Novel Algorithms and Benchmarks
for Computational Protein Design

by

Noah Ollikainen

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Biological and Medical Informatics

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA, SAN FRANCISCO
Acknowledgements and Dedication

Computational biology is a highly interdisciplinary field and I have been fortunate to work with outstanding experimental and computational collaborators who have had a significant impact on my graduate research. Here I would like to thank these individuals.

During the beginning of my graduate career, I worked with Prof. Andreas Kuhlmann, Dr. Ellen Sentovich and Dr. Carlos Coelho at Cadence Research Laboratories. I would like to thank Andreas, Ellen and Carlos for imparting on me the optimistic view that for every solution to a problem, there exists another solution that is orders of magnitude more efficient.

I would like to thank all present and past members of the Kortemme lab at UCSF. In particular, I thank Cristina Melero for collaborating with me on multiple projects that showed me the potential of combining experimental and computational approaches. I am extremely grateful to have worked with Prof. Tanja Kortemme, my thesis advisor, who has been tremendously influential in my development as a scientist and I thank Tanja for her guidance and mentorship.

Near the end of my graduate career, I worked with Dr. Rene de Jong, Dr. Jan-Metske Laan-van-der and Dr. Jan Leeuwen-van at DSM Food Specialties in the Netherlands. I would like to thank Rene, Jan-Metske and Jan for showing me the broad, practical applications of my research and helping me develop methods to make these applications possible.

I would like to thank Prof. Patricia Babbitt and Prof. Andrej Sali for their guidance and for serving on my thesis committee as well as Prof. James Fraser for his encouragement.

Most importantly, I would like to thank Dr. Brittany Belin for her love and support, for her enduring friendship, for inspiring me, for motivating me, for countless fascinating conversations, for traveling the world with me, and for helping me become a better person. I dedicate this dissertation to Brittany, whose brilliant mind has shaped how I view the world.
Previously Published Work

A substantial amount of the work presented in this dissertation has previously been published. Below are the citations for this previously published work:


Abstract

Computational protein design aims to predict protein sequences that will fold into a given three-dimensional structure and perform a desired function. Though significant accomplishments in computational protein design have been achieved in the past several years, including the design of novel enzymes and protein–protein interactions, the accuracy of computational protein design is relatively low and many designed sequences must be experimentally tested in order to obtain a successful design. Moreover, successful designs often require directed evolution to achieve catalytic activities or binding affinities similar to naturally occurring proteins. A major challenge in computational protein design that limits its accuracy is the inability to sufficiently sample protein sequence and conformational space at a high resolution. Sampling is difficult due to the combinatorially large number of possible protein sequences and the inherent flexibility of the protein backbone, which may change its conformation upon changes in sequence. To address the issue of sampling a large number of sequences, I developed a deterministic computational protein design algorithm that identifies all sequences within a given energy of the global minimum energy sequence. To identify an accurate method of modeling backbone flexibility, I created a benchmark that evaluates designed sequences based on their similarity to natural sequences with respect to amino acid covariation. Lastly, I developed a novel method of coupling side-chain and backbone sampling. I applied this method to re-designing enzyme substrate specificity and showed a substantial improvement in accuracy over previous computational protein design methods. Taken together, these results demonstrate the importance of modeling protein backbone flexibility and provide new tools to enable higher accuracy computational protein design.
# Table of Contents

List of Tables v
List of Figures viii

Chapter 1: Introduction 1
Chapter 2: SAT-based Protein Design 4
Chapter 3: Flexible Backbone Sampling Methods to Model and Design Protein Alternative Conformations 12
Chapter 4: Computational Protein Design Quantifies Structural Constraints on Amino Acid Covariation 39
Chapter 5: Computational Re-design of Enzyme Substrate Specificity Using Coupled Side-chain and Backbone Moves 77
Chapter 6: Conclusion 85
List of Tables

Table 1: Results of using ProtSAT for designing core positions of 10 PDZ domains. 10
Table 2: Protein domains used for covariation analysis. 44
Table 3: Results of enzyme substrate specificity benchmark. 82
Table 4: Comparison of coupled move protocol with fixed backbone design. 83
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Amino acid side-chain degrees of freedom.</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>Nine representative conformations for Valine.</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>SAT encoding search tree.</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>Runtime and number of conformations searched by ProtSAT.</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>Progress in energy minimization during SAT search.</td>
<td>10</td>
</tr>
<tr>
<td>6</td>
<td>Runtime and number of unique amino acid sequences for Erbin PDZ domain.</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>Cartoon representation of Erbin PDZ domain.</td>
<td>10</td>
</tr>
<tr>
<td>8</td>
<td>Closeup of Erbin design of Figure 7 showing four low energy sequences.</td>
<td>10</td>
</tr>
<tr>
<td>9</td>
<td>Sequence logos for ε-distance sequences for Erbin and PDZ superfamily.</td>
<td>11</td>
</tr>
<tr>
<td>10</td>
<td>The backrub move and its applications in Rosetta.</td>
<td>15</td>
</tr>
<tr>
<td>11</td>
<td>Backrub moves improve prediction of mutant side-chain conformations.</td>
<td>18</td>
</tr>
<tr>
<td>12</td>
<td>Backrub moves improve prediction of alternative side-chain conformations.</td>
<td>21</td>
</tr>
<tr>
<td>13</td>
<td>Backrub moves can model functionally relevant alternative conformations.</td>
<td>26</td>
</tr>
<tr>
<td>14</td>
<td>Rosetta generates near native ensembles using backrub sampling.</td>
<td>30</td>
</tr>
<tr>
<td>15</td>
<td>Rosetta design methods capture evolutionary amino acid covariation.</td>
<td>34</td>
</tr>
<tr>
<td>16</td>
<td>Flow chart of strategy to compare native &amp; designed amino acid covariation.</td>
<td>47</td>
</tr>
<tr>
<td>17</td>
<td>Effects of the magnitude of structural variation on sequence properties.</td>
<td>49</td>
</tr>
<tr>
<td>18</td>
<td>Effects of the mechanism of structural variation on sequence properties.</td>
<td>53</td>
</tr>
<tr>
<td>19</td>
<td>Energetic effects of forcing amino acid covariation onto fixed backbones.</td>
<td>55</td>
</tr>
<tr>
<td>20</td>
<td>Natural and designed covarying amino acid pair preferences are correlated.</td>
<td>57</td>
</tr>
<tr>
<td>21</td>
<td>Covariation mechanisms of natural and designed covarying pairs.</td>
<td>61</td>
</tr>
<tr>
<td>22</td>
<td>Distinguishing features of natural and designed covarying pairs.</td>
<td>64</td>
</tr>
<tr>
<td>23</td>
<td>Flow chart of Monte Carlo simulation for enzyme specificity design.</td>
<td>79</td>
</tr>
<tr>
<td>24</td>
<td>Boltzmann weighted rotamer sampling.</td>
<td>80</td>
</tr>
<tr>
<td>25</td>
<td>Ten enzyme substrate specificity benchmark cases.</td>
<td>81</td>
</tr>
</tbody>
</table>
Chapter 1: Introduction

Proteins have undergone millions of years of evolution to achieve a remarkable diversity of three-dimensional structures and biological functions. Some naturally occurring proteins have been repurposed for a wide range of practical applications, including medicine, food manufacturing, waste management and renewable energy. However, these applications can differ considerably from the protein's native biological function, often resulting in suboptimal performance. Computational protein design seeks to bypass millions of years of evolution and create new proteins that are optimized for any given application. To accomplish this, computational protein design methods predict protein sequences that will fold into a given three-dimensional structure and perform a desired function. These methods consist of two basic components: 1) a search algorithm that will sample protein sequences and 2) an energy function that will score protein sequences. The goal of these methods is typically to find low energy protein sequences for a given protein backbone structure. These sequences can then be tested experimentally to see if they obtain the designed structure and perform the desired function. Unfortunately, despite much progress in this field, state-of-the-art computational protein design methods still have high failure rates, requiring many designs to be tested before a successful design is found. The work in this dissertation aims to improve the accuracy of computational protein design methods through developing novel algorithms and benchmarks.

One major challenge in computational protein design is sampling the combinatorially large number of possible protein sequences for a given protein backbone. To overcome this challenge, I collaborated with researchers at Cadence Research Laboratories in Berkeley, CA during the summer of my first year at UCSF. The goal of this collaboration was to apply
advanced combinatorial optimization algorithms used for integrated circuit design to computational protein design. To do this, we formulated protein design as a Boolean satisfiability (SAT) problem and applied a state-of-the-art SAT-solver to identify the global minimum energy sequence for a set of protein design problems. This algorithm eliminates large areas of sequence space that provably do not contain the lowest energy sequence and is described in detail in Chapter 2. We extended this method to identify all sequences within a given energy of the global minimum energy sequence, which is useful for creating libraries of designs that can be screened experimentally.

Another major challenge in computational protein design is accurately modeling protein backbone flexibility. Unlike amino acid side-chains, whose conformations can be conveniently discretized into rotamer bins, protein backbone motions are much more continuous and therefore difficult to sample efficiently. Consequently, many computational protein design methods assume a fixed backbone and only optimize amino acid side-chains. An underlying hypothesis of my work is that this fixed backbone assumption leads to inaccuracies in design and that incorporating protein backbone flexibility can result in more accurate models and predictions. Chapter 3 highlights a number of experiments that compare fixed backbone and flexible backbone methods in predicting mutant conformations, modeling alternative conformations and predicting sequence tolerance. These experiments demonstrate that flexible backbone can lead to substantial improvements in these applications of computational protein design. Chapter 4 builds on these observations and establishes a large-scale computational benchmark of protein design methods based on comparing computationally designed protein sequences with naturally occurring protein sequences. This benchmark enabled us to evaluate the accuracy of different mechanisms and magnitudes of protein backbone flexibility and conclude that incorporating
small, localized backbone movements into design resulted in the greatest improvements over fixed backbone design.

In order to extend the utility of computational protein design to new applications, I collaborated with researchers at DSM Food Specialties in the Netherlands. These researchers were particularly interested in re-designing enzyme substrate specificity. Their goal was to predict mutations in naturally occurring enzymes that would allow them to catalyze their reaction on novel substrates, thus enabling these enzymes to be used for various applications in the food manufacturing industry. While at DSM, I developed a computational protein design method that predicts mutations that switch the specificity of a given enzyme toward a new substrate. This method incorporates protein backbone flexibility by optimizing side-chain and backbone conformation simultaneously and is described in detail in Chapter 6. To test this method, I created a benchmark of known specificity altering mutations from 10 enzymes and demonstrated that my protocol significantly increased the accuracy of predicting these mutations relative to traditional fixed backbone protein design.

The algorithms and benchmarks described in this dissertation provide significant progress towards increasing the accuracy of computational protein design and lay the foundation for future protein design approaches that incorporate backbone flexibility. Through the comparison of fixed and flexible backbone methods, I have identified many cases where fixed backbone design fails and flexible backbone design succeeds because subtle backbone movements are necessary in order to accommodate a given mutation. Future comparisons like these, which allow us to learn from our failures, will be critical to continuously improve computational protein design to the point where it can achieve high accuracy performance.
Chapter 2: SAT-based Protein Design

Abstract

Computational protein design can be formulated as an optimization problem, where the objective is to identify the sequence of amino acids that minimizes the energy of a given protein structure. In this paper, we propose a novel search-based approach that utilizes a Boolean function to encode the solution space where the function’s onset represents the sequences considered during the search. We first present a dead-end-elimination (DEE) based method for the initial setup of the Boolean function and then describe a branch-and-bound algorithm that employs the search and deduction engine of a modern Boolean Satisfiability (SAT) solver. Its fast implication processing and conflict-based learning provide an efficient framework for the overall algorithm. Our results indicate that the presented approach can efficiently find the guaranteed optimum solution for protein core design problems. Furthermore, since our method is complete and symbolic, it can find all solutions that are within an ε-distance from the global minimum. This capability allows further analysis, such as identifying common sequence patterns of close-to-optimum solutions. Lastly, the SAT-based encoding of the search space provides a flexible mechanism to take complex design constraints into account, such as enforcing dependencies for amino acid choices at different positions or optimizing a single amino acid sequence to be simultaneously consistent with multiple input structures.

1 Introduction

Proteins are organic macromolecules composed of amino acids linked together in a chain and folded into a three-dimensional structure. There are 20 different types of common amino acids, and all amino acids contain amine and carboxyl functional groups, allowing the connection of adjacent amino acids via peptide bonds that form the backbone of a protein. Each type of amino acid contains a unique side-chain group, which varies in physical properties such as size, charge and polarity. The linear sequence of amino acids in a protein generally determines the three-dimensional structure into which the protein folds, and this structure in turn determines the function of the protein. Proteins perform a vast array of functions and participate in all major cellular processes. Types of proteins include enzymes, antibodies, structural components, motors, receptors, signaling factors and transporters. All of these proteins are formed from different sequences of the 20 amino acids.

The goal of protein design is to predict sequences of amino acids that will enable a protein to perform a desired function. This field has a wide range of applications, including engineering protein therapeutics for the treatment of diseases and designing novel enzymes for drug synthesis or the production of biofuels. Consequently, a significant amount of effort has been put into the development of computational protein design methods, and numerous successes have demonstrated the potential of computer-aided protein design [1, 2, 3]. These computational methods commonly start with a desired protein backbone and employ combinatorial optimization algorithms to predict the sequence of amino acids that will be most energetically stable on the given backbone. In general, this involves identifying the sequence of amino acids that minimizes the energy of the protein structure.

To solve this optimization problem, several approximations are commonly made. First, the protein backbone is kept rigid. Second, to approximate the flexibility of amino acid side-chains, the set of all possible amino acid side-chain conformations is discretized into statistically representative conformations called rotamers. A rotamer is the set of three-dimensional atomic coordinates for an amino acid side-chain at a given position on the backbone. The goal now becomes to identify the set of rotamers, one for each position on the backbone, that minimizes an empirically derived energy function. Third, the energy function itself is approximated by including only terms for rotamer-backbone and pairwise rotamer-rotamer interactions. The total energy of the protein can be calculated as follows:

$$E_{total} = E_{backbone} + \sum_i E(r(i)) + \sum_{i,j} E(r(i), r(j))$$

where $E_{backbone}$ is the energy of the backbone, $E(r(i))$ is the interaction energy between rotamer $r(i)$ at position $i$ and the backbone, and $E(r(i), r(j))$ is the interaction energy between rotamers $r(i)$ and $r(j)$ at positions $i$ and $j$, respectively. Even with these approximations, the optimization problem is NP-hard [4].

Identifying the global minimum energy conformation (GMEC) for a given protein design problem is challenging due to the enormous size of the solution space. A protein consisting of 100 amino acids, for instance, could have $2^{1000} \approx 10^{330}$ possible amino acid sequences. If there were on average 10 rotamers per amino acid for this protein, the solution space would contain $2^{1000} \approx 10^{330}$ possible solutions. Since enumeration of all solutions is impossible, search algorithms such as Monte Carlo [5, 6], genetic algorithms [7, 8, 9], self-consistent mean-field [10, 11], dead-end elimination (DEE) [12, 13, 14, 15, 16, 1] and branch-and-bound search [17, 18] have previously been used to find optimal or near-optimal solutions. In general, there is trade-off between speed and accuracy [19], where stochastic approaches, such as Monte Carlo, do not guarantee optimality but run much faster than approaches that guarantee optimality, such as DEE and branch-and-bound search. For practical reasons, it can be more useful to have an ensemble of near-minimum solutions rather than one global minimum solution. Due to inaccuracies in the energy function and the approximations made in formulating the protein design problem, it is possible that the global minimum solution will not yield the desired structure when tested experimentally. Thus, to increase the likelihood of identifying a sequence with the desired structure, the set of solutions within a specified energy distance ($\varepsilon$) from the global minimum could be examined. To efficiently identify the GMEC, as well as these $\varepsilon$-optimal solutions, we have developed a
branch-and-bound algorithm that uses a Boolean function to symbolically encode the search space and applies a Boolean Satisfiability (SAT) solver for performing a fast search within that space.

2 Preliminaries and Previous Work

2.1 Terminology and Notation

Let $\Pi = \{1, 2, \ldots, n\}$ denote the set of $n$ positions of the protein backbone to be optimized and $\Lambda = \{ A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y \}$ be the set of amino acids that can be chosen for each position. Furthermore, let $\chi_1, \chi_2, \chi_3, \chi_4$ be the four dihedral angles of an amino acid that determine its specific 3D conformation. Figure 1 provides a simple example of a protein fragment with four positions. The $\chi$ dihedral angles indicate the degrees of freedom to form different protein conformations for this chain of amino acids. The $\chi$ angles define the degrees of freedom to form different amino acid conformations. For example, amino acid Valine has only a simple side-chain and its conformations can vary only in $\chi_1$.

Let $\Omega : \Pi \rightarrow \Lambda \times \chi_1 \times \chi_2 \times \chi_3 \times \chi_4$ denote a protein conformation that assigns to each position $i$ a unique amino acid $\Lambda(i)$ and unique values $\chi_1(i), \chi_2(i), \chi_3(i), \chi_4(i)$ for its dihedrals. In general, there is an infinite number of possible conformations for a protein of length $n$ since the dihedrals can be given with arbitrary precision. However, in practical protein design, for each amino acid only a representative discrete subset of dihedrals is considered, which dramatically reduces the search space. The representative set is determined by statistical analysis of existing proteins and therefore reflects conformations that occur in nature. Moreover, the dihedral angles are grouped into three sets, gauche$^+ (-120^\circ \leq \chi < 0^\circ)$, gauche$^- (0^\circ \leq \chi < 120^\circ)$, and trans($120^\circ \leq \chi < 240^\circ$). Assuming bond lengths are ideal, the set of dihedral angles for a rotamer determines its 3D coordinates relative to its position on the backbone. Figure 2 shows the nine representative $\chi$ angles for the amino acid Valine and its grouping into the three sets.

Let $\{r_1, r_2, \ldots, r_m\}$ be the set of rotamers with $r_k \in \Pi \times \Lambda \times \chi_1 \times \chi_2 \times \chi_3 \times \chi_4$ denoting the assignment of an amino acid and representative dihedrals $\chi$ to a specific position. The nine representative conformations for Valine shown in Figure 2 correspond to nine possible rotamers. Let $r(i)$ denote the chosen rotamer at position $i$. In protein design, the total energy of a protein conformation is determined by an empirically derived energy function. This function uses the 3D atomic coordinates of the rotamers and the protein backbone to compute the sum of all atom-atom interactions.

\[
E(r) = E(r_v) + \sum_{j \neq i} \left[ \min_{r_n \in \Psi^j} E(r_n, r_v) - \max_{r_n \in \Psi^j} E(r_n, r_v) \right] > 0
\]

where the min sum refers to the energy of the best conformation of $r_v$ with its neighbors and the max sum refers to the worst conformation of $r_v$ with its neighbors. This criterion states that $r_v$ can be eliminated if the energy of its best conformation is greater than the energy of the worst conformation of an alternative rotamer $r_v$ at the same position. With $n$ positions in the protein, and assuming $p$ is the average number of rotamers per position, the complexity of this computation is $O(n^2 p^3)$. This criterion is conservative as it does not identify all instances where one rotamer is always energetically more favorable than another—however, it is inexpensive to compute. A more powerful criterion was introduced by Goldstein [13]:

\[
E(r_v) - E(r_v) + \sum_{j \neq i} \min_{r_n \in \Psi^j} E(r_n, r_v) - \max_{r_n \in \Psi^j} E(r_n, r_v) > 0
\]

With this criterion, $r_v$ is eliminated if the total energy of the protein is always lowered by using $r_v$ instead of $r_v$. The complexity
of evaluating this criterion $O(n^2 p^3)$. Goldstein DEE can also be applied to eliminate pairs of rotamers, but the complexity grows to $O(n^3 p^3)$. As with the original DEE criterion, Goldstein DEE can only eliminate $r_u$ when there exists a single alternative $r_v$ that is more energetically favorable in all possible conformations.

Another DEE criterion, called split DEE [14], overcomes this limitation by splitting conformational space into partitions and determining whether there exists an alternative $r_v$ in each partition that dominates $r_u$. To perform split DEE, a splitting position $k$ is chosen, and conformation space is divided such that each partition uses a different $r_u \in \Psi^k$. The Goldstein criterion is then applied to determine whether $r_u$ is dominated by an alternative $r_v$ within each partition. The criterion is expressed as follows:

$$E(r_u) - E(r_v) + \sum_{j,k: g^j l^k \in \Psi^k} \min_{r_u, r_v} [E(r_u, r_v) - E(r_v, r_s)] + [E(r_u, r_w) - E(r_v, r_w)] > 0$$

(4)

If $r_u$ is dominated in all $r_u$ partitions, then $r_u$ can be eliminated.

If $r_u$ fails to be dominated in all $r_u$ partitions, then for each partition where $r_u$ is dominated, the rotamer pair $(r_u, r_v)$ can be eliminated. This fact makes split DEE especially powerful, as it shares the same complexity as Goldstein DEE but can lead to the elimination of a large number of rotamer pairs. These eliminated rotamer pairs provide valuable restrictions to the solution space.

Upon completion of split DEE, a set of rotamers have been eliminated completely, and pairs of rotamers that are not feasible are saved for later use as logical constraints in our SAT-based approach.

### 2.3 Boolean Satisfiability Solver

Given a set of Boolean variables $x = (x_1, x_2, \ldots, x_n), x_i \in \{0, 1\}$ the onset $ON(f)$ of a Boolean function $f(x)$ is defined as the set of assignments to $x$ which evaluate $f$ to 1, i.e., $ON(f) = \{ x \mid f(x) = 1 \}$. A function is said to be satisfiable (SAT) if $ON(f) \neq \emptyset$.

Determining whether a Boolean function is satisfiable is a classic NP-complete problem [20]. The Conjunctive Normal Form (CNF) of a Boolean function is a function formula with the following structure:

$$\phi = \bigwedge c_i$$

$$c_i = \bigvee l_j$$

where the $c_i$ are referred to as clauses and each literal $l_j$ represents the variable $x_j$ or its negation $\bar{x}_j$. In the past 15 years significant improvements have been made to develop efficient SAT solvers for many practical problem classes [21, 22, 23, 24]. Current solvers are able to routinely solve instances with many thousand variables and millions of clauses.

Most contemporary SAT solvers are based on the Davis-Putnam-Logeman-Loveland (DPLL) procedure [25, 26] which performs a systematic search by interleaving decisions for attempted new variable assignments with Boolean constraint propagation (BCP). BCP determines all necessary variable assignments that can be deduced from the current partial assignment. If a conflicting assignment is encountered, e.g. $x$ is assigned to 1, and BCP determines that for $f$ to be SAT $x$ must be 0, the algorithm backtracks and continues with the next decision. GRASP [21] introduced efficient “learning of no-goods” [27] which was later refined in [22] and other follow-up works. Learning provided a critical component leading to the power of today’s SAT solvers. It analyzes each conflict encountered and derives a conflict clause which prevents the solver from visiting parts of the solution space which are isomorphic with respect to the conflict. Algorithm 1 provides the general DPLL algorithm with learning as implemented by Chaff [22].

### Algorithm 1 TestSat

1. while (MakeDecision () ≠ COMPLETE)
2. while (BCP () = CONFLICT) // find stable assignment
3. if (AnalyzeConflictAndBacktrack () = CONFLICT)
4. return UNSAT
5. return SAT

MakeDecision() heuristically picks the next unassigned variable and assigns a value to it. It returns COMPLETE if all variables are already assigned and therefore a satisfiable solution is found. BCP() determines all implied assignments by finding unit-clauses (i.e., unsatisfied clauses with only one unassigned literal) and processing them. If there is a conflicting assignment, CONFLICT is returned. In this case, AnalyseConflictAndBacktrack() will trace the implications from the previous decision and add a learned conflict clause to the CNF formula with only one variable whose assignment was caused by this decision. After backtracking, i.e., undoing all assignments on this and possibly previous levels, this clause will trigger the opposite variable assignment during the next BCP() step on line 2. If an empty clause is learned, i.e., a clause with no variables, the problem is UNSAT.

### 2.4 SAT-based Optimization

The SAT search algorithm presented in Section 2.3 can be extended for an optimization problem of the form

$$f(x) = 1$$

$$g(x) \rightarrow \min$$

where the variables $x$ provide a Boolean encoding of the solution space, $f$ is a Boolean predicate that must hold for all solutions, and $g$ gives an objective function to be minimized.

In principle, a Pseudo-Boolean constraint solver (PB solver) can be applied to handle the cases where $g$ is a linear function of the form $g = \sum a_i \cdot l_i; a_i \in \mathbb{R}; l_i \in \{ x_i, \bar{x}_i \}$. A PB solver extends the SAT search mechanics by also handling constraints of the form $\sum a_i \cdot l_i \leq c; c \in \mathbb{R}$ in BCP() and AnalyseConflictAndBacktrack() [28]. For optimization the solver is repeatedly invoked with gradually tighter bounds for $c$. A PB solver is not well suited for the given application for two reasons. First the objective function is non-linear and would require the introduction of many new variables to linearize it, and second, it has a large number of terms which would make a PB search extremely slow.

A better approach for SAT-based protein optimization is to interleave SAT with a branch-and-bound scheme. Algorithm 2 provides a modified version of the SAT search of Algorithm 1.

### Algorithm 2 OptSat

1. best.score = $\infty$
2. repeat
3. if (MakeDecision () = COMPLETE)
4. best = current
5. GenerateBlockingClauseAndBacktrack ()
6. while (BCP () = CONFLICT or Estimate () ≥ best.score)
7. if (AnalyzeConflictAndBacktrack () = CONFLICT)
8. return best
The variable best holds the best solution found thus far and its score is initialized to $\infty$. On line 6 the score of the current best solution "best" is compared with a lower-bound estimate for all solutions with the current partial assignment determined by Estimate(). If this comparison shows that no solution with these assignments can beat the best found thus far, the search branch is terminated. In this case or in case of a conflicting variable assignment found in BCP(), AnalyseConflictAndBacktrack() generates a conflict clause and drives the search forward. Everytime the search generates a complete assignment (indicated by MakeDecision() = COMPLETE on line 3) it reflects that a new best solution is found. This solution is then recorded and a blocking clause with the corresponding complete assignment is generated in GenerateBlockingAndBacktrack(). For example, if the assignment \{$x_1 = 1, x_2 = 0, x_3 = 1$\} is found as the current best solution, the blocking clause \((\neg x_1 \lor x_2 \lor \neg x_3)\) is added to \(\phi\). The following backtrack and BCP() step will continue the search until AnalyseConflictAndBacktrack() generates an empty clause indicating that the entire solution space has been covered.

There are multiple advantages of utilizing a SAT-based approach for protein design. First, the use of a binary encoding for the selection of rotamers and CNF clauses for excluding invalid rotamer combinations (including conformations that have been pruned by DEE), offers a compact format to represent the solution space. Second, the BCP process provides a fast mechanism to process necessary rotamer assignments that are implied by decisions during the search. Third, the learning mechanism helps to compactly collect inferior parts of the solution space and prevents the search from revisiting them. Moreover, the constraint-based nature of the SAT encoding allows the addition of complex design requirements. For example, the context of the protein design frequently imposes complex restrictions on the choices of amino acids and dihedrals; such restrictions can easily be encoded as CNF clauses and taken into account during search.

3 SAT-based Protein Design

3.1 Problem Formulation

3.1.1 Basic SAT Encoding

Recall, each rotamer \(r_k \in \{r_1, \ldots, r_m\}\) corresponds to an assignment of an amino acid and its dihedrals to a particular position. For the basic SAT encoding of our protein optimization approach, we simply use Boolean variables \(r_k\) to reflect the inclusion of the rotamer in the protein design. The assignment \(r_k = \text{true}\) indicates that rotamer \(r_k\) has been selected for its position, whereas \(r_k = \text{false}\) means it has not been selected. The initial setup of formula \(\phi\) includes two sets of constraints. First, we need to ensure that exactly one rotamer is selected for each position. This can be achieved by enforcing one-hot encoding of the variables of each position \(i\). Let \(\Psi_i \subseteq \Psi^i\) denote the subset of rotamers at position \(i\) that have not been pruned by DEE, then the following set of clauses ensures that the only solutions considered during search have exactly one rotamer selected for each position:

\[
(\bigvee_{r_k \in \Psi^i} r_k) \land \forall_{r_n, r_r \in \Psi^i} (\neg r_n \lor \neg r_r)
\]  
(5)

Second, we can reduce the search space by adding additional constraints that correspond to pairs of rotamers that have been eliminated during DEE. Let \((r_u, r_v)\) be a pair of rotamers that has been eliminated by DEE, then the clause

\[
(\neg r_u \lor \neg r_v)
\]
ensures that no solution is considered that includes both rotamers. Similarly, if we detected implications during DEE, e.g. the presence of rotamer \(r_u\) requires the presence of another rotamer \(r_v\), then the clause

\[
(\neg r_u \lor r_v)
\]
ensures this for the solution space.

Using the SAT-based search algorithm described in Section 2.4, this encoding performs a flat branch-and-bound search by making decisions for the rotamer assignments at each position. The BCP process facilitates quick implications in the search based on the pruned rotamers and previously visited parts of the search space.

3.1.2 Advanced SAT Encoding

We found that a more advanced encoding which supports hierarchical decision making performs significantly faster and scales better for larger protein design problems. For this, we employ a scheme that first decides which amino acid is assigned to a position followed by the decision of angle ranges for the four dihedrals (if present) and finally the selection of the actual rotamer.

For this, in addition to the Boolean variables \(r_1, \ldots, r_m\) we use the variables \(A', C', D', \ldots, Y'\) to one-hot encode the choice of amino acid at position \(i\), e.g., \(A' = \text{true}\) indicates that amino acid \(A\) is selected for position \(i\). The Boolean variables \(\chi^i_j, \chi^i_{\bar{j}}\) with \(j \in \{1, 2, 3, 4\}\) one-hot encode the choice of the dihedrals of the four levels \(j\) such that \(\chi^i_j = \text{true}\) indicated that the dihedral range at level \(j\) of position \(i\) is equal to gauche\(^+\); \(\chi^i_{\bar{j}} = \text{true}\) indicated that this range is gauche\(^-\), and \(\chi^i_{\bar{j}} = \text{true}\) sets it to trans.

In order to enforce consistency for this more advanced encoding, additional clauses are added to \(\phi\). First, for each position \(i\) we need to enforce the one-hot encoding for each of the five sets: \(\{A', C', \ldots, Y'\}\), \(\{\chi^i_1, \chi^i_\bar{1}\}\), \(\{\chi^i_2, \chi^i_\bar{2}\}\), \(\{\chi^i_3, \chi^i_\bar{3}\}\), and \(\{\chi^i_4, \chi^i_\bar{4}\}\), by using the same scheme as shown in formula (5). For example, for the first dihedral of position \(5\) this encoding is:

\[
(\chi^5_1 \lor \chi^5_\bar{1}) (\neg \chi^5_1 \lor \neg \chi^5_\bar{1}) (\neg \chi^5_1 \lor \neg \chi^5_\bar{1}) (\neg \chi^5_1 \lor \neg \chi^5_\bar{1})
\]

For each set, a first clause enforces that at least one variable is assigned to true, and a set of clauses formed by pairs of literals ensures that no two variables are simultaneously assigned to true.

The second set of constraints ensures consistency of the dihedrals settings with respect to the amino acids. As shown for Valine in Figure 2, not all amino acids have all four dihedrals. In these cases, we simply assume a default assignment \(\chi^i_j = \text{true}\) by adding a corresponding implication to \(\phi\). For example, the additional clauses for the Valine amino acid at position \(7\) of Figure 1 are:

\[
(\neg V^7 \lor \chi^7_1) (\neg V^7 \lor \chi^7_2) (\neg V^7 \lor \chi^7_2)
\]

leaving only \(\chi_1\) open for choice.

The third set of constraints ensures that the selections for the amino acids and \(\chi\) dihedrals are consistent with the rotamers. For this we add clauses such that the selection of a rotamer implies its corresponding amino acid and dihedrals. For example, suppose the rotamer shown for the amino acid Glutamate at position 10 of Figure 1 is denoted by \(r_9\) and corresponds to dihedrals \(\chi_1 = \text{gauche}^+, \chi_2 = \text{gauche}^-, \chi_3 = \text{trans}\). Then consistency is enforced by the following constraints:

\[
(\neg r_9 \lor \chi_1^{10}) (\neg r_9 \lor \chi_2^{10}) (\neg r_9 \lor \chi_3^{10}) (\neg r_9 \lor \chi_4^{10}) (\neg r_9 \lor \chi_5^{10})
\]
3.2 Implementation of Dead-End Elimination

Prior to the Boolean encoding, we use split DEE to significantly reduce the number of possible rotamers and consequently the number of possible solutions. Only rotamers that have not been eliminated by DEE are encoded as Boolean variables. Split DEE eliminates pairs of rotamers in addition to single rotamers, and each pair of eliminated rotamers \( r_k \) and \( r_i \) is encoded as a clause \((\neg r_u \lor \neg r_s)\) and added to \( \Phi \). To reduce the number of these clauses, we only add one such clause when an eliminated pair is shared between all rotamers of the same amino acid or \( \chi \) group. For instance, if all \( r_k \in V^i \) have an eliminated pair with \( r_s \), then we add the single clause \( (\neg V^i \lor \neg r_s) \) to \( \Phi \) instead of a clause for each \( r_k \in V^i \).

We use a simple ordering heuristic to reduce the time spent on DEE. This heuristic takes advantage of the observation that some rotamers are more successful at eliminating than others [16]. For each rotamer \( r_u \) at position \( i \), we calculate the following:

\[
E_{\text{min}}(r_u) = E(r_u) + \sum_{j \in \Pi^i, j \neq i} \min_{r_s \in \Psi^i} E(r_u, r_s)
\]

\[
E_{\text{max}}(r_u) = E(r_u) + \sum_{j \in \Pi^i, j \neq i} \max_{r_s \in \Psi^i} E(r_u, r_s)
\]

where \( E_{\text{min}} \) and \( E_{\text{max}} \) are the energies of best and worst possible interactions involving \( r_u \), respectively. In general, rotamers with low \( E_{\text{max}} \) are more successful at eliminating and rotamers with high \( E_{\text{min}} \) are more likely to be eliminated. Accordingly, we sort rotamers by their \( E_{\text{min}} \) in descending order, and apply the split DEE criterion to each rotamer using a list of alternative rotamers sorted by their \( E_{\text{max}} \) in ascending order. This is repeated until no further eliminations are made. At this point, we construct the encoding described in 3.1.2 and begin the branch-and-bound search.

3.3 Implementation of Branch-and-Bound Search

Our branch-and-bound search approach is outlined in Algorithm 2. This method searches the solution space by making decisions on amino acids, \( \chi \) angle and rotamer Boolean variable assignments. We refer to the assignment of all variables as a complete assignment and the assignment of a subset of variables as a partial assignment. Branch-and-bound prunes the solution space by evaluating lower energy bounds for partial assignments and generating conflict clauses for partial assignments whose lower energy bound is greater than \( E_{\text{best}} \), the energy of the best complete assignment that has been encountered during the search. These conflict clauses invalidate the partial assignment, and the algorithm backtracks to the previous valid partial assignment. When a complete assignment is made, it necessarily has an energy that is equal to or lower than \( E_{\text{best}} \) and \( E_{\text{best}} \) is assigned to this energy. A conflict clause is then triggered that invalidates the complete assignment to prevent it from being visited again during the search. Branch-and-bound terminates once all possible assignments have been invalidated.

To compute lower energy bounds, we use a bounding expression described in [18]. This expression requires rotamer energy \( E_{\text{best}} \) terms to be folded into the pairwise interaction energy \( E_{\text{best}} \) terms as follows:

\[
E'(r_u, r_s) = \frac{E(r_u) + E(r_s)}{2(n-1)} + \frac{E(r_u, r_s)}{2}
\]

where \( n \) is the number of positions. The total energy of the protein can now be calculated by summing all pairwise \( E'(r_u, r_s) \) energies:

\[
E_{\text{total}} = E_{\text{backbone}} + \sum_{i} \sum_{j \neq i} E'(r_u, r_s)
\]

This formula is used to calculate the energy for complete assignments. For partial assignments, we define the set of constrained positions \( \Pi^i \) to be those with a rotamer assigned to true and the set of unconstrained positions \( \Pi^i \) to be those without a positive rotamer assignment. The lower energy bound for a partial assignment can be calculated as follows:

\[
E_{\text{bound}} = \sum_{i \in \Pi^i} \sum_{j \not\in \Pi^i} E'(r_u, r_s) + \sum_{i \in \Pi^i} \min_{r_s \in \Psi^i} \left[ 2 \sum_{j \in \Pi^i} E'(r_u, r_s) + \sum_{j \not\in \Pi^i} \min_{r_s \in \Psi^i} E'(r_u, r_s) \right]
\]

This bounding expression consists of several components. The first term is the sum of the interaction energies for each rotamer pair of the constrained positions. Next, for each rotamer \( r_u \) of an unconstrained position \( i \in \Pi^i \), an energy is calculated that represents the minimum energetic contribution of \( r_u \) to the constrained rotamers. This energy consists of the interaction energy of \( r_u \) with each constrained rotamer and the sum of the best possible interactions between \( r_u \) and the other rotamers at unconstrained positions. Finally, the summation of the lowest minimum energetic contributions is taken over the unconstrained positions and added to the first term to obtain the lower energy bound of the partial assignment.

The assignment order for the amino acids, \( \chi \)'s and rotamers is chosen by a decision heuristic. Our heuristic ranks the decisions at each unconstrained position based on minimum energetic contributions computed during the previous calculation of the lower energy bound. Let \( \text{rank}(d) \) denote the rank of the decision \( d \).
Amino acid, \( \chi \) and rotamer decisions are ranked as follows:

\[
\text{rank}(r_k) = 2 \sum_{j \in \Pi_1} E'(r_k, r_s) + \sum_{r_i \in \Psi} \text{min} E'(r_k, r_s)
\]

\[
\text{rank}(\chi'_j) = \text{min} \text{rank}(r_k) \quad r_k \in \chi'_j
\]

\[
\text{rank}(N^j) = \text{min} \text{rank}(r_k) \quad r_k \in N^j
\]

where a lower rank corresponds to a better decision. Our approach chooses to make decisions with the minimum rank at the position with the maximum difference between its two lowest ranked decisions. For a given position, amino acid decisions always precede \( \chi \) decisions and \( \chi \) decisions always precede rotamer decisions. This enforces the hierarchical search structure illustrated in Figure 3.

3.4 \( \varepsilon \)-Distance Solutions

The first step in obtaining all solutions within \( \varepsilon \) energy of the global minimum is to identify the global minimum energy via DEE and our SAT-based branch-and-bound algorithm. Upon termination, we restart the algorithm using a modified DEE criterion and bounding expression that prevent the elimination of \( \varepsilon \)-minimal solutions. The \( \varepsilon \)-DEE criterion simply compares the left side of Equation 4 to \( \varepsilon \) instead of 0. Consequently, a rotamer can only be eliminated if there exist alternative rotamers that lower the total energy by at least \( \varepsilon \) when used in place of \( r_k \) for all possible conformations.

Next, we calculate the lower energy bound for each rotamer and pair of rotamers and compare it with \( E_{\text{best}} \), the global minimum energy. A rotamer is eliminated if the following condition is satisfied:

\[
E_{\text{bound}} > E_{\text{best}} + \varepsilon
\]

where \( E_{\text{bound}} \) is the lower energy bound of the partial assignment \( r_k = \text{true} \). We also use this condition to eliminate rotamer pairs whose lower energy bound exceeds \( E_{\text{best}} + \varepsilon \). To prevent \( \varepsilon \)-distance solutions from being pruned during the branch-and-bound search, we modify the comparison of the lower energy bound for partial assignment with \( E_{\text{best}} \) such that it contains \( \varepsilon \) as shown in Equation 9.

These modifications allow us to identify all possible conformations within \( \varepsilon \) energy of the global minimum. However, we are interested in all unique sequences within \( \varepsilon \) energy of the minimum. To identify these sequences and prevent the enumeration of a prohibitively large number of conformations, we modified our branch-and-bound search to generate a conflict clause that invalidates the current set of amino acid assignments upon finding an \( \varepsilon \)-distance solution. This conflict clause prevents the search engine from visiting any additional conformations for the current set of amino acids. The algorithm then backtracks to the previous valid set of amino acid assignments and continues the search until all \( \varepsilon \)-optimal sequences have been found.

4 Computational Results

We developed a protein design tool ProtSAT based on the presented approach. For this, we implemented the DEE algorithm from scratch and built the branch-and-bound search on top of Minisat version 1.14 [24]. We applied our method to re-design the cores of 10 PDZ domains. We chose this application because PDZ domains are common protein structures found in many signaling proteins and hence there is a lot of information on naturally occurring sequences to which we can compare our design results. The backbone of the PDZ domain of Erbin is shown in Figure 7(a) in a cartoon representation that illustrates structural elements such as \( \alpha \)-helices and \( \beta \)-strands as helices and arrows, respectively. Each PDZ domain has a similarly structured backbone, with variations due to differences in amino acid sequence.

We constructed 20 benchmark cases to design 10 and 17 core positions for each of the 10 PDZ domains. Designing 10 to 20 positions is common in typical protein core or protein interface design applications. The 17 core positions of Erbin are shown in pink in Figure 7(b). The energy tables containing \( E_{\text{backbone}} \), \( E(r_k) \) and \( E(r_s, r_t) \) terms were generated with an energy function dominated by a van-der-Waals potential, an orientation-dependent hydrogen bonding term [30] and an implicit pairwise-additive solvation potential [31]. Rotamers were generated using the Dunbrack backbone-dependent rotamer library [32] as previously described in [6]. Since the core of a PDZ domain is a hydrophobic environment, we restricted the set of amino acids for each position to the hydrophobic amino acids \{A, C, F, I, L, V, W, Y\}. This is a common restriction made in previous protein design efforts [14, 15].

The performance of ProtSAT on each benchmark is shown in Table 1. Our approach found the GMEC in less than 2 seconds for each 10 position case and less than 1 minute for each 17 position case. The total run-time for 8 separate runs with 10 to 17 core positions of Erbin is shown in Figure 4. The progress of the branch-and-bound search for the 17 position Erbin case is plotted in Figure 5 and the cartoon representation of the GMEC is illustrated in Figure 7(b).

After identifying the GMEC for 17 core positions of Erbin, we applied our \( \varepsilon \)-distance approach to obtain all sequences of amino acids for these 17 positions that have a conformation within an \( \varepsilon \) of the global minimum. We used values for \( \varepsilon \) ranging from 0 to 3 and plotted the number of unique amino acid sequences found for each epsilon in Figure 6. To illustrate how a set of different amino acid sequences can result in similar energies, we show four \( \varepsilon \)-distance sequences for three positions in Figure 8.

Finally, we compared our \( \varepsilon \)-distance sequences with those found in nature. PDZ domain sequences were obtained from the Pfam superfamily alignment [33]. Redundant sequences and sequences with \(< 30\%\) sequence identity to Erbin were removed, resulting in 188 unique PDZ domain sequences. We constructed sequence logos [34] of 2808 core sequences found using \( \varepsilon = 3 \) and of the cores for the 188 PDZ domain sequences, shown in Figure 9. The overall height of each column displays the sequence conservation of that position and the height of each individual letter indicates the frequency of the corresponding amino acid at that position. For 14 out of the 17 positions, the \( \varepsilon \)-distance sequences share the same most frequent amino acid with the natural PDZ domains.

Figure 4: Runtime and number of conformations searched by ProtSAT for protein Erbin for varying number of designed positions.
and at several positions the second or third most frequent amino acid is also shared. The similarity between these two sequence logos suggests that our ε-distance approach can successfully identify an ensemble of realistically low energy sequences.

All calculations were done on an Intel Xeon Processor W3520 (8M Cache, 2.66GHz) running CentOS 5.3 with 4GB RAM.

5 Conclusions

We have described an approach to the computational protein design problem that uses DEE as an efficient pre-processing step, followed by a branch-and-bound algorithm that employs an efficient SAT engine to help prune the search. We demonstrated that this method can successfully identify the GMEC for protein cores as well as all amino acid sequences within ε of the GMEC. This ε-distance capability is a promising approach for a variety of new protein design applications. For example, these sequences can highlight covariation between amino acid types allowed at certain positions that may be crucial for maintaining a defined protein structure. They can also be used as a guide for the design of sequence

<table>
<thead>
<tr>
<th>Protein</th>
<th>PDB ID</th>
<th># Designed Positions</th>
<th># Rotamers Before</th>
<th>Rotation (sec)</th>
<th>Runtime (sec)</th>
<th># Variables</th>
<th># Clauses</th>
<th>SAT</th>
<th># Decisions</th>
<th>Runtime (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Erbin</td>
<td>1MPG</td>
<td>10</td>
<td>1305</td>
<td>200</td>
<td>1.0</td>
<td>370</td>
<td>12402</td>
<td>1789</td>
<td>44</td>
<td>0.02</td>
</tr>
<tr>
<td>CASK</td>
<td>1KWA</td>
<td>10</td>
<td>1387</td>
<td>108</td>
<td>0.8</td>
<td>276</td>
<td>4240</td>
<td>1366</td>
<td>15</td>
<td>0.08</td>
</tr>
<tr>
<td>nNOS</td>
<td>1QAU</td>
<td>10</td>
<td>1304</td>
<td>71</td>
<td>0.8</td>
<td>233</td>
<td>2293</td>
<td>1030</td>
<td>7</td>
<td>0.04</td>
</tr>
<tr>
<td>PSD-95</td>
<td>1BE9</td>
<td>10</td>
<td>1359</td>
<td>153</td>
<td>2.9</td>
<td>319</td>
<td>7734</td>
<td>1651</td>
<td>38</td>
<td>0.01</td>
</tr>
<tr>
<td>InaD</td>
<td>1IHJ</td>
<td>10</td>
<td>1187</td>
<td>80</td>
<td>1.0</td>
<td>234</td>
<td>2020</td>
<td>1033</td>
<td>9</td>
<td>0.03</td>
</tr>
<tr>
<td>GRIP1</td>
<td>1N7F</td>
<td>10</td>
<td>1327</td>
<td>49</td>
<td>0.6</td>
<td>200</td>
<td>891</td>
<td>806</td>
<td>13</td>
<td>0.03</td>
</tr>
<tr>
<td>Par-6</td>
<td>1RZX</td>
<td>10</td>
<td>1064</td>
<td>79</td>
<td>0.4</td>
<td>239</td>
<td>2059</td>
<td>1090</td>
<td>16</td>
<td>0.04</td>
</tr>
<tr>
<td>Tamalin</td>
<td>2EIQN</td>
<td>10</td>
<td>971</td>
<td>54</td>
<td>0.3</td>
<td>204</td>
<td>911</td>
<td>834</td>
<td>15</td>
<td>0.02</td>
</tr>
<tr>
<td>MPDZ</td>
<td>2PNE</td>
<td>10</td>
<td>1451</td>
<td>215</td>
<td>1.5</td>
<td>385</td>
<td>12799</td>
<td>2183</td>
<td>18</td>
<td>0.01</td>
</tr>
<tr>
<td>PCK1</td>
<td>2GZV</td>
<td>10</td>
<td>1390</td>
<td>63</td>
<td>0.6</td>
<td>213</td>
<td>1038</td>
<td>900</td>
<td>9</td>
<td>0.04</td>
</tr>
</tbody>
</table>

Table 1: Results of using ProtSAT for designing 10 and 17 core positions of 10 PDZ domains.

Figure 5: Progress in energy minimization during SAT search for designing 17 core positions of Erbin using ProtSAT.

Figure 6: Runtime and number of unique amino acid sequences found during ε-distance search for designing 17 positions for Erbin.

Figure 7: Cartoon representation of the PDZ domain Erbin: (a) plain backbone with open positions for design, (b) minimum energy conformation computed by ProtSAT. The window indicates the closeup area shown in Figure 8.

Figure 8: Closeup of Erbin design of Figure 7 showing four possible amino acids assignments for positions 17, 19, 37 that are within the ε-distant minimal energy: (a) FIV, (b) FVL, (c) FII, (d) MIV.
libraries that may then be experimentally tested for structural and functional characteristics [35]. As GMEC solutions are not always functional, broadening the number of sequences predicted to be compatible with a given structure or function can facilitate difficult engineering problems. Finally, it may be desirable to optimize protein sequences to be consistent with several different functional requirements, such as binding to different partners or adopting multiple conformations. The latter may broaden applications of recent successes in the engineering of novel biocatalysts by computational protein design [2, 3] by optimizing enzymes to act on sets of alternative substrates or enhancing catalysts by optimizing for the ability to adopt multiple conformations during a reaction cycle.

References


Chapter 3: Flexible Backbone Sampling Methods to Model and Design Protein Alternative Conformations

ABSTRACT

Sampling alternative conformations is key to understanding how proteins work and engineering them for new functions. However, accurately characterizing and modeling protein conformational ensembles remains experimentally and computationally challenging. These challenges must be met before protein conformational heterogeneity can be exploited in protein engineering and design. Here, as a stepping stone, we describe methods to detect alternative conformations in proteins and strategies to model these near-native conformational changes based on backrub-type Monte Carlo moves in Rosetta. We illustrate how Rosetta simulations that apply backrub moves improve modeling of point mutant side chain conformations, native side chain conformational heterogeneity, functional conformational changes, tolerated sequence space, protein interaction specificity, and amino acid co-variation across protein-protein interfaces. We include relevant Rosetta command lines and RosettaScripts to encourage the application of these types of simulations to other systems. Our work highlights that critical scoring and sampling improvements will be necessary to approximate conformational landscapes. Challenges for the future development of these methods include modeling conformational changes that propagate away from designed mutation sites and modulating backbone flexibility to predictively design functionally important conformational heterogeneity.
INTRODUCTION

Proteins are constantly fluctuating between alternative conformations (Frauenfelder et al., 1991). Processes including folding (Korzhev et al., 2010), ligand binding (Boehr et al., 2009), and enzymatic catalytic cycles (Nagel and Klinman, 2009) depend on the movement across the energy landscape. While protein folding is generally driven by a large energy gap between the “native” state and the unfolded ensemble, functionally essential conformations within the “native” state are often separated by smaller energy differences (Fleishman and Baker, 2012).

Computational modeling of the “native” state can result in either a representative single structure or a limited ensemble of conformations. Several straightforward global and local metrics have been developed to compare computational predictions of a representative single structure to an experimentally derived X-ray structure (MacCallum et al., 2011). In contrast, modeling conformational heterogeneity within the “native” state presents significant complications. For example, conformational heterogeneity present in NMR structural ensembles can result from a lack of restraints, limitations in sampling methods, or genuine heterogeneity (Rieping et al., 2005; Schneider et al., 1999). Additionally, comparisons to simulations are often necessary to distinguish between multiple motional models suggested by NMR dynamics observables including Residual Dipolar Couplings (Meiler et al., 2001), CPMG relaxation dispersion (Bouvignies et al., 2011), and side chain order parameters ($S^2$) (Li et al., 1996). X-ray crystallography, which is traditionally interpreted in terms of a single static structure, can also contain information about protein conformational heterogeneity (Best et al., 2006; Furnham et al., 2006; Lang et al., 2010; Levin et al., 2007). All of these experimental data types can be integrated to improve the computational modeling of protein conformational ensembles.
Ultimately, to connect protein conformational dynamics to function, simulations must be leveraged to provide structural mechanisms consistent with the experimental data. Molecular dynamics simulations present the most obvious solution to identify the structural mechanisms of conformational heterogeneity (Maragakis et al., 2008). However, other than in exceptional cases (Kelley et al., 2008; Shaw et al., 2010), the timescales accessible to molecular dynamics often preclude sampling functional motions. The computational requirements of molecular dynamics simulations also make it prohibitive to simultaneously sample sequence space for protein design.

Monte Carlo simulations, for example as used in Rosetta (Leaver-Fay et al., 2011), can also be used to sample protein conformations, but rely on having moves that result in energetically accessible conformations. Fixed backbone Monte Carlo simulations, where side chain conformations are sampled based on a rotamer library, can provide some indications of local flexibility (DuBay and Geissler, 2009). However, it is clear that both side chain and backbone flexibility are necessary to describe and design protein conformational heterogeneity (Friedland et al., 2008; Mandell and Kortemme, 2009). Backbone conformations are less easily discretized compared to side chain rotamers, leading to problems in both creating and validating Monte Carlo backbone moves. Many strategies to efficiently search through backbone space have been implemented in Rosetta including fragment insertion (Simons et al., 1997), loop closure with cyclic coordinate decent (CCD) (Canutescu and Dunbrack, 2003), local torsion sampling with kinematic loop closure (KIC) (Mandell et al., 2009), and backrub (Davis et al., 2006; Smith and Kortemme, 2008). Additionally, these moves can be combined and iterated with sequence design to enrich for proteins with desired conformational and functional properties that could not be explored without backbone flexibility.
Here, we describe the application and validation of the backrub sampling move, which was initially inspired by observations using high resolution X-ray crystallography (Davis et al., 2006), in Rosetta (Smith and Kortemme, 2008) (Figure 10 A-C). To test whether these moves accurately represent protein conformational heterogeneity, we provide example command lines and scripts that can be run using Rosetta version 3.5. These commands and scripts examine how backrub sampling affects predictions of mutant structures, alternative conformations observed by X-ray crystallography, peptide-ligand binding specificities, and evolutionary properties (Figure 10D). The continued development of flexible backbone sampling methods that agree with diverse experimental and evolutionary data will improve our ability to design and engineer new protein functions that depend on and exploit conformational heterogeneity.

Figure 10. The backrub move and its applications in Rosetta. (A) The Richardson group originally described the “Backrub” move as a rotation around the Ca_{i-1} and Ca_{i+1} axis by \( \tau_{\text{disp}} \), along with simultaneous peptide plane rotations (\( t_1 \) and \( t_2 \)), without disturbing other surrounding atom coordinates. (B) By shifting the Ca-Cb bond vector, this move can couple side chain rotameric changes with small local backbone adjustments. (C) In Rosetta, the generalized
backrub move is a single rotation that can also include longer intervals and other backbone atom types as pivots for the rotations. (D) Implementing backrubs as a Monte Carlo move in Rosetta enables a variety of flexible backbone prediction and design applications that are described in this paper: predicting mutant conformations (Figure 11), modeling alternative conformations (Figures 12 and 13), coupling conformational and sequence plasticity (Figure 14), and designing amino-acid co-variation at protein interfaces (Figure 15).

**ROSETTA MOVES TO MODEL ALTERNATIVE CONFORMATIONS IN X-RAY DENSITY**

*Modeling the Richardson Backrub in Rosetta*

Unlike other methods for flexible backbone sampling implemented in Rosetta, which are based on fragment insertion or geometric constraints, the backrub move derives its motional model from conformational variation observed in high-resolution X-ray data (Davis et al., 2006). The Richardson group observed electron density consistent with a concerted backbone reorientation that moves a central side chain perpendicular to the main chain direction for 3% of total residues in a dataset of ultra-high resolution crystal structures. They noted that this move changed the accessible side chain conformations while leaving flanking structure undisturbed.

In Rosetta, the backrub consists of a rotation about an axis defined by the flanking backbone atoms that changes 6 internal backbone degrees of freedom in the protein, namely the $\phi$, $\psi$, and the N-Cα-C bond ($\alpha$) angles at both pivots (Smith and Kortemme, 2008). In the Richardson formulation, the pivots were Cα atoms surrounding a single residue (Figure 10A), but the move can be performed over any backbone atom type over varying length scales (Figure 10C). Bond angle, rotational angle and Cβ/Hα placement constraints are included to eliminate the need for costly minimization steps after ever move. In addition, the backrub move in Rosetta can be adapted so that it obeys detailed balance (Smith and Kortemme, 2008). One notable aspect of the backrub move is that it makes certain side chain conformations, which would not be
accessible in the starting backbone conformation, accessible in the newly accepted backbone conformation (Figure 10B). Such moves alter the potential to accommodate new side chain conformations and mutations at the “backrubbed” position and its local neighbors.

**Modeling the Response to Mutations**

Subtle backbone adjustments are often necessary to accommodate differences between the wild type and mutant side chain. An initial test of the backrub move in Rosetta was to compare its performance to fixed backbone sampling in predicting the conformation of mutated side chains (Figure 11). Based on a template of the wild type structure, a successful prediction of a mutant structure would generate both the conformation of the mutant side chain and any changes that propagate away from the mutated residue. In general, backrub moves decrease the RMSD between the prediction and conformation observed in the mutant crystal structure (Figure 11). Particularly dramatic successes are achieved when there would be a clash to neighboring atoms that is relieved by a small backbone adjustment or when a local backbone move changes the probability of accessing a new conformation from a backbone dependent rotamer library.
Figure 11. Backrub sampling improves the prediction of mutant side chain conformation compared to fixed backbone simulations. (A) Example predictions of mutant conformations given the wild type structure, with the indicated PDB codes and point mutations. The mutant crystal structure is shown in yellow compared to the wild type crystal structure (green, left), prediction based on fixed backbone simulations (magenta, center), or prediction based on backrub flexible backbone sampling (cyan, right). (B) The overall quantification of the results of fixed backbone and backrub predictions over a set of 136 buried (SASA <5%) side chains with conformations differing by more than 0.2 Å between mutant and wild type. The median RMSD decreases from 1.17 Å to 0.98 Å. Shown are box plots with the median as a black line and the 25-75th percentiles in the shaded box with outlier-corrected extreme values as dashed lines. (C) A scatter plot representation of the data in (B) shows that for many mutant structure predictions backrub leads to large improvements compared to fixed backbone simulations.

Due to the broad utility of this application for predicting the results of single or multiple point mutations, we have created a webserver that automates this task:

https://kortemmelab.ucsf.edu/backrub/ (Lauck et al., 2010). On the server, the user must enter the desired PDB, the site of mutation and the new amino acid identity. By default, 10 independent simulations are performed, but this can be adjusted to 2-50 simulations. For each simulation, the server will attempt 10,000 moves that include backrub rotations of various lengths and angles centered on the mutated residue, and side chain moves in a 6Å shell around the mutated residue. The resulting conformations are scored with the Rosetta scoring function and accepted or rejected according to the Metropolis criterion using a kT of 0.6. The lowest scoring conformation
out of all the simulations is returned to the user as the best prediction. Advanced users can exert greater control over these parameters by using the “backrub” Rosetta command line program or RosettaScripts (Fleishman et al., 2011). An example command line for point mutation prediction is as follows:

```bash
~/rosetta/rosetta_source/bin/backrub.linuxgccrelease -database ~/rosetta/rosetta_database/ -s 1CV1.pdb -ex1 -ex2 -extrachi_cutoff 0 -use_input_sc -backrub:ntrials 10000 -nstruct 10 -resfile 1CV1_M111I.resfile -pivot_residues 84 99 102 103 106 107 108 109 110 111 112 113 114 115 118
```

where the “resfile” sets positions to be mutated and repacked (allowing rotamer changes), and the pivot residues denote pivots allowed for backrub moves (necessary files to run the command line with Rosetta version 3.5 are included in the Supplementary Materials as example S1; results shown in Figure 11 were obtained with Rosetta revision 18013). For details about command line flags and the resfile syntax, see the Rosetta manual at [http://www.rosettacommons.org/](http://www.rosettacommons.org/).

**Discovering and Modeling Alternative Conformations from X-ray data**

The backrub move was inspired by manual examination of ultra-high (sub 1Å) resolution electron density maps (Davis et al., 2006) suggesting that conformational heterogeneity in X-ray data can be used to develop and validate new sampling methods. Subsequently, Alber and colleagues developed a method, Ringer (Lang et al., 2010) to automate the discovery of alternative side chain conformations in high (sub 2Å) resolution electron density maps by sampling around side chain dihedral angles (Figure 12A). Despite the limitation that Ringer uses a fixed backbone to define the sampling radius, they showed that 18% of side chains have evidence for unmodeled alternative conformations at electron density levels of 0.3-1s. Concurrently, van den Bedem et al. (van den Bedem et al., 2009) developed a complementary method, qFit, which includes local backbone and side chain flexibility to compute an optimal fit
to the electron density for each residue. The resulting 1-3 backbone and side chain conformations per residue are merged together in a multiconformer qFit model that improves R/Rfree and maintains excellent geometry statistics. Remarkably, despite an entirely different search procedure from the original observation by the Richardson group, many alternative conformations identified by qFit can be related by backrub-like moves (Figure 12B). Collectively, these studies suggest that electron density maps can provide a more informative representation of the “native” state than traditionally offered by static X-ray structures.

By examining 30 pairs of matched room temperature and cryogenic X-ray datasets, Fraser et al. used Ringer and qFit to show that room temperature X-ray data increase the evidence for alternative conformations compared to data collected at conventional cryogenic temperatures (Fraser et al., 2011). We reasoned that sampling between these experimentally visualized alternative conformations would assess the ability of fixed backbone or backrub simulations to access a representative set of conformations that are significantly populated in the “native” state. To test this idea, we considered the A and B alternative conformations (Figure 12C) from 30 room temperature X-ray multiconformer models refined by qFit.
Figure 12. Backrub sampling improves the prediction of alternative side chain conformation observed in protein crystal structures. (A) Electron density sampling by Ringer around the c1 of R29 from PDB 1KWN reveals high electron density for the primary conformation 60° and a secondary peak (indicated by the black arrow), above the 0.3s threshold that enriches for alternative conformations over hydrogen atoms (shaded green area), near the 180° rotameric bin. (B) 2mFo-DFc electron density surrounding R29 from PDB 1KWN contoured at 1s (blue mesh) and 0.3s (cyan mesh). The original PDB model is shown in yellow, with an alternative conformation identified by Ringer and modeled with qFit at 25% occupancy shown in green. (C) Example predictions with Rosetta, with the indicated PDB codes and residues. Sampling of side chain conformations (yellow) starting from alternative conformations (green, right) is improved by flexible backbone backrub moves (cyan, right) compared to fixed backbone side chain only sampling (magenta, center). (D) The overall quantification of the
results, showing that backrub sampling increases identification of discrete side chain local
minima modeled as alternative conformations by qFit compared to fixed backbone models over a
set of 152 side chains with solvent accessibility less than 30%. The median RMSD decreases
from 0.47 to 0.33. Box plots are shown as in Figure 11C. (E) A scatter plot representation of the
data in (D) shows that backrub leads to large improvements compared to fixed backbone for
many alternative conformation predictions.

We focused our analysis on alternative conformations with Cb deviations of 0.2 Å or
greater, relative SASAs less than or equal to 30%, and different c1 rotameric bins (152 side
chains). First, we split the multiconformer model into two separate PDB files, containing all
residues without alternative conformations and either the “A” conformations or “B”
conformations (for example, 1kwn_A.pdb and 1kwn_B.pdb. Next, we ran a RosettaScripts
protocol that moves between the starting conformation specified by the flag –s (in this example,
1kwn_A.pdb) and the target alternative conformation specified by the flag –in:file:native (in this
example, 1kwn_B.pdb). At the beginning of the protocol, the c angles of a central side chain are
switched from the starting conformation to the target conformation. This script tests whether
changes in the surrounding side chains (fixed backbone) or both the backbone and surrounding
side chains (backrub) better accommodate the new c angles and find a side chain conformation
close in RMSD to the target conformation. We tested this protocol in both directions between the
A and B conformations. In the simple example below, the variables c1 through c4 provide the
target χ angles (here the χ angles of conformation B), and piv1 through piv3 define the positions
around the central residue allowed to be pivots for the backrub move.

The command to run the protocol is:

~/rosetta/rosetta_source/bin/rosetta_scripts.linuxgccrelease --database ~/rosetta/rosetta_database
-s 1kwn_A.pdb -in:file:native 1kwn_B.pdb -parser:protocol model_alternate_conformation.xml
-parser:script_vars pos=29 chi1=-145.522 chi2=-160.509 chi3=81.9108 chi4=175.816 piv1=28
piv2=29 piv3=30

The contents of model_alternate_conformation.xml are:

<ROSETTASCAPRTS>
<SCOREFXNS>
  Include the bond angle potential scoring term
  <score12_backrub weights=score12_full>
    <Reweight scoretype=mm_bend weight=1/>
  </score12_backrub>
</SCOREFXNS>

<TASKOPERATIONS>
  Define the restrictions on the sidechain moves that will occur in the simulation
  <ExtraRotamersGeneric name=extra_rot ex1=1 ex2=2 extrachi_cutoff=0/>
  <IncludeCurrent name=input_sc/>
  <DesignAround name=neighbors_only allow_design=0 design_shell=6.0 repack_shell=6.0 resnums=%%pos%%/>
  <RestrictToRepacking name=repack_only/>
  <PreventRepacking name=fix_central_residue resnum=%%pos%%/>
</TASKOPERATIONS>

<FILTERS>
  Calculates the side-chain RMSD before and after simulation
  <SidechainRmsd name=rmsd threshold=10 include_backbone=0 res1_res_num=%%pos%% res2_res_num=%%pos%%/>
</FILTERS>

<MOVERS>
  Set the chi angles of the residue of interest
  <SetChiMover name=setchi1 chinum=1 resnum=%%pos%% angle=%%chi1%%/>
  <SetChiMover name=setchi2 chinum=2 resnum=%%pos%% angle=%%chi2%%/>
  <SetChiMover name=setchi3 chinum=3 resnum=%%pos%% angle=%%chi3%%/>
  <SetChiMover name=setchi4 chinum=4 resnum=%%pos%% angle=%%chi4%%/>

  Set backrub moves to only occur near residue of interest
  <Backrub name=backrub pivot_residues=%%piv1%%,%%piv2%%,%%piv3%% min_atoms=3 min_atoms=7/>

  Set side-chain moves to only include residues within 6 angstrom shell
  <Sidechain name=sidechain task_operations=extra_rot,input_sc,fix_central_residue, neighbors_only,repack_only/>

  During Monte Carlo, alternate between backrub moves (75%) and side-chain moves (25%)
  <ParsedProtocol name=backrub_protocol mode=single_random>
    <Add mover_name=backrub apply_probability=0.75/>
    <Add mover_name=sidechain apply_probability=0.25/>
  </ParsedProtocol>

  Set up Monte Carlo simulation with 10,000 steps and kT=0.6
  <GenericMonteCarlo name=backrub_mc mover_name=backrub_protocol scorefxn_name=score12_backrub trials=10000 temperature=0.6 preapply=0/>
</MOVERS>

<PROTOCOLS>
  Set the residue of interest to the desired chi angles
  <Add mover_name=setchi1/>
  <Add mover_name=setchi2/>
  <Add mover_name=setchi3/>
  <Add mover_name=setchi4/>

  Calculate RMSD before simulation
  <Add filter_name=rmsd/>

  Run backrub simulation
  <Add mover_name=backrub_mc/>

  Calculate RMSD after simulation
Similarly to the mutation data set (Figure 11), backrub moves significantly improve the predictions (Figure 12D,E). Additionally, we tested the effect of including larger backrub moves or using C and N atoms as pivots in place of the normal Cα pivot. Applying these larger moves or using additional pivot atoms did not significantly affect the modeled alternate side-chain RMSD over the dataset. However, there may be certain residue types, secondary structures, or local environments that benefit from distinct move sets.

These results suggest that backrub moves help to model “native” state heterogeneity. While above we have described the validation of this procedure on high resolution room temperature X-ray data, similar strategies can be applied to cryogenic data, where models will likely contain fewer alternative conformations, or to low resolution data, where the electron density maps do not reveal discrete alternative conformations. Therefore, flexible backbone sampling strategies in Rosetta may help to improve the description of the “native” state offered by conventional or low-resolution X-ray crystallography experiments (Tyka et al., 2011). Such sampling methodologies will have many applications including flexible receptor docking in drug discovery (Sherman et al., 2006).

**Sampling Functional Alternative Conformations in Cyclophilin A**

While the preceding examples utilized backbone flexibility centered on a single residue, many protein motions require movement of a neighborhood of residues that may potentially spread across multiple elements of secondary structure. These movements can create loop, rigid body domain, or side chain rearrangements that are crucial for the biological mechanism.
The difficulty of discovering and simulating correlated motions is exemplified in the intrinsic conformational exchange of the proline isomerase cyclophilin A (CypA) (Fraser et al., 2009) (Figure 13A). Previous NMR studies by the Kern group identified a collective exchange process extending from the active site into the core of the protein and established a link between the rate of conformational exchange and the catalytic cycle of the enzyme (Eisenmesser et al., 2002; Eisenmesser et al., 2005). Room temperature X-ray crystallography and electron density interpretation using a combination of Ringer and manual inspection were used to reveal that the exchange was due to a coupled network of alternative side chain conformations (Fraser et al., 2009). The functional importance of the alternative conformation was tested by demonstrating a parallel reduction in dynamics and catalysis upon mutation of a residue outside the active site (Fraser et al., 2009). Intriguingly, the backbone movement of Phe113 renders the alternative conformation undetectable by Ringer, which is limited to fixed backbone sampling (Figure 13B).

Due to the millisecond timescale of this correlated motion, the side chain conformational changes cannot be sampled in conventional molecular dynamics simulations. However, recent accelerated molecular dynamics simulations that reduce torsional barriers have recapitulated several key elements of the conformational dynamics during catalysis (Doshi et al., 2012). As an initial test of the ability of Rosetta to model correlated motions between neighboring side-chains in CypA, we modified the RosettaScripts protocol used to sample alternative conformations. We defined multiple positions that are allowed to be pivots for backrub and included a call to a “resfile” that specifies the positions whose side-chains can be repacked. This protocol improves sampling the alternative conformation over fixed backbone approaches (Figure 13B).
Figure 13. Backrub sampling can be used to model functionally relevant alternative conformations. (A) NMR relaxation experiments detect that residues in a dynamic network (cyan transparent surface) undergo a collective exchange between a major and minor conformation with and without substrate present (Eisenmesser et al., 2002; Eisenmesser et al., 2005; Fraser et al., 2009). Room temperature X-ray data collection and qFit multiconformer refinement identify a major (green) and minor (yellow) conformation providing a structural basis for the NMR observations. Additional alternative conformations are shown in orange. (B) Rosetta simulations can access to the alternative conformation starting from either state (yellow/green) using backrub (right, cyan), but not fixed backbone (middle, magenta) sampling methods.
The command to run the protocol is:

```
~/rosetta/rosetta_source/bin/rosetta_scripts.linuxgccrelease -database ~/rosetta/rosetta_database/
-s 3K0N_A.pdb -in:file:native 3K0N_B.pdb -parser:protocol
model_alternate_conformation_F113.xml -resfile F113.resfile -parser:script_vars chi1=-53.763
chi2=-41.4727 chi3=0 chi4=0
```

The contents of model_alternate_conformation_F113.xml are:

```xml
<ROSETTASCRIPr T S>
  <SCOREFXNS>
    Include the bond angle potential scoring term
    <score12_backrub weights=score12_full>
      <Reweight scoretype=mm_bend weight=1/>
    </score12_backrub>
  </SCOREFXNS>
  <TASKOPERATIONS>
    <ExtraRotamersGeneric name=extra_rot ex1=1 ex2=2 extrachi_cutoff=0/>
    <ReadResfile name=read_resfile filename="F113.resfile"/>
  </TASKOPERATIONS>
  <FILTERS>
    Calculates the side-chain RMSD before and after simulation
    <SidechainRmsd name=rmsd threshold=10 include_backbone=0 res1_pdb_num=113A
    res2_pdb_num=113A/>
  </FILTERS>
  <MOVERS>
    Set the chi angles of the residue of interest
    <SetChiMover name=setchi1 chinum=1 resnum=113A angle=%%chi1%%/>
    <SetChiMover name=setchi2 chinum=2 resnum=113A angle=%%chi2%%/>
    <SetChiMover name=setchi3 chinum=3 resnum=113A angle=%%chi3%%/>
    <SetChiMover name=setchi4 chinum=4 resnum=113A angle=%%chi4%%/>
    Set backrub moves to only occur near residue of interest
    <Backrub name=backrub
    Set side-chain moves to only include residues within 6 angstrom shell
    <Sidechain name=sidechain task_operations=read_resfile,extra_rot/>
    During Monte Carlo, alternate between backrub moves (75%) and side-chain moves (25%)
    <ParsedProtocol name=backrub_protocol mode=single_random>
      <Add mover_name=backrub apply_probability=0.75/>
      <Add mover_name=sidechain apply_probability=0.25/>
    </ParsedProtocol>
    Set up Monte Carlo simulation with 10,000 steps and kT=0.6
    <GenericMonteCarlo name=backrub_mc mover_name=backrub_protocol
    scorefxn_name=score12_backrub trials=10000 temperature=0.6 preapply=0/>
  </MOVERS>
  <PROTOCOLS>
    Set the residue of interest to the desired chi angles
    <Add mover_name=setchi1/>
  </PROTOCOLS>
</ROSETTASCRIPr T S>
```
Here, we have specified neighboring residues that can undergo backrub moves. Similarly, the ability of backrub to sample functionally important loop conformations has been demonstrated for triosephosphate isomerase (TIM) (Smith and Kortemme, 2008). To efficiently sample these enzymatic motions, we used prior knowledge of residues that need conformational adjustments. Therefore, these strategies present an immediate challenge: sampling large correlated motions without prior knowledge of what residues are involved in the motion. One approach is to use unbiased simulations to identify flexible regions, as was retrospectively shown in the TIM study. Another intermediate on the road to this goal is to include constraints from NMR relaxation dispersion experiments, which specify residues that are experiencing an exchange in chemical environment, but do not provide direct structural information about the exchange. Recent work using T4 Lysozyme (Bouvignies et al., 2011) suggests that Rosetta fragment insertion methods biased by experimental chemical shifts can generate structural descriptions of alternative conformations discovered by NMR. The success of backrub moves in sampling the enzyme motions of CypA and TIM indicate that “native” state sampling using backrub moves can likely be exploited in a similar fashion to link conformational dynamics discovered by NMR relaxation dispersion experiments with structural mechanisms.
SEQUENCE PLASTICITY AND CONFORMATIONAL PLASTICITY ARE
INTERTWINED

The improvements offered by backrub moves in predicting point mutant structures (Figure 11) suggest that subtle backbone rearrangements can significantly alter the prediction of tolerated mutations. It follows that conformational ensembles created through backrub moves would also change the potential for sequences predicted to be consistent with a given protein fold. Indeed, incorporating backbone flexibility increased the overlap between sequences predicted to be consistent with the ubiquitin fold and the evolutionary record (Friedland et al., 2009). These results provided further evidence that the relationship between sequence and structural variability can be leveraged to develop and validate new conformational sampling methods. Both sequence alignments of orthologous proteins (natural selection) and sequences enriched in high-throughput binding experiments, such as phage display or peptide arrays (artificial selection), can been used to define the sequence variability that design methods can target.

Modeling Peptide Binding Specificity

In addition to predicting sequences tolerated by single protein fold, flexible backbone methods can improve the prediction of binding specificity in peptide binding domains such as PDZ, SH3 and WW domains. Phage display coupled with next-generation sequencing techniques can generate experimental position weight matrices (PWMs) based on large numbers of potential sequences (Huang and Sidhu, 2011). A challenge for interpreting these datasets is to define the structural basis for specificity in binding pockets that are quite similar. To test how well Rosetta can recapitulate the binding specificities discovered by these experiments, we sampled
conformations of both the peptide and receptor protein using backrub moves. As observed previously for sampling the ubiquitin fold family sequences, the temperature parameter is key for controlling the conformational diversity sampled by the ensemble (Figure 14A). For applications where there is a larger degree of backbone flexibility and corresponding sequence variability, higher temperatures can be explored. For PDZ domain-peptide interactions, a temperature of 0.6 kT was used (Smith and Kortemme, 2010, 2011). After generating an ensemble using backrub moves, design can be used to sample sequence changes of either the receptor or the peptide.

**Figure 14. Rosetta generates near native ensembles using backrub sampling.** (A) Ca cartoons of Rosetta generated conformational ensembles using backrub sampling at different temperatures, compared to the fixed backbone (top left). Greater temperatures increase the conformational diversity and can increase agreement to experimental data. The PDZ domain structure is shown in green and peptide in orange. (B) Sequence tolerance protocol to predict peptide specificity for 4 PDZ domains (DLG1-2, MPDZ-12, MPDZ-13 and Erbin) and 1 PDZ domain point mutant (Erbin V83K); peptide positions are indicated using the standard nomenclature for PDZ domain motifs, with 0 denoting the C-terminal residue, followed by -1, -2, etc. Without backbone flexibility Rosetta fails to predict important residues preferences observed in experimental phage display selections, such as valine at the 0 position or tryptophan at the -1 position for DLG1-2 and Erbin.

We have automated the sequence tolerance protocol for using flexible backbone ensembles and sequence design for predicting peptide binding specificity on a webserver:  
[https://kortemmelab.ucsf.edu/backrub/](https://kortemmelab.ucsf.edu/backrub/) (Lauck et al., 2010). Users can also download a “protocol capture” of the sequence tolerance method, complete with example input/output and scripts, in the Supplementary Materials accompanying (Smith and Kortemme, 2011). Given a peptide-
bound structure, backrub sampling methods are used to generate an ensemble of conformations. For each conformation, a genetic algorithm is used to design sequences for high affinity binding. In this protocol, the interface energy is given greater weight (Smith and Kortemme, 2010, 2011). To compare these predictions to experimental data, we generate a sequence logo based on the positional frequencies in the resulting designed sequences (Figure 14B). Compared to fixed backbone methods, backrub sampling increases the agreement at several positions. Given the adaptable nature of many protein-protein interfaces, it is clear that flexible backbone methods will provide great insight into the structural and energetic basis for binding specificity. Additionally, as more datasets on mutant binding domains are collected, there is potential to look for co-variation between the sequences tolerated between receptor and peptide positions (Ernst et al., 2010).

**Co-variation and Interface Design in Two-component Signaling**

Testing computational protein design methods based on comparison with experimental PWMs is informative. However, it involves evaluating amino acid positions independently from each other and therefore may overlook some of the intricate details of pair-wise interactions between designed residues. In order to assess how well flexible backbone design protocols capture dependencies between designed residues, we directly compared designed residue co-variation to native residue co-variation. We chose to examine co-variation within the bacterial two-component signaling system, since it has previously been shown that sensor histidine kinases (HK) and their cognate response regulators (RR) exhibit significant intermolecular co-variation at their protein-protein interface (White et al., 2007).
Designed sequences were obtained by generating a backrub conformational ensemble of 500 structures starting from the co-crystal structure of HK853 and RR468 from *Thermotoga maritima* (Casino et al., 2009) PDB 3DGE). Since bacterial HK and RR sequences are highly divergent, we used a temperature of 1.2 kT to produce a conformational ensemble that would yield sufficiently diverse designed sequences. We then performed sequence design using Monte Carlo simulated annealing on each structure, which resulted in 500 designed HK and RR sequences.

The command lines for this protocol are as follows:

**Backrub ensemble generation:**

```bash
~/rosetta/rosetta_source/bin/backrub.linuxgccrelease -database ~/rosetta/rosetta_database/ -s 3DGE.pdb -resfile NATAA.res -ex1 -ex2 -extrachi_cutoff 0 -backrub:mc_kt 1.2 -backrub:ntrials 10000 -nstruct 500 -backrub:initial_pack
```

**Sequence design:**

```bash
~/rosetta/rosetta_source/bin/fixbb.linuxgccrelease -database ~/rosetta/rosetta_database/ -s 3DGE_0001_last.pdb -resfile ALLAA.res -ex1 -ex2 -extrachi_cutoff 0 -nstruct 1 -overwrite -linmem_ig 10 -no_his_his_pairE -minimize_sidechains
```

To compare the sequence features from interface design with those observed in naturally interacting proteins, we collected alignments of natural HK and RR sequences from Pfam (PF000512 for HK and PF00072 for RR) and concatenated all pairs of HKs and RRs that were adjacent in a particular genome (i.e., pairs with GI numbers differing by 1). To avoid bias from closely related sequences, we filtered the joint HK/RR alignment for redundancy using an 80% sequence identity cutoff. We quantified residue co-variation of all intermolecular pairs of amino acid positions in designed and natural sequences using a mutual information based statistic (Dickson et al., 2010).

We observed significant overlap between the designed and natural highly co-varying intermolecular pairs within the HK/RR complex (Figure 15A). Mapping the residue pairs that
were highly co-varying in both designed and natural sequences onto the structure of the complex revealed that all of these pairs are localized to the HK/RR interface (Figure 15B). A closer examination of these pairs shows that each pair forms a physical interaction across the HK/RR interface (Figure 15C), suggesting that these pairs may be important for determining specificity in bacterial two-component signaling systems. Indeed, several of these positions have previously been mutated to alter the specificity of HK-RR interactions: HK-Thr in Pair 1, HK-Tyr in Pair 9 and HK-Val in Pair 11 (Skerker et al., 2008). The remaining pairs, including those that highly co-vary in designed sequences but not natural sequences, represent potential opportunities for rewiring two-component signaling specificity using computational protein design.

FUTURE CHALLENGES

The success of flexible backbone sampling methods in predicting mutant side chain (Figure 11) and alternative (Figure 12) conformations indicates the broad utility of these methods in designing sequences compatible with a target “native” structure. Previous studies have used Rosetta to provide structural mechanisms for NMR measures of protein dynamics (Friedland et al., 2009; Friedland et al., 2008) and to design mutations that stabilize specific conformations from a dynamic ensemble (Babor and Kortemme, 2009; Bouvignies et al., 2011). Additionally, the comparisons to naturally and artificially selected sequence data suggest that flexible backbone methods can be leveraged to design libraries for generating proteins with new or improved functions (Friedland and Kortemme, 2010).

Despite these successes, exploiting backbone flexibility to design conformational heterogeneity, in contrast to design of a single target structure, remains largely unaddressed. A major challenge in the coming years will be to adapt these methods to design functionally important protein conformational dynamics. Examples of these design challenges include:
designing loops to sample multiple conformations that exclude water and permit substrate flux during an enzymatic catalytic cycle, creating peptide binding domains where specificity is encoded by distinct binding modes, or generating coupled networks of side chain conformations that respond to an allosteric binding event.

Figure 15. Rosetta backrub design methods capture features of evolutionary amino-acid co-variation. (A) Comparison between designed and natural intermolecular amino acid co-variation for histidine kinases (HK) and their cognate response regulators (RR). Each point represents a
pair of amino acid positions. Natural co-variation was quantified using a mutual information based metric for all pairs of positions in a multiple sequence alignment of HKs concatenated to their cognate RRs. A backrub ensemble of 500 structures was generated for a HK/RR complex (PDB ID 3DGE) and RosettaDesign was used to predict one low energy sequence for each structure in the ensemble. Designed co-variation was quantified for all pairs of positions in the resulting multiple sequence alignment of 500 sequences. The red lines indicate the threshold cut-off for the top 30 designed co-varying intermolecular pairs (horizontal) and the top 30 natural co-varying pairs (vertical). The 12 intermolecular pairs of positions that are highly co-varying in both designed and natural sequences are highlighted in green. (B) The structure of a HK/RR complex with amino acids that are involved in highly co-varying intermolecular pairs in both natural and designed sequences are shown in green and stick representation. (C) Close-up of the 12 intermolecular co-varying pairs. Each of these 12 pairs of amino acids forms a physical interaction across the interface of the complex.

To meet these lofty challenges, scoring functions must be sensitive to the small gaps that separate these conformations on the energy landscape (Fleishman et al., 2011). In addition, to avoid having populations biased by the sampling algorithm and provide better estimates of conformational entropy, the Monte Carlo move sets must obey detailed balance (Hastings, 1970). In addition to improvements in scoring and thermodynamics, more sophisticated sampling protocols will likely be needed. Here, we have primarily focused on backrub moves around Ca. However, sampling the “native” state of some protein environments may benefit from different strategies or iterations through a combination of sampling moves. Indeed, we have recently had success at modeling conformational changes that propagate away from a designed mutation by iteratively switching between different sampling and scoring strategies during the course of a single simulation (Kapp et al., 2012). Learning from the successes and failures of these new strategies will be essential to improve both protein design and our understanding of the relationship between protein conformational dynamics and function.

REFERENCES

Babor, M., Kortemme, T., 2009. Multi-constraint computational design suggests that native sequences of germline antibody H3 loops are nearly optimal for conformational flexibility. Proteins 75, 846-858.


molecular dynamics simulation shows effect of slow loop dynamics on backbone amide
to the dynamic interpretation of residual dipolar couplings in globular proteins. J Am Chem
Soc 123, 6098-6107.
Rieping, W., Habeck, M., Nilges, M., 2005. Inferential structure determination. Science 309,
303-306.
protein NMR structures: derivation of realistic model distance data from a long molecular
Shaw, D.E., Maragakis, P., Lindorff-Larsen, K., Piana, S., Dror, R.O., Eastwood, M.P., Bank,
modeling ligand/receptor induced fit effects. J Med Chem 49, 534-553.
structures from fragments with similar local sequences using simulated annealing and
Skerker, J.M., Perchuk, B.S., Siryaporn, A., Lubin, E.A., Ashenberg, O., Goulian, M., Laub,
M.T., 2008. Rewiring the specificity of two-component signal transduction systems. Cell
133, 1043-1054.
protein conformational variability and improves mutant side-chain prediction. J Mol Biol
380, 742-756.
Smith, C.A., Kortemme, T., 2010. Structure-based prediction of the peptide sequence space
Smith, C.A., Kortemme, T., 2011. Predicting the tolerated sequences for proteins and protein
Tyka, M.D., Keedy, D.A., Andre, I., Dimaio, F., Song, Y., Richardson, D.C., Richardson, J.S.,
J Mol Biol 405, 607-618.
Crystallogr 65, 1107-1117.
Chapter 4: Computational Protein Design Quantifies Structural Constraints on Amino Acid Covariation

ABSTRACT

Amino acid covariation, where the identities of amino acids at different sequence positions are correlated, is a hallmark of naturally occurring proteins. This covariation can arise from multiple factors, including selective pressures for maintaining protein structure, requirements imposed by a specific function, or from phylogenetic sampling bias. Here we employed flexible backbone computational protein design to quantify the extent to which protein structure has constrained amino acid covariation for 40 diverse protein domains. We find significant similarities between the amino acid covariation in alignments of natural protein sequences and sequences optimized for their structures by computational protein design methods. These results indicate that the structural constraints imposed by protein architecture play a dominant role in shaping amino acid covariation and that computational protein design methods can capture these effects. We also find that the similarity between natural and designed covariation is sensitive to the magnitude and mechanism of backbone flexibility used in computational protein design. Our results thus highlight the necessity of including backbone flexibility to correctly model precise details of correlated amino acid changes and give insights into the pressures underlying these correlations.
AUTHOR SUMMARY

Proteins generally fold into specific three-dimensional structures to perform their cellular functions, and the presence of misfolded proteins is often deleterious for cellular and organismal fitness. For these reasons, maintenance of protein structure is thought to be one of the major fitness pressures acting on proteins. Consequently, the sequences of today’s naturally occurring proteins contain signatures reflecting the constraints imposed by protein structure. Here we test the ability of computational protein design methods to recapitulate and explain these signatures. We focus on the physical basis of evolutionary pressures that act on interactions between amino acids in folded proteins, which are critical in determining protein structure and function. Such pressures can be observed from the appearance of amino acid covariation, where the amino acids at certain positions in protein sequences are correlated with each other. We find similar patterns of amino acid covariation in natural sequences and sequences optimized for their structures using computational protein design, demonstrating the importance of structural constraints in protein molecular evolution and providing insights into the structural mechanisms leading to covariation. In addition, these results characterize the ability of computational methods to model the precise details of correlated amino acid changes, which is critical for engineering new proteins with useful functions beyond those seen in nature.

INTRODUCTION

Evolutionary selective pressures on protein structure and function have shaped the sequences of today’s naturally occurring proteins [1-3]. As a result of these pressures, sequences of natural proteins are close to optimal for their structures [4]. Natural protein sequences therefore provide an excellent test for computational protein design methods, where the goal is to
predict protein sequences that are optimal for a desired protein structure and function [5]. It is often assumed that given a natural polypeptide backbone conformation, an accurate protein design algorithm should be able to predict sequences that are similar to the natural protein sequence. This test is commonly referred to as native sequence recovery [4] and it has been used extensively to evaluate various protein design sampling methods and energy functions [6-8].

Beyond simply recovering the native sequence, a further challenge in computational protein design is to predict the set of tolerated sequences that are compatible with a given protein fold and function [9-13]. Predicting sequence tolerance is important for applications such as characterizing mutational robustness [14,15], predicting the specificity of molecular interactions [16-20], and designing libraries of proteins with altered functions [21,22]. Recent methods developed for this goal involve generating an ensemble of backbone structures similar to the native structure and then designing low energy sequences for the different structures in the ensemble [9,16,19,23-25]. These flexible backbone design methods can produce sequences that are highly divergent from the native sequence but may still fold into the desired structure, which makes simple native sequence recovery a poor indicator for the accuracy of these methods. A more useful computational test of these approaches involves comparing designed sequences with a set of reference sequences, either naturally occurring or experimentally derived, that share the desired protein fold. This comparison can be based on sequence profile similarity, which involves quantifying the difference between the frequencies of observing each amino acid at corresponding positions in the designed and reference sequences [16,17,19].

While high similarity between designed and reference sequence profiles can be informative to gauge the accuracy of a protein design method, it does not guarantee that the method will predict sequences that fold into the desired structure. This is because sequence
profile comparisons evaluate amino acid positions independently from each other and therefore ignore the details of amino acid interactions that are critical for protein structure and function. Naturally occurring protein structures are formed cooperatively and each amino acid can physically interact with multiple neighboring amino acids. Evolutionary selective pressures have acted upon these interactions, resulting in the patterns of amino acid covariation that can be observed within today’s naturally occurring protein families. Accordingly, previous studies have used information theoretic methods to detect amino acid covariation in multiple sequence alignments of many different protein families [26-28] and have used contact prediction based on covariation to dramatically improve the accuracy of protein structure modeling [29].

Despite the clear occurrence of amino acid covariation in natural protein sequences, the extent to which different selective pressures have shaped amino acid covariation in diverse protein families is unknown. Additionally, it is difficult to dissect to what extent phylogenetic bias has influenced the observations of amino acid covariation. Previous work has indicated that networks of covarying amino acids play a role in allosterically linking distant functional sites, suggesting that amino acid covariation is driven by protein functional constraints [30,31]. However, other studies have shown in two test cases that computational protein design can recapitulate naturally occurring covariation in the cores of SH3 domains [4,13,32] and for two-component signaling systems [33]. These results indicate that constraints imposed by protein structure have played a role in producing the covariation in the studied examples, but it has not yet been shown that these observations are general.

In this paper, we use computational protein design to measure the extent to which protein structure has shaped amino acid covariation in a diverse set of 40 protein domains. Since computational protein design predicts sequences that are energetically optimal based on protein
structure alone, we expect that pairs of amino acids that highly covary in both designed and natural sequences to have likely covared to maintain protein structure. We find significant overlap in the sets of highly covarying amino acid pairs between designed and natural sequences for all 40 domains examined, suggesting that maintenance of protein structure is a dominant selective pressure that constrains the evolution of amino acid interactions in proteins. Our analysis furthermore quantifies the extent to which different types of interactions explain the observed covariation. Finally, we demonstrate the utility of amino acid covariation recapitulation as a sensitive test for evaluating different protein design methods. We find that flexible backbone design significantly improves covariation recapitulation relative to fixed backbone design and that recapitulation of amino acid covariation is exquisitely sensitive to both the magnitude and mechanism of backbone flexibility. Taken together, these results provide fundamental insights into the physical nature of amino acid co-evolution and, more practically, provide a new benchmark that may help improve the accuracy of computational protein design methods.

RESULTS

Computational protein design recapitulates natural amino acid covariation

To compare amino acid covariation in natural and predicted designed protein sequences, we selected 40 protein domains that were diverse with respect to their secondary structure composition and fold class (Table 2). We then quantified natural amino acid covariation for each domain by creating a multiple sequence alignment for the domain, followed by computing covariation between every pair of columns in the multiple sequence alignment by using a mutual information based method [28] (see Methods). Pairs of amino acid positions with a covariation
score that is two standard deviations above the mean or greater were considered to be highly covarying pairs.

<table>
<thead>
<tr>
<th>Pfam ID</th>
<th>Pfam Name</th>
<th>Total # of sequences</th>
<th>PDB ID</th>
<th>Amino acid length</th>
<th>SCOP class</th>
</tr>
</thead>
<tbody>
<tr>
<td>PF00226</td>
<td>DnaJ</td>
<td>19122</td>
<td>2O37</td>
<td>60</td>
<td>a</td>
</tr>
<tr>
<td>PF00249</td>
<td>Myb_DNA-binding</td>
<td>9398</td>
<td>1GUU</td>
<td>47</td>
<td>a</td>
</tr>
<tr>
<td>PF00439</td>
<td>Bromodomain</td>
<td>4483</td>
<td>3JVL</td>
<td>89</td>
<td>a</td>
</tr>
<tr>
<td>PF00486</td>
<td>Trans_reg_C</td>
<td>35180</td>
<td>2ZXJ</td>
<td>77</td>
<td>a</td>
</tr>
<tr>
<td>PF00550</td>
<td>PP-binding</td>
<td>28748</td>
<td>1T8K</td>
<td>68</td>
<td>a</td>
</tr>
<tr>
<td>PF01029</td>
<td>NusB</td>
<td>5275</td>
<td>1TZV</td>
<td>127</td>
<td>a</td>
</tr>
<tr>
<td>PF01035</td>
<td>DNA_binding_1</td>
<td>5230</td>
<td>3GVA</td>
<td>87</td>
<td>a</td>
</tr>
<tr>
<td>PF01627</td>
<td>Hpt</td>
<td>7684</td>
<td>2A0B</td>
<td>88</td>
<td>a</td>
</tr>
<tr>
<td>PF12844</td>
<td>HTH_19</td>
<td>9622</td>
<td>3FYM</td>
<td>70</td>
<td>a</td>
</tr>
<tr>
<td>PF00072</td>
<td>Response_reg</td>
<td>103232</td>
<td>1MVO</td>
<td>111</td>
<td>a / b</td>
</tr>
<tr>
<td>PF00085</td>
<td>Thioredoxin</td>
<td>16281</td>
<td>1FB0</td>
<td>104</td>
<td>a / b</td>
</tr>
<tr>
<td>PF00581</td>
<td>Rhodanese</td>
<td>19885</td>
<td>1GN0</td>
<td>92</td>
<td>a / b</td>
</tr>
<tr>
<td>PF00582</td>
<td>Usp</td>
<td>15546</td>
<td>2Z3V</td>
<td>137</td>
<td>a / b</td>
</tr>
<tr>
<td>PF01451</td>
<td>LMWPc</td>
<td>5201</td>
<td>1JL3</td>
<td>130</td>
<td>a / b</td>
</tr>
<tr>
<td>PF00013</td>
<td>KH_1</td>
<td>11484</td>
<td>1WVN</td>
<td>63</td>
<td>a + b</td>
</tr>
<tr>
<td>PF00076</td>
<td>RRM_1</td>
<td>31837</td>
<td>2X1B</td>
<td>71</td>
<td>a + b</td>
</tr>
<tr>
<td>PF00111</td>
<td>Fer2</td>
<td>11941</td>
<td>1CZP</td>
<td>76</td>
<td>a + b</td>
</tr>
<tr>
<td>PF00179</td>
<td>UQ_con</td>
<td>6107</td>
<td>1Z2U</td>
<td>138</td>
<td>a + b</td>
</tr>
<tr>
<td>PF00240</td>
<td>Ubiquitin</td>
<td>8316</td>
<td>2BWF</td>
<td>69</td>
<td>a + b</td>
</tr>
<tr>
<td>PF00254</td>
<td>FKBP_C</td>
<td>11034</td>
<td>2PPN</td>
<td>95</td>
<td>a + b</td>
</tr>
<tr>
<td>PF00327</td>
<td>Ribosomal_L30</td>
<td>3324</td>
<td>1BXY</td>
<td>52</td>
<td>a + b</td>
</tr>
<tr>
<td>PF00381</td>
<td>PTS-HPr</td>
<td>5246</td>
<td>1PTF</td>
<td>84</td>
<td>a + b</td>
</tr>
<tr>
<td>Domain ID</td>
<td>Domain Name</td>
<td>PDB ID</td>
<td>Chain Length</td>
<td>SCOP</td>
<td>Design Protocol</td>
</tr>
<tr>
<td>-----------</td>
<td>---------------</td>
<td>--------</td>
<td>--------------</td>
<td>------</td>
<td>----------------</td>
</tr>
<tr>
<td>PF00542</td>
<td>Ribosomal_L12</td>
<td>1CTF</td>
<td>68</td>
<td>a + b</td>
<td></td>
</tr>
<tr>
<td>PF00691</td>
<td>OmpA</td>
<td>1OAP</td>
<td>96</td>
<td>a + b</td>
<td></td>
</tr>
<tr>
<td>PF00708</td>
<td>Acylphosphatase</td>
<td>3BR8</td>
<td>89</td>
<td>a + b</td>
<td></td>
</tr>
<tr>
<td>PF04002</td>
<td>RadC</td>
<td>2QLC</td>
<td>123</td>
<td>a + b</td>
<td></td>
</tr>
<tr>
<td>PF00018</td>
<td>SH3_1</td>
<td>2O9S</td>
<td>48</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>PF00041</td>
<td>Fn3</td>
<td>1TEN</td>
<td>81</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>PF00168</td>
<td>C2</td>
<td>3F04</td>
<td>87</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>PF00169</td>
<td>PH</td>
<td>1UNQ</td>
<td>103</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>PF00313</td>
<td>CSD</td>
<td>3I2Z</td>
<td>67</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>PF00355</td>
<td>Rieske</td>
<td>1FQT</td>
<td>95</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>PF00364</td>
<td>Biotin_lipoyl</td>
<td>2EVB</td>
<td>68</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>PF00498</td>
<td>FHA</td>
<td>3GQS</td>
<td>67</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>PF00595</td>
<td>PDZ</td>
<td>2H3L</td>
<td>87</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>PF01833</td>
<td>TIG</td>
<td>3MQI</td>
<td>83</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>PF02823</td>
<td>ATP-synt_DE_N</td>
<td>1AQT</td>
<td>80</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>PF07679</td>
<td>I-set</td>
<td>1U2H</td>
<td>91</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>PF07686</td>
<td>V-set</td>
<td>2PND</td>
<td>116</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>PF08666</td>
<td>SAF</td>
<td>1UCS</td>
<td>60</td>
<td>b</td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Protein domains used for covariation analysis. Forty diverse protein domains were selected from Pfam. This table contains the Pfam information for each domain, the total number of sequences assigned to this domain according to Pfam, the PDB ID of the domain crystal structure used for design, the domain length and the SCOP classification.

We predicted designed protein sequences for each of the 40 domains using RosettaDesign [4,34]. We first used the standard RosettaDesign fixed backbone protocol [34], which takes a crystal structure as input and runs Monte Carlo simulated annealing, to predict 500 designed sequences for each domain structure. We then quantified amino acid covariation in the designed sequences and compared it to natural amino acid covariation for each domain. We calculated the
similarity between designed and natural covariation based on the percent overlap of the highly
covarying pairs in each set (see Methods). We found this overlap to be significant ($p < 0.001$)
for all 40 domains.

**Magnitude of structural variation affects covariation similarity**

Given the observation that fixed backbone protein design can recapitulate a significant
fraction of naturally covarying amino acid pairs, we next aimed to understand how incorporating
backbone flexibility into the design protocol affects this recapitulation. To accomplish this, we
generated a conformational ensemble of 500 backbone structures for each domain using the
“backrub” method [35] in Rosetta [36], which iteratively applies local backbone perturbations
throughout the protein structure combined with adjustments in side-chain conformations. We
then used RosettaDesign to predict a low energy sequence for each backbone structure in the
ensemble, resulting in 500 designed sequences. Figure 16 shows a flow chart of this approach
applied to an SH3 domain.
Figure 16. Flow chart of the computational strategy to compare natural and designed amino acid covariation. For each domain family (the SH3 domain in the example), a crystal structure of the domain is obtained from the Protein Data Bank. This structure is used as input to a protocol that generates a conformational ensemble of protein structures. Each structure in this ensemble is then input to a protocol that designs a low energy sequence consistent with the structure. Amino acid covariation is calculated for every pair of positions in the designed sequences, and the designed covariation is compared to the covariation seen among naturally occurring sequences with the same protein domain.
To investigate the effect of the magnitude of backbone flexibility in the design protocol, we varied the temperature parameter in the Monte Carlo simulations used in the backrub protocol to generate conformational ensembles with different amounts of structural variation (Figure 17A). We designed sequences for each ensemble (kT = 0.3, 0.6, 0.9, 1.2, 1.8, 2.4) and quantified similarity to natural covariation for each set of sequences. We compared these results with sequences designed using the fixed backbone design protocol described above (“Fixed”). Figure 17B shows a significant increase in covariation similarity for the flexible backbone simulations relative to the fixed backbone simulation. Moreover, the distributions of covariation similarity for the 40 domains show that there is an optimal degree of structural variation, as low-temperature and high-temperature simulations perform significantly worse than mid-temperature simulations. We observed this same trend when we repeated this analysis using a different method for quantifying covariation [37], suggesting that our results are not dependent on the method used to quantify covariation.

To better understand the basis of this trend, we examined several other sequence and structural characteristics: sequence recovery, sequence profile similarity, sequence entropy and structural variation (see Methods). The resulting distributions for these characteristics are shown in Figure 17C. Sequence entropy and sequence profile similarity showed similar trends to covariation similarity (sequence entropy is most similar to natural sequences and profile similarity is highest at 0.9 kT), suggesting that backbone flexibility allows for sampling diverse sequences with native-like properties. These trends are consistent with the observation that sequence recovery decreases with increasing amounts of backbone flexibility. As diversity within a set of sequences increases, those sequences tend to become more dissimilar to any individual sequence, including the native sequence of the crystal structure used as input for
design. Structural variation in the 0.3, 0.6, 0.9 and 1.2 kT simulations is less than the structural variation among naturally occurring protein structures with these domains, which could be due to the fact that natural proteins use additional mechanisms of generating structural variation that are not being modeled, such as the insertion or deletion of amino acids in loop regions. Taken together, these results suggest that a moderate degree of backbone flexibility allows for the accommodation of sequences that differ from the native sequence and yet are similar to naturally occurring sequences with respect to their sequence profiles, sequence entropies and patterns of amino acid covariation.
**Figure 17. Effects of the magnitude of structural variation on sequence properties.** A) Representative structures of conformational ensembles generated using backrub Monte Carlo simulations with different temperature parameters (shown for an SH3 domain). B) Box plot showing the distributions of covariation similarity values between natural sequences and sequences designed using conformational ensembles generated with different temperature parameters (including “Fixed” backbone design sequences). Each distribution contains 40 covariation similarity values, one for each of the 40 protein domains. For each box plot in this study, the top and bottom whiskers denote the maximum and minimum values, respectively. The top and bottom of the box indicates the 75th and 25th percentiles, respectively, and the bold line denotes the 50th percentile. C) Box plots showing the distributions of sequence and structural characteristics for each temperature. The sequence profile similarity, sequence recovery and structural variation distributions each contain 40 values, one for each of the 40 protein domains. The sequence entropy distributions each contain 2778 values, one for each position in the 40 protein domains. “Native” distributions for structural variation and sequence entropy of the natural proteins were included as well.

**Mechanism of structural variation affects covariation similarity**

Next we examined whether or not these results were specific to the method used to generate the conformational ensembles for design. We tested two other Monte Carlo based methods that iteratively perform perturbations to the backbone. One method performed Kinematic Closure (“KIC”), which involves randomizing phi/psi torsions in a local region of the backbone while keeping the rest of the backbone fixed, thus introducing a chain break, and then using inverse kinematics to solve for the torsions that will close the chain [38]. The other method performs potentially non-local moves by perturbing the phi and psi torsions of residues by a random small angle (“Small”) [39]. We ran both of these methods for the same number of trials and for the same values of kT as the backrub protocol. The resulting distributions of covariation similarity show the same trend we observed previously with the backrub simulations, where mid-range temperature simulations result in an optimal degree of covariation similarity.

While the optimal simulation temperature parameter was comparable for each of the methods tested, the methods achieved a different optimum level of covariation similarity with the natural sequences. We found that the two local move simulations (KIC and Backrub)
outperformed the non-local move simulation (Small). To test if this observation holds true more generally, we tested two additional methods of generating conformational ensembles that make non-local moves. These methods included FastRelax ("Relax"), which consists of multiple rounds of side-chain repacking and all-atom minimization while increasing the weight of the repulsive term in the Lennard–Jones (LJ) potential from 2% to 100% of its default value, and AbInitioRelax ("AbRelax"), which performs fragment-based \textit{ab initio} structure prediction followed by FastRelax [40]. As an additional control, we also designed sequences using a fixed backbone structure with an energy function that dampens the weight of the repulsive LJ term ("Soft"). The resulting covariation similarity distributions show that recapitulation of natural amino acid covariation is sensitive to the method used to generate conformational ensembles (Figure 18A). Both local move simulations (KIC, Backrub) achieved higher median covariation similarities than the non-local move simulations (Small, AbRelax, Relax) and the fixed backbone simulations (Fixed, Soft).

We also evaluated each of these methods using the other metrics described above: native sequence recovery, sequence profile similarity, sequence entropy and structural variation (Figure 18B). Unexpectedly, the AbRelax method, which resulted in conformational ensembles with the greatest structural variation, achieved the highest sequence profile similarity with the natural sequences of any method tested. A possible explanation for this behavior is that local interactions are preserved in AbRelax generated structures, but the overall topology of the protein is incorrect. To test this hypothesis, we examined covariation similarity in the AbRelax sequences by splitting all covarying pairs into the following two sets: pairs separated by fewer than 10 residues in sequence ("Near") and pairs separated by greater than 10 residues in sequence ("Far"). This analysis revealed that whereas AbRelax sequences have relatively high
covariation similarity with natural sequences for pairs close in sequence, they have low
covariation similarity for pairs that are distant in sequence (Figure 18C). In contrast, covariation
similarity for “near” and “far” pairs were similar for simulations using backrub ensembles.
These results suggest that AbRelax can model local interactions within a secondary structural
element or between adjacent secondary structures, but it does not correctly capture non-local
interactions that are likely critical for achieving a cooperatively folded, stable tertiary structure.
This observation demonstrates the importance of using amino acid covariation to evaluate the
accuracy of protein design methods, since it is possible to obtain deceptively high sequence
profile similarity scores with highly divergent tertiary structures as long as local interactions are
maintained. Of all the flexible backbone design methods tested, Backrub, kT = 0.9 resulted in
sequences most similar to the natural sequences with respect to covariation similarity and
sequence profile similarity. Using the assumption that a method that gives higher similarity to
natural sequences will better capture the mechanisms underlying covariation, we used Backrub,
kT = 0.9 as the representative flexible backbone sequences for the remainder of the study.
Figure 18. Effects of the mechanism of structural variation on sequence properties. A) Box plot showing the distributions of covariation similarity values between natural sequences and sequences designed using conformational ensembles generated with different methods. The Backrub, KIC and Small simulations shown here were run with kT values of 0.9, 1.2, and 1.2, respectively (which represents the optimal temperature for covariation similarity in each case, see Figures 17B). B) Box plots showing the distributions of sequence and structural characteristics for each method of generating conformational ensembles. C) Covariation similarity distributions for subsets of covarying pairs that are “near” in sequence (separated by 10 residues or fewer) or “far” in sequence (separated by greater than 10 residues).
Backbone flexibility allows for amino acid interactions that fixed backbones cannot accommodate

To understand how backbone flexibility influences the extent of covariation similarity between designed and natural sequences, we identified all pairs of amino acid positions that highly covaried in both the natural sequences and a set of flexible backbone sequences (Backrub, kT = 0.9) but did not highly covary in the fixed backbone sequences. We then took all pairs of amino acids at these positions that were not sampled in the fixed backbone simulation and designed them onto the crystal structure backbone using fixed backbone design. For each pair of these positions, we calculated mean interaction energies and compared these energies between fixed and flexible backbone design structures (Figure 19A). We calculated both one-body energies, which include the interaction of an amino acid residue with itself, and two-body energies, which include the interactions between two amino acid residues in the protein (see Text S1 for description of the components of Rosetta one-body and two-body energies). We found both the one-body and two-body energies of these pairs to be generally greater in the context of fixed backbones relative to flexible backbones. Splitting the energies into their component terms revealed that the backbone-dependent Dunbrack rotamer energy (fa_dun) and Lennard-Jones repulsive (fa_rep) terms resulted in greater energy increases in the one-body and two-body energies, respectively, than any other term in the energy function. These results suggest that amino acid pairs that covary in flexible backbone simulations but do not covary in fixed backbone simulations generally cannot be accommodated on fixed backbones without resulting in steric clashes or rotamers that are unfavorable for the given backbone. Simply modifying the energy function by using a “soft” repulsive potential that reduces the energy of clashes does not increase sequence diversity or covariation similarity (Figure 18B), suggesting that backbone
movements are required to accommodate these amino acid interactions. Figure 19B shows representative cases where some degree of backbone flexibility is required to correctly model the precise interaction details of specific amino acid pairings.
Figure 19. Energetic effects of forcing amino acid covariation onto fixed backbones. A) Scatter plots of covarying pair energies in the context of fixed or flexible backbones. Each dot represents a pair of positions that was found to be highly covarying in the flexible backbone sequences (Backrub, kT = 0.9) and the natural sequences but not in the fixed backbone sequences. Pairs of amino acids at these positions that were found in flexible backbone designs but not in fixed backbone designs were forced onto fixed backbones taken from X-ray crystal structures and their one and two-body energies were calculated. The left plot shows a comparison of one-body energies and the right plot shows a comparison of two-body energies for these pairs. B) Representative examples of pairs of amino acids that require backbone movements to achieve low-energy interactions. Models from flexible backbone design (Backrub, kT = 0.9) are shown in cyan and models from fixed backbone design are shown in magenta. The top case shows a ring stacking interaction, the middle case shows a hydrogen bonding interaction and the bottom case shows a salt bridge interaction. Red disks represent steric clashes, where the radius and number of the disks is proportional to the magnitude of the clash.

Natural and designed amino acid pair propensities are highly correlated

We have thus far compared amino acid covariation between natural and predicted designed sequences based on the extent of overlap between the sets of highly covarying pairs. However, it is also important to consider the amino acid pair propensities at covarying positions to test whether the natural and designed covarying pairs utilize the same types of amino acid interactions. To accomplish this, we calculated amino acid propensities at pairs of positions that covary in both the natural and designed sequences (Figure 20A). Over-represented amino acid pairs in both designed and natural sequences included those with opposite charges, hydrophobic pairs and hydrogen-bonding pairs. Differences in the designed and natural amino acid pair propensities included the over-representation of cation-pi pairs in the natural sequences but not in the designed sequences (such as W-R). These differences highlight shortcomings of the energy function used for design, which does not currently account for cation-pi interactions.

To quantify the similarity between the natural and designed covarying pair propensities, we calculated the correlation coefficients between the natural and designed propensities for all sets of designed sequences. We found these correlations to be dependent on both the magnitude
and mechanism of backbone flexibility, as we previously observed with the overlap in covarying pairs. The comparison between natural and designed pair propensities for fixed backbone sequences and for a set of flexible backbone sequences (Backrub, kT = 0.9) are shown in Figure 20B, again supporting the conclusion that backbone flexibility improves recapitulation of amino acid covariation.

Figure 20. Correlation of amino acid pair propensities between natural and designed covarying pairs. A) Heat maps of amino acid pair propensities at pairs of positions that are
highly covarying in both designed and natural sequences. Red pairs are over-represented at covarying positions and blue pairs are under-represented at covarying positions. The values are shown as Z-scores, which denote the number of standard deviations above or below the mean. B) Correlation of amino acid pair propensity Z-scores between designed and natural sequences. The left plot shows the correlation from flexible backbone design sequences and the right plot shows the correlation from fixed backbone design sequences. A Pearson correlation coefficient (r) is shown for each plot.

**Mechanisms underlying covariation in natural and design sequences include**

**complementary changes in amino acid size, charge or hydrogen bonding**

While similar pair propensities between natural and designed covarying pairs demonstrate that the same types of amino acid interactions occur in both natural and designed sequences, they do not show that the mechanisms underlying covariation are the same in both cases. To investigate this, we first classified the mechanism of covariation for all pairs that covary in both designed and natural sequences and then quantified how often the same mechanism is used. Figure 21A shows an illustration of three of the covariation mechanisms: size, hydrogen bonding and charge. Classifying each of these mechanisms requires examining the transition from one amino acid pair to another. For example, the transitions depicted in Figure 21A are IA–VV, AP–SS, RE–DR. Covariation due to size involves a decrease in the size of one amino acid and an increase in the size of the other (IA–VV). Covariation due to hydrogen bonding involves a hydrogen bond that exists in one pair but not the other (AP–SS). Covariation due to charge involves a pair of amino acids with opposite charges that either swap sign (RE–DR) or become uncharged amino acids. We also defined covariation mechanisms based on cation-pi interactions, pi-pi interactions, and other interactions not falling into any of the previous categories that we classify as hydrophobic, hydrophilic or mixed hydrophobic and hydrophilic (see Methods for a detailed definition).
For each pair of positions that covaried in both the designed and natural sequences, we computed the ten most significant transitions between amino acid pairs at those positions and classified each transition based on the mechanism of covariation. The resulting distributions of covariation mechanisms for the designed and natural pairs are shown in Figure 21B. The designed and natural covariation mechanisms distributions share similar properties, including covariation due to charge being the most common mechanism, whereas cation-pi, pi-pi and other (hydrophilic) covariation mechanisms are more rare. In both natural and designed distributions, hydrogen bonding and size covariation together account for approximately 30% of the total mechanisms. However, a number of quantitative differences exist in the distributions, including charge occurring more frequently in the designed pairs, suggesting that the design method may be over-predicting charged interactions. Additionally, in the natural pairs, size covariation is more common than hydrogen bonding covariation while the opposite is true in designed pairs. The “other” categories are also more common in the natural pairs than in the designed pairs. To better understand these differences, we split the pairs up based on the extent of their burial and compared the distributions of covariation mechanisms. This analysis revealed that covariation mechanism is dependent on the extent of pair burial and that buried pairs have the most significant differences between natural and designed covariation mechanisms. In natural buried pairs, the most common covariation mechanisms are size and other (hydrophobic), whereas the most common mechanisms in designed buried pairs are hydrogen bonding and size. This likely occurs due to insufficient penalization of buried polar groups during the design protocol, resulting in over-predicting polar amino acids at buried positions and therefore incorrect predictions of covariation mechanism.
To quantify how often the same covariation mechanism is used for specific pairs of positions in the designed and natural sequences, we calculated the percent of pairs sharing the same classification type in both the natural and designed sequences (percent overlap) for each type of covariation mechanism (Figure 21C). Covariation due to charge has the highest percent overlap between the designed and natural pairs, followed by hydrogen bonding, size, other (hydrophobic) and other (mixed), which have roughly equal percent overlaps. Covariation due to cation-pi and pi-pi interactions have relatively low percent overlaps between the designed and natural sequences, likely due to the fact that these types of interactions are not explicitly accounted for in the design energy function. We repeated this analysis using fixed backbone design sequences and found a decrease in the percent overlaps for size and other (hydrophobic) interactions, indicating that backbone flexibility may aid in modeling these types of covariation mechanisms. Taken together, this analysis provides insights into the mechanisms underlying amino acid covariation in naturally occurring proteins. Overall, the analysis shows considerable agreement between naturally occurring and designed covariation mechanisms. In some cases, it exposes pathologies in the design methods (such as the over-representation of polar amino acids in cores under-representation of cation-pi and pi-pi interactions) that can be addressed in future work using naturally occurring covariation as a reference point.
Figure 21. Covariation mechanisms of natural and designed covarying pairs. A) Representative examples of covariation mechanisms in both natural and designed sequences. Models from flexible backbone design (Backrub, kT = 0.9) are shown in cyan and x-ray crystal structures are shown in yellow. The left panel shows covariation due to size, the middle panel shows covariation due to hydrogen bonding and the right case shows covariation due to charge. The top and bottom rows for each panel show different amino acid pairs in the same positions but in different proteins. B) Pie charts showing the distribution of covariation mechanisms for pairs that covary in both natural and designed sequences. The left pie chart shows covariation mechanisms for natural pairs and the right shows covariation mechanisms for designed pairs. See the methods for a definition of the mechanism classification. C) Bar plot showing the percent overlap between natural and designed pairs for each covariation mechanism.
Covarying pairs not modeled by design are more distant in three-dimensional structure and differ in amino acid pair propensities

While computational protein design can model a significant fraction of naturally occurring covarying amino acid pairs, there remain pairs of amino acids that are highly covarying in the natural sequences but not in the designed sequences (nature-specific pairs). Moreover, there also exist pairs that highly covary in designed sequences but not in natural sequences (design-specific pairs). Figure 22A shows the classification of nature-specific, design-specific and overlap pairs for the SH3 domain. To understand the basis for these differences, we first compared these sets of pairs based on their distances in three-dimensional structure (Figure 22B). We found the design-specific and overlap covarying pairs to be significantly closer in structure than the nature-specific pairs. These results are consistent with the all-atom energy function used for generating the design sequences, which is most sensitive at short distances. The long distances in the nature-specific pairs could result from a number of factors, including interactions that bridge monomers in an oligomeric complex [37], interactions that exist in alternative conformations [37], long-range correlations in protein dynamics or from phylogenetic bias in the natural sequences. Another possibility is that in naturally occurring proteins, destabilizing substitutions (that occur in functional sites) co-vary with compensating stabilizing mutations in the protein that could be far away from the functional site.

In addition to analyzing design-specific and nature-specific pairs with respect to pair distance, we compared them based on extent of amino acid burial, the presence in interfaces or active sites, and amino acid pair propensity. We observed a slight decrease in the percent of exposed pairs in the designed-specific pairs relative to the nature-specific pairs, which may be due to the difficulty of accurately modeling solvent exposed interactions in protein design. We
observed no difference in the design-specific and nature-specific pairs with respect to their presence in interfaces or active sites, suggesting that the constraints imposed by known functional sites are not responsible for the inability to model the nature-specific pairs. We observed that the amino acid pair propensities of nature-specific and overlap pairs were different, while the amino acid pair propensities of design-specific pairs were highly correlated to those of the overlap pairs (Figure 22C). The latter observation indicates that the energetic interactions leading to design-specific and overlap pairs may be similar to each other. A simple explanation may be that the design-specific pairs are equally compatible with the given protein structure, but may simply not have been sampled by nature. Such design-specific pairs may provide opportunities for engineering proteins with novel amino acid interactions, such as re-designing the specificity of protein-protein interactions.
Figure 22. **Distinguishing features of natural and designed covarying pairs.** A) Example comparison of natural and designed covariation for an individual protein domain (SH3 domain). Each dot represents an amino acid pair. Dashed red lines indicate the thresholds used to identify pairs as highly covarying (two standard deviations above the mean). The indicated quadrants contain the design-specific, overlap and nature-specific pairs, respectively. B) Box plot of distances between amino acid pairs in the nature-specific, design-specific and overlap sets. Pair distances are measured as the minimum distance between heavy-atoms of two amino acids in the representative crystal structure of the domain. C) Correlation of amino acid pair propensity Z-scores between different sets of covarying pairs. The left plot shows the correlation between design-specific and overlap pairs and the right plot shows the correlation between nature-specific and overlap pairs. A Pearson correlation coefficient (r) is shown for each plot.
DISCUSSION

Our study tested the hypothesis that the structural constraints imposed by protein architecture are a major determinant of amino acid covariation in naturally occurring proteins. If true, we reasoned that computational design methods that design sequences based on protein structure alone should be able to recapitulate amino acid covariation, provided that design predictions are sufficiently accurate. Confirming these ideas, we found a significant overlap between amino acid covariation in natural and designed protein sequences across a set of 40 diverse protein domains. These results quantify the influential role of the selective pressures for maintaining protein structure on shaping amino acid covariation. Therefore, even though correlated changes are undoubtedly important to evolve new activities and regulatory mechanisms [30,31] the presence of covariation alone may not necessarily indicate a functional role.

Our study also illustrates how recapitulation of amino acid covariation serves as a stringent test for the ability of computational protein design methods to capture precise details of interactions between amino acids. We demonstrate that modeling backbone flexibility significantly increases the similarity between natural and designed covariation, and that this similarity is exquisitely sensitive to the mechanism used to model backbone changes. These findings indicate that protein backbone motions are required for allowing precise adjustments in amino acid interactions that enable covariation. Moreover, simulations that perform local backbone movements (Backrub and KIC) result in sequences with more natural-like covariation than simulations that perform non-local backbone movements (AbRelax, Relax, Small). Proteins may have undergone local motions similar to Backrub and KIC moves to accommodate new mutations and amino acid interactions during evolution [24,35,36,41]. Such motions could have
provided proteins with a mechanism to allow subtle, incremental changes to their structures without adversely affecting protein structure or protein function.

While local motions may be a common mechanism for proteins to accommodate point mutations, larger structural adjustments may be necessary for dealing with insertions or deletions. In this study, we found that a moderate degree of backbone flexibility best recapitulated natural amino acid covariation, however, the magnitude of structural variation produced by this degree of backbone flexibility was less than the structural variation among naturally occurring protein families.

This discrepancy is likely due to the assumption in the design method that the protein remains a fixed length. This is not true in naturally occurring sequences; in fact, all 40 domains in our benchmark include loop regions that have varying lengths. Mutations that change the length of a flexible loop could allow for secondary structure elements to re-orient themselves and slightly alter the tertiary structure. The accumulation of mutations in loop regions can produce significant structural diversity that cannot be modeled using a protein design method that keeps the number of amino acids in a protein constant. Future protein design methods, particularly those involving loop regions such as protein-protein interaction design or enzyme specificity design, could potentially benefit from incorporating moves that both change the conformation and length of the protein backbone.

In addition to observing significant similarity between the sets of natural and designed highly covarying amino acid pairs, we observed a high correlation in the amino acid propensities of these covarying pairs and showed that the structural mechanisms underlying covariation are similar for both natural and designed sequences. Differences between natural and designed covarying pairs highlight areas for improvement in the energy function used for protein design.
For instance, cation-pi interactions, which are not explicitly accounted for in the energy function used in this study, have high propensities among naturally covarying pairs but not in designed covarying pairs. Similarly, polar amino acid pairs are more frequent in the cores of designed proteins than in naturally occurring proteins. Interestingly, we found differences in the pair propensities between nature-specific pairs and pairs that highly covary in both natural and design sequences. We also observed that nature-specific pairs tend to be more distant in three-dimensional structure. These results have implications for the field of contact prediction, as combining amino acid covariation with amino acid pair propensity information could improve the prediction of three-dimensional contacts in protein structures compared to using amino acid covariation alone. Improving methods of contact prediction would increase the accuracy of recent protein structure prediction algorithms that use amino acid covariation [29].

Unlike nature-specific pairs, design-specific pairs have amino acid propensities that are highly correlated with the amino acid propensities of pairs that covary in both natural and designed sequences. These design-specific pairs represent candidate positions for engineering amino acid interactions that have not been sampled by natural protein evolution. A practical application of this is the re-wiring of protein interaction specificity to design orthogonal protein-protein interactions for use in synthetic biology. Natural intermolecular covariation has previously been exploited to alter specificity in two component signaling systems [42]. Future work could exploit designed intermolecular covariation to re-engineer protein interactions with novel specificities that are orthogonal from naturally occurring protein-protein interactions [43] and therefore useful for synthetic applications.
METHODS

Preparation of natural protein sequences

The protein domains used in this study were selected from the Pfam database [44] based on the following criteria: 1) at least one crystal structure of a protein containing the domain was available from the Protein Data Bank (PDB) [45], 2) at least 500 sequences of proteins from the domain were available from Pfam and 3) the domain was equal to or less than 150 amino acids in length. We selected a total of 40 domains that represented a diverse set of protein folds (Table 2). The seed alignment and the full alignment for each domain were obtained from Pfam. In order to remove highly divergent sequences with uncommon insertions or deletions, we first removed sequences from the seed alignment if they had either of the following: 1) a gap in a position where 90% of the sequences in the seed alignment did not have a gap or 2) an amino acid in a position where 90% of the sequences in the seed alignment had a gap. Next, we aligned each sequence in the full alignment to the seed alignment using MUSCLE [46] and we discarded any sequences that resulted in the creation of gaps that were not in the seed alignment. This resulted in an alignment without sequences containing uncommon insertions or deletions. Finally, we used CD-HIT [47] to filter the sequence alignments by removing sequences with 80% redundancy or greater.

Generation of designed protein sequences

For each of the 40 protein domains, the highest resolution crystal structure of the domain was obtained from the PDB. This structure was used as a template for all the design simulations. The design method used in this study consisted of two steps: 1) the generation of a conformational ensemble and 2) the design of sequences onto each structure in the ensemble.
using RosettaDesign. For each of the 40 domains, 500 structures were generated for the conformational ensemble and 500 sequences were designed, one for each structure in the ensemble. Descriptions of each protocol used for generating conformational ensembles and for designing sequences are provided in Text S1 along with the corresponding Rosetta command lines.

**Amino acid covariation**

Amino acid covariation was quantified using a mutual information based metric called Zpx [28]. First, the Shannon entropy is calculated at each position $i$ as follows:

$$ H_i = - \sum_x P_x \log_2 P_x $$

where $P_x$ is the frequency of amino acid $x$ at position $i$. The joint entropy is calculated between all pairs of positions as follows:

$$ H_{i,j} = - \sum_x \sum_y P_{x,y} \log_2 P_{x,y} $$

where $P_{x,y}$ is the frequency of amino acid $x$ and $y$ and positions $i$ and $j$, respectively. The mutual information (MI) between each pair of columns in a multiple sequence alignment, $i$ and $j$, was calculated as the difference between the individual entropies and the joint entropy:

$$ MI_{i,j} = H_i + H_j - H_{i,j} $$

Next, the background mutual information due to random noise and shared ancestry is subtracted to obtain the product corrected mutual information (MIP) [27]:

$$ MIP_{i,j} = MI_{i,j} - \frac{(\bar{MI}_i \times \bar{MI}_j)}{\bar{MI}} $$
where $\overline{MI}$ is the mean MI of position $i$ with all other positions and $\overline{M}$ is the overall mean. This value is converted to two $Z$-scores, one for each column, which are multiplied together:

$$Z_{ixy} = \frac{MI_{i,j} - \overline{MI}_i}{\sigma(MI_i)} \times \frac{MI_{i,j} - \overline{MI}_j}{\sigma(MI_j)}$$

The final score, called $Z_{px}$, is the square root of the absolute value of $Z_{ixy}$. If $Z_{ixy}$ is negative, then $Z_{px}$ is multiplied by -1. This normalization of $MI_p$ was demonstrated to reduce the sensitivity to misaligned regions in multiple sequence alignments, which otherwise result in artificially high mutual information scores [28]. Calculation of $Z_{px}$ was implemented in Python. Direct coupling analysis (DCA) was calculated using Matlab code provided by its authors [37].

**Covariation similarity**

To compare amino acid covariation between natural and designed multiple sequence alignments, $Z_{px}$ was first computed for all pairs of ungapped positions in each alignment. The mean $Z_{px}$ for each alignment was calculated and residue pairs with values greater than two standard deviations above the mean $Z_{px}$ were considered to be covarying residue pairs. The covariation similarity between the natural and designed covarying amino acid pairs was calculated as the percent of overlap, $2C / (A+B)$, where $A$ and $B$ are the total numbers of natural and designed covarying pairs, respectively, and $C$ is the number of pairs that covary in both natural and designed sequences. The same approach was used to calculate covariation similarity using DCA.

**Sequence recovery, entropy and profile similarity**

Sequence recovery was calculated as the mean percent identity of the designed sequences to the sequence of the crystal structure used as input for the design protocol. Sequence entropy was calculated for each position as $H$, defined above. Sequence profile similarity was calculated as
the mean prof_sim score [48] between each position in the natural and designed alignments. Briefly, prof_sim is the product of two scores: 1) the estimated probability that two amino acid frequency distributions represent the same source distribution and 2) the a priori probability of the source distribution. Using this metric, positions in designed sequences receive high prof_sim scores if both 1) their amino acid distribution is similar to the amino acid distribution at the corresponding position in the natural alignment and 2) their amino acid distribution is different than the background amino acid distribution. Calculation of sequence recovery, entropy and profile similarity was implemented in Python.

**Structural variation**

Structural variation was calculated as the mean pair-wise RMSD between 10 randomly selected structures in each conformational ensemble. Natural structural variation was computed for all domains with at least 10 crystal structures in the PDB. The following 20 domains were used to compute natural structural variation: PF00013, PF00018, PF00041, PF00072, PF00076, PF00085, PF00111, PF00168, PF00169, PF00179, PF00254, PF00355, PF00439, PF00550, PF00581, PF00582, PF00595, PF01833, PF07679, PF07686. Structural alignments and RMSD calculations were performed using PyMol [49].

**Amino acid pair propensities**

Amino acid pair propensities (PP) were calculated as the ratio between observed pair frequencies and the expected individual amino acid frequencies:

\[
PP_{x,y} = \log_{20} \frac{P_{x,y}}{P_x P_y}
\]

To compare amino acid pair propensities between two sets of covarying pairs, we computed the Z-score for each pair amino acid pair x,y. The Pearson correlation coefficient r between the two
sets of Z-scores was then calculated using R [50]. Cysteines were excluded from this analysis because they rarely appear in the designed sequences.

**Covariation mechanisms**

To classify the mechanisms of covariation for a pair of positions, we first computed a correlation coefficient $\phi_{r,s}$ for each amino acid pair $x,y$ [32]. We then calculated a score for all possible amino acid pair transitions (PT) between one pair $x,y$ and another pair $a,b$ as follows:

$$PT(x,y \leftrightarrow a,b) = \phi_{x,y} + \phi_{a,b} - \phi_{x,b} - \phi_{a,y}$$

This pair transition score quantifies the significance of the transition between the amino acid pair $x,y$ and the pair $a,b$. The most significant transitions are defined as those that highly favor pairs $x,y$ and $a,b$ but highly disfavor pairs $x,b$ and $a,y$. For each pair of positions, ten pair transitions with the greatest scores were assigned one of eight classes in the following order: charge, cation-pi, pi-pi, size, hydrogen bonding, other (hydrophobic), other (hydrophilic) and other (mixed).

Charge transitions involve a pair with opposite charges that either swap sign or become uncharged. A charge transition is also assigned to pair transitions that avoid like charges, for example, if $x$ and $b$ (or $y$ and $a$) are like charges. Cation-pi transitions involve one pair with a potential cation-pi interaction but no cation-pi interaction in the other pair. Similarly, pi-pi transitions involve one pair with a potential pi-pi interaction but no pi-pi interaction in the other pair. Size transitions involve a decrease in the size of one amino acid by at least $18 \text{ Å}^3$ (the volume of a methyl group) and an increase in the size of the other amino acid by at least $18 \text{ Å}^3$. Hydrogen bonding transitions involve a potential hydrogen bonding interaction (hydrogen bond acceptor and donor) in one pair but not in the other pair. The three other classes are used to assign pair transitions that do not fit any of the above criteria. Other (hydrophobic) transitions are those where both pairs contain only hydrophobic amino acids, other (hydrophilic) transitions
are those where both pairs contain only hydrophilic amino acids, and other (mixed) transitions are those with both hydrophobic and hydrophilic amino acids. Similarity between natural and designed was quantified using the percent overlap (defined above) for each covariation mechanism.

**Amino acid burial**

Amino acid burial was defined for each position based on the number of C$\beta$ atoms within 8 Å of the C$\beta$ atom of the given position as follows: exposed 0–8, intermediate 9–14 and buried > 14. We defined pairs of positions that were buried/buried or buried/intermediate as buried pairs, exposed/buried or intermediate/intermediate as intermediate pairs, and exposed/intermediate or exposed/exposed as exposed pairs.

**Interface and active site positions**

For domains with known protein–ligand or protein–protein interface information, we defined all positions with a heavy-atom within 6Å of any heavy-atom on the binding partner as an interface position. The domains with interface information were PF00013, PF00439, PF00498, PF00691, PF00072, PF00018, PF00076, PF00249, PF00327, PF01035, PF00169, PF00550 and PF00595. For domains with known active sites, we defined all positions with a heavy-atom within 6Å of any heavy-atom on a catalytic residue as an active site position. The domains with active site information were PF00085, PF00111, PF00355, PF00708, PF00581, and PF01451.

**REFERENCES**

conformational entropy is not a strong determinant of amino acid environmental
in the Enumeration of Allowed Sequences for Different Structural Classes. J Mol Biol
193: 775–791.
Effects of Protein Mutations Appear to be Universally Distributed. J Mol Biol 369: 1318–
1332.
of Mutational Tolerance in HIV-1 Protease and Reverse Transcriptase Using Flexible
Identification of a novel class of farnesylation targets by structure-based modeling of
19. Smith CA, Kortemme T (2011) Predicting the Tolerated Sequences for Proteins and
libraries of fluorescent proteins evaluated by preservation and diversity of function. Proc
Natl Acad Sci USA 104: 48–53.
Specificity Using Computational Design of a Defined-Sequence Library. Chem Biol 17:
1306–1315.
Diversity Using Flexible Backbone Computational Protein Design. Structure 16, 1777–
1788.


41. Keedy DA, Georgiev I, Triplett EB, Donald BR, Richardson DC, et al. (2012) The role of


49. The PyMOL Molecular Graphics System, Version 1.2b0 Schrödinger, LLC.

Chapter 5: Computational Re-design of Enzyme Substrate Specificity Using Coupled Side-chain and Backbone Moves

INTRODUCTION

Naturally occurring enzymes have evolved to catalyze a particular chemical reaction on a specific set of substrate molecules. The ability to control and re-design the specificity of enzymes is important for a wide range of applications. For example, enzymes that are used in bioremediation need to specific to certain pollutants in order to break them down and remove them from the environment [1]. Enzyme specificity is critical in the food manufacturing industry where it can ultimately affect the taste and appearance of food products. Synthetic biology approaches such as metabolic pathway engineering require fine-tuning enzyme specificity in order to optimize the production of small molecule products such as drugs or biofuels [2]. Enzyme specificity is also important in therapeutic strategies such as suicide gene therapy, in which a therapeutic enzyme must convert a specific pro-drug into a cytotoxic compound in order to kill cancer cells [3,4]. A generalized and automated method for re-designing enzyme substrate specificity that could achieve any of these goals would therefore be tremendously useful. In this study, we evaluate computational protein design methods on their ability to predict known specificity altering mutations in a diverse set of enzymes. We introduce a new sampling algorithm that combines backbone flexibility, side-chain sampling and substrate flexibility in order to predict the set of tolerated amino acid sequences for a given enzyme–substrate interaction. Finally, we demonstrate that this new method significantly improves our ability to re-design enzyme specificity over traditional fixed backbone design methods.
ENZYME SPECIFICITY RE-DESIGN PROTOCOL

We aimed to develop a protocol would accurately predict mutations that re-design enzyme substrate specificity towards a non-native, desired substrate. This protocol would take as input the desired substrate as well as a structure of a wild-type enzyme bound to its native substrate, and it would output a set of mutant enzymes that have their specificities re-designed toward the desired substrate. To develop such a protocol, we faced the challenge of having to efficiently sample a large conformational search space at a high resolution. The degrees of freedom in this problem include the rotation and translation of the substrate relative to the enzyme, the flexibility of the substrate itself, the sequence and side-chain conformations of amino acids that constitute the active site, and the flexibility of the protein backbone, whose conformation may change upon introducing mutations.

Our strategy for sampling these many degrees of freedom at a high resolution was to couple multiple types of moves together. For example, in order to model the flexibility of the protein, we can combine backbone and side-chain moves into a single move rather than sampling them independently. This coupled move first makes a small, local backrub move and then re-adjusts the side-chain to accommodate the change in the backbone conformation. The side-chain move can either be a “repack” move that optimizes its conformation or a “design” move that allows it to mutate to a different amino acid. This coupled move strategy can be readily extended to model substrate flexibility. Substrate coupled moves consist of a random rigid-body rotation and translation followed by optimization of the substrate internal degrees of freedom. Protein and substrate coupled moves can be combined into a single protocol via a Monte Carlo simulation, as illustrated in Figure 23.
Figure 23. Flow chart of Monte Carlo simulation for enzyme specificity design. This simulation starts by picking a position at random. If that position is on the protein, a local backrub move is made followed by designing or repacking the side-chain. If the position is the substrate, a rigid-body translation and rotation is made followed by repacking the substrate. The move is then accepted or rejected based on the change in energy of the move. If the energy decreases, the move is accepted automatically, but if the energy increases, the move is accepted with some probability that is inversely proportional to the energy increase.

While first testing this protocol, one problem that we encountered was an accumulation of alanines and glycines in the output designs. This problem occurred because whenever a position was repacked or design, the rotamers and amino acids were selected completely at random. This makes it unlikely to accept mutations to larger or more flexible amino acids, since they are more likely to result in a clash than an alanine or glycine, which are the smallest amino acids. To overcome this problem, we developed a strategy for biasing side-chain sampling based on energy. This method first iterates over each rotamer at a given position and calculates its energy.
in the context of its environment. Given these rotamer energies, a Boltzmann weight for each rotamer can be calculated to determine the relative probability of that rotamer. We then select a rotamer using these probabilities, such that low energy rotamers are more likely to be chosen, allowing for more conformations and sequences to be sampled than traditional approaches that simply select the lowest energy rotamer.

![Energy and Selection Probability](image)

\[ P(E_i) = \frac{1}{\sum_i e^{-\frac{E_i}{kT}}} \]

**Figure 24. Boltzmann weighted rotamer sampling.** The bar-plot on the left shows the energy for each rotamer of an example amino acid. The equation on the bottom shows how the probability of a given rotamer \( i \) is calculated based on its energy \( E_i \). The bar-plot on the right shows the calculated probabilities for each rotamer.

We further extended this Boltzmann weighted rotamer sampling method to amino acid design by using it select a rotamer for each of the twenty amino acids. A probability for each amino acid is then calculated based on the energy of its selected rotamer. Application of Boltzmann weighted rotamer sampling reduces the number of alanines and glycines in the designs and significantly
increases the acceptance rate in Monte Carlo simulations, allowing more sequences to be sampled.

**ENZYME SPECIFICITY BENCHMARK RESULTS**

To evaluate the accuracy of our enzyme specificity re-design protocol, we compiled a set of ten enzyme–substrate pairs with known specificity altering mutations. For each of the pairs in this benchmark, there are available crystal structures of both the wild-type enzymes bound to the original substrate and the mutant enzymes bound to the new substrate. Figure 25 depicts the ten enzyme–substrate interactions in this benchmark.

![Figure 25. Ten enzyme substrate specificity benchmark cases](image)

**Figure 25. Ten enzyme substrate specificity benchmark cases.** Wild-type enzymes with their native substrates or substrate analogs are shown in green. Mutant enzymes with mutations that alter their substrate specificity are shown in magenta. PDB IDs are shown for the wild-type enzyme in each case.
To use our protocol to predict substrate specificity altering mutations, we first ran a Monte Carlo simulation on the wild-type enzyme with its original substrate, allowing the positions interacting with the substrate to sample any amino acid, and we recorded all sequences that were accepted during the simulation. We then replaced the original substrate with the new substrate and repeated the simulation. This resulted in two sets of sequences, one specific to the original substrate and the other specific to the new substrate. We then ranked all individual mutations by how enriched they were in the new substrate sequences relative to the original substrate sequences. Table 3 shows the results of our coupled move protocol on this benchmark. Overall, we identify 11 out of 17 known specificity altering mutations, and a majority of these identified mutations rank in the top 90\textsuperscript{th} percentile relative to other possible mutations.

<table>
<thead>
<tr>
<th>Mutation #</th>
<th>PDB ID</th>
<th>Mutation</th>
<th>Found Mutation?</th>
<th>Enrichment Percentile</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2FZN</td>
<td>Y540S</td>
<td>YES</td>
<td>94%</td>
</tr>
<tr>
<td>2</td>
<td>1FCB</td>
<td>L230A</td>
<td>YES</td>
<td>72%</td>
</tr>
<tr>
<td>3</td>
<td>1ZK4</td>
<td>G37D</td>
<td>NO</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>3KZO</td>
<td>E92A</td>
<td>YES</td>
<td>100%</td>
</tr>
<tr>
<td>5</td>
<td>3KZO</td>
<td>E92S</td>
<td>YES</td>
<td>97%</td>
</tr>
<tr>
<td>6</td>
<td>3KZO</td>
<td>E92P</td>
<td>NO</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>3KZO</td>
<td>E92V</td>
<td>NO</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>2O7B</td>
<td>H89F</td>
<td>YES</td>
<td>76%</td>
</tr>
<tr>
<td>9</td>
<td>1PK7</td>
<td>M64V</td>
<td>YES</td>
<td>79%</td>
</tr>
<tr>
<td>10</td>
<td>1K70</td>
<td>D314S</td>
<td>YES</td>
<td>92%</td>
</tr>
<tr>
<td>11</td>
<td>1K70</td>
<td>D314G</td>
<td>YES</td>
<td>8%</td>
</tr>
<tr>
<td>12</td>
<td>1K70</td>
<td>D314A</td>
<td>NO</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>2H6F</td>
<td>W602T</td>
<td>NO</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>1A80</td>
<td>K232G</td>
<td>YES</td>
<td>65%</td>
</tr>
<tr>
<td>15</td>
<td>1A80</td>
<td>R238H</td>
<td>YES</td>
<td>97%</td>
</tr>
<tr>
<td>16</td>
<td>3HG5</td>
<td>E203S</td>
<td>YES</td>
<td>99%</td>
</tr>
<tr>
<td>17</td>
<td>3HG5</td>
<td>L206A</td>
<td>NO</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 3. Results of enzyme substrate specificity benchmark. Green rows indicate cases where the coupled move protocol successfully identified a known specificity altering mutations, whereas red rows indicate cases where the mutation was not identified.
To quantify the benefit of using our coupled move protocol to predict specificity altering mutations, we compared the results with predictions based on traditional fixed backbone sequence design. As its name suggests, fixed backbone sequence design assumes a fixed backbone and does not model substrate rigid-body movements or substrate flexibility. The fixed backbone design protocol only identifies 2 out of the 17 specificity altering mutations. These results suggest that sampling more degrees of freedom, such as protein backbone flexibility, substrate rigid-body movements and substrate flexibility, can result in more accurate predictions.

<table>
<thead>
<tr>
<th>Mutation #</th>
<th>PDB ID</th>
<th>Mutation</th>
<th>Coupled Moves</th>
<th>Fixed Backbone</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2FZN</td>
<td>Y540S</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>2</td>
<td>1FCB</td>
<td>L230A</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>3</td>
<td>1ZK4</td>
<td>G37D</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>4</td>
<td>3KZO</td>
<td>E92A</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>E92S</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>E92P</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>E92V</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>8</td>
<td>2O7B</td>
<td>H89F</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>9</td>
<td>1PK7</td>
<td>M64V</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>10</td>
<td>1K70</td>
<td>D314S</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>D314G</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>D314A</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>13</td>
<td>2H6F</td>
<td>W602T</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>14</td>
<td>1A80</td>
<td>K232G</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>15</td>
<td></td>
<td>R238H</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>16</td>
<td>3HG5</td>
<td>E203S</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>17</td>
<td></td>
<td>L206A</td>
<td>NO</td>
<td>NO</td>
</tr>
</tbody>
</table>

Table 4. Comparison of coupled move protocol with fixed backbone design. The coupled move protocol described in this chapter predicts 11 out of 17 specificity altering mutations, whereas traditional fixed backbone design predicts 2 out of the 17 mutations.

FUTURE DIRECTIONS

While the results in the chapter demonstrate progress in re-designing enzyme substrate specificity, the tools developed here can be broadly applied to more general computational
protein design problems. One area of immediate interest is the prediction of the entire set of sequences the enable a protein to bind a given small molecule. Evaluating the accuracy of such predictions requires creating a benchmark set that consists of naturally occurring protein families with a large number of members that all bind the same molecule. I have written software to automatically detect protein families with these characteristics and I have found that the most useful systems are enzymes with conserved small molecule cofactors. Despite the fact that members of these protein families all interact with the same small molecule, their sequences are remarkably diverse and therefore provide ideal cases to test our sampling methods. To assess the accuracy of our methods, I will predict a set of binding site sequences for each protein family and directly compare these sequences with the naturally occurring sequences of that protein family. For comparison, I will also evaluate the accuracy of fixed backbone design as well as the flexible backbone design methods described in Chapter 4. Finally, I will test a number of variants of the method described in this chapter, including incorporating more subtle motions via torsion angle minimization, to determine which method results in the most accurate predictions. These results will be extremely valuable for future applications of computational protein design that involve interactions between proteins and small molecules.

REFERENCES

Chapter 6: Conclusion

Enormous progress has been made in the field of computational protein design over the past several years. Each successful design encourages us to attempt new and ambitious design goals, and this process has dramatically broadened the possible applications of computational protein design. Despite these broadened applications, the basic methods used in computational protein design have not fundamentally changed in this time. These basic methods involve sampling amino acid sequences and side-chain conformations while keeping the protein backbone fixed. One of the major goals of the work in this dissertation is to break this fixed backbone paradigm by demonstrating the benefits of incorporating backbone flexibility in design. Chapter 3 showed how modeling backbone flexibility improved the prediction of point mutant conformations, the modeling of alternative side-chain conformations and the prediction of the tolerated sequences in a protein–protein interaction. Chapter 4 further demonstrated that backbone flexibility improved the recapitulation of natural sequence properties including amino acid covariation, sequence profiles and sequence entropy. Finally, Chapter 5 introduced a novel algorithm that simultaneously optimized backbone and side-chain conformations and showed that this method substantially increased the accuracy of predicting enzyme substrate specificity altering mutations relative to fixed backbone design. The algorithms and benchmarks described in this dissertation provide the tools to both expand the range of existing computational protein design applications that can benefit from modeling backbone flexibility and to further broaden the possible applications of computational protein design.
Publishing Agreement
It is the policy of the University to encourage the distribution of all theses, dissertations, and manuscripts. Copies of all UCSF theses, dissertations, and manuscripts will be routed to the library via the Graduate Division. The library will make all theses, dissertations, and manuscripts accessible to the public and will preserve these to the best of their abilities, in perpetuity.

Please sign the following statement:
I hereby grant permission to the Graduate Division of the University of California, San Francisco to release copies of my thesis, dissertation, or manuscript to the Campus Library to provide access and preservation, in whole or in part, in perpetuity.

[Signature]
Author Signature

[Date]
3-11-14
Date