from above (pdbid = 1AF7) is very interesting. Here the $\overline{\text{PG}}$ identifies a main rigid cluster that is somewhat "dented" (cf. Fig. 4.4c), whereas the VPG in panel (d) overestimates the cluster size. Also, the VPG

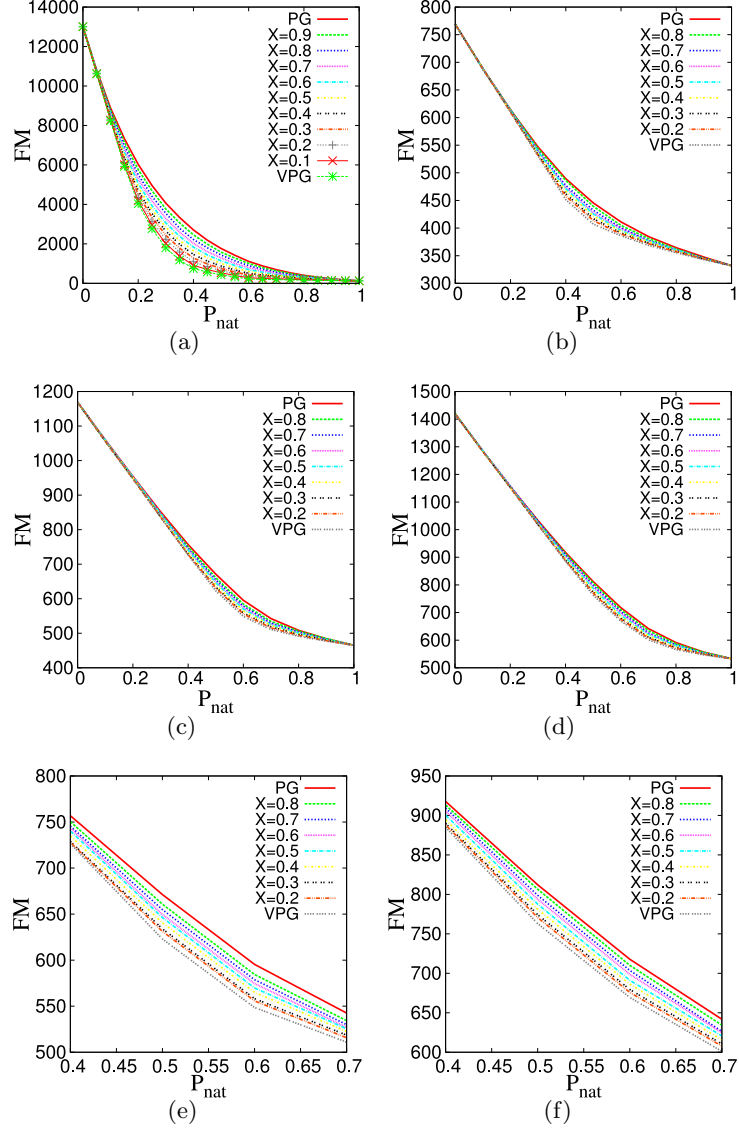


Figure 4.3: The number of floppy modes (FM) is plotted versus P_{nat} . In each panel, the $\overline{\text{PG}}$ and VPG results are shown, which represent two extremes. In panel (a), the results for a cubic lattice network are presented, whereas the others correspond to protein structure examples: (b) lipoprotein, (c) oxidoreductase and (d) methyltransferase. The critical region of panels (c) and (d) are respectively zoomed in on panels (e) and (f). In these latter two panels, it is clear that the VPG- x results systematically transition from VPG-like at small values of x to $\overline{\text{PG}}$ -like at larger values of x .

overestimates the three off-diagonal “parallel” lines that correspond to a four stranded β -sheets that are also part of the main rigid cluster. The VPG- x correctly suppresses rigidity in all of these regions, returning results more consistent with the $\overline{\text{PG}}$ results (c). In this case, x is again equal to 0.5, whereas R has been increased to 200 due to its larger size. While the VPG overestimates the size of the main rigid cluster in the methyltransferase

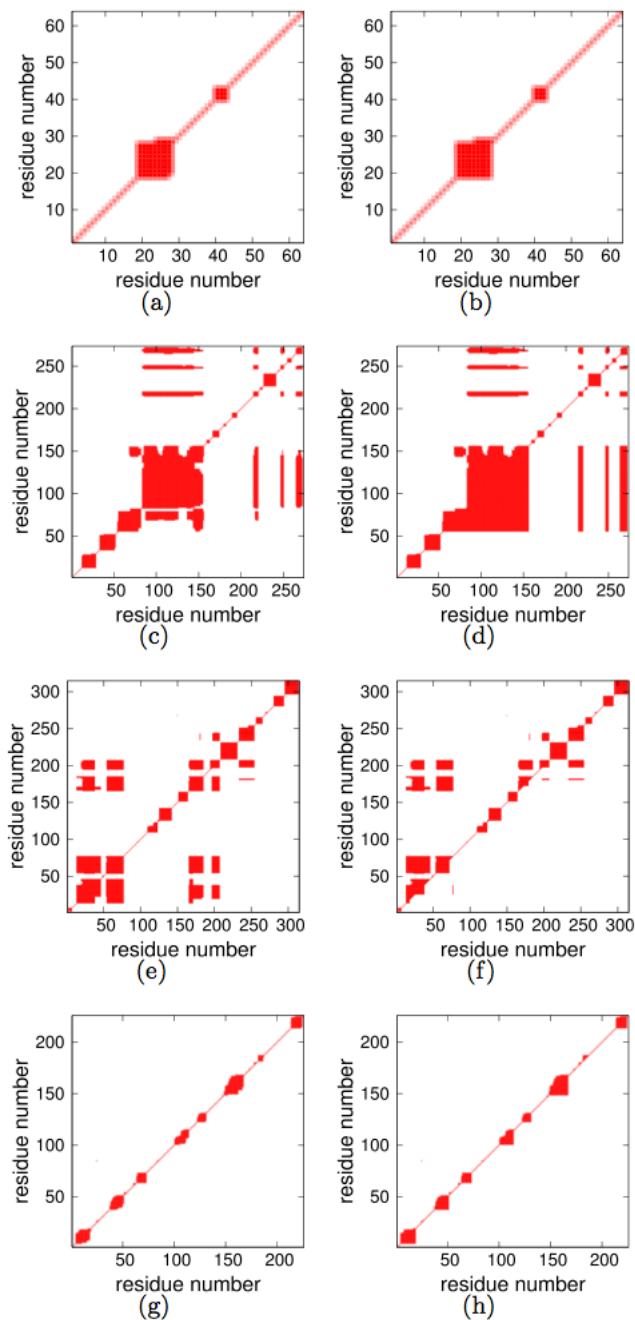


Figure 4.4: Rigid cluster maps (RCM) identify all co-rigid (red) residue pairs in four example protein structures. In each, the \overline{PG} results are plotted in the upper triangle, whereas the lower triangle corresponds to the VPG- x (left column) and VPG (right column) variants. The considered proteins are: (a-b) a neurotoxin structure at $P_{nat} = 0.63$, (c-d) a methyltransferase structure at $P_{nat} = 0.72$, (e-f) a FLAP endonuclease structure at $P_{nat} = 0.78$ and (g-h) an oxidoreductase structure at $P_{nat} = 0.57$. In each case, the reported P_{nat} values correspond to the lowest value of the Rand Measure comparing the \overline{PG} and VPG results.

example, it actually underestimates it in FLAP endonuclease (pdbid = 1A76) [36] (cf. Fig. 4.4f). Nevertheless, the VPG- x algorithm restores the off-diagonal co-rigidity, again closely approximating the $\overline{\text{PG}}$ behavior (cf. Fig. 4.4e). In this example, values of $x = 0.2$ and $R = 30$ are considered. Finally, the disulfide oxidoreductase from above (pdb = 1A8L) is shown in the last row using $x = 0.5$ and $R = 100$. As with the neurotoxin structure, the lack of crosslinking H-bonds makes all three rigidity characterizations very similar.

Based on the RCM descriptions, it is clear that the VPG- x better approximates the $\overline{\text{PG}}$ results. However, this point is further underscored by the MCM comparisons. Using the same format and parameter values as above, Fig. 4.5 compares across the same four example protein structures. In these plots, slight “shadowing” occurs in many off-diagonal regions that are simply colored white in the RCM plots, meaning that the number of DOF shared by the corresponding pairs is relatively small. In fact, in the two simplest RCM examples without any off-diagonal components, there is slight (neurotoxin structure) and extensive (oxidoreductase) shadowing in the $\overline{\text{PG}}$ descriptions. In both cases, the VPG- x algorithm does a much better job of reproducing the $\overline{\text{PG}}$ results compared to the VPG. Similarly, the VPG- x variant again does a better job of describing the $\overline{\text{PG}}$ descriptions in the two more complex examples, meaning it clearly detects subtle properties that the VPG misses.

Mapping the identified rigid clusters to structures further highlights the differences of the PG variants. Again using the large methyltransferase structure and parameters from above as an example, Fig. 4.6 color-codes all rigid clusters identified by the VPG and VPG- x variants, meaning spatially contiguous co-rigid residues are colored the same, whereas color changes indicate flexible DOF between clusters. Grey regions specify flexible regions that do not belong to a rigid cluster. Across the pair, color-coding is conserved to facilitate comparisons. In both cases, the structure is primarily composed of one large rigid cluster (shown in red). However, as highlighted above, there were some disagreements between the two probabilistic treatments, which are highlighted by the black circles corresponding to the “dented” $\overline{\text{PG}}$ squares in Fig. 4.4c-d. Across the $\overline{\text{PG}}$ ensemble, there is a general lack of consensus in the descriptions of the small β -sheets and the capping end of the identified α -helix due to maximal rigidity fluctuations therein. These fluctuations are completely missed

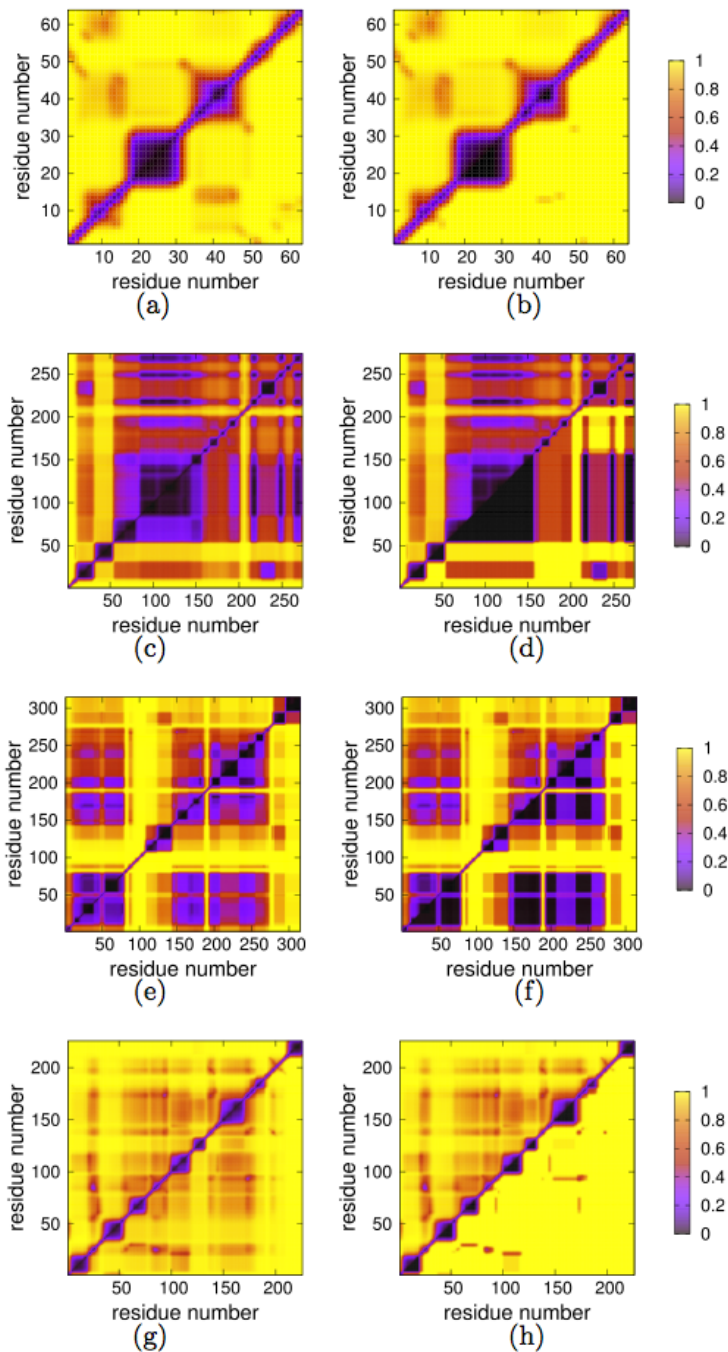


Figure 4.5: Mechanical coupling maps (MCM) provide a more nuanced view of co-rigidity. Juxtaposed to the all-or-nothing RCM plots above, MCM values quantitatively indicate how many spare pebbles are present within each residue pair. As a result, the power of the VPG- x algorithm is further underscored due to its ability to detect subtle properties missed by the VPG. The organization of this plot and the parameters used are exactly the same as Fig. 4.4.

by the VPG, resulting in both regions to be included in the primary (red) rigid cluster, but the VPG- x algorithm correctly identifies them as, on average, distinct rigid clusters.

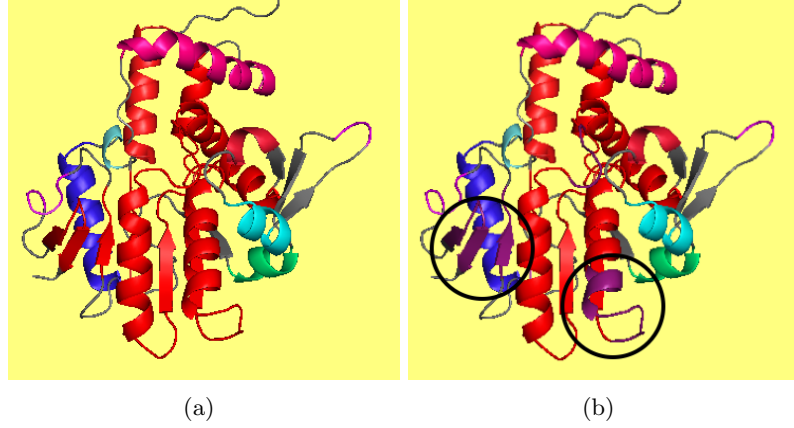


Figure 4.6: Rigid Cluster decomposition for VPG (a) and VPG- x (b) for a large methyltransferase structure (1AF7). The black circles in the bottom panel highlight regions of disagreement between the algorithms.

We have shown that VPG- x provides a better description of rigidity compared to the VPG. However, the parameters x and R have been chosen somewhat arbitrarily. A natural question is thus, “What values of x and R maximize improvement?”. In this regard we calculated the root mean square distance (RMSD) between the $\overline{\text{PG}}$ and VPG- x MCM plots (at different values of x and R) versus an “optimal” MCM map, which is defined as $\overline{\text{PG}}$ averaged over 1000 samples. Fig. 4.7 shows how well the various PG variants that rely on sampling reproduces the “optimal” behavior as a function of the number of samples. Here, we use three examples from above (the neurotoxin, lipoprotein and oxidoreductase structures) and introduce one new example from our dataset that is particularly biomedical important, which is the structure (pdbid = 1A1X) of the oncogene MTCP-1 [35]. In each case, as more PG samples are considered, the $\overline{\text{PG}}$ better approximates the “optimal” behavior. In the same way, the VPG- x results also improve as a function of the number of samples in a systematic and expected way. However, the partial mean-field treatment of the VPG- x algorithm prevents it from ever exactly reproducing the $\overline{\text{PG}}$ descriptions. Further, the more $\overline{\text{PG}}$ -like, the better each VPG- x variant does, which is consistent with the results presented above. While obvious and expected trends are found, no optimal values for x and R emerge from this analysis. A noteworthy characteristic of the VPG- x algorithms, which

is inherited from the VPG, is that it converges very rapidly at lower values of x . Conversely, at high values of x , the number of realizations needed to converge makes the speedup over $\overline{\text{PG}}$ largely negligible.

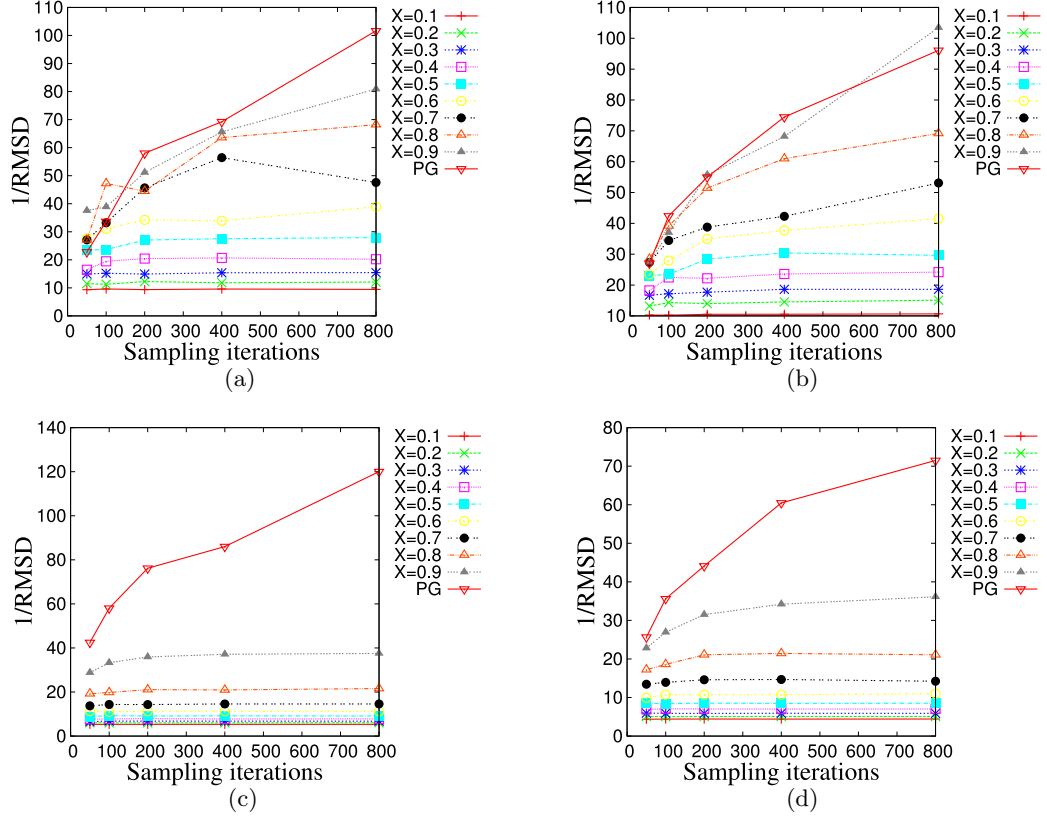


Figure 4.7: Accuracy vs. computational cost. The inverse root mean square distance (1/RMSD) between the MCM plots from each PG variant and an “optimal” $\overline{\text{PG}}$ using the corresponding number of samples is plotted. In all cases, the accuracy improves as more samples are considered. Similarly, accuracy also improves as the VPG- x variants become more $\overline{\text{PG}}$ -like. The presented examples are: (a) structure of a neurotoxin, (b) structure of an oncogene, (c) structure of a lipoprotein and (d) structure of an oxidoreductase enzyme.

4.4 Conclusions

In this chapter, we introduced the VPG- x algorithm. The VPG- x algorithm is based on a hybrid probabilistic/sampling algorithm where a portion of the interactions are treated in a VPG-like way and the remaining are statistically sampled using a $\overline{\text{PG}}$ -like treatment. The results from this hybrid approach clearly provide a more nuanced and accurate description of network rigidity within protein structures compared to those from the VPG, which suppresses fluctuations. In addition, the results intuitively and systematically transition

from VPG-like to $\overline{\text{PG}}$ -like as the variable x is adjusted. These same conclusions are arrived at regardless of which similarity metric is considered. As expected, the VPG- x algorithm converges quickly at small values of x , but the accuracy fails to approach that of $\overline{\text{PG}}$ because only a few interactions are sampled. Conversely, at high values of x , the VPG- x algorithm converges slowly and, more critically, it still fails to reproduce the accuracy of $\overline{\text{PG}}$. Thus, in situations where maximal accuracy is required, the original $\overline{\text{PG}}$ variant remains the best option. However, when such extreme accuracy is not needed, which is frequently the case, the VPG- x algorithm provides a significant accuracy increase over the VPG at limited computational cost.

4.5 Future work

The primary purpose of this chapter is to introduce the concept of the VPG- x algorithm and demonstrate its promise. While the VPG- x algorithm has generated encouraging results, we believe this approach can be significantly improved by two variations. The first is based on the fact that we are not intelligently determining where to apply the sampling, meaning we are currently randomly assigning each interaction to the VPG-like versus $\overline{\text{PG}}$ -like treatments. As such, we are often sampling in regions where the mean-field treatment is sufficient and often applying mean-field in a critical region where sampling is necessary. Eliminating both of these contra-situations is expected to greatly improve performance of a hybrid method. In future work, we will develop a bootstrapping method to identify critical regions where sampling is most necessary, and then only apply the sampling within those critical regions. A “background” VPG-like description will correspond to non-critical regions. While the details have not yet been finalized, these critical regions typically occur near *isostatic* regions where the network is marginally rigid, thus leading to frustrated behavior that suppresses VPG accuracy. These areas can be initially determined by the VPG as an initial step. Then some regions will be subjected to sampling, keeping a VPG-like description to the remainder of the structure. After a relatively small number of samples are taken, a more accurate determination can be made to identify additional regions to sample. In this way, the increased computational cost of the VPG- x algorithm can be kept in check. Far away from the rigidity transition, the total number of samples will be very small. As

the rigidity transition is approached, the number of samples will increase as fluctuations play more of an important role, perhaps naturally reaching the limit of the $\overline{\text{PG}}$.

The second variation is related to our uniform treatment of P_{nat} , which is an extreme (unphysical) assumption that makes the results presented here the worst case. By using context dependent H-bond probabilities lifted from [24], the results of the VPG and VPG- x will much more readily match with the $\overline{\text{PG}}$ results because most H-bond probabilities are near 1 or 0, where the VPG and $\overline{\text{PG}}$ give nearly identical results (actually this is shown in next chapter). The two variations can be combined, since the latter variation is a passive improvement that should automatically occur when the context dependence of a H-bond is taken into account. The nature of both variations is to focus on a much smaller number of H-bond that are critical to the flexibility within a protein. Therefore, the work toward development of a more efficient algorithm is tied to identifying a critical set of hydrogen bonds having great influence on protein stability and function.

CHAPTER 5: COMPARISON OF A MEAN FIELD ALGORITHM AGAINST AN ENSEMBLE-BASED ALGORITHM USING PROBABILITIES FOR HYDROGEN BONDS DERIVED FROM THEIR RESPECTIVE ENERGY

5.1 Introduction

In chapters two and three we compared the Pebble Game (PG) and the Virtual Pebble Game (VPG) algorithms using different test cases such as disordered lattices and an extensive 272 protein dataset (non-redundant at SCOP [33] family level). According to a vast number of metrics that we employed, there is strong evidence to suggest that VPG is a very good predictor of the mechanical responses calculated by the ensemble of PG descriptions ($\overline{\text{PG}}$). This result holds optimistic expectations for using the VPG on real-life applications. Nonetheless, it is important to note that several abstractions have been made when generating the test cases. One such abstraction is the uniform probability of existence for all the fluctuating interactions for any given ensemble. The parameter that controls the existence of the fluctuating interactions is the variable P_{nat} . When we compared both algorithms at a particular P_{nat} we considered that all the fluctuating interactions were present with the same probability regardless its strength. This assumption led to a slightly unphysical test-case that, based on the following discussion, actually represents the worst-case scenario. We refer to this model hence forth as *uniform*.

When we analyzed the calculation of degrees of freedom (DOF) in chapter two and the RAND measure in chapter three, we observed a region where the difference between both approaches was maximal. This particular point very well corresponds with the rigidity threshold, where the protein (or lattice) transitions from being globally flexible to globally rigid. The major discrepancies occur at this point given that the fluctuations are maximized and that the VPG suppresses fluctuations. As a counterpart, both algorithms yield identical calculations when P_{nat} is away from the rigidity transition (near a value of zero or one). In proteins, the fluctuating interactions are going to range from very weak (barely perceptible) to very strong (ever present) energy levels. Considering that the probabilities of existence

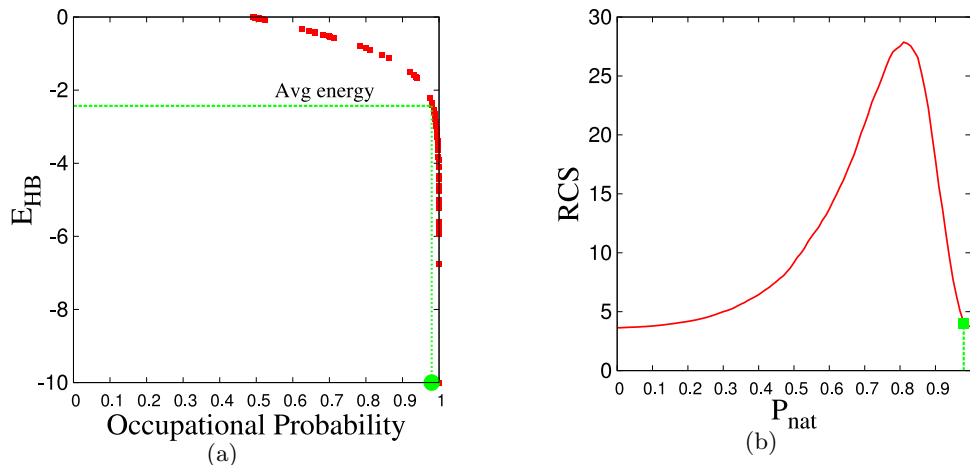


Figure 5.1: Potential energy of each H-bond versus its corresponding occupational probability (a) and the RCS curve (b) for a neurotoxin structure.

for these interactions are directly proportional to their energy, we could expect that a differentiation in their probabilities may improve the similarity between PG and VPG. We are going to call this model *differentiated*.

Fig. 5.1 a) shows the energy and corresponding probability of existence (occupational probability) for all H-bonds in a neurotoxin protein (1AHO). The stronger the energy (more negative) the more likely an H-bond will be present within the network at any given moment. A basic and intuitive approach to know how much fluctuations (on average) correspond to this particular set of H-bonds implies to calculate the average energy for all H-bonds (~ -2.43 kcal/mol) and its respective probability $P_{avg} = 0.98$. Panel (b) shows the RCS curve for the same protein under the uniform model, the rigidity transition occurs at the peak of the curve, $P_t = 0.81$. According to the RCS curve, we can observe that P_{avg} does not correspond to the point with largest fluctuations, suggesting that a differentiated model may do a good job driving away from this situation. An important feature of the differentiated model is that most of the H-bond probabilities will be near 1 or 0 (sharp peak functions), where the VPG and \overline{PG} give nearly identical results.

The differentiated model also leads to a more realistic treatment of protein networks. Figure 5.2 exemplifies this model with a network of eight vertices and three fluctuating interactions. Assuming that each fluctuating interaction has a different energy and consequently a different probability of existence, note that some fluctuating interactions are

always present within the PG realizations while others are intermittently or barely present. Interestingly, some degree of homogeneity in these networks is observed, this is of particular relevance given that from chapter two we know that VPG is very accurate when the networks have this property.

In this chapter, we move to a more realistic treatment for the fluctuating interactions within each protein network. For this purpose we derive the probabilities of each interaction based on its respective energy. The lowest the energy the highest the probability of existence for that particular interaction. Using this model we compared both algorithms and we show through the calculation of DOF, RCMs, MCMs and Rand measure that, as expected, the VPG algorithms performs better than in the other less realistic test cases. This result is of great importance given that it offers a more quantifiable perspective of the potential application of the VPG.

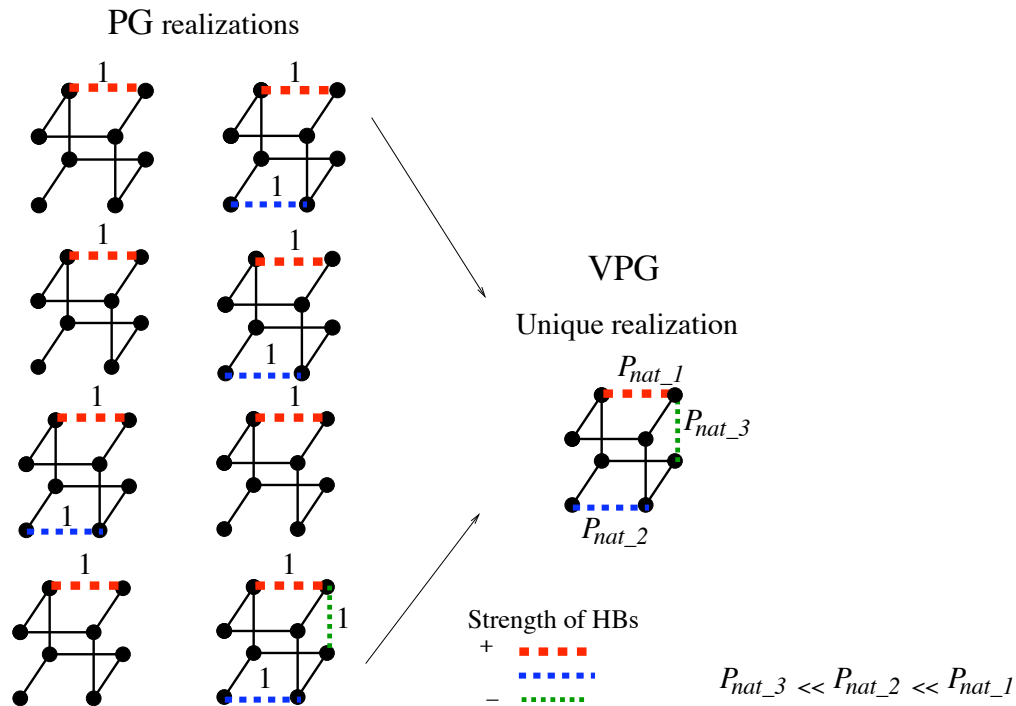


Figure 5.2: The probability of existence of a fluctuating interaction is based on its respective energy. Based on this model, for PG several network topologies are created whereas for the VPG just one topology is needed.

5.2 Materials and Methods

For comparison purposes we again use the 272 protein dataset presented previously. This dataset has shown to encompass a variety of protein structures which has helped generalize our results. To compare the PG and VPG over different ensembles of protein networks, having each ensemble the same average number of hydrogen bonds (order parameter) we apply algorithm 8. To define the variables in the algorithm, be hb_{max} the total number of hydrogen bonds for a specific protein, μ a lagrange multiplier, E_{hb} a unidimensional vector of size hb_{max} that stores the energy of each hydrogen bond and P_{hb} a unidimensional vector of size hb_{max} that stores the probability of each hydrogen bond.

When there are hb hydrogen bonds on average in the protein network, the algorithm calculates the probability of each particular hydrogen bond to exist, meaning that the sum over all the hydrogen bond probabilities must be equal to hb . For example, to calculate the occupational probabilities in Fig. 5.1a, hb was set to 65 H-bonds ($P_t \times 80$ H-bonds), since we were comparing the uniform and differentiated models at the point with maximal fluctuations. The *for* loop sweeps all the possible (average) number of hydrogen bonds in the network, except both end points (when $hb = 0$ and $hb = hb_{max}$), to avoid bad edge effects. Note that to calculate the mechanical variables, PG or VPG must be included in step 5, when all the probabilities were already assigned. The function *get_new_μ*($hb_{max}, E_{hb}, hb, old_μ$) calculates the lagrange multiplier μ that makes the Boltzmann sum ($\sum_i^{hb_{max}} e^{-\beta(E_{hb_i} - \mu)}$) of all the probabilities of hydrogen bonds equal to hb . This function starts off with an initial guess for the lagrange multiplier (*old_μ*) that is constantly updated until the average number of hydrogen bonds is hb . Afterwards, the function *get_hb_prob*($hb_{max}, E_{hb}, P_{hb}, \mu$) assigns to each hydrogen bond its respective weight (probability) on the sum for which the lagrange multiplier was previously calculated.

Algorithm 8 *get_probabilities_from_energy*

```

1: for  $hb = 1$  to  $hb_{max} - 1$  do
2:    $\mu \leftarrow \text{get\_new\_}\mu(hb_{max}, E_{hb}, hb, old\_μ)$ 
3:    $old\_μ \leftarrow \mu$ 
4:    $P_{hb} \leftarrow \text{get\_hb\_prob}(hb_{max}, E_{hb}, \mu)$ 
5:   MECHANICAL ASSESMENT
6: end for
```

5.2.1 Network Similarity Metrics

In chapter three you can find a detailed description of the network similarity metrics that we employ here. We compare PG and VPG at the calculation of available degrees of freedom (DOF) in the complete network. This calculation is made once all the constraints were added to the system. According to the number of constraints and the number of degrees of freedom that they consume we can analyze the excess of pebbles and therefore identify flexible, constrained and over-constrained regions in the proteins. On average, the number of DOF is a monotonically decreasing function with respect to the number of constraints in the system, formerly represented by the variable P_{nat} and in this chapter represented by the average number of hydrogen bonds in the network.

A more elaborate comparison is based on the identification of rigid clusters of atoms. A rigid cluster represents a rigid body in space, meaning that all of its atoms (elements) move together as a single body. The constraints that are placed within a rigid cluster have exhausted the excess of pebbles, just leaving the six trivial DOF for its reference in 3D space. The Rand Measure (RM) [39] is a well suited metric to compare clusters of elements. When we compare the cluster distribution of two networks, a RM equal to one will represent that the clusters of vertices in both networks have exactly the same elements. On the contrary, a value of zero will represent that both networks have a totally different cluster distribution. Interestingly, the latter corresponds to the case where one network is totally formed by *one*-element clusters (globally flexible) while in the other there is a unique *n*-element cluster (globally rigid).

A way to visualize rigid clusters of elements is provided by the Rigid Cluster Map (RCM). Two residues are co-rigid if the maximum number of DOF that can be gathered between them is six. If this is the case, the RCM will color red (1) the intersection of those residues in the map. On the opposite side, when there are more than six pebbles for a pair of residues, representing some degree of flexibility, the RCM will leave blank the intersection (0). A more nuanced view with respect to the number of DOF between a pair of vertices is given by the Mechanical Coupling Map (MCM). The MCM shows through a color scale the maximum number (level of flexibility) of DOF for a given pair of residues. This scale has

been normalized for practical reasons, i.e., a value of one represents the maximum excess of DOF (six pebbles) for any given pair of residues. Note that the DOF for two elements that have the full amount of flexibility will be twelve.

5.3 Results and Discussion

First, we compare both approaches regarding the total number of available DOF in the system. We show eight proteins where we perform this comparison. These proteins with their respective PDB code are: an oncogene (1A1X [35]), a FLAP endonuclease (1A76 [36]), an oxidoreductase (1A8L [50]), an apolipoprotein (1AEP [56]), methyltransferase (1AF7 [49]), neurotoxin (1AHO [34]), spasmolytic polypeptide (2PSP [58]) and a subtilisin BPN' (2SIC [57]). This information is shown in Figure 5.3. For PG, the average (over 200 realizations) value is reported, standard deviation bars are also shown for comparison purposes. Most of the figures confirm that the VPG calculation is within, or slightly below, one PG standard standard deviation. The difference between both approaches in the region where most fluctuation occur is not as dramatic as when we compared both algorithms using disordered lattices in chapter two. Overall, VPG shows a remarkable approximation with respect to PG for the calculation of available DOF.

After the calculation of DOF we compared both algorithms at the identification of co-rigid residues for the same proteins from above. The RCMs that are shown in Fig. 5.4 contrast PG and VPG at the particular point (worst case) where the difference in the calculation of degrees of freedom is the largest across the average number of hydrogen bonds. The RCM compares side by side both approaches, in the left upper triangle the $\overline{\text{PG}}$ calculation is shown, VPG is opposite. Note that to present the information for $\overline{\text{PG}}$ and since the RCM is a binary plot, the rigidity calculation corresponds to a majority rule, e.g. if for most of the individual PG calculations ($> 50\%$) a pair of residues is rigid, then they are going to be co-rigid in the $\overline{\text{PG}}$ RCM. In general, there is a similar rigidity pattern for most of the proteins. Although these panels show all the possible outcomes for the VPG rigidity assessment (under/overestimation) there are minor differences with respect to PG. The methyltransferase structure (panel e) presents a “mixed” rigidity calculation for VPG, some regions of the largest rigid cluster are under/overestimated. This cluster is formed by

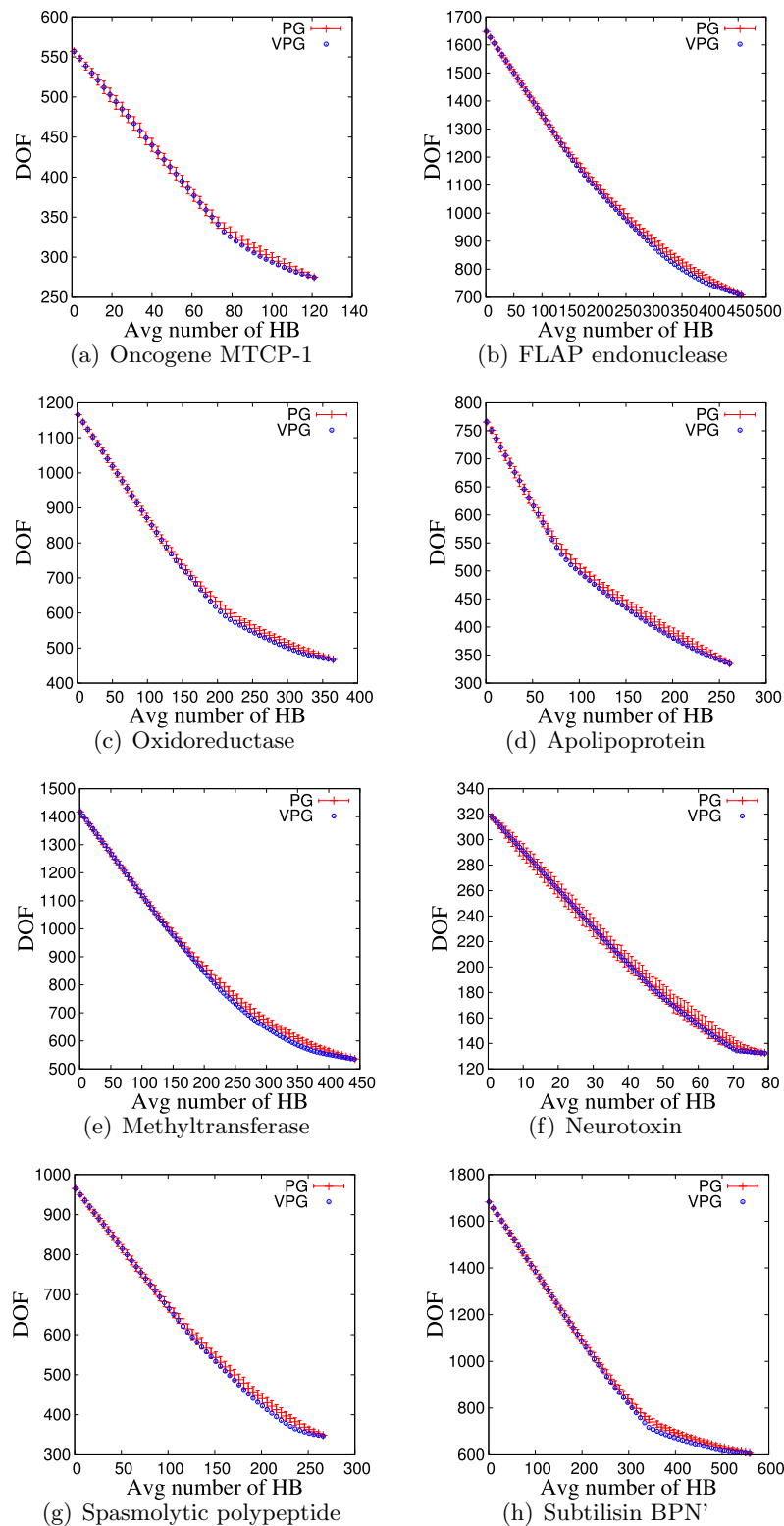


Figure 5.3: Comparison of the calculation of the total number of available DOF for eight proteins.

five α -helices (shown over the diagonal) and six β -sheets (off diagonal elements). Some regions of the α -helices are underestimated by VPG while sections of the β -sheets are overestimated. The subtilisin BPN' structure (panel h) presents the largest contrast, VPG mainly identifies some α -helices as rigid, while PG markedly found vast co-rigid regions with respect to the same α -helices. This particular comparison takes us to the next analysis.

Based on the RCMs, *how is it possible for the VPG to underestimate rigidity if it always under-predicts the DOF (cf. 5.4 a,h vs. 5.3 a,h)?* Moreover, this question is even more on-point considering the RCMs are compared at the average number of H-bonds with maximal deviations between both algorithms. Two possible answers do exist. The first one is based on the calculation of DOF on the flexible regions (blank areas) in the RCM. Being RCM a binary plot it is susceptible to small perturbations of pebbles, i.e. there is no difference whatsoever to have 1×10^{-9} or 6 pebbles, for a pair of co-flexible residues. Be this the case, VPG could have an almost perceptible number of pebbles on “flexible” regions, giving the false impression of global flexibility while a global count of DOF will offer an opposite conclusion. The second answer is based on the ensemble nature of PG. Since both plots (calculation of DOF and RMC) show an average calculation, it could be the case that most of the PG individual calculations are indeed more rigid than VPG, but those where the PG is flexible, it is far greater more flexible than VPG. Leading to a counting of DOF greater than the one of VPG, but also, to a more rigid view in the RCM. Our next analysis addresses this question.

A more nuanced view of the rigidity composition is detected by the Mechanical Coupling Maps (MCM). Fig. 5.5 shows the MCM for $\overline{\text{PG}}$ and VPG over the same proteins from above (same parameters as in RCM). Unlike the RCM, the MCM gives a global and detailed view of flexibility/rigidity throughout the protein network. If the rigidity pattern that was detected by the RCM is clear, the MCMs confirm it and make it even more evident. As an example of this latest statement consider the “very contrasting” (in RCM) subtilisin BPN' protein (Fig. 5.4 h). In its RCM, this particular protein showed a characteristic behavior that even made us placed an interesting question regarding VPG underestimating rigidity. While in the MCM (Fig. 5.5 h) we can clearly detect that, for VPG, the majority of its residue pairs exhibit some lack of flexibility, very well approaching the PG calculation. This point

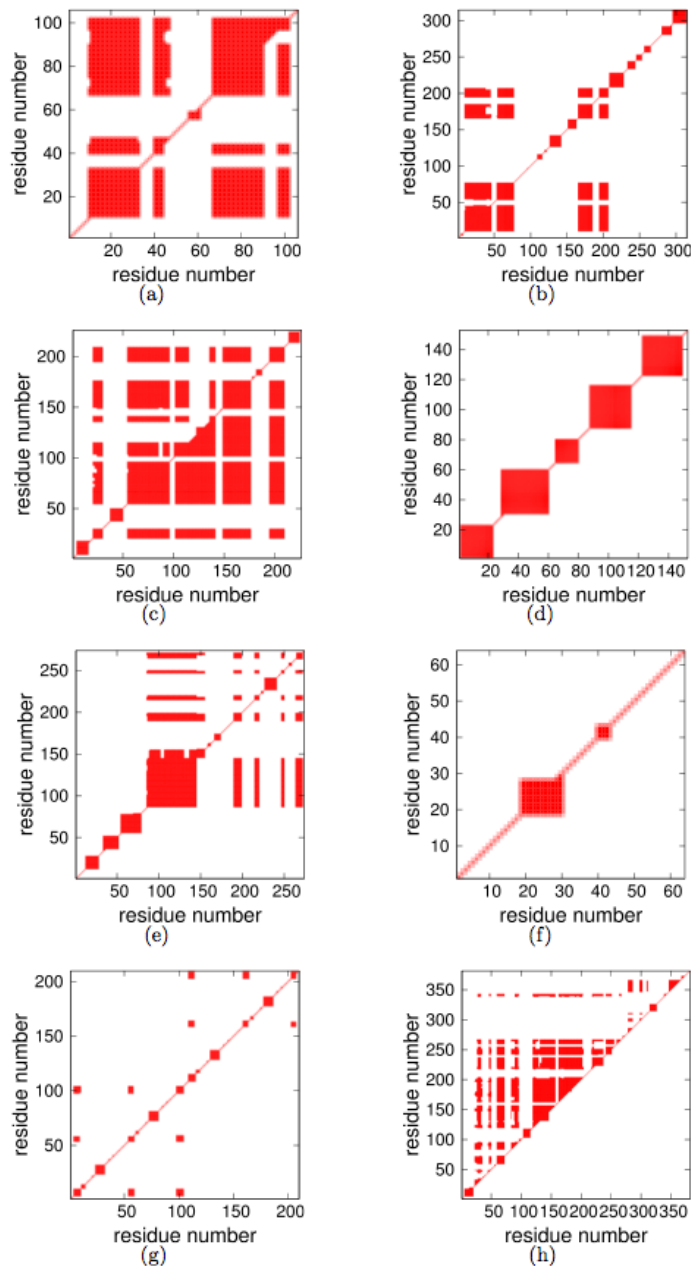


Figure 5.4: Rigid Cluster Maps (RCM) for eight proteins. In each panel a $\overline{\text{PG}}$ -VPG comparison is made for: a) an oncogene, b) a FLAP endonuclease, c) an oxidoreductase, d) an apolipoprotein, e) methyltransferase, f) neurotoxin, g) spasmodic polypeptide and h) a subtilisin BPN'. $\overline{\text{PG}}$ rigidity assessment is shown in the left upper triangle, VPG is opposite.

supports one of the reasons we previously gave to explain this phenomenon.

For all the proteins, the MCMs complement the rigidity view offered by the RCMs, confirming that VPG offers excellent approximations to the ones of PG under this model. According to the MCM plots, the VPG shows a remarkable agreement to the calculation

and distribution of DOF throughout the protein networks. This fact is also emphasized given that the comparison was made where the largest difference between both approaches do occur.

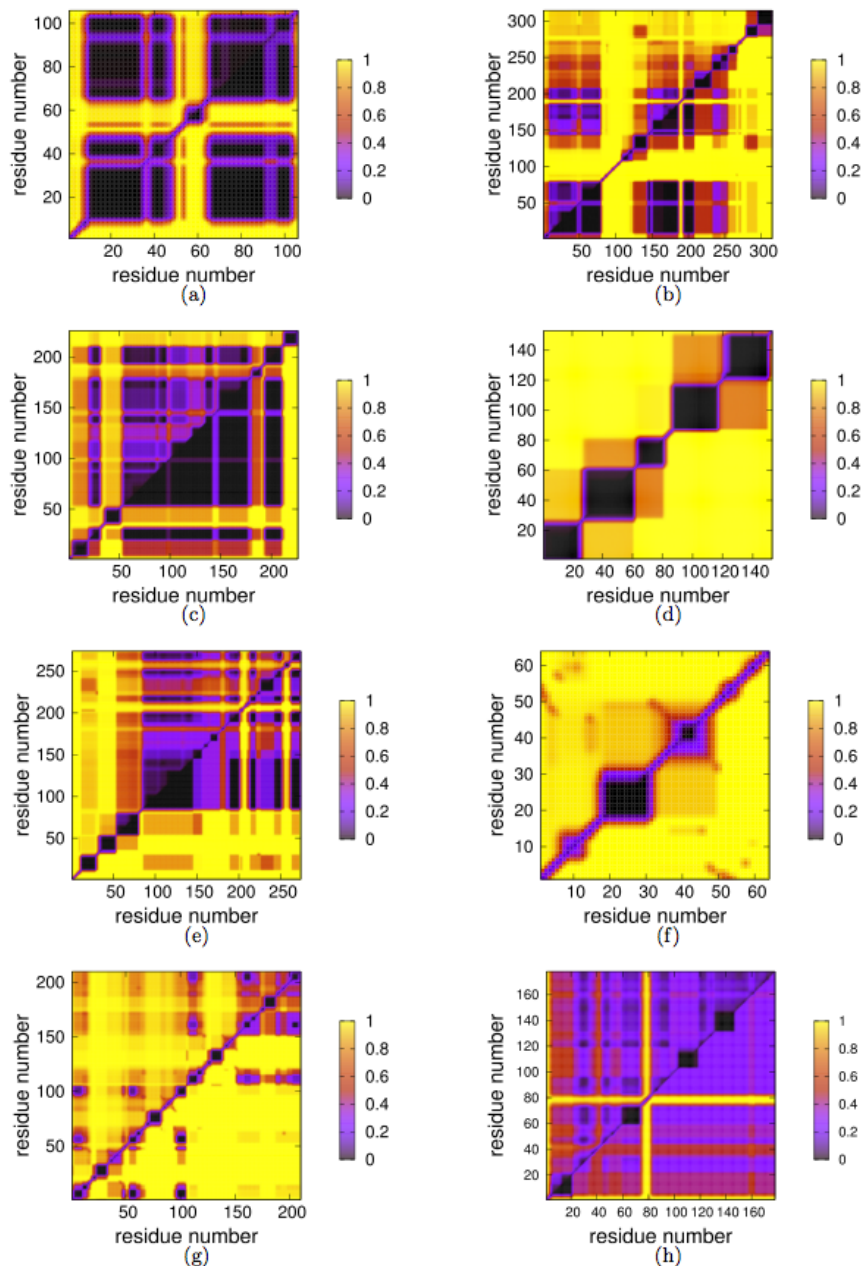


Figure 5.5: Mechanical Coupling Maps (MCM) for eight proteins. In each panel a $\overline{\text{PG}}$ -VPG comparison is made for: a) an oncogene, b) a FLAP endonuclease, c) an oxidoreductase, d) an apolipoprotein, e) methyltransferase, f) neurotoxin, g) spasmolytic polypeptide and h) a subtilisin BPN'. $\overline{\text{PG}}$ rigidity assessment is shown in the right upper triangle, VPG is opposite.

5.3.1 Physicality of three bars per hydrogen bond

The model to generate protein networks from PDB files that is explained in detail in chapter three, considers the representation of a hydrogen bond with three distance constraints. This representation is based on the abstraction that two degrees of freedom are required for each, the donor and the acceptor, while the one that is remaining considers the angular constraint. The rigidity analysis using this model and its respective conclusions have been satisfactory, finding in most of the cases ($> 80\%$) a rigidity transition for the protein networks. Rigid cluster susceptibility curves for some exemplar proteins are shown in chapter 3 (Figure 3.9), note that the peak of the curve represents the rigidity threshold. According to this figure, we can observe that effectively the proteins transit from a globally flexible state to a rigid state under this model.

Within the community of protein rigidity analysis, the topic about the number of distance constraints to represent hydrogen bonds is far from being settled. Actually, current software packages to analyze rigidity in proteins [59, 60] leave this parameter open for the modeler, so that any number of distance constraints can be chosen. In this matter, private discussions [61] have suggested an alternative to represent a hydrogen bond with five distance constraint as in other works [13, 14]. It is important to remark some differences that can occur when the number of distance constraints is increased. The obvious consequence is that using five bars (distance constraints) will lead to a more rigid structures, given the requirement for more pebbles when covering a hydrogen bond. In this way, it is expected to see the rigidity transition for a larger number of protein structures in our dataset. Another consequence is that the number of fluctuations is increased given the cooperativity nature of the constraints. This last point proves to be relevant for our studies, given that we know deviation between the VPG and PG is related to the extent of fluctuations within the ensemble. As such, our next set of experiments characterize H-bonds with five constraints. We first analyze the rigidity calculation of the PG algorithm on both models, with three (PG_3) and five constraints (PG_5), over 200 realizations. Fig. 5.6 shows the calculation of degrees of freedom for this comparison (standard deviations are also shown) over an oncogene (a), oxidoreductase (b), apolipoprotein (c), neurotoxin (d), spasmodic polypeptide

(e) and subtilisin BPN' (f). As expected, the five-bar model consume a larger number of DOF. Nonetheless, a qualitatively similar behavior for both models is observed. Based on this results, we hypothesize that an analysis over a larger set of protein networks will lead to similar results. Particular deviations between both models are dependent upon specific network characteristics, such as localization of fluctuating interactions.

After the previous comparison, a natural question would be: *how does the VPG compare to the PG under the five-bar model?* The last experimental section where we compared both algorithms under the five-bar model was in chapter two, using disordered square lattices. In those comparisons, VPG was several standard deviations away from PG near the rigidity transition. It is fair to comment that uniform probabilities for fluctuating interactions were used in that comparison. Given the regularity of square lattices, the calculations of PG showed very little standard error. Fig. 5.7 shows the comparison between the VPG and PG under the five-bar model using the same six proteins from above. Interestingly, from this figure we can observe that although the fluctuations are increased the deviations of VPG from PG are relatively small, even in some cases falling within one standard deviation away from PG (similar to the three-bar model comparison). It is encouraging that VPG maintains its DOF calculation close to the one from PG. Overall, the VPG keeps offering a very good alternative to the calculation of mechanical responses in comparison to the traditional PG algorithm.

5.4 Future work

This chapter is the seed of a journal paper. The results presented support our hypothesis regarding the good approximation to PG given by VPG. Nonetheless, we are planning to extend the experimental section by adding more metrics to this analysis. We would like to compare both algorithms regarding:

1. Visualization of rigid clusters over the protein structures
2. Comparison of rigidity at dihedral angle level (agreement measure)
3. A statistical analysis of the VPG rigidity assessment with respect to all the PG individual rigidity assessments

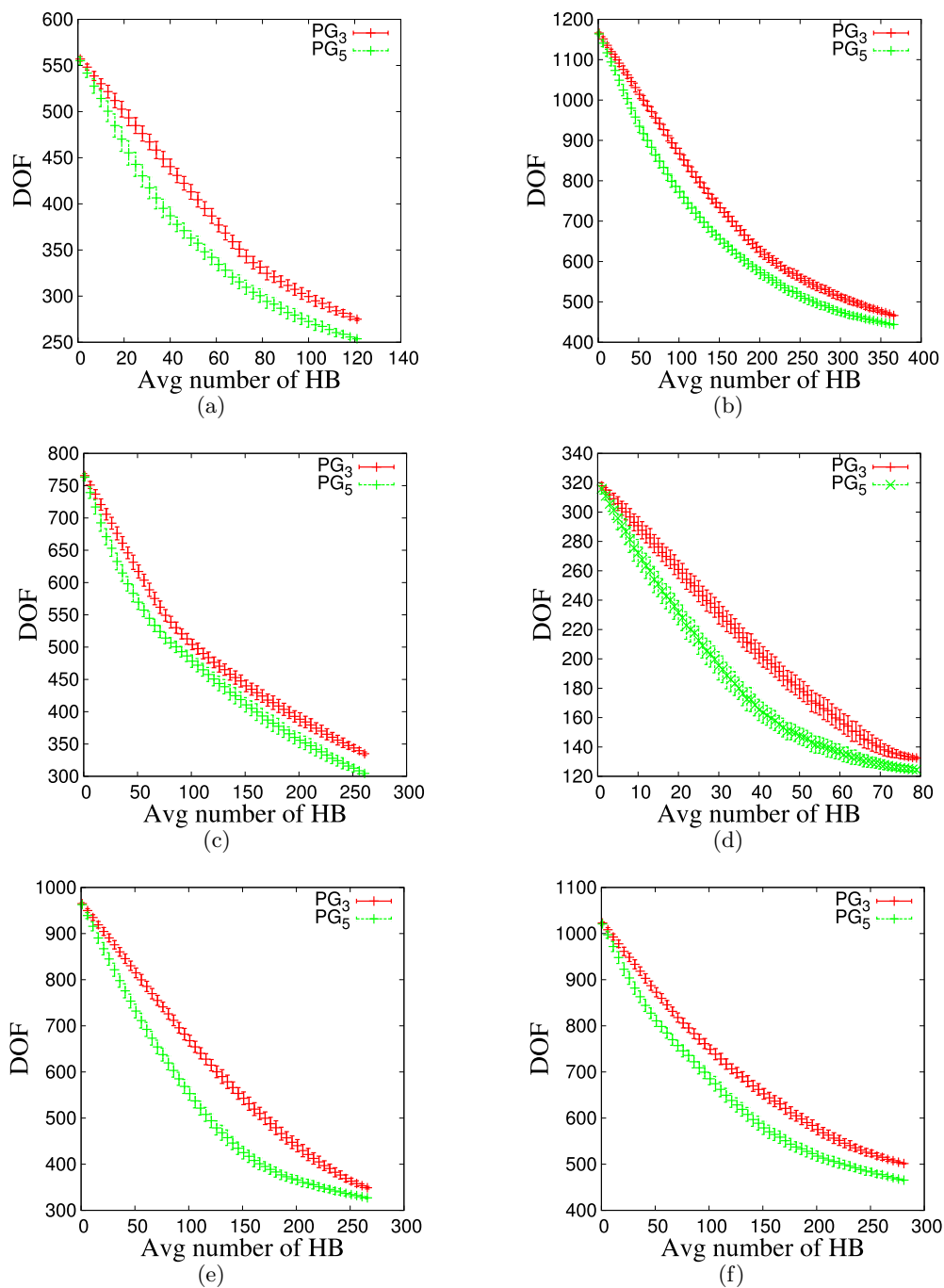


Figure 5.6: Calculation of DOF by PG when using three versus five distance constraints per hydrogen bond. The protein networks are: a) oncogene, b) oxidoreductase, c) apolipoprotein, d) neurotoxin, e) spasmodic polypeptide and f) subtilisin BPN'

4. The calculation of the rigid cluster susceptibility (RCS) curve and related variables (average cluster size and largest cluster)

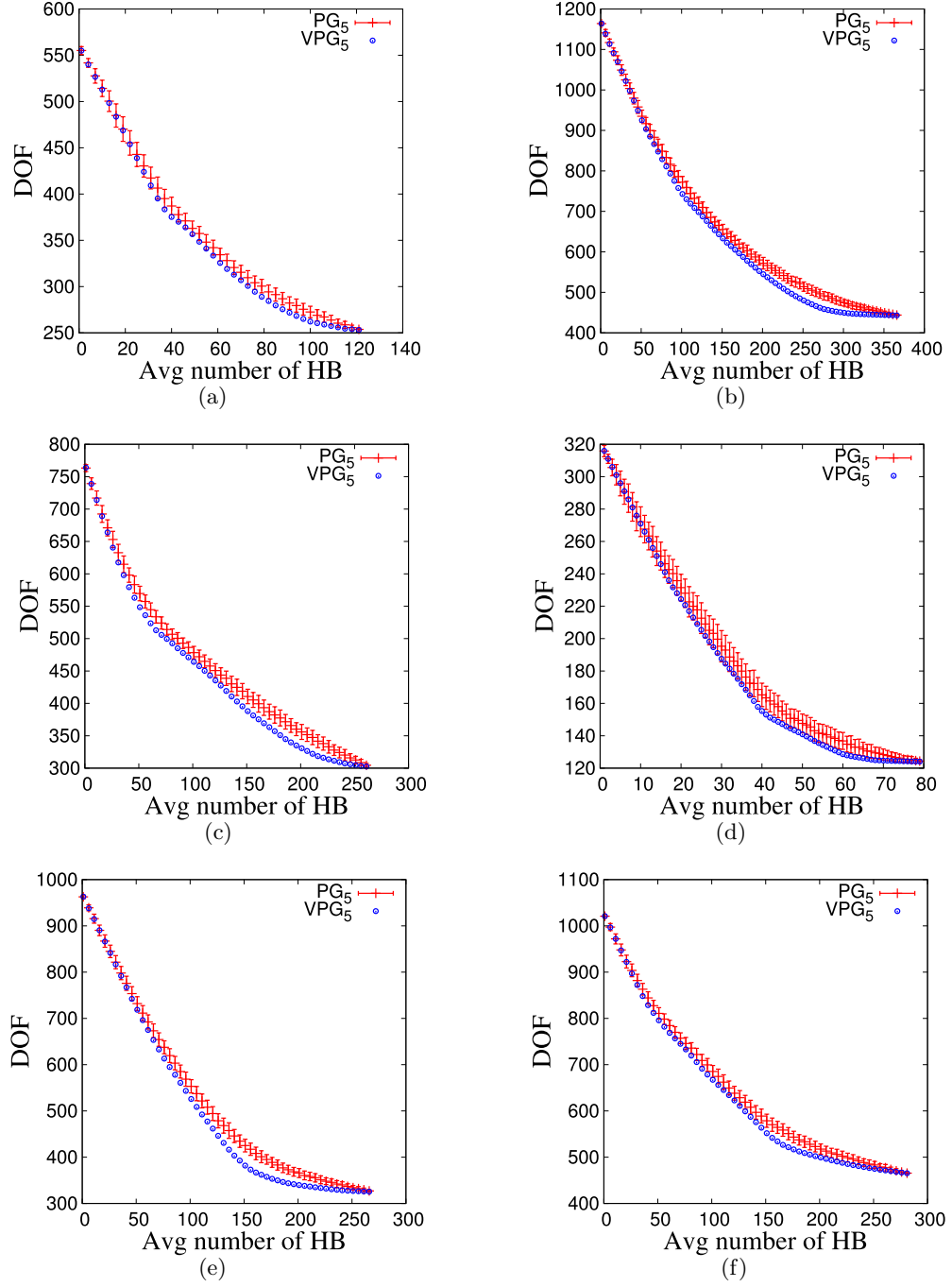


Figure 5.7: Comparison of the calculation of DOF by PG and VPG using the five-bar model. The protein networks are: a) oncogene, b) oxidoreductase, c) apolipoprotein, d) neurotoxin, e) spasmodic polypeptide and f) subtilisin BPN'

5.5 Conclusions

In this chapter, we used a biologically relevant model to compare the PG and VPG algorithms. To use this model we selected a 272 protein dataset that was introduced in

a previous chapter. Before this chapter, the probabilities of existence of fluctuating interactions were kept uniform, meaning they were treated equally throughout the rigidity calculations. In the experimental set up, the probability of each fluctuating interaction was derived based on its particular strength (energy potential).

Several metrics were employed for comparison purposes. We first compared both algorithms regarding their calculation of DOF. The comparison was made at different average number of hydrogen bonds in every particular network. Along this comparison we observed that for most of the cases the VPG calculation fell within one standard deviation of those of PG.

After the calculation of DOF we were able to detect the point where the difference between both approaches was maximized, we chose this point to perform the next analysis. We employed the RCM to compare the identification of residues that are co-rigid throughout the protein network. Using the RCM we were able to determine how well VPG approached the rigidity patterns found by PG. To complement this information, we also compared both algorithms using MCMs. In a more nuanced comparison, we observed that VPG is able to capture subtle DOF that allow a pair of residues to have some degree of flexibility.

Based on other works, we moved from representing a hydrogen bond with three distance constraints to five. We compared both algorithms over this model and we conclude that VPG still continues offering a very good approximation for the calculation of DOF within protein networks.

CHAPTER 6: DISCUSSION

Through the previous chapters we have quantitatively measured the accuracy of the VPG algorithm. In this chapter, we would like to complement those analyses by providing further discussion on several topics of interest.

6.1 Heuristics for the VPG

From an implementation point of view, the VPG and the PG have the same complexity, because they rely on virtually the same set of algorithms related to pebble searches and condensation. In practice, the similarity is even greater because a finite precision can be placed on real variables, allowing the VPG to be integer based. For example, if a desired precision is to be 10^{-6} , we represent each pebble using a million fractional pebbles. We have implemented the VPG with double precision variables and with different finite precision levels, and established consistency in all results. However, the finite precision version that uses only integers is simpler to implement, and runs faster.

It is important to note that although the VPG does not give the exact answer for rigidity properties in general, it is exact when no fluctuating constraints are considered. In this case, the VPG is algorithmically identical to the PG. When there are no fluctuating constraints each realization for the network topology is exactly the same for the VPG and the PG. In this limit, no fractional pebble capacity occurs within the network, and as a result the allowed number of pebbles on a vertex or edge covering is always an integer ranging from 0 to 6. However, the body-bar PG as implemented by Jacobs [13] treats the network as a multi-bar network, and performs individual pebble searches for each distance constraint. As such, it may require 6 different pebble searches to find 6 pebbles on a vertex. In the VPG, multiple pebbles are searched for and collected simultaneously if possible. This change in the implementation is a trivial step, but does make the performance about 20% times faster in typical applications.

The interesting question is what happens when there are fluctuating edges? The VPG provides a lower bound estimate to the number of DOF within the network. The heuristic for the VPG is based on running many PG in parallel over an ensemble of networks differing only within fluctuating edges, and the pebble flow through a fluctuating edge will self-average over the ensemble. Then, within the VPG, the flow of pebbles through a fluctuating edge will be restricted by an average pebble capacity that is determined by the ensemble average. The structure of the VPG and the PG algorithms govern the flow of pebbles while maintaining local conservation rules. Consequently, when viewing the VPG as tracking the average flow of pebbles, it immediately translates to the flow of probability that a pebble is present.

We know from rigidity percolation [10] that the DOF within a network is very good at self-averaging away from the rigidity transition, which is why MCC works well in many applications involving networks with a near uniform density of constraints, and minimal fluctuations. However, near a rigidity transition where fluctuations are large, the self-averaging approach (or mean field treatment) breaks down. Self-averaging is a property that tends to become more robust in higher dimensions. As was described on chapter two, self-averaging holds up much better in 3D than 2D as expected, yet the VPG performs quite well in general (even in 2D).

By assigning the predetermined average pebble capacity (of $c \times p$) to each fluctuating edge, the final results are independent of the ordering of how the edges are placed (quenched or fluctuating edges). This feature is very important because it means that a single VPG provides a definitive answer. We tested the property that the answer is independent in a very stringent way. Not only did we scramble the ordering of the edges placed to show that all orderings give the same answer, we also subdivided edges into a stack of 10 contributions with randomized capacities that add up to $c \times p$ for fluctuating edges, and c_{full} for quenched edges (depending on the edge-type c_{full} could either represent five or six distance constraints). We then scrambled the order of all these partial contributions. The VPG under this condition runs slower, but the results are found to be independent of the ordering of all these components. Although we cannot rule out that there may exist an alternative averaging-rule that can better capture density fluctuations and provide definitive results

independent of the edge ordering, we know that the VPG defined above retains this special and desirable property important in practical applications.

We conclude this section with some final remarks about the correctness of the VPG as a new algorithm to estimate the average number of DOF within an ensemble of body-bar networks. Even in the PG, the pebble covering of an edge is actually fractionalized. This can be seen using the data structures that have been implemented in the VPG, and looking at the special cases where there is an exact mapping between the VPG and PG algorithms. We see that if an edge represents 5 distance constraints, the PG allows 0, 1, 2, 3, 4, and 5 of these bars (distance constraints) to be redundant. In this sense, the pebble covering within a PG is fractionalized (or quantized) to $\{0, 1/5, 2/5, 3/5, 4/5, 1\}$ of the maximum capacity allowed when the edge is fully independent. When viewed in this way, the finite precision implementation of the VPG that we use simply makes these fractions more refined. For example, representing each pebble as 200 fractional pebbles, then the allowed pebble fractions for covering an edge in the VPG are $\{0, 1/1000, 2/1000, \dots, 998/1000, 999/1000, 1\}$. Nonetheless, we agree that the rigor of the VPG as being faithful to generic rigidity for body-bar networks is an aspect that deserves a formal mathematical treatment.

Based on our heuristic arguments, the reason why the VPG achieves only a lower bound estimate for the number of DOF within a network (not the exact value) is because the averaging procedure suppresses fluctuations within the network. Thus the problem is related to statistics, and the neglecting of correlations that exist within individual realizations that the PG can capture. This phenomenon is a common occurrence in physical systems whenever mean-field theory is employed to describe the interactions. Our conjecture is that the amount of statistical fluctuations that are suppressed characterizes the severity of the approximation. We already tested this idea on chapter two, the degree of intrinsic fluctuations was quantified by a heterogeneity index, and we considered another type of PG where fluctuations play less of a role to see if the VPG provides better estimates. Our empirical results strongly suggested that the errors generated by the VPG are directly related to the mean-field nature of the approximation. Moreover, in chapter three, the VPG algorithm was found faithful in reproducing rigid cluster decompositions within protein networks that are statistically consistent with the typical response over the ensemble of

networks determined by the PG.

6.2 On the various ways to improve the accuracy of the VPG

The VPG which assigns an average pebble capacity of $c \times p$ (c and p represent the capacity and probability) to each fluctuating edge is one possible way to perform the self-averaging of the ensemble of all networks that the PG must be averaged over. This assignment is based on the very important property that the value of $c \times p$ is intrinsic to the edge itself, and does not depend on the state of the system (i.e. any other edge, or what is flexible or rigid within the network). Since each fluctuating edge could have 0 distance constraints or c distance constraints with probability $(1 - p)$ and p , respectively, the average value of $c \times p$ is a simple and intuitive averaging procedure. However, we have considered other ways to perform this averaging while the network is being built up. In this section we elaborate about different rules to averaging for the VPG algorithm. These algorithms were tested under the cubic lattice model, therefore the capacity c , for any fluctuating or quenched edge, equals five distance constraints.

The first algorithm that we implemented was called `VPG_Five_WithOut_Condensation` (*VPG5WOC*). The approach *VPG5WOC* (algorithm 9) looks for five pebbles to cover a given edge regardless its actual capacity c ($0 < c \leq 5$). Be *collected_pebbles* the maximum amount of pebbles found. If five pebbles are found, then the edge is going to be covered by its current capacity (that may be less than five), otherwise the edge is going to be covered with $\text{collected_pebbles} \times p$ pebbles. Note that if $c \leq \text{collected_pebbles} < 5$, the amount of pebbles to cover the edge would be $\text{collected_pebbles} \times p$. By implementing this idea, we tried to spend on average less pebbles on covering an edge since we could end up with $(\text{collected_pebbles} \times p) < c$. In this version no condensation is applied.

Algorithm 9 VPG5WOC

```

1: if collected_pebbles  $\geq$  5 then
2:   cover the current capacity  $c$  of the edge
3: else
4:   cover the edge with  $\text{collected\_pebbles} \times p$  pebbles
5: end if

```

The second algorithm that we proposed is called *VPG5M* (algorithm 10). This version

is named “5M” after “**M**ixing” the rules of the original VPG (classic) and VPG5WOC (previous algorithm). When covering an edge, if $\text{collected_pebbles} \geq 5$ then the edge is going to be covered with its actual capacity c (new and classic rules are the same). If $(5 \times p) \leq \text{collected_pebbles} < 5$ then the edge is going to be covered with $\text{collected_pebbles} \times p$ pebbles (new rule), otherwise the edge is going to be covered with collected_pebbles and condensation is applied (classic rule).

Algorithm 10 VPG5M

```

1: if  $\text{collected\_pebbles} \geq 5$  then
2:   cover the full capacity  $c$  of the edge
3: else
4:   if  $\text{collected\_pebbles} \geq c$  then
5:     cover the edge with  $\text{collected\_pebbles} \times p$  pebbles
6:   else
7:     cover the edge with  $\text{collected\_pebbles}$ 
8:     apply condensation
9:   end if
10: end if

```

The third algorithm was called *VPG5T* (algorithm 11), where ‘T’ stands for “Threshold”. In this version if $\text{collected_pebbles} \geq c$ then the capacity of the edge will be covered in full. If $T \leq \text{collected_pebbles} < c$ then the edge is going to be covered with $\text{collected_pebbles} \times p$, otherwise the edge will be covered with collected_pebbles and condensation will be applied. Note that T plays an important role in this algorithm, since it indicates the level of “tolerance” to the number of pebbles to find. Given the nature of this method, a follow up question is, “what value of T shall we use?”. We tested empirically different threshold values and took into consideration the speed and accuracy of the method to get the optimum $T = 0.15$.

6.2.1 Results for the comparison of the VPG methods

To make the comparison between these methods and the VPG and PG algorithms we run a set of experiments for two different system sizes at $L = 5$ and $L = 10$ (L being the number of vertices per side in the lattice). We followed the procedure to generate the test cases already described in chapter two. Fig. 6.1 shows the results for systems of size $L = 5$ at different q_{fix} and q_{fluct} parameters. We can note that there is no significant improvement

Algorithm 11 VPG5T

```

1: if collected_pebbles  $\geq$  capacity  $c$  then
2:     cover the full capacity  $c$  of the edge
3: else
4:     if collected_pebbles  $\geq T$  then
5:         cover the edge with collected_pebbles  $\times p$  pebbles
6:     else
7:         cover the edge with collected_pebbles pebbles
8:         apply condensation
9:     end if
10: end if

```

between any of the VPG versions with respect to PG, in fact the difference among the VPG versions themselves is imperceptible.

An interesting point is that the classic VPG rules are the only ones that give the exact same answer regardless the placement order of the edges. This fact affects the execution time of the other algorithms, since several runs should be implemented to average over.

Even when we increase the system size the behavior among the algorithms remains consistent. Fig. 6.2 shows qualitatively similar results for systems of size $L = 10$ with respect to $L = 5$. We can conclude that none of these approaches showed significant difference with respect to VPG. The execution times for one run of all the algorithms are comparable (data not shown), except for the VPG5WOC where the no-use of condensation makes it slower.

Despite much effort in finding an alternative rule that would provide a substantially better estimate than the VPG, no alternative averaging rule could be found to give a noticeable improvement to warrant the extra complexity of programming, or in some cases longer run times. For all the alternative rules we explored, we found a consistent undesirable property that the estimated DOF within the network depends on the ordering of how the edges are placed when building the network up recursively one edge at a time. Moreover, we could not find a preferred ordering that would give us the greatest lower bound. In short, all alternative averaging rules we considered generated non-definitive answers.

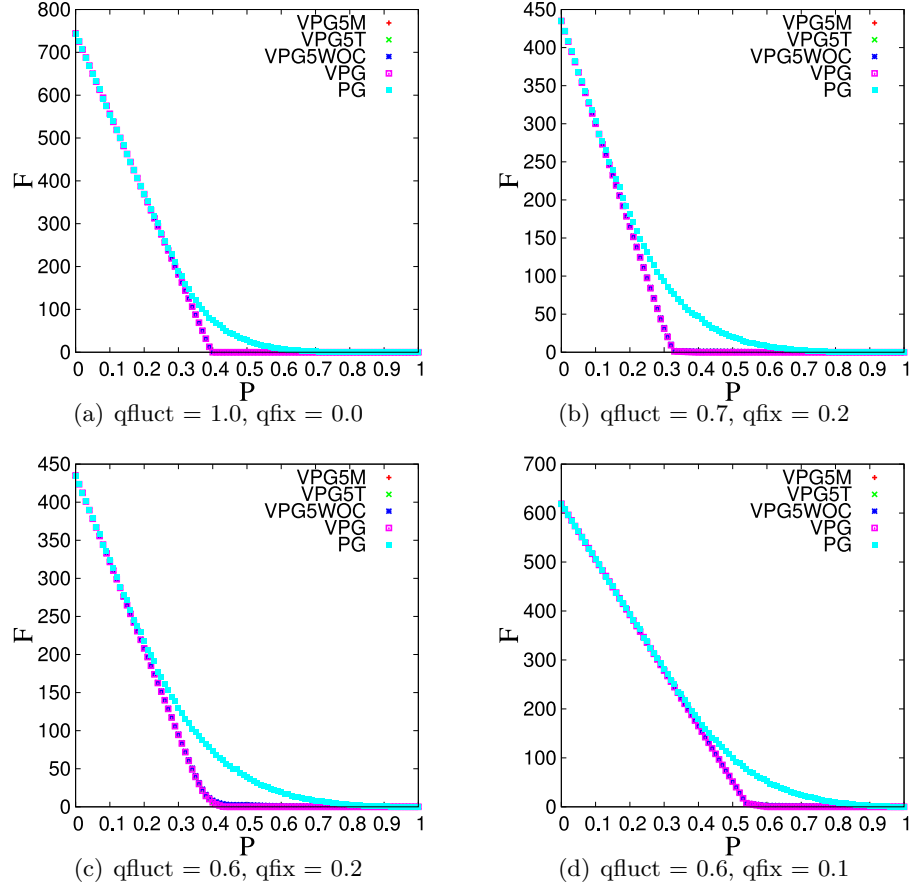


Figure 6.1: The free DOF in the system for $L = 5$ as a function of the probability of the fluctuating edges for algorithms *VPG5M*, *VPG5T*, *VPG5WOC*, *VPG* and *PG*. In general, we can see that the performances of the *VPG-flavor* algorithms are similar. The version 5WOC stands for the "WithOut Condensation" version.

6.3 Future work

Some of the chapters in this dissertation include their own future work, where very specific follow up projects are proposed. In a more general way, in this section we envision several interesting projects that can be derived from this Ph.D dissertation.

1. A comprehensive analysis of the VPG on a very well studied set of proteins. A question that motivates this project is: to what extent does the rigidity assessment made by VPG correlate with the mechanics in the protein? It would be very interesting to find out the biological basis of the rigid clusters and flexible regions in proteins as identified by VPG. For this purpose we propose the analysis of a very well studied set of proteins to shed light on the biological importance of the findings of VPG.

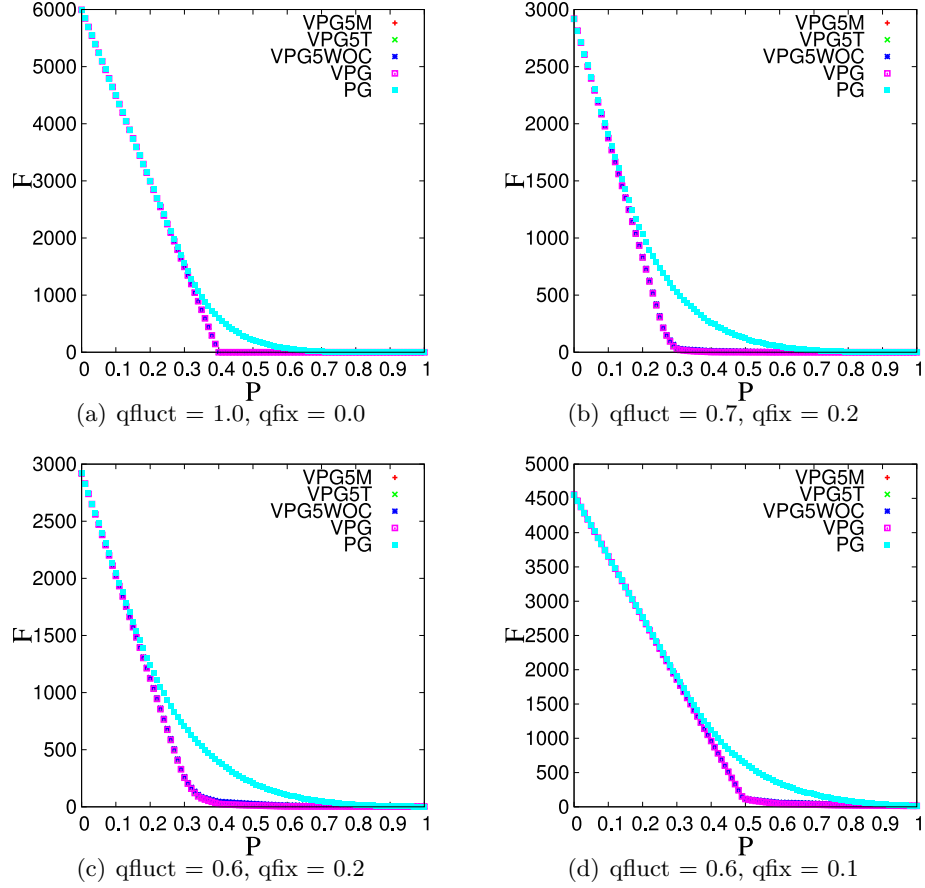


Figure 6.2: The free DOF in the system for $L = 10$ as a function of the probability of the fluctuating edges for algorithms *VPG5M*, *VPG5T*, *VPG5WOC*, *VPG* and *PG*. In general, we can see that the performances of the *VPG-flavor* algorithms is similar.

2. Calculation of the probabilities of H-bonds based on molecular dynamics. There are many ways in which we can derive the probabilities of existence for fluctuating interactions within a protein. We propose to consider the employment of statistical sampling through Molecular Dynamics experiments. Starting from an input structure we can generate statistical meaningful probabilities for the presence of hydrogen bonds within a given conformation. Then, we can average the hydrogen bond presence in the structure via several realizations and obtain their probabilities. This project will enrich the characterization of the VPG and it will help evaluate our current energy-derivation method (based on Boltzmann weights).
3. A mathematical analysis of the VPG algorithm. Mathematically speaking, in this dissertation we introduced a new approach into the field of rigidity theory [62]. Recent

studies have been carried out to set the theory of $(k - l)$ -sparse graphs [18, 19], which helped prove the correctness of the PG algorithm. The VPG algorithm although intuitively similar to PG must be mathematically analyzed to understand its theoretical basis.

6.4 Products of the dissertation

The caliber or impact of a research project can be measured by published articles. From this Ph.D. dissertation we are planning to have several articles published. According to the organization of this document, we submitted for publication chapters two, three, four and currently working on the fifth.

1. Luis C. González, Hui Wang, Dennis R. Livesay and Don J. Jacobs. Improving Protein Flexibility Predictions by Combining Statistical Sampling with a Mean-Field Virtual Pebble Game. ACM Conference on Bioinformatics, Computational Biology and Biomedicine, ACM-BCB 2011. (In press.)
2. Luis C. González, Hui Wang, Dennis R. Livesay and Don J. Jacobs. Calculating Ensemble Averaged Descriptions of Protein Rigidity without Sampling (submitted to journal).
3. Luis C. González, Hui Wang, Dennis R. Livesay and Don J. Jacobs. A virtual pebble game to ensemble average graph rigidity (submitted to journal).

In addition, chapter five of this dissertation is expected to be published as well.

CHAPTER 7: CONCLUSIONS

In this dissertation, we developed a mean field approach called the body-bar Virtual Pebble Game algorithm (VPG) to analyze ensembles of protein networks without the costly process of sampling. In previous models, sampling is required due to the presence of fluctuating interactions that are intermittently present within the network. The VPG accomplishes this goal by representing average properties within the ensemble by fractional degrees of freedom (DOF), and recasting the algorithm as a network flow problem. Compared to the original body-bar Pebble Game algorithm (PG), the accuracy of the VPG is characterized by comparing a large number of quantitative and visual network rigidity properties. Our results indicate that the VPG provides satisfactory mechanical characterizations of the network, especially when not considering the rigidity transition. The four primary conclusions of this dissertation are:

1. Across both square-lattice and protein networks, the Virtual Pebble Game algorithm satisfactorily describes ensemble-averaged network rigidity properties by mean-fielding at the level of individual edges, rather than globally across the entire network as Maxwell Constraint Counting does.
2. As expected, the Virtual Pebble Game algorithm presents its worst performance when ensemble fluctuations are maximized (i.e., the rigidity transition). Fortunately, the rigidity transition rarely represents the domain of interest. Rather, the primary region of interest corresponds to the native protein structure where the VPG closely reproduces the $\overline{\text{PG}}$ behavior.
3. As with the original PG algorithm, the Virtual Pebble Game execution time scales linearly with the number of vertices in the network.
4. The VPG- x algorithm successfully combines the advantages of mean field and sampling approaches, thus resulting in improved network rigidity characterizations.

We present a brief overview of the information that supports every conclusion.

The first conclusion is supported at two levels, the calculation of degrees of freedom (DOF) and the identifications of rigid clusters of atoms by the two algorithms. It was shown that VPG offers a very accurate quantification of the number of DOF available throughout the network after having placed all the distance constraints, most of the time this calculation falls within one standard deviation away from the ensemble average PG values ($\overline{\text{PG}}$). The statistical comparison, based on boxplots, of the Mechanical Coupling Maps (MCM) suggest that VPG approximates $\overline{\text{PG}}$ better than the vast majority of the single PG realizations. This point is strengthened given that this comparison was made at the worst Rand measure score. The identification of rigid clusters of atoms by the two algorithms presents very high similarity as quantified by the Rand Measure, in fact the point with the largest disagreement corresponds to a $\text{RM} \approx 0.80$. The rigidity estimation at the dihedral angle level indicates that both algorithms coincide more than 92% of the time, this is a very compelling result of the rigidity agreement of both VPG and PG. All these results combined show that the Virtual Pebble Game algorithm effectively represents an ensemble of protein networks by a single network.

We calculated the number of available DOF throughout different networks constructed on cubic lattices with periodic boundary conditions. We compared the approximation to PG given by VPG and Maxwell Constraint Counting (MCC). The results show that MCC overestimates the minimum number of constraints needed for the network to become rigid. The VPG results are much closer to PG calculations than MCC specially on the rigid side of the rigidity transition. Relative to MCC, the improved accuracy of the VPG occurs because it applies mean field approximation locally at the edge level. To prove this last point, we performed another set of experiments with a variation of the PG algorithm called bar-PG. In the bar-PG, each of the five distance constraints representing a fluctuating edge are placed independent of each other, each with probability p . In this model, the fluctuations, given by the cooperativity nature of the original group of five constraints, are drastically reduced, therefore yielding results that are markedly close to the VPG. Consequently, we conclude that the Virtual Pebble Game algorithm provides an accurate mean-field approximation to the Pebble Game algorithm at the level of individual edges, rather than across the entire

network (Maxwell Constraint Counting).

Regarding the second conclusion, we characterized the network topologies to analyze the performance of VPG on cubic lattices. We firstly identified the maximum deviation of the internal DOF per vertex between the two algorithms. The maximum differences occurs when $q_{\text{fluct}} = 1$, the difference at this point is ~ 0.6 pebbles, which is about 10% error considering there are six DOF per vertex. It is important to remark that the region with maximal fluctuations corresponds to the rigidity transition, but this point rarely represents the domain of interest. Rather, the primary region of interest corresponds to the native protein structure where the VPG closely reproduces the $\overline{\text{PG}}$ behavior. To quantify the heterogeneous character of the network topology, the heterogeneity index h_I was introduced. The h_I is defined as the standard deviation of the coordination number across all vertices throughout the network. The average number of internal DOF estimated by the VPG is less than 5% error when h_I is below a critical value, which is $h_I^c \approx 0.7$ for the 2D square lattice, and $h_I^c \approx 1.1$ for the 3D cubic lattice. Beyond these critical values the VPG errors increase, but are always less than 10%.

For the third conclusion, we demonstrated that the execution time of the VPG is comparable to a single PG run in a fixed network. We show that for a case with uniform values of p , $q_{\text{fix}} = 0$ and $q_{\text{fluct}} = 1$ the execution time is linear with respect the number of vertices in the system. This time benchmarking holds up for all regions of parameter space $\{q_{\text{fix}}, q_{\text{fluct}}, p\}$, except exceedingly close to the rigidity transition where the VPG scales as $O(N^2)$. It is generally found from simulation on networks that model molecular structure that condensation improves the scaling of the VPG from $O(N^2)$ to $O(N)$ above the rigidity transition.

For the last conclusion, we implemented the VPG- x algorithm that effectively combines mean field and sampling. While tuning the parameter x , that controls the amount of fluctuating edges to sample (the rest are mean field treated), we were able to better approach the $\overline{\text{PG}}$ results than just using the VPG algorithm. The results intuitively and systematically transition from VPG-like to $\overline{\text{PG}}$ -like. Contrary to the single calculation by VPG, the use of the VPG- x algorithm is limited by an increase in the number of realizations that is directly proportional to the amount of sampling needed.

REFERENCES

- [1] M. Gerstein, A.M. Lesk, and C. Chotia. Structural mechanism for domain movements in proteins. *Biochemistry*, 33:6739–6749, 1994.
- [2] K. Sugihara. On redundant bracing in plane skeletal structures. *Bull. Electrotech. Lab.*, 44:376–386, 1980.
- [3] H. Imai. On combinatorial structures of line drawings of polyhedra. *Discrete Appl. Math*, 10(1):79–92, 1985.
- [4] H. N. Gabow and H. H. Westermann. Forests, frames and games: Algorithms for matroid sums and applications. In *20th Annual ACM Symposium on the theory of Computing*, pages 407–421, 1988.
- [5] H.N. Gabow and H.H. Westermann. Forests, frames, and games: Algorithms for matroid sums and applications. *Algorithmica*, 7:465–497, 1992.
- [6] B. Hendrickson. Conditions for unique graph realizations. *SIAM J. Comput.*, 21(1):65–84, 1992.
- [7] G. Laman. On graphs and rigidity of plane skeletal structures. *J. Eng. Math.*, 4(4):331–340, 1970.
- [8] S. Bereg. Certifying and constructing minimally rigid graphs in the plane. *SCG '05: Proceedings of the twenty-first annual symposium on Computational Geometry*, pages 73–80, 2005.
- [9] D.J. Jacobs and B. Hendrickson. An algorithm for two-dimensional rigidity percolation: The pebble game. *J. of Comput. Phys.*, 137:346–365, 1997.
- [10] D. J. Jacobs and M. F. Thorpe. Generic rigidity percolation: The pebble game. *Phys. Rev. Lett.*, 75(22):4051–54, 1995.
- [11] D. J. Jacobs. Generic rigidity in three-dimensional bond-bending networks. *J. Phys. A: Math. Gen.*, 31(31):6653–6668, 1998.
- [12] D. J. Jacobs, A. J. Rader, L. A. Kuhn, and M. F. Thorpe. Protein flexibility predictions using graph theory. *Proteins: Struct. Funct. and Genet.*, 44(2)(2):150–165, Aug 2001.
- [13] B. M. Hespenheide, D. J. Jacobs, and M. F. Thorpe. Structural rigidity in the capsid assembly of cowpea chlorotic mottle virus. *J. Phys.: Condens. Matter*, 16:S5055–64, 2004.
- [14] Walter Whiteley. Counting out to the flexibility of molecules. *Phys Biol*, 2(4):S116–S126, Nov 2005.
- [15] W. Whiteley. The union of matroids and the rigidity of frameworks. *SIAM J. Disc. Math.*, 1:237–255, 1988.
- [16] T-S Tay and W. Whiteley. Recent progress in the rigidity of frameworks. *Structural Topology*, 9:31–38, 1984.

- [17] N. White and W. Whiteley. The algebraic geometry of motions of bar and body frameworks. *SIAM J. Alg. Disc. Math.*, 8:1–32, 1987.
- [18] Audrey Lee and Ileana Streinu. Pebble game algorithms and (k, l) -sparse graphs. *DMTCS proc. AE*, pages 181–186, 2005.
- [19] Audrey Lee and Ileana Streinu. Pebble game algorithms and sparse graphs. *Discrete Mathematics*, 308:1425–1437, 2008.
- [20] D. J. Jacobs, L. A. Kuhn, and M. F. Thorpe. Flexible and rigid regions in proteins. In M. F. Thorpe and P. M. Duxbury, editors, *Rigidity theory and applications*, pages 357–384. Plenum Publishing, NY, 1999.
- [21] M. F. Thorpe, M. Chubynsky, B. Hespenheide, S. Menor, D. J. Jacobs, L. A. Kuhn, M. I. Zavodszky, M. Lei, A. J. Rader, and W. Whiteley. *Current topics in physics*, chapter 6, pages 97–112. Imperial College Press, London, 2005.
- [22] D.J. Jacobs, S. Dallakyan, G.G. Wood, and A. Heckathorne. Network rigidity at finite temperature: Relationships between thermodynamic stability, the nonadditivity of entropy, and cooperativity in molecular systems. *Phys. Rev. E*, 68, 061109:1–21, 2003.
- [23] D. R. Livesay, S. Dallakyan, G. G. Wood, and D. J. Jacobs. A flexible approach for understanding protein stability. *FEBS Lett.*, 576(3):468–476, 2004.
- [24] D. J. Jacobs and S. Dallakyan. Elucidating protein thermodynamics from the three-dimensional structure of the native state using network rigidity. *Biophys. J.*, 88(2)(2):903–915, Feb 2005.
- [25] J.C. Maxwell. On the calculation of the equilibrium and stiffness of frames. *Philos. Mag.*, 27:294–299, 1864.
- [26] D.J. Jacobs and M.J. Fairchild. Thermodynamics of a beta-hairpin to coil transition elucidated by constraint theory. *Biopolymer Research Trends Ed: Pablo C. Snchez. Nova Publishers, NY*, 2007.
- [27] O. K. Vorov, D. R. Livesay, and D. J. Jacobs. Helix/coil nucleation: A local response to global demands. *Biophys. J.*, 97(11):3000–3009, 2009.
- [28] O. K. Vorov, D. R. Livesay, and D. J. Jacobs. Nonadditivity in a conformational entropy upon molecular rigidification reveals a universal mechanism affecting folding cooperativity. *Biophys. J.*, 100(4):1129–38, 2011.
- [29] J.G. Oxley. *Matroid Theory*. Oxford University Press, 1992.
- [30] L. R. Ford and D. R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.
- [31] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. MIT Press and McGraw-Hill, 1991.
- [32] T.-S. Tay. Rigidity of multi-graphs. i. linking rigid bodies in n -space. *Journal of Combinatorial Theory, Series B*, 36(1):95–112, 1984.

- [33] A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chotia. Scop: A structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.*, 247(4):536–540, 1995.
- [34] G. D. Smith, R. H. Blessing, S. E. Ealick, J. C. Fontecilla-Camps, H. A. Hauptman, D. Housset, D. A. Langs, and R. Miller. Ab initio structure determination and refinement of a scorpion protein toxin. *Acta Crystallogr D Biol. Crystallogr.*, 53:551–557, 1997.
- [35] Z.Q. Fu, G.C. Du Bois, S.P. Song, I. Kulikovskaya, L. Virgilio, J.L. Rothstein, C.M. Croce, I.T. Weber, and R.W. Harrison. Crystal structure of mtc-1: implications for role of tcl-1 and mtc-1 in t cell malignancies. *Proc. Natl. Acad. Sci. USA*, 95:3413–3418, 1998.
- [36] K.Y. Hwang, K. Baek, H.Y. Kim, and Y. Cho. The crystal structure of flap endonuclease-1 from methanococcus jannaschii. *Nat. Struct. Biol.*, 5:707–713, 1998.
- [37] M. Hong, M. X. Fitzgerald, S. Harper, C. Luo, D. W. Speicher, and R. Marmorstein. Structural basis for dimerization in dna recognition by gal4. *Structure*, 16:1019–1026, 2008.
- [38] A. J. Rader, Brandon M. Hespenheide, Leslie A. Kuhn, and M. F. Thorpe. Protein unfolding: Rigidity lost. *PNAS*, 99(6):35403545, 2002.
- [39] William M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.
- [40] Andrei Y Istomin, M. Michael Gromiha, Oleg K Vorov, Donald J Jacobs, and Dennis R Livesay. New insight into long-range nonadditivity within protein double-mutant cycles. *Proteins*, 70(3)(3):915–924, Feb 2008.
- [41] B. M. Hespenheide, A. J. Rader, M. F. Thorpe, and L. A. Kuhn. Identifying protein folding cores from the evolution of flexible regions during unfolding. *J. Mol. Graphics Modeling*, 21(3):195–207, 2002.
- [42] A. J. Rader, G. Anderson, B. Ising, H. G. Khorana, I. Bahar, and J. Klein-Seetharam. Identification of core amino acids stabilizing rhodopsin. *PNAS*, 101(19):7246–7251, 2004.
- [43] A. J. Rader and I. Bahar. Folding core predictions from network models of proteins. *Polymer*, 45(2):659–668, 2004.
- [44] H. Tan and A. J. Rader. Identification of putative, stable binding regions through flexibility analysis of hiv-1 gp120. *Proteins*, 74(4):881–894, 2008.
- [45] S. A. Wells, J. E. Jimenez-Roldan, and R. A. Romer. Comparative analysis of rigidity across protein families. *Phys Biol*, 6(4), 2009.
- [46] S. Radestock and H. Gohlke. Protein rigidity and thermophilic adaptation. *Proteins*, 79(4):1089–1108, 2011.
- [47] S. Fulle and H. Gohlke. Constraint counting on rna structures: linking flexibility and function. *Methods*, 49(2):181–188, 2009.

- [48] N. Fox and I Streinu. Redundant interactions in protein rigid cluster analysis. In *1st IEEE International Conference on Computational Advances in Bio and medical Sciences (ICCABS)*, 2011.
- [49] S. Djordjevic and A. M. Stock. Crystal structure of the chemotaxis receptor methyltransferase chcr suggests a conserved structural motif for binding s-adenosylmethionine. *Structure*, 5(4):545–558, 1997.
- [50] B. Ren, G. Tibbelin, D. de Pascale, M. Rossi, S. Bartolucci, and R. Ladenstein. A protein disulfide oxidoreductase from the archaeon *pyrococcus furiosus* contains two thioredoxin fold units. *Nat Struct. Biol.*, 5(7):602–611, 1998.
- [51] Dennis R. Livesay and Donald J. Jacobs. Conserved quantitative stability/flexibility relationships (qsfr) in an orthologous rna h pair. *Proteins*, 62(1)(1):130–143, Jan 2006.
- [52] Donald J. Jacobs, Dennis R. Livesay, Jeremy Hules, and Maria Luisa Tasayco. Elucidating quantitative stability/flexibility relationships within thioredoxin and its fragments using a distance constraint model. *J. Mol. Biol.*, 358(3)(3):882–904, May 2006.
- [53] D. R. Livesay, D. Huynh, S. Dallakyan, and D. J. Jacobs. Hydrogen bond networks determine emergent mechanical and thermodynamic properties across a protein family. *Chem. Central J.*, 2(17), 2008.
- [54] J. M. Mottonen, M. Xu, D. J. Jacobs, and D. R. Livesay. Unifying mechanical and thermodynamic descriptions across the thioredoxin protein family. *Proteins*, 75(3):610–627, 2009.
- [55] T. Nakatsu, H. Kato, and J. Oda. Crystal structure of asparagine synthetase reveals a close evolutionary relationship to class ii aminoacyl-trna synthetase. *Nat. Struct. Biol.*, 5:15–19, 1998.
- [56] D.R. Breiter, M.R. Kanost, M.M. Benning, G. Wesenberg, J.H. Law, M.A. Wells, I. Rayment, and H.M. Holden. Molecular structure of an apolipoprotein determined at 2.5-Å resolution. *Biochemistry*, 30:603–608, 1991.
- [57] Y. Takeuchi, Y. Satow, K. T. Nakamura, and Y. Mitsui. Refined crystal structure of the complex of subtilisin bpn’ and streptomyces subtilisin inhibitor at 1.8 resolution. *J. Mol. Biol.*, 221:309–325, 1991.
- [58] T. N. Petersen, A. Henriksen, and M. Gajhede. Structure of porcine pancreatic spasmolytic polypeptide at 1.95 Å resolution. *Acta Crystallogr. Sect. D*, 52:730–737, 1996.
- [59] Stephen Wells, Scott Manor, Brando Hespenheide, and Michael F. Thorpe. Constrained geometric simulations of diffusive motion in proteins. *Phys. Biol.*, 2:127–136, 2005.
- [60] D. W. Farrell, S. Kirill, and M. F. Thorpe. Generating stereochemically acceptable protein pathways. *Proteins*, 78:2908–2921, 2010.
- [61] Luis C. Gonzalez, Donald J. Jacobs, and Dennis R. Livesay. Personal communication, 2011.
- [62] Henry Crapo. Structural rigidity. *Topologie Structurale*, 1:26–45, 1979.