Positioning residues for function in designed proteins

by

Cody Krivacic

DISSERTATION

Submitted in partial satisfaction of the requirements for degree of

DOCTOR OF PHILOSOPHY

in

Bioengineering

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA, SAN FRANCISCO

AND

UNIVERSITY OF CALIFORNIA, BERKELEY

Approved:

DocuSigned by:

Tanja Kortemme — Tanja Kortemme

74ABEB009F65402...

Chair

DocuSigned by:

William DeGrado — William DeGrado

B6C7C7EB4D15441...

DocuSigned by:

John Dueber — John Dueber

936187F1D37D483...

Committee Members

Acknowledgments

I feel incredibly fortunate to have had the opportunity to study at an institute as vibrant and full of brilliance as UCSF, and to have been able to make my modest mark on one of the most exciting and promising fields I can imagine.

I will start by thanking my advisor, Tanja Kortemme, for giving me the opportunity to grow as a scientist in this incredible field, for regularly sharing deep and unexpected insights, for having the supernatural ability to tell that a protein is broken and how to fix it just by looking at it, and for encouraging me to pursue the many ideas that excited me. I must also acknowledge the members of the Kortemme lab for providing the perfect environment for my scientific and personal growth. In particular, I want to thank Anum Glasgow, my first mentor in the lab, for getting my foot in the door and always saying something unexpected; Kale Kundert, who deserves as much credit for Chapter 2 of this dissertation as I do, who explained many protein design concepts in ways that made them immediately intuitive, and from whom I still borrow many coding conventions; Xingjie Pan, who brought joy and insight to every lab meeting, and whose work on fragment-kinematic closure, along with Roland Pache and all of the authors of the paper on which Chapter 2 is based, made Pull Into Place in its current form possible. I'd also like to thank Tina Perica, who taught me how to pronounce my name and who brought a liveliness to every interaction, James Lucas and Kyle Barlow for their general career advice, Rob Alberstein and Amanda Loshbaugh for letting me use their dogs as support animals, and Benjamin Orr and Dom Grisingher for benchmarking and piloting HELIX. Most of all, I'd like to thank Chris Mathy for the many intense scientific discussions about completely nonscientific topics; you kept me (debatably) sane all these years.

I am honored to be able to call myself a member of the Rosetta community, which has been welcoming, fun, and full of crazy ideas that somehow work. I would also like to thank my committee members, John Dueber and Bill DeGrado, who's scientific and career advice have been invaluable. And to the many members of the Bioengineering program, it has been incredible to be a part of such a fun and diverse group of scientists.

My journey would not have been possible without the mentors that came before. Kalju Kahn taught me more in one year than I had learned in my entire life up to that point. I thank him for his rigorous teaching methods and for believing in me. Norbert Reich gave me my first taste of real biochemical research. His endless enthusiasm and expert guidance motivated me to push myself. Karen Maegley was an incredible mentor who always had my back. She encouraged me to go the distance when I was unsure if pursuing my doctorate was the right move, and I will be forever thankful for that.

Of course, none of this would have been possible without my family and friends. I would like to thank my wonderful parents and sister, Fidele Galey, Bob Krivacic, and Kittie Krivacic, who always believed I could do anything, and made me believe it as well. I also thank all my friends near and far for filling my free time with great memories and listening to me drone on about proteins. And finally, I give my warmest thanks to Laurel, who has been my rock through all this – thank you for your love and support, and for making me stop and smell the houseplants.

Contributions

Chapter 2 is adapted from the following published manuscript:

Krivacic, C.; Kundert, K.; Pan, X.; Pache, R. A.; Liu, L.; O Conchúir, S.; Jeliazkov, J. R.; Gray, J. J.; Thompson, M. C.; Fraser, J. S.; Kortemme, T. Accurate Positioning of Functional Residues with Robotics-Inspired Computational Protein Design. *Proceedings of the National Academy of Sciences* **2022**, *119* (11), e2115480119. https://doi.org/10.1073/pnas.2115480119.

Positioning residues for function in designed proteins

Cody Krivacic

Abstract

Computational protein design is poised to bring forth many advances in areas ranging from personalized medicine to biofuel production. While protein engineers have made huge strides towards the design of structure, proteins with new functions remain extremely difficult to design *in silico*, owing largely to the precise geometric requirements they demand. In this dissertation, I present two new methods for the design of functional proteins that focus specifically on enabling the precise sidechain geometries that are required for functions such as binding and catalysis. Pull Into Place (PIP) accurately places key amino acid sidechains by creating and stabilizing new irregular backbone geometries that are compatible with a provided functional interaction, utilizing new fragment-based local structure design and prediction methods. We experimentally validate PIP on a ketosteroid isomerase model system by redesigning its active site loop such that it uses a glutamate, rather than an aspartate, to perform proton extraction on its substrate, a task requiring accuracy on the scale of a carbon-carbon bond.

Where PIP enables one or a few functional sidechain interactions requiring irregular backbone structures, Helical ELements Interface eXplorer (HELIX), a new method for *de novo* protein-protein interface design, instead attempts to connect highly regular secondary structure elements to facilitate many functional sidechain interactions at once. It does this by matching tertiary structural motifs from the PDB with a library of proteins containing systematically reshaped α-helices. HELIX results in computational designs with a high degree of shape complementarity, many of which contain difficult-to-design polar interaction networks. Together, these two works encompass several new tools and approaches that enable the design of new functions.

Table of contents

List of Figures

List of Tables

Chapter 1

Introduction: Computational Protein Design

Proteins are a class of biological polymers that are uniquely situated for interacting with the atomic world. In nature, they perform a broad range of functions, including chemical synthesis and degradation, regulation, signaling, and structural roles. This variety in functions arises due to several features that differentiate proteins from other biological macromolecules: unlike the simple but inflexible hydrogen bonding-mediated interactions of DNA base pairing, proteins fold into and switch between complex structures by utilizing a breadth of chemical phenomena. The *Escherichia coli* lactose repressor, for instance, makes use of the energetic penalty resulting from burying a charged lysine residue in its hydrophobic core to facilitate a critical conformational change [1], combining charge interactions, relative permittivity, and the hydrophobic effect to impart function. By contrast, unlike the flexible but nonspecific interactions of lipids, proteins can reach deep and narrow energy minima, bestowing them distinctive structures that enable specificity and affinity in their interactions; indeed, preorganization of the same lactose repressor's DNA-binding domain results in tighter binding to its target [2]. It is the intricate and varied structures of proteins, as well as their tunable flexibility, that gives them the affinity, specificity, and adaptability needed to carry out complex biological functions. This also makes them attractive targets for engineers looking to harness the power of biology; computational protein engineering, then, is the rapidly evolving field of designing proteins *in silico* that fold into a desired structure or perform a user-defined task *in vitro* or *in vivo*.

The variety of chemical properties available to proteins that makes them attractive targets for engineering also represents the biggest challenge facing computational protein design. Since each position in a protein is comprised of one of twenty canonical amino acids, a 100-residue protein, smaller than most natural proteins [3], represents roughly 1.3×10^{130} (or 20^{100}) possible sequences, which is many orders of magnitude larger than the number of atoms in the observable universe [4]. Exhaustive sampling of sequences for even a single backbone is therefore impossible in practice, and the same is true of sampling conformational space for even a single sequence. Given the enormous search spaces required, it is no surprise that a myriad of limitations still impede the era of bespoke molecular machines.

1.1 Designing functional proteins

Sampling limitations are especially pronounced for the design of proteins with new functions. Such a task requires (a) functional interactions with a target molecule, (b) a structure that facilitates those interactions, and (c) a sequence that folds into the functional structure. These requirements result in a very small solution space within a vast combined sequence-conformation-energy landscape, so protein engineers who wish to design new functions must find clever ways to navigate it. Because the requirements for (a) are the most strict, design approaches typically start by defining one or more functional motifs. Next, (b) is addressed by finding a structure which can facilitate the functional motif. Finally, a sequence is designed (c) which stabilizes the design structure. Using this general framework, the design of functions such as ligand and metal binding [5], [6], nucleotide deamination [7], kemp elimination [8], small molecule sensing [9], and protein switching [10] have achieved success, often leveraging the extensive framework for modeling and design laid out by the Rosetta macromolecular modeling suite [11].

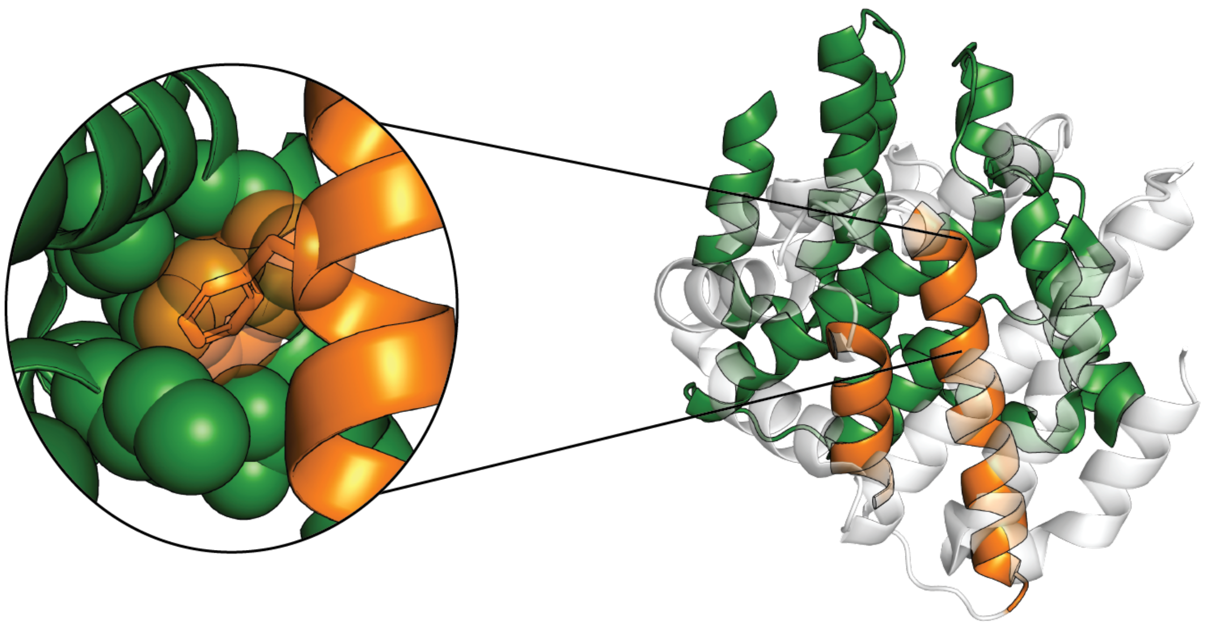


Figure 1.1 Geometric requirements in functional proteins. Specific residue-level interactions (left, orange and green spheres) must be accommodated by compatible backbone geometries (orange, cartoon), and often constellations of many such interactions must be accommodated by several noncontiguous structural units (right, orange). These, in turn, must be held together by a suitable protein scaffold (right, white).

Finding scaffolds that are compatible with a cluster of functional residues is often the most difficult step in designing function. These sampling difficulties are sometimes addressed by matching functional motifs into natural protein scaffolds [8], [9], or by modifying an existing protein to perform a new function [7], taking advantage of the myriad of functional structures already selected by nature. Natural protein folds are thought to be more fundamentally compatible with function than random protein geometries, as evidenced by the divergent and convergent evolution of domains [12], as well as the reuse of smaller geometric motifs [13], even among proteins with unrelated functions. However, it may sometimes be desirable to use completely *de novo* folds, as they are less likely to interact with natural proteins unintentionally.

In other cases, the scaffold problem is addressed by grafting functional residues into helical bundles [5], [10]. This approach constrains the design space significantly by only considering a single, highly designable topology; because helices self-satisfy their internal backbone hydrogen-bonding donors and acceptors, a helical bundle can be stabilized primarily via hydrophobic packing, which is a far easier design task than satisfying buried polar groups [14]. This makes stable noncontiguous interactions much easier to design into helical bundles than folds which contain a variety of secondary structures. However, the palette of functions that are compatible with helical bundles is likely limited, and other approaches are needed to fully unlock the functional potential of designed proteins.

Finally, machine learning approaches have recently emerged as a new paradigm in computational protein engineering [15], [16]. Here, recent advances in protein structure prediction via deep learning can be leveraged to design protein structures that are predicted to be well-folded. In principle, functional constraints can be incorporated as a loss function measuring the difference between the predicted structure and some target geometry, the value of which can be used as a Metropolis criterion in a Monte-Carlo search through sequence space. Alternatively, when the predicted structure is close to satisfying the design goals, the differential of the loss function can be backpropagated all the way to the input sequence [17]. This approach has seen some success in creating stable structures [18], but only computational results exist for the design of function [19], and it is unclear whether machine learning models that learned to predict natural protein structures would perform well with completely *de novo* geometries.

Regardless of the approach, most computationally designed functional proteins fail to reach the affinities or catalytic rate enhancements seen in nature. One of the fundamental difficulties in the design of functions such as binding and catalysis is that they can require constellations of precise sidechain interactions with a ligand or binding partner, and these interactions must be facilitated by often-irregular backbone geometries. Indeed, success rates for the design of function are significantly lower than for the design of structure, yet functional proteins that are largely helical fare much better than the general pool of functional designed proteins [20]. This highlights the difficulty in stabilizing interactions that are incompatible with helical bundles, and many functions are likely to require combinations of secondary structure and irregular loop geometries to position key functional residues; therefore, methods for designing new backbone geometries that enable specific sidechain-level interactions are highly valuable.

1.2 New methods for designing functional proteins

In this dissertation, I outline two methods that address the problem of positioning functional residues: Pull Into Place (PIP) and Helical ELements Interface eXplorer (HELIX). PIP positions key residues by generating and optimizing the sequence of new local backbone geometries, utilizing fragment-based design and prediction algorithms to design new structures that are compatible with function. These geometries can consist of secondary structures or irregular loops, making PIP well-suited for optimizing functions that are less compatible with highly regular geometries. HELIX enables the creation of protein-binding proteins by focusing first on optimizing many local interactions via positioning of short helical fragments, and then searching for scaffolds that enable the positioning of several of these helices. This simultaneously addresses the problems of functional residue placement and scaffold selection. By focusing on enabling residue-level functional interactions, these methods address what is perhaps the most fundamental problem in the design of functional proteins. I believe this work will continue to provide utility to protein engineers, as well as a basis for progressing the field of computational protein design.

1.3 References

[1] H. Zhan, Z. Sun, and K. S. Matthews, “Functional Impact of Polar and Acidic Substitutions in the Lactose Repressor Hydrophobic Monomer·Monomer Interface with a Buried Lysine,” *Biochemistry*, vol. 48, no. 6, pp. 1305–1314, Feb. 2009, doi: 10.1021/bi801357f.

[2] A. Glasgow, H. T. Hobbs, Z. R. Perry, S. Marqusee, and T. Kortemme, “Ligand-induced changes in dynamics mediate long-range allostery in the <em>lac</em> repressor,” *bioRxiv*, p. 2021.11.30.470682, Jan. 2021, doi: 10.1101/2021.11.30.470682.

[3] L. Brocchieri and S. Karlin, “Protein length in eukaryotic and prokaryotic proteomes,” *Nucleic Acids Res.*, vol. 33, no. 10, pp. 3390–3400, 2005, doi: 10.1093/nar/gki615.

[4] P. Collaboration *et al.*, “Planck 2015 results. XIII. Cosmological parameters,” *Astron. Astrophys.*, vol. 594, p. A13, Sep. 2016, doi: 10.1051/0004-6361/201525830.

[5] N. F. Polizzi and W. F. DeGrado, “A defined structural unit enables de novo design of small-molecule–binding proteins,” *Science*, vol. 369, no. 6508, pp. 1227–1233, Sep. 2020, doi: 10.1126/science.abb8330.

[6] G. M. Bender *et al.*, “De Novo Design of a Single Chain Diphenylporphyrin Metalloprotein,” *J. Am. Chem. Soc.*, vol. 129, no. 35, pp. 10732–10740, Sep. 2007, doi: 10.1021/ja071199j.

[7] P. M. Murphy, J. M. Bolduc, J. L. Gallaher, B. L. Stoddard, and D. Baker, “Alteration of enzyme specificity by computational loop remodeling and design,” *Proc. Natl. Acad. Sci.*, vol. 106, no. 23, pp. 9215–9220, Jun. 2009, doi: 10.1073/pnas.0811070106.

[8] D. Röthlisberger *et al.*, “Kemp elimination catalysts by computational enzyme design,” *Nature*, vol. 453, no. 7192, pp. 190–195, May 2008, doi: 10.1038/nature06879.

[9] A. A. Glasgow *et al.*, “Computational design of a modular protein sense/response system,” *bioRxiv*, p. 648485, May 2019, doi: 10.1101/648485.

[10] R. A. Langan *et al.*, “De novo design of bioactive protein switches,” *Nature*, vol. 572, no. 7768, Art. no. 7768, Aug. 2019, doi: 10.1038/s41586-019-1432-8.

[11] A. Leaver-Fay *et al.*, “Chapter nineteen - Rosetta3: An Object-Oriented Software Suite for the Simulation and Design of Macromolecules,” *Methods Enzymol.*, vol. 487, pp. 545–574, Jan. 2011, doi: 10.1016/B978-0-12-381270-4.00019-6.

[12] R. Helling *et al.*, “The designability of protein structures,” *J. Mol. Graph. Model.*, vol. 19, no. 1, pp. 157–167, Feb. 2001, doi: 10.1016/S1093-3263(00)00137-6.

[13] F. Zheng, J. Zhang, and G. Grigoryan, “Tertiary structural propensities reveal fundamental sequence/structure relationships,” *Struct. Lond. Engl. 1993*, vol. 23, no. 5, pp. 961–971, May 2015, doi: 10.1016/j.str.2015.03.015.

[14] P. B. Stranges and B. Kuhlman, “A comparison of successful and failed protein interface designs highlights the challenges of designing buried hydrogen bonds,” *Protein Sci. Publ. Protein Soc.*, vol. 22, no. 1, pp. 74–82, Jan. 2013, doi: 10.1002/pro.2187.

[15] M. Baek *et al.*, “Accurate prediction of protein structures and interactions using a 3-track neural network,” *Science*, vol. 373, no. 6557, pp. 871–876, Aug. 2021, doi: 10.1126/science.abj8754.

[16] J. Jumper *et al.*, “Highly accurate protein structure prediction with AlphaFold,” *Nature*, vol. 596, no. 7873, Art. no. 7873, Aug. 2021, doi: 10.1038/s41586-021-03819-2.

[17] “Protein sequence design by conformational landscape optimization | PNAS.” https://www.pnas.org/doi/10.1073/pnas.2017228118 (accessed May 11, 2022).

[18] I. Anishchenko *et al.*, “De novo protein design by deep network hallucination,” *Nature*, vol. 600, no. 7889, Art. no. 7889, Dec. 2021, doi: 10.1038/s41586-021-04184-w.

[19] M. Jendrusch, J. O. Korbel, and S. K. Sadiq, “AlphaDesign: A de novo protein design framework based on AlphaFold,” *bioRxiv*, p. 2021.10.11.463937, Jan. 2021, doi: 10.1101/2021.10.11.463937.

[20] X. Pan and T. Kortemme, “Recent advances in de novo protein design: Principles, methods, and applications,” *J. Biol. Chem.*, vol. 296, p. 100558, Mar. 2021, doi: 10.1016/j.jbc.2021.100558.

Chapter 2

Accurate positioning of functional residues using robotics-inspired protein design

2.1 Abstract

Proteins achieve their complex functions, such as molecular recognition with high affinity and specificity, through intricate three-dimensional geometries in functional sites. To engineer new protein functions, accurate positioning of amino acid functional groups is therefore critical but has remained difficult to achieve by computational methods because of current limitations in the design of new conformations with arbitrary user-defined geometries. Here we introduce two computational methods capable of generating and predicting new local protein geometries: fragment kinematic closure (FKIC) and loophash kinematic closure (LHKIC). FKIC and LHKIC integrate two approaches: robotics-inspired kinematics of protein conformations and insertion of peptide fragments. We show that FKIC and LHKIC predict native-like conformations at atomic accuracy and with up to 140-fold improvements in sampling efficiency over previous approaches. We then integrate these methods into a new design protocol, pull-into-place (PIP), to position functionally important sidechains via design of new backbone conformations. We validate PIP by remodeling a sizeable active site region in an enzyme and confirming the engineered new conformations of two designs with crystal structures. The described methods can be applied broadly to the design of user-defined geometries for new protein functions.

2.2 Introduction

Advances in computational protein design [1], [2] promise to create new proteins to impact current and future challenges in biotechnology and medicine. Computationally designed proteins already enable important applications as modular sense/response systems to control precise biological responses [3]; as nanoparticles for potent protein vaccines[4]; and as protein therapeutics with minimal side effects [5]. However, while new “idealized” protein structures consistent primarily of regular secondary structure elements connected by short loops can now often be designed rather robustly[6], design of new functions remains more difficult [2], [7].

A key challenge lies in the difficulty of designing the fine-tuned protein geometries necessary for function with atomic accuracy. Many functions involve considerable deviations from the idealized highly stable *de novo* designed structures that are much easier to design [8], [9]. Further difficulties arise both from the small energy gaps between functional and non-functional conformations [10] and the formidable problem of sampling the enormous space of possible sequence/structure combinations [11]. Taken together, these issues complicate the accurate positioning of amino acid functional groups for many applications involving specific molecular recognition.

Accurate positioning of key amino acid side chain functional groups by computational design is particularly challenging in cases where the desired geometry cannot be achieved by simply placing new side chains on an existing or slightly modified backbone, but instead requires generation and design of substantially altered backbone conformations. Despite the importance of this capability for designing proteins with new user-defined functions, as well as prior work on local alterations of active sites [12], [13], this problem has remained generally unsolved.

Here we describe and experimentally validate a new approach for designing substantially altered protein conformations that accurately position user-defined functional groups in proteins, called Pull Into Place (PIP). The PIP protocol has three steps: (i) generation of new backbone conformations, where functional groups of interest are gently pulled towards their desired positions using harmonic restraints, (ii) sequence design using fixed-backbone side-chain optimizations with the same restraints, and (iii) structure prediction using unrestrained flexible-backbone simulations to identify designs predicted to adopt the desired new backbone conformation. We demonstrate that PIP is capable of accurately placing side chains and designing the required considerable alterations of the protein backbone by solving crystal structures of two designs. Detailed characterization of one successful design reveals a robustness to mutation, suggesting that multiple interactions contribute to the conformation of the remodeled region. The design methods described here advance the engineering of new proteins by allowing the accurate positioning of functional groups critical for many aspects of protein function, such as specific recognition of binding partners.

2.3 Results

We set out to develop a method (PIP) to accurately position amino acid functional groups in proteins by designing new local backbone geometries. The PIP algorithm required 3 components (**Fig. 2.1**): (1) a method to generate designable backbone conformations that could precisely position defined functional groups, (2) a way to stabilize these new backbone and side chain conformations by finding sequences optimal for the desired structure, and (3) a method to predict the new conformation given a sequence, to assess whether the desired structure is also optimal for the designed sequence.

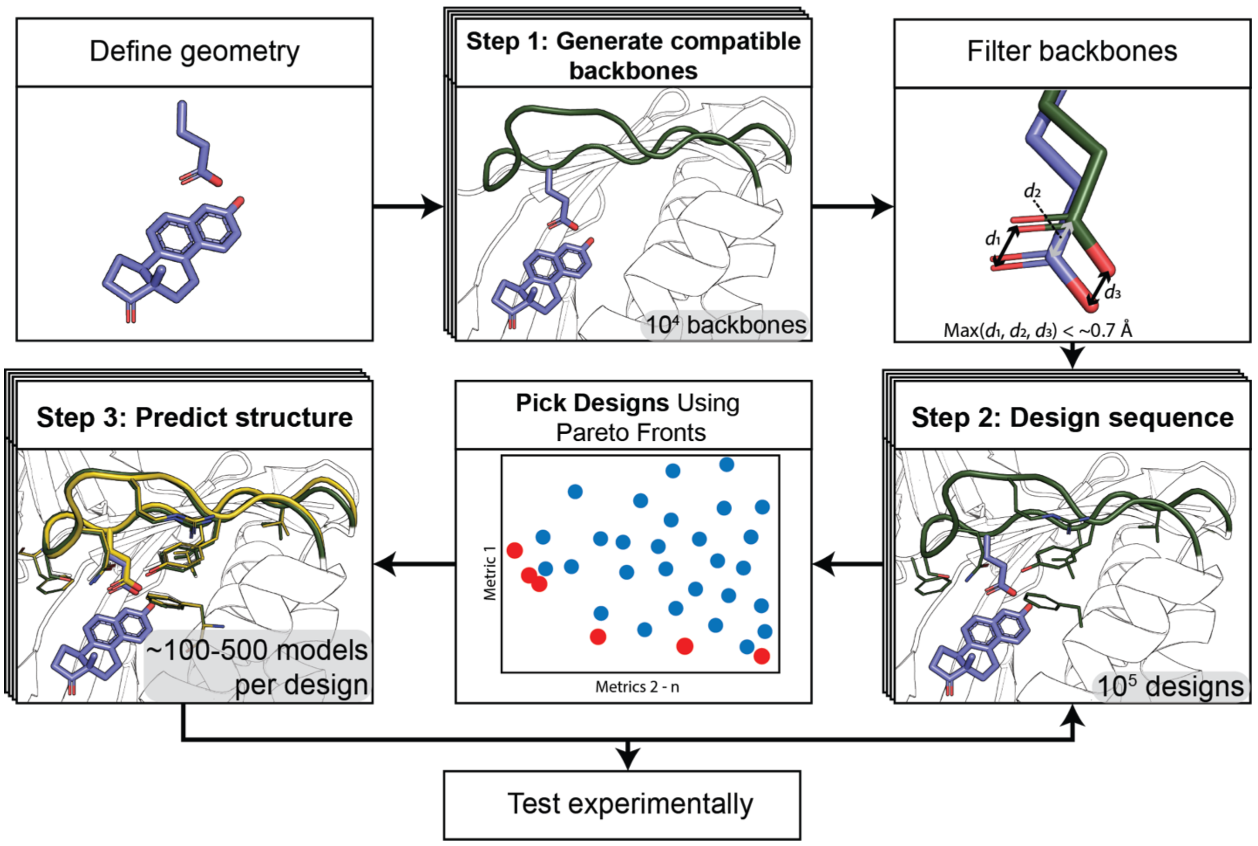

Define geometry
Step 1: Generate compatible backbones
10^4 backbones
Filter backbones
d_2
d_1
d_3
Max(d_1, d_2, d_3) < ~0.7 Å
Step 3: Predict structure
~100-500 models per design
Pick Designs Using Pareto Fronts
Metric 1
Metrics 2 - n
Step 2: Design sequence
10^5 designs
Test experimentally

Figure 2.1. Steps of the PIP protocol. Top left: functional geometry is defined. Top middle: new backbone conformations (green) are generated to satisfy the geometric restraints. Top right: Backbones are filtered based on their ability to satisfy the geometric restraints. d1, d2, and d3 refer to the distances of the atoms in a positioned carboxyl group to their defined ideal positions. Bottom right: Sequences are designed to stabilize the *de novo* backbone. Bottom middle: Designs are selected based on multiple computational quality metrics using Pareto fronts (**Methods**). Red: Pareto-efficient designs; blue: other designs. Bottom left: For selected sequences, Rosetta structure prediction method are applied to predict the lowest-energy structure (yellow). Illustrations use the KSI model system detailed in **Fig. 2.3**.

We first describe two improved computational methods, fragment-kinematic closure (FKIC) and loophash-kinematic closure (LHKIC), to generate new backbone conformations (step 1) and to predict their structures accurately given designed sequences (step 3). These methods are particularly suited to problems where (i) target structures do not exclusively adopt regular secondary structure geometries, (ii) there are no protein homologs that can be used as templates for modeling, and (iii) there are no multiple sequence alignments to guide current deep learning structure prediction methods [14], [15], since we aim to design new structures and sequences. We then describe the application of the entire PIP protocol in the program Rosetta to a design problem in which we reshape the backbone geometry of a model protein, ketosteroid isomerase (KSI), to replace a functional aspartate with a glutamate residue (not found in any KSI homologs) such that the carboxyl groups align. We chose this design problem as a proof-of-concept because it presents a particularly challenging positioning problem that cannot be solved with (near-)fixed-backbone design, for which no solution was known in a homologous protein, and that requires accuracy on the length scale of a carbon-carbon bond.

2.3.1 FKIC and LHKIC algorithms

FKIC and LHKIC integrate two concepts that have separately led to considerable advances in protein modeling: sampling preferred combinations of backbone torsions from fragments of proteins in the protein structure databank (PDB) [16], and improved sampling of segments without regular secondary structure or template information with an inverse kinematic closure algorithm termed 'KIC' [17] borrowed from the field of robotics [18]. KIC determines 'mechanically accessible' conformations for internal protein segments of given lengths by sampling the phi/psi torsion degrees of freedom in the segment. In each KIC move, three Cα atoms of an N-residue segment are designated as pivots, leaving N-3 non-pivot Cα atoms. In the standard implementation of KIC in Rosetta [17], non-pivot torsions are sampled from a residue type-specific Ramachandran map. In the new FKIC method (see **Methods** for details), non-pivot degrees of freedom are taken from peptide fragments that are picked from the PDB using the sequence of the target segment [19] (**Fig. 2.2a**, **left**); KIC is then used to determine the values of the pivot torsions that close the resulting chain break. We reasoned that FKIC would combine the

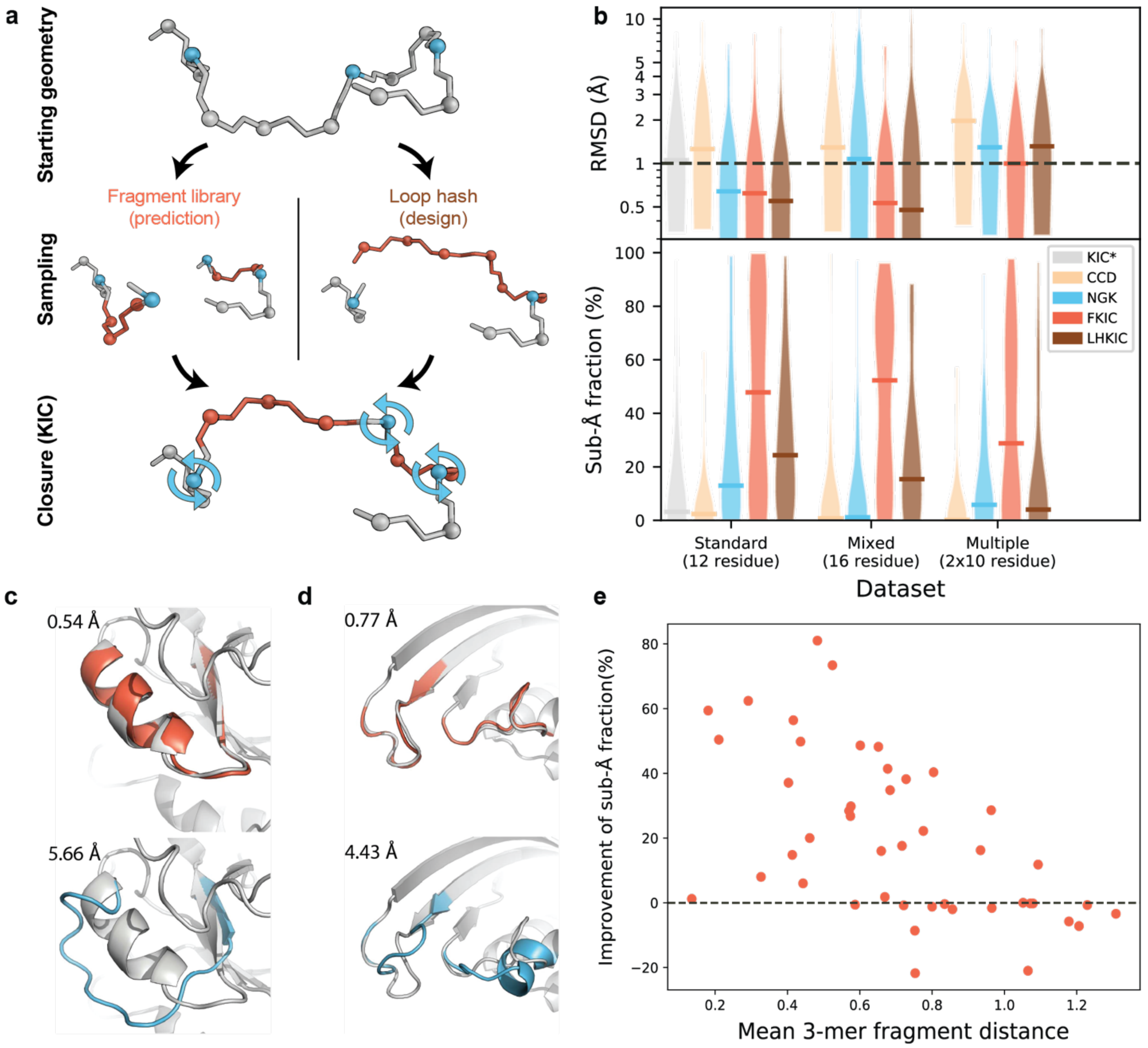


Figure 2.2. FKIC improves prediction of conformations of local backbone segments. (a) Individual FKIC/LHKIC move. Three Cα atoms (blue) on the target segment to be modeled (grey) are picked randomly as pivots. Fragment insertion (FKIC) or loop hash (LHKIC) is applied to sample torsion degrees of freedom at non-pivot atoms (red), which breaks the chain. The KIC algorithm is then used to close the chain by determining appropriate values for the pivot torsions. **(b)** Comparison of performance of different methods for three datasets: (i) Standard dataset described in ref. [17], and 2 new sets: (ii) A "Mixed Segment" dataset with 30 16-residue regions that contain both loops and segments of regular secondary structure and (iii) a "Multiple Segments" dataset of 30 cases with 2 separate 10-residue regions that are interacting. KIC [17]: grey; CCD [20]: orange; NGK [21]: blue; FKIC: red; LHKIC: brown. Upper panel: violin plot of RMSD of lowest energy (best) model across each dataset. Horizontal bars indicate the median lowest-energy RMSD. FKIC is the only method that provides predictions with atomic accuracy (≤ 1Å median RMSD) for all datasets. Lower panel: violin plot of fraction of predicted models in each dataset that have sub-Å accuracy. FKIC leads to considerable improvements

over previous methods. Asterisk indicates data from ref. [42]; all other simulations were run with the ref2015 Rosetta energy function [21]; methods using fragments (CCD and FKIC) used identical fragment libraries that excluded fragments from structural homologs to the target proteins. **(c,d)** FKIC accurately predicts geometries from sequence where the previous state-of-the-art method, NGK, fails. Shown are examples from the Mixed Segment **(c)** and Multiple Segments dataset **(d)**. Experimentally determined structures: grey; predictions from FKIC: red, top; predictions from NGK: blue, bottom. RMSDs to the experimentally determined structures are given in each panel in Å. **(e)** The fraction of sub-Å predictions is negatively correlated with the mean 3-mer fragment distance (**Methods**). Each data point represents a protein from the standard 12-residue dataset.

improved prediction accuracy of KIC demonstrated previously [17] with improved sampling efficiency because of the reduction of degrees of freedom by using coupled torsion angles from fragments (in contrast to sampling all non-pivot torsions independently from Ramachandran space as in KIC, which is unlikely for example to sample regular secondary structures).

To mimic a design case where the sequence of the modeled segment is not known a priori, we also developed a variation on the method, LHKIC, which uses the loophash protocol [20] to pick fragments that simultaneously sample structures and sequences of the target segments. The loophash protocol uses the 6D transformation between the residue before the first pivot and the residue after the last pivot as a query key to find peptide fragments from the PDB that approximately close the gap between these two residues (**Fig. 2.2a**, **right**). After insertion of a fragment, KIC determines the pivot torsions that close the gap. For design cases, LHKIC can optionally mutate remodeled residues to the amino acids from the inserted fragment to improve local sequence-structure compatibility. Individual FKIC or LHKIC sampling moves (**Fig. 2.2a**) are then followed by optimization of side chain conformations in and around the altered backbone region and integrated into a Monte Carlo minimization protocol (**Fig. 2.5**); sampled conformations are evaluated with Rosetta's all-atom energy function [21], [22].

2.3.2 Local structure prediction performance

We tested the ability of FKIC to recapitulate the local conformations of protein segments, given their sequences, on three benchmark sets. The first is a benchmark of 45 12-residue loops [23] previously used to evaluate KIC[17] ("Standard" set), to enable comparisons with published work. We also used two new sets representing more challenging problems closer to design applications: a new set of 30 16-residue-segments where each segment contains both regular secondary structure elements and loop regions ("Mixed Segment" set), and a new set of 30 pairs of interacting 10-residue segments ("Multiple Segments" set). As controls, we applied methods that use KIC and fragment insertion (CCD) [24], [25] alone to the same datasets using an otherwise identical protocol in Rosetta. We used two performance metrics: The first quantifies prediction accuracy by determining the RMSD of the model with the lowest (best) predicted Rosetta energy to each native structure and then taking the median RMSD value across each dataset. The second metric quantifies sampling efficiency by measuring the fraction of native-like (correct) models generated for each protein case, where native-like is defined as <1Å ("sub-Å") RMSD to the native structure, and again taking the median for each dataset (**Methods** and **Table 2.2a**).

The Rosetta KIC method had previously been shown [17] to be comparable to a state-of-the-art molecular mechanics method[23]. The next-generation KIC (NGK) update [26] led to improved performance over KIC, and had comparable performance to GalaxyLoop-PS2 [27], RCD+ [28], Sphinx [29], LEAP [30] and FREAD [31], [32] when tested on identical datasets. Here we show that FKIC improves structure prediction accuracy over CCD, KIC and NGK with the largest changes for the two new datasets (**Fig. 2.2b**, **top** and **Table 2.2a**). On the 16-residue Mixed

Segment dataset, which tests the ability of FKIC to predict conformations of protein segments with arbitrary secondary structure composition, the median accuracy improved to 0.53Å RMSD with FKIC compared to 1.29Å and 1.07Å with CCD and NGK alone, respectively. For the Multiple Segments dataset, which tests the ability of FKIC to predict conformations of discontinuous interacting segments, FKIC was the only method that yielded atomic (1Å) median accuracy, compared to 1.97Å and 1.29Å with CCD and NGK alone, respectively (**Fig. 2.2b, top** and **Table 2.2a**). Representative examples where FKIC correctly predicted protein conformations while NGK failed are shown for the Mixed Segment and Multiple Segments datasets in **Figs. 2.2c,d** and details are given in **Tables 2.3** and **2.4**. The improvements on the Standard dataset were smaller (median RMSD was 0.62Å with FKIC compared to 0.64Å for NGK, **Table 2.2a**), but for 35/45 proteins FKIC finds lower energy structures than NGK (**Table 2.5**). Cases where FKIC predictions did not lead to the identification of sub-Å accuracy lowest-scoring models can be attributed to both sampling and energy function limitations (**Table 2.6, Fig. 2.6**, and **Appendix II: Supplementary Note 1**).

FKIC also considerably improved sampling efficiency, which we quantified by how frequently FKIC generated conformations that are <1Å RMSD from the crystallographic conformation (**Fig. 2.2b**, **bottom**). For the Mixed Segment set, the median fraction of sub-Å predictions for FKIC was 52.3%, which was 45- and 105-fold higher than for NGK and CCD, respectively. For the Multiple Segments dataset, the median fraction of sub-Å predictions was 28.5% with FKIC, which was 5-fold higher than with NGK (5.5%) and 143-fold higher (0.2%) than with CCD (**Table 2.2a**). In several cases, FKIC was able to find correct solutions for even larger conformational sampling problems such as a set with 2 interacting 12-residue segments

(**Appendix II: Supplementary Note 2** and **Tables 2.7-8**). These improvements in sampling efficiency are important in particular for design, since they reduce the computational time needed to predict the conformation of a reshaped backbone segment, allowing for more designs to be evaluated.

We also tested the ability of LHKIC to predict local protein conformations on the three benchmark sets. LHKIC performed similarly to FKIC in terms of RMSD (**Fig. 2.2b, top**, **Table 2.2a**, and **Appendix II: Supplementary Note 3**). However, in this structure prediction task LHKIC sampling efficiency was lower than for FKIC (**Fig. 2.2b, bottom**), since LHKIC does not use information on the target sequence for picking fragments. LHKIC is therefore intended for design applications where sequence and structure are sampled simultaneously rather than for structure prediction tasks where the sequence is known and fixed.

Overall, the improvement of the fraction of sub-Å predictions is negatively correlated with the mean 3-mer fragment distance from the native structure (**Fig. 2.2e**, **Methods**). This observation shows that high quality fragments focus the sampling on native-like conformations. While both CCD and FKIC sample from the same fragment set, FKIC performs considerably better (**Fig. 2.2b**). This difference between the two fragment-based structure prediction methods could at least partly be attributed to the fact that when CCD closes a chain break, it modifies all torsions along the inserted fragment, while KIC maintains more conformational information from the inserted fragment by only modifying the three pivot residues. While high quality fragments could be derived from homologous structures, for both CCD and FKIC benchmark simulations we excluded fragments from homologs to test the ability to predict structures of regions for which there are no templates. However, we also repeated our simulations with fragments from homologs present in the database. As expected, both prediction accuracy and the median fraction of sub-Å predictions improved further when homologous structures are included in FKIC simulations (**Table 2.2b**).

2.3.3 Application of the PIP protocol

With improved methods for sampling and prediction of backbone conformations in hand, we set out to test the entire PIP protocol (**Fig. 2.1**) in a design application. We chose *Pseudomonas testosteroni* ketosteroid isomerase (KSI) as a model system (**Fig. 2.3a**). KSI uses a catalytic aspartate at position 38 to abstract a proton from a steroid substrate to catalyze an energetically favorable double-bond rearrangement. Here we set out to replace aspartate 38 with glutamate while maintaining the precise placement of the side chain carboxyl group (**Fig. 2.3a**) by reshaping a sizable region of the protein backbone (11-12 residues, **Fig. 2.3b**). To test our designs before solving atomic-resolution structures, we reasoned that KSI activity provides a convenient way to estimate the accuracy of functional group positioning, because KSI activity is sensitive to perturbations of the functional site geometry on the length scale of a carbon-carbon bond: With 5(10)-estrene-3,17-dione as a substrate, mutating aspartate 38 in KSI to a glutamate reduces the protein's k_{cat} by approximately 103-fold (**Table 2.1**) (this value is similar to previous work that reported a reduction of 240-fold in the D38E mutant compared wild-type [33]). This reduction in k_{cat} is attributed to the misplacement of the side chain carboxyl group that is common to glutamate and aspartate due to the additional methylene group in the glutamate sidechain. We note that the PIP design protocol is not geared towards optimizing catalytic activity, as the protocol does not specifically consider requirements of catalysis other than positioning of functional groups. However, enzyme activity is still a useful proxy to probe for

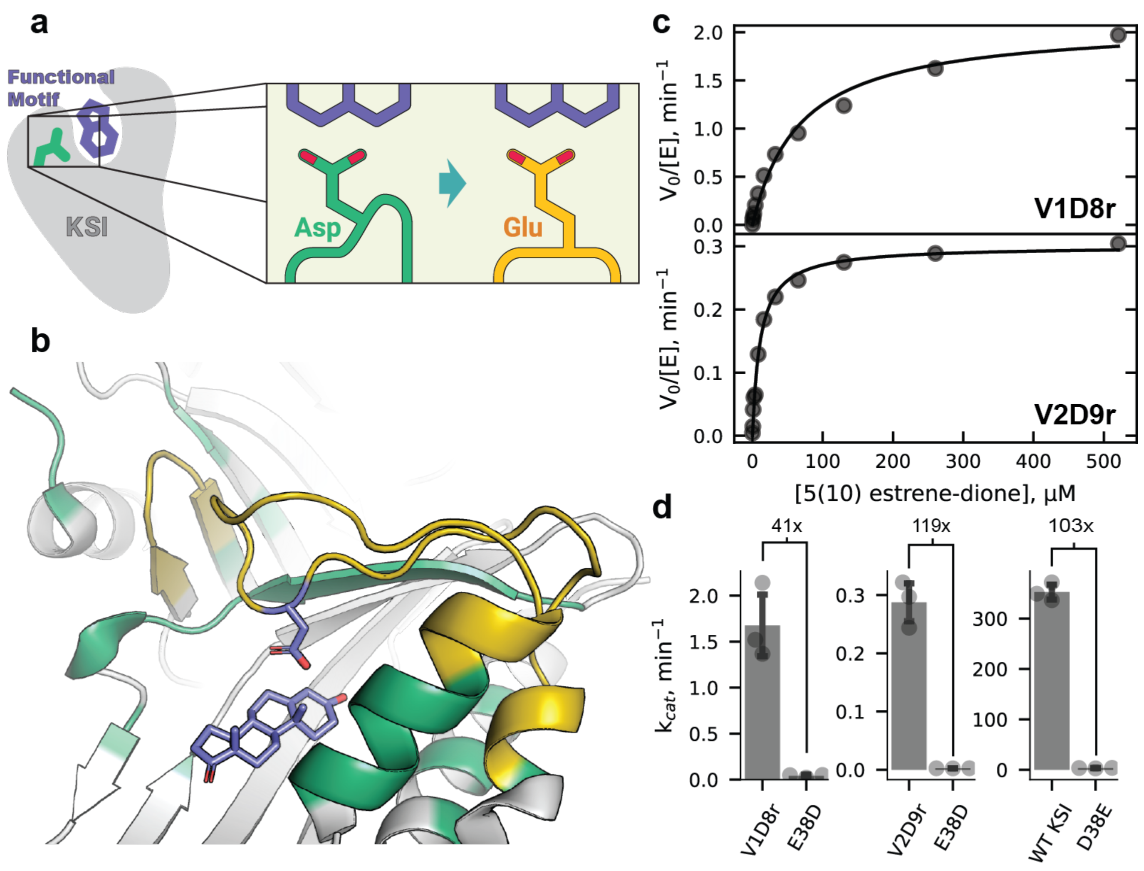


Figure 2.3. Functional characterization of designs V1D8r and V2D9r. (a) Schematic of design goal for KSI. Green: wild-type KSI with catalytic aspartate. Yellow: Designed KSI variant with reshaped active site to position the glutamate carboxyl group in place of the wild-type aspartate carboxyl group. **(b)** KSI wild-type structure (PDB 1QJG), showing the active site regions to be remodeled. Residues allowed to change identity (design) or conformation (repack) during the design process (PIP version 2) are shown in yellow or green, respectively, and static positions are shown in grey. **(c)** Representative Michaelis-Menten curves for design V1D8r (top) or V2D9r (bottom). **(d)** Bar plots showing the k_{cat} values of V1D8r (top), V2D9r (middle), or wild-type KSI (bottom) and their E38D or D38E active site mutations. Values show the fold-change in k_{cat} between the respective D/E active-site residue pairs. Standard deviation of independent triplicate experiments are shown as error bars with individual measurements shown as points.

accurate positioning when comparing aspartate to glutamate. Moreover, no known homologs of KSI contain a glutamate at the catalytic position [34]. Thus, any designed solutions would be novel, and a fragment-based design protocol would not be able to rely on naturally occurring homologs that have already solved this particular problem.

Our PIP design protocol for KSI (**Fig. 2.1**) proceeded in three steps: In step 1, we built 20,000 *de novo* backbone conformations that positioned the functional carboxyl group using harmonic coordinate restraints defined by the amide atoms of asparagine 38 (an inactivating mutation for the catalytic D38 that enables a transition state mimic to be crystallized) in PDB file 1QJG in place of the catalytic D38. We selected the 1,600 or 4,000 conformations(numbers are for two rounds of the protocol, see **Methods**) that best matched the desired geometry based on their restraint satisfaction, which we defined as the maximum distance of any restrained atom in the model to the atom's ideal position (**Fig. 2.1**, top right panel).

In the second step, these new backbone conformations were stabilized by redesigning the local environment, where all residues of the new backbone segments as well as residues in the environment were redesigned using design methods in Rosetta (see **Appendix I: Supplementary Methods**). This process resulted in 10-50 designs per input structure. We then selected 200 or 422 design models for structure prediction in step 3. These designs were selected based on how close the modeled catalytic residue carboxyl group atoms were to their desired positions, and several computational design quality metrics including Rosetta score terms, hydrogen bond satisfaction, and metrics for sequence-structure compatibility (see **Methods** for details).

While step 2 (design) aims to find sequences that are optimal for the targeted new conformations, step 3 (structure prediction) aims to assess whether these sequences indeed fold into the targeted conformation (i.e. is the conformation also optimal given the sequence). Steps 2 and 3 were iterated to further optimize sequence-structure combinations. In particular, designed sequences that produced structure prediction models that correctly placed the functional carboxyl group but were not the lowest-scoring model generated by the structure prediction protocol were fed back to step 2 for further sequence optimization.

2.3.4 Selection of designed KSI variants

We created designs using two versions of the PIP protocol, denoted versions 1 and 2 (see **Methods** and **Supplementary Methods** for details regarding differences in implementation of the PIP steps). In total, 33-39 and 29-30 residue positions were designed (allowed to change amino acid residue) in versions 1 and 2, respectively. We selected 32 designs for experimental testing, 14 from version 1 and 22 from version 2. Designs were named according to the version of PIP used to create them (V1 and V2), a design number (D1, D2, ...), and an appended "r" to indicate if any mutations were reverted to the wild-type residue based on visual inspection (for details see **Tables 2.9-10, Figs. 2.7-8**). We chose designs that maximized the gap in Rosetta score between models that correctly place the catalytic residue (<1 Å restraint satisfaction, defined as the maximum distance between a restrained atom and its defined position) and models which do not correctly position the catalytic residue (>2 Å restraint satisfaction). We also chose designs that were predicted to have few buried unsatisfied hydrogen bond donors or acceptors, and that did not have significant sequence and structural similarity to other selected designs. Selected designs contained between 12 (V2D6r, V2D9r) and 32 (V1D7) mutations. For PIP version 1, all selected designs expressed in the insoluble fraction after cell lysis and had to be purified from inclusion bodies, as is common with KSI mutants[35]. Of the designs purified from inclusion bodies, half were soluble after refolding. We selected one design to characterize in further detail based on an initial screen of catalytic activity (**Table 2.11**), V1D8r. For version 2 we obtained one design that expressed in the soluble fraction, V2D9r. Both designs V1D8r and V2D9r were stable after purification as assessed by Circular Dichroism spectroscopy (**Fig. 2.9**).

2.3.5 Functional characterization of designed KSI variants

Both designs V1D8r and V2D9r showed robustly measurable enzymatic activity when using 5(10)-estrene-3,17-dione as a substrate (**Fig. 2.3c**), enhancing catalysis by 4 to 5 orders of magnitude when compared to the water-catalyzed isomerization of the similar 5-androstene-3,17-dione [36]. To test for the ability of PIP to accurately position functional groups, we reverted the glutamate in the designs back to the original wild-type aspartate. Because of the sensitivity to functional group positioning observed in wild-type when adding a methylene group going from aspartate to glutamate, and if we indeed correctly positioned the new glutamate in the design, we expected a considerable drop in catalytic activity in the design upon subtracting the methylene group again. As predicted by this model, for both designs V1D8r and V2D9r, we found a substantial reduction in k_{cat} in the E38D reversion mutant; the activities of both E38D mutants were near the detection limit of the assay, and were reduced compared to the designs with E38 by at least 41-fold and 119-fold for V1D8r and V2D9r, respectively (**Fig. 2.3d**). This reduction was not simply due to loss of protein stability as both E38D reversion mutants in the design background were folded (**Fig. 2.9b-d**). Notably, these fold changes are similar to the 103-fold-change in k_{cat} between wild-type KSI and the D38E mutation (**Fig. 2.3d**, **Table 2.1**). Taken together, these results suggest that the designed backbone geometries successfully altered the enzyme's preference for its catalytic residue. We note that the designs were overall less active than both wild-type and D38E KSI (**Table 2.1**). There are many potential reasons for this finding (**Supplementary Note 4**), including the observation that the designs are monomeric at the concentrations of the enzyme assay, whereas wild-type KSI functions as a dimer (**Supplementary Fig. 5e**). Additionally, our designs contain a large number of mutations (19 and 12 for V1D8r and V2D9r, respectively) that could affect active site electrostatics important for catalysis[37]. Predicting the energetics of polar interactions making up protein functional sites with sufficient accuracy is a formidable problem, and we note that PIP (like other computational design methods) does not consider possible requirements of catalysis other than positioning (see **Discussion**). However, our analysis suggests that the positioning of the catalytic residue's carboxyl moiety, which PIP optimized for, is still an important determinant of catalytic activity. In particular, the design V2D9r has approximately the same fold reduction when changing glutamate to aspartate as wild-type KSI when changing aspartate to glutamate.

2.3.6 Structural characterization of designed KSI variants

To assess whether V1D8r and V2D9r indeed adopted the designed new backbone conformations, we determined crystal structures of the two designs V1D8r and V2D9r. Both structures contained a ligand in the active site. For V1D8r, we observed density from deoxycholate retained from the purification process. V2D9r was co-crystallized with equilenin (which was present in the structure that was used as a basis for design) but also contained some residual density for deoxycholate (see further below and **Methods**). For both designs the electron density of the reshaped backbone region (residues 34-45 for V1D8r and 34-46 for V2D9r) was well-resolved (**Fig. 2.10a-b**). Importantly, the backbone geometries of the reshaped backbone region in V1D8r and V2D9r were within 1.39 and 1.15 Å RMSD (N, C, Cα, and O backbone atoms)

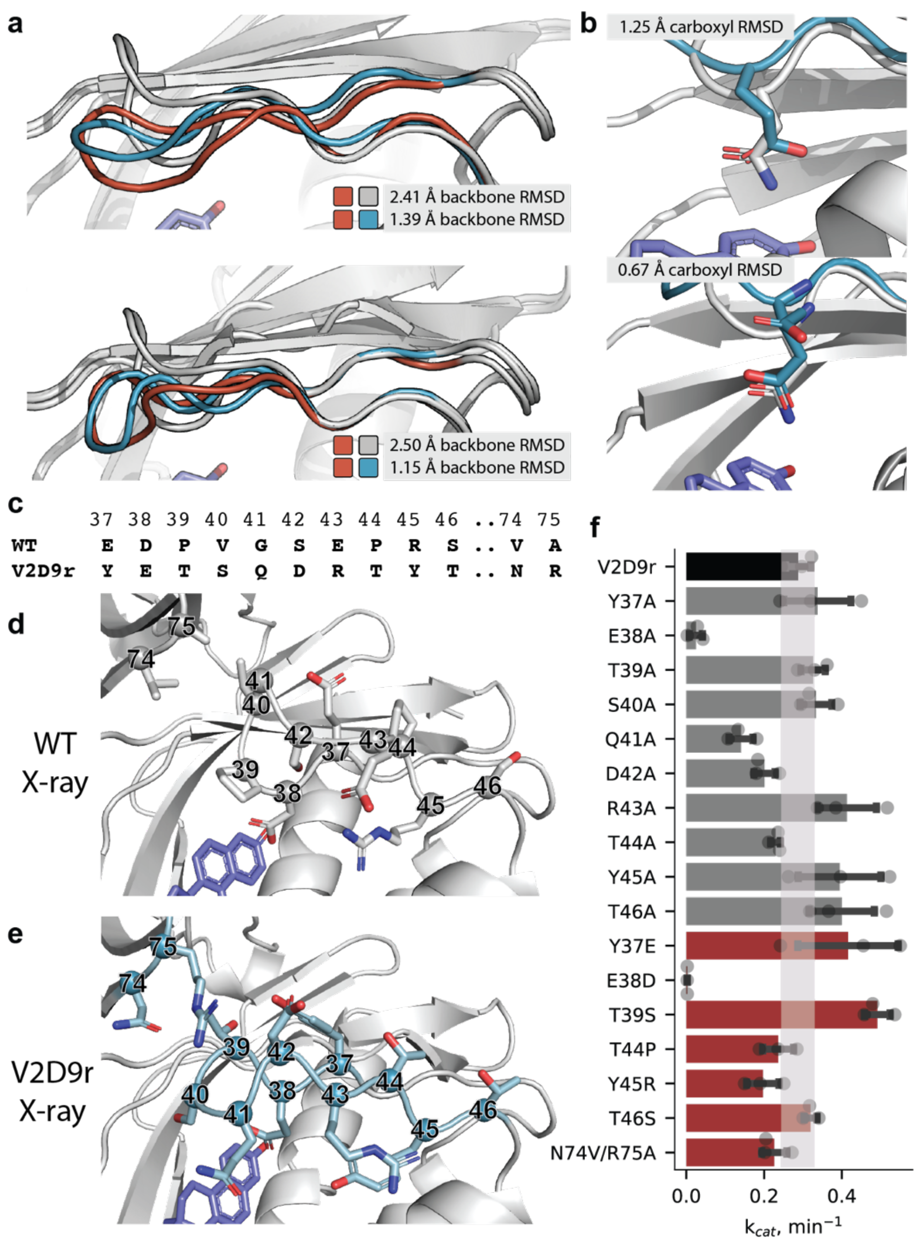


Figure 2.4. Structural characterization of designs V1D8r and V2D9r. (a) Overlay of wildtype KSI crystal structure (grey), lowest-energy predicted models for V1D8r (orange, top) and V2D9r (orange, bottom), and crystal structures for V1D8r (blue, top) and V2D9r (blue, bottom). **(b)** Crystal structure (blue) of V1D8r (top) and V2D9r (bottom) showing the catalytic glutamate's placement relative to the amide in the KSI starting structure (PDB 1QJG) used to define the catalytic position (grey). RMSD values between compared structures are indicated in the different panels. (**c-f**) Mutational analysis of differences between wildtype KSI and design V2D9r: sequence alignment (**c**), comparison between the active site region in the crystal structures of wild-type KSI (**d**) and in design V2D9r (**e**), and **(f)** bar graph of k_{cat} values for design V2D9r (black), alanine scan mutants (grey), and reversion / selected mutants (red). In (**f**), standard deviation of independent triplicate experiments are shown as error bars with individual measurements shown as points. The k_{cat} error range for V2D9r is shown as a shaded bar.

Table 2.1. Kinetic parameters of wild-type (WT) KSI, WT D38E, and designs. Ranges are based on the standard deviation of three independent experiments.

Enzyme	k_{cat} (min^{-1})	K_M (μM)	k_{cat}/K_M ($μM^{-1}$ min^{-1})
WT	350 ± 18	120 ± 32	2.9 ± 0.79
WT D38E	3.4 ± 0.50	37 ± 4.9	0.092 ± 0.018
V1D8r	1.7 ± 0.41	67 ± 15	0.025 ± 0.0084
V2D9r	0.29 ± 0.0040	9.0 ± 2.0	0.032 ± 0.0084

of the corresponding lowest-energy design models (**Fig. 2.4a**). For comparison, both the design structures and the computational models had conformations considerably different from the wild-type backbone (**Fig. 2.4a**). In the reshaped region, the design model of V1D8r and V2D9r differed from wild-type by 2.41 Å and 2.50 Å backbone RMSD, respectively. If considering the most variable segment (residues 37-42 for V1D8r and 37-43 for V2D9r), the design models for V1D8r and V2D9r differed from wild-type by 3.49 and 3.34 Å RMSD respectively.

Next, we examined sidechain positioning, especially the catalytic glutamate carboxyl group. V1D8r and V2D9r (which was co-crystallized with equilenin) placed the catalytic carboxyl within 1.25 and 0.7 Å RMSD of the wild-type aspartate carboxyl, respectively (**Fig. 2.4b).** The overall heavy-atom RMSDs for buried designed residues in the reshaped segment (solvent-accessible surface area (SASA) of less than 40 Å^2) were 1.43 and 1.07 Å for V1D8r and V2D9r, respectively (**Fig. 2.10c-d)**. As noted above, the crystal structures of both designs showed at least partial occupancy of deoxycholate in the ligand-binding site, and it is conceivable that the bulky carboxyl moiety of the ligand changed the placement of the catalytic carboxyl group. This hypothesis is supported by the observation that the V2D9r crystal had partial occupancy of deoxycholate in 3 out of 4 asymmetric units, and the positioning of the catalytic residue in those monomers was significantly worse than in the asymmetric unit that contained only equilenin. In the asymmetric unit which contained only equilenin, we observed two distinct possibilities for the placement of the carboxyl of E38, which we modeled as alternate conformations (**Fig. 2.10e)**. Despite the apparent flexibility of E38, the crystallographic data support the conclusion that the designed backbone is indeed capable of supporting the desired functional site geometry, as one of the alternate conformations is close to the wild-type carboxyl placement (0.75 Å restraint satisfaction, 0.67 Å carboxyl heavy-atom RMSD compared to the amide group of 38N of 1QJG, **Fig. 2.4c**). Taken together, the structural analysis shows that PIP can design novel backbone conformations that differ by over 3 Å from their native counterparts with high accuracy, and the functional analysis demonstrates successful switching of the specificity in the designs from aspartate to glutamate.

Finally, we tested the robustness of V2D9r's redesigned backbone segment to mutation. To determine whether the activity of V2D9r was dependent on any particular residue in the redesigned region (**Fig. 2.4c-e**), we performed an experimental alanine scan along all mutated residues. We also made reversion mutants for residues whose backbone atoms did not move significantly between the wild-type and the design conformations, as well as a T39S mutant that we hypothesized might alleviate problems with buried polar groups. No mutation affected the k_{cat} more than two-fold except for the catalytic glutamate (**Fig. 2.4f, Appendix II: Supplementary Note 5**), suggesting that the designed new backbone conformation depends on several interactions to adopt a catalytically competent conformation, as well as the glutamate as a general base.

2.4 Discussion

We introduced and validated methods to accurately position amino acid functional groups in proteins by computational design in cases that require substantial alterations of the protein backbone (**Fig. 2.1**). We first developed and benchmarked two new robotics-inspired sampling methods, FKIC and LHKIC, that generate and predict the structures of new backbone conformations with high accuracy (**Fig. 2.2**). We then integrated these methods into a new design protocol, PIP, to accurately position sidechain functional groups by remodeling the backbone (**Fig. 2.1**), and validated the approach experimentally by functional analysis and solving crystal structures of designs with reshaped backbone regions (**Fig. 2.3**, **2.4**).

FKIC leads to considerable improvements over the two approaches it combines, the fragment-independent loop modeling method NGK [26] and the fragment-insertion based prediction approach CCD [24] (**Fig. 2.2b**). In addition to sub-Å structure prediction accuracy, our results demonstrate that FKIC provides up to ~140- fold improvement in sampling native-like conformations on the challenging problems of modeling local protein conformations with multiple segments and arbitrary secondary structure composition. This key advance in sampling efficiency paves the way to use FKIC in combination with LHKIC to design new local backbone geometries not seen in nature. Our results provide a first proof-of-concept for such a design application.

We note FKIC and LHKIC are conceived for generation, design and prediction of new local backbone conformations and not for homology modeling that may require additional non-local changes in protein structure. Therefore, applications of FKIC to homology modeling of naturally occurring proteins may require integration of FKIC with more aggressive remodeling in the entire protein, not just a local region (**Appendix II: Supplementary Note 6**, **Table 2.12**). It will be interesting in the future to test whether deep learning methods for protein structure predictions [14], [15] could be used to predict structures of designed sequences more rapidly than the robotics-methods assessed here, while also achieving sub-Angstrom accuracy. To our knowledge there are not yet systematic studies benchmarking the accuracy of deep learning methods on protein regions with irregular structures in the absence of multiple sequence alignments and structures of homologous proteins, as will be the case when designing conformations not seen in nature.

Despite the success with positioning a functional group that required reshaping of a sizeable backbone region, our results also highlight the considerable challenges faced when designing functional proteins. PIP in its current implementation optimizes positioning and is hence more suitable to designing specific geometries for binding rather than catalysis (which may require consideration of other determinants of catalysis not considered in current computational design methods and sometimes not even fully known). Moreover, certain functions may require switching between two or more approximately isoenergetic conformations. Such a scenario is much more challenging to engineer than optimizing for one deep energy minimum, which is sufficient for successful *de novo* design of protein structures. While the achieved carboxylate positioning in our designs is encouraging, it is not perfect, and accurately estimating the relative free energies of different alternative conformations in proteins is a current challenge common to all state-of-the-art atomistic modeling methods.

Kinetic analysis of V2D9r failed to reveal any specific residues that were key to stabilizing the catalytically competent loop conformation (**Fig. 2.4**, **Appendix II: Supplementary Note 5**), highlighting an important challenge in the design and modeling of precise local protein conformations: The energetic contributions to the stabilization of a particular backbone geometry may be distributed among many residues, which, combined with enormous sequence and conformational landscapes, makes it difficult to arrive at a minimum via successive single-residue substitution. Efficient sequence and conformational sampling are therefore crucial to the design of functional geometries. LHKIC addresses this challenge by pairing the structure search with sequence information from natural proteins, favoring local sequence/structure compatibility.

Naturally occurring proteins are often only marginally stable, so when reengineering them for new functions it is often challenging to maintain stability. One approach to avoid this problem is to start with highly stable entirely *de novo* designed proteins into which to build desired structural features [38]. However, the idealized geometries of current *de novo* proteins may not be optimal for specific new functions such as molecular recognition. Here, FKIC/LHKIC and other methods[39] could provide a new way to systematically reshape local regions to endow *de novo* designed proteins with new functions. The ability to sample both conformational and sequence space afforded by the robotics-inspired approaches and protocols presented here should help address current limitations, and be useful in both the design and modeling of novel backbone conformations that enable specific functional geometries for binding or conformational switching [40] in *de novo* designed proteins.

2.5 Methods

2.5.1 Structure prediction simulations

Fragment-sampled KIC (FKIC) Overview

FKIC is based on the KIC protocol [17], but, instead of sampling non-pivot ϕ/ψ torsions probabilistically from Ramachandran space, FKIC uses coupled ϕ/ψ/ω degrees of freedom from consecutive residues of protein fragments of size nine, three or one to sample conformational space. During the low- and high-resolution sampling stages (**Fig. 2.5**), each KIC move in the original KIC protocol is replaced by an FKIC move (**Fig. 2.2a**). An FKIC move consists of the following sequence of steps: (i) a fragment library (see "Generation of fragment libraries" section in **Appendix I: Supplementary Methods**) is chosen at random from all available libraries (i.e. 9mers, 3mers and 1mers), (ii) the chosen fragment library is searched for fragment alignment frames that (at least partially) overlap with the given target sub-segment, (iii) one of the alignment frames is chosen at random, (iv) one of the 200 fragments contained in the given alignment frame is chosen at random, (v) the ϕ/ψ/ω torsions of that fragment are applied to the respective overlapping region of the given target sub-segment, and (vi) the segment is closed using kinematic closure. Fragment libraries used for FKIC are the same as for the CCD protocol used in benchmark comparisons. Importantly, for benchmarking purposes, we ran simulations using fragment libraries that excluded homologs to the given query sequence (**Appendix I: Supplementary Methods**).

Loophash-sampled KIC (LHKIC) Overview

LHKIC and FKIC share the same overall simulation protocol (**Fig. 2.2**, **Fig. 2.5**). In LHKIC, the non-pivot ϕ/ψ/ω degrees of freedom are sampled from fragments picked by the loophash algorithm [20]. At each KIC sampling step, we calculate the 6D transformation from the residue before the first pivot to the residue after the last pivot. We use the 6D transformation to query a pre-compiled loophash database (see "Generation of loophash databases" in **Appendix I: Supplementary Methods**). One 6D transformation query can return multiple loops. Torsions of a random loop from the returned loops are applied to the residues between the pivot residues.

Rosetta Simulations

FKIC and NGK benchmarking simulations were performed using the Rosetta macromolecular modeling and design suite (https://www.rosettacommons.org/software), revision 59052. The LHKIC method was developed later and used Rosetta revision 60022. KIC simulation results reported in **Fig. 2.2b** were taken from ref. [26]. The Rosetta 'CCD' loop modeling method using fragment insertion and the cyclic coordinate descent closure technique[24] is described in ref. [25]. The NGK loop modeling method is described in ref.[26]. For FKIC simulations, NGK was modified to sample torsions from the generated fragment libraries. Similarly, for LHKIC, NGK was modified to sample torsions from loops picked using loophash[20]. For control simulations that use native bond lengths and angles as input, we replaced the input structure with the native structure and disabled the randomization of torsions at the beginning of the simulation. Since the publication of the original KIC method, the Rosetta energy function has undergone several revisions, including the changes described in the "talaris2013" and "talaris2014" versions [41] and the latest improvements made in the "ref2015" version [21]. The ref2015 energy function [21] was used for all benchmarks unless otherwise noted. Compared to the other energy functions, ref2015 showed a consistent performance improvement (**Table 2.13**). For each test protein in each benchmark set (see below), we generated 500 models with FKIC and calculated the backbone heavy atom root mean square deviation (RMSD) of each target segment after aligning the protein without the modeled segment to its crystal structure. We also measured the median run time to determine whether any increased sampling performance increases computational cost (**Table 2.2a**).

Full descriptions of RosettaScripts code and command lines can be found in **Appendix I: Supplementary Methods**.

Benchmark Datasets

The 12-Residue "*Standard*" benchmark dataset was as previously described [17], [26], [42]. The 16-Residue *"Mixed Segment"* dataset consists of 30 structures from the PDB containing 16-residue target segments where each segment has 5 to 11 residues that contain alpha helices or beta strands. The 10-Residue "*Multiple Segments*" dataset consists of 30 structures from the PDB each containing a pair of 10-residue interacting target segments. We also constructed two analogous sets that contain either two 8-residue segments or two 12-residue segments (**Appendix II: Supplementary Note 2; Table 2.7** and **Table 2.8**). More details on the benchmark datasets are in **Appendix I: Supplementary Methods**.

Preparation of benchmark input structures

To exclude information on the native conformation of the target segment(s) for all benchmark datasets, all side chains in the segment(s) as well as side chains within 10Å of the segment(s) (based on all-atom pairwise distance measurements) were removed. The backbone information was removed by changing the segment into an extended conformation with idealized bond lengths and angles. The datasets were constructed with an openly available script: https://github.com/Kortemme-Lab/benchmark_set_construct.

Fragment distance calculation in structure prediction

The chord distance [43] was calculated between pairs of fragments. The chord distance between two angles is defined as: $D^2(\theta_1, \theta_2) = 2 - 2\cos(\theta_1 - \theta_2)$. In our case, this value was calculated for backbone dihedral angles and summed over paired residues between fragments and target loops: $\langle D \rangle = \frac{1}{n}\sum_i^n \left(\frac{1}{2}D^2\left(\phi_1^i, \phi_2^i\right) + \frac{1}{2}D^2\left(\psi_1^i, \psi_2^i\right)\right)$, with n=3 defining 3mer fragments for example. $\langle D \rangle$ will have a minimum of 0 if the angles match exactly and a maximum of 4 if the angles differ by 180 degrees.

2.5.2 Pull Into Place (PIP) design protocol

Overview

We created designs using two versions of the PIP protocol, denoted versions 1 and 2, which differed in several details. Version 1 was developed before FKIC and LHKIC, and therefore used NGK for both model generation (step 1) and structure prediction (step 3). In step 1, we varied the length of the remodeled active site region from 0 to -6 residues (relative to its native length). In subsequent steps, we made comparisons only between segments of the same length, to avoid biases towards longer segments that can make more favorable interactions at the expense of loss of conformational entropy not considered in Rosetta. Sequence design (step 2) was done using fixed-backbone rotamer sampling. Residues within 4Å of the active site backbone segment were designed (i.e. allowed to change amino acid identity) excluding residues Y14, F54, D99, A114 and F116 that are important for catalysis. In total, 33-39 residues were allowed to design, depending on segment length. Designs from step 2 to be evaluated in step 3 were selected with a probability proportional to their Boltzmann-weighted Rosetta total scores. This approach was intended to improve the diversity of the selected designs, while still selecting more favorable (low-scoring) designs. The designs selected by this procedure for experimental testing either retained the native segment length or shortened the segment by one residue.

In version 2, we made several changes: We used LHKIC for model generation (step 1) and FKIC for structure prediction (step 3). This strategy takes advantage of the ability of LHKIC to sample both sequence and structure simultaneously in step 1 (as fragment picking in LHKIC is independent of the starting sequence). Conversely, FKIC is better suited to predicting conformations given a sequence in step 3, since FKIC picks fragments based on the input sequence. In step 2, we incorporated a small degree of backbone flexibility into the design process by using the Rosetta FastDesign method, which iterates fixed-backbone sequence design and fixed-sequence structure minimization. Because this design algorithm is more computationally expensive than that from version 1, we made fewer designs per backbone model (10 instead of 50). Based on the results from version 1, we only considered two segment lengths: the native length and a one-residue deletion. We also allowed a different (and smaller) set of residues to design: 25 – 26 residues in the active site segment and 4 residues in a small β-hairpin (residues 74 – 77 in the dimer partner) that make inter-chain contacts with the active site segment. To select designs for step 3, we incorporated knowledge from additional metrics besides Rosetta score and functional group positioning. We used metrics including the number of buried unsatisfied and oversaturated hydrogen bonds, a fragment quality filter, total solvent-accessible surface area, and Rosetta's foldability metric (**Appendix I: Supplementary Methods**).

Because it is unclear *a priori* how to prioritize these metrics, we used Pareto fronts consisting of the above metrics to choose designs for computational structure prediction (**Fig. 2.11**). We also selected more designs than in version 1 (up to 422 instead of 200) for structure prediction in early iterations of step 3, taking advantage of the fact that FKIC requires fewer simulations than NGK to make sub-Ångstrom predictions.

In comparison, both versions of PIP used similar robotics-inspired approaches to conformational sampling, but PIP version 2 placed an additional emphasis on fragment-based sampling using FKIC / LHKIC, and analysis of fragment quality using Pareto fronts. Fragment quality measures how well designs conform to local sequence / structure relationships observed in naturally occurring proteins, and designs with a better fragment quality might be expected to be more stable[44]. Attention to fragment quality as a design metric may have resulted in several beneficial characteristics in design V2D9r, which was both more soluble when expressed in *E. coli*, and had a higher apparent T_M (**Fig. 2.9**) than design V1D8r. However, our design sample is small and further exploration of the impact of fragment-based design on design success would be interesting.

Rosetta version.

PIP was run using Rosetta commit 10b6f2f8e20d70757e6b510def2ddcbeef172538 (PIP version 1) or revision 60048 (PIP version 2). We used the latest available score function for each PIP version, which were talaris2013 for PIP version 1 or ref2015 for PIP version 2.

More details on the PIP protocol are in the **Appendix I: Supplementary Methods**.

2.3.3 Experimental characterization

Cloning and purification.

The 14 designs chosen for experimental tests from PIP version 1 were ordered from GenScript pre-cloned into the pET-21a expression vector. For PIP version 2, and for characterization of V1D8r and the wild-type protein, we used an expression vector using parts from the modular yeast cloning toolkit[45] which was similar to pET-21a, except that the cloning resulted in a glycine-serine genetic scar at the C-terminus. Full sequences of ordered designs and vectors can be found in **Supplementary Data 1**. Proteins were expressed in *E. coli* BL21(DE3) cells. Wild-type KSI and design V2D9r were purified essentially as described previously [35], [46] with minor differences (**Appendix I: Supplementary Methods**).

Activity assay.

Purified KSI variants were tested for catalytic activity using an absorbance assay. 5(10)-Estrene-3,17-dione was solubilized at 2.1 mM and serial-diluted two-fold down to 0.51 μM in 100% DMSO. 115 μL enzyme, prepared in 40 mM potassium phosphate, 2 mM DTT, and 1 mM EDTA at pH 7.2, was then added to 5 μL of substrate for final substrate concentrations between 520 and 0.51 μM. K_M and k_{cat} values for the WT enzyme and designs V1D8r and V2D9r were measured at enzyme concentrations between 0.5 and 18 μM. For reversion and alanine scan mutations, k_{cat} values were measured in triplicate at 512 μM substrate. Absorbance at 248 nm was tracked for 5 minutes at room temperature in a Varian Cary 50 Bio UV-Visible spectrophotometer using a 1cm path length. The first 30-60s of each reaction were excluded to allow the reaction to reach steady-state.

X-ray crystallography.

Designed proteins were crystallized in 1 M ammonium sulfate (design V1D8r) or 1.6 M ammonium sulfate, 50 mM potassium phosphate, pH 7.2 (design V2D9r) using the hanging drop method. For design V2D9r, an equal volume of 2 mM equilenin (CAS 517-09-9 from Steraloids

Inc., catalog ID E0400-000) was added to each drop. For details on X-ray data collection and processing see **Appendix I: Supplementary Methods**.

Structure determination.

We obtained initial phase information for calculation of electron density maps by molecular replacement using the program Phaser [47], as implemented in the PHENIX suite [48]. For the V1D8r structure, we identified a single copy of the protein in the asymmetric unit using the coordinates from a previous KSI model, and for the V2D9r structure we identified four copies of the protein in the asymmetric unit. Both solutions were consistent with an analysis of Matthews probabilities for the observed unit cell and molecular weight of the protein [49], [50].

We manually rebuilt the molecular replacement solutions using the resulting electron-density maps, followed by iterative refinement of atomic positions, individual atomic displacement parameters (B-factors) with a TLS model, and occupancies, using riding hydrogen atoms and automatic weight optimization, until the model reached convergence. Throughout the course of manual model building, electron density corresponding to several ligand molecules became apparent, which we were able to model. In the V1D8r structure, we observed electron density for two steroid-like molecules, one occupying the KSI active site, and a second nestled at a crystal contact. These densities were modeled using deoxycholate, which was present in one of the purification buffers used to prepare the crystallization samples. Additionally, we identified two phosphate ions in this structure. In the V2D9r structure, we also saw density for steroid ligands in the active sites of each of the four copies of the enzyme. In this case, the modeling was challenging, because the samples were exposed to both deoxycholate (during purification) and equilenin (post-purification), and electron density features suggested that there could be a mixture of both ligands represented in the electron density. We attempted to model various combinations of the ligands into the active site densities, and found that the electron density features could best be described by modeling equilenin in one active site (chain B), deoxycholate in one active site (chain D), and a mixture of both ligands in the other two active sites (chains A and C). Our choice to model the ligand densities in this way is based on both reduction of refinement R-factors, as well as on overall flatness of residual Fo-Fc difference density maps around the modeled ligands. In the V2D9r structure, we also modeled 12 sulfate ions. All model building was performed using Coot [51] and refinement steps were performed with phenix.refine within the PHENIX suite [48], [51]. Restraints for the ligands were calculated using phenix.elbow [52]. Further information regarding model building and refinement is presented in **Table 2.14**.

RMSD and SASA Calculations.

For all RMSD calculations, structures were aligned to all residues except those involved in the RMSD calculation. To calculate backbone RMSDs that involved comparing the shorter V1D8r segment to the full-length WT protein, we had to exclude one residue in WT structure. We chose to exclude residue 38, as this resulted in the lowest RMSD between the design and the WT protein. Per-residue SASA was calculated using the SasaCalc class in PyRosetta version 2021.12+release.ed6a5560506, which uses the LeGrand approximation of molecular surface area [53].

2.6 Appendix I: Supplementary Methods

2.6.1 Generation and prediction of new conformations (FKIC and LHKIC)

Rosetta scripts & command line for CCD

We used the following Rosetta scripts to run the CCD benchmark simulations:

```
<ROSETTASCRIPTS>
    <TASKOPERATIONS>
        <RestrictToLoops name="loop" loops_file="%%loop_file%%"/>
    </TASKOPERATIONS>
    <MOVERS>
```

```
        <LoopmodelWrapper name="modeler" loops_file="%%loop_file%%"
fast="%%fast%%"/>
    </MOVERS>
    <PROTOCOLS>
        <Add mover_name=”modeler”/>
    </PROTOCOLS>
</ROSETTASCRIPTS>
```

Let the home directory of Rosetta be */path/to/rosetta/main*; the input structure be *my_structure.pdb*; the loop file be *my_structure.loop*; the fragment files be *my_structure.200.9mers.gz* and *my_structure.200.3mers.gz*; and the Rosetta script XML file be *loopmodel.xml*. Then the command to run one simulation is:

```
/path/to/rosetta/main/source/bin/rosetta_scripts.linuxgccrelease -database
/path/to/rosetta/main/database -in:file:s my_structure.pdb -parser:protocol
loopmodel.xml -parser:script_vars loop_file=my_structure.loop fast=no -
out:prefix prefix -overwrite -loops:remodel quick_ccd -loops:refine
refine_ccd -ex1 -ex2 -loops:frag_sizes 9 3 1 -loops:frag_files
my_structure.200.9mers.gz my_structure.200.3mers.gz
```

Rosetta scripts & command line for NGK

We used the following Rosetta script to run the NGK benchmark simulations:

```
<ROSETTASCRIPTS>
    <TASKOPERATIONS>
        <RestrictToLoops name="loop" loops_file="%%loop_file%%"/>
    </TASKOPERATIONS>
    <MOVERS>
        <LoopModeler name="modeler" config="kic" loops_file="%%loop_file%%"
fast="%%fast%%" />
    </MOVERS>
    <PROTOCOLS>
        <Add mover_name=”modeler”/>
    </PROTOCOLS>
</ROSETTASCRIPTS>
```

Let the home directory of Rosetta be */path/to/rosetta/main*; the input structure be *my_structure.pdb*; the loop file be *my_structure.loop*; and the Rosetta script XML file be *loopmodel.xml.* Then the command line to run one simulation is:

```
/path/to/rosetta/main/source/bin/rosetta_scripts.mysql.linuxgccrelease -
database /path/to/rosetta/main/main/database -in:file:s my_structure.pdb -
parser:protocol loopmodel.xml -parser:script_vars loop_file=my_structure.loop
fast=no -out:prefix prefix -overwrite
```

Rosetta scripts & command line for FKIC

We used the following Rosetta script to run FKIC benchmark simulations:

```
<ROSETTASCRIPTS>
    <TASKOPERATIONS>
        <RestrictToLoops name="loop" loops_file="%%loop_file%%"/>
    </TASKOPERATIONS>
    <MOVERS>
        <LoopModeler
            name="modeler"
            config="kic_with_frags"
            loops_file="%%loop_file%%"
            fast="%%fast%%">
        </LoopModeler>
    </MOVERS>
    <PROTOCOLS>
        <Add mover_name="modeler"/>
    </PROTOCOLS>
</ROSETTASCRIPTS>
```

Let the home directory of Rosetta be */path/to/rosetta/main*; the input structure be *my_structure.pdb*; the loop file be *my_structure.loop*; the fragment files be *my_structure.200.9mers.gz* and *my_structure.200.3mers.gz*; and the Rosetta script XML file be *loopmodel.xml.* Then the command line for one simulation is:

```
/path/to/rosetta/main/source/bin/rosetta_scripts.mysql.linuxgccrelease -
database /path/to/rosetta/main/database -in:file:s my_structure.pdb -
parser:protocol loopmodel.xml -parser:script_vars loop_file=my_structure.loop
fast=no -out:prefix prefix -overwrite  -loops:frag_sizes 9 3 1 -
loops:frag_files my_structure.200.9mers.gz my_structure.200.3mers.gz
```

Generation of loophash databases.

We generated the loophash databases using the *loophash_createfiltereddb* application and the VALL database distributed with Rosetta. The command line is:

```
mpirun -np 32 loophash_createfiltereddb.mpi.linuxgccrelease -lh:db_path
loophash_db/ -in:file:vall
path_to_rosetta_tools_repository/tools/fragment_tools/vall.jul19.2011.gz -
lh:loopsizes  3 4 5 6 7 8 9 10 11 12 13 14  -lh:num_partitions 32 -
lh:createdb_rms_cutoff 4.5 6 7.5 9 10.5 12 13.5 15 16.5 18 19.5 21
```

where *loophash_db* is the output path.

Rosetta scripts & command line for LHKIC

We used the following Rosetta script to run the LHKIC benchmark simulations:

```
<ROSETTASCRIPTS>
    <MOVERS>
        <LoopModeler name="modeler" config="loophash_kic"
loops_file="%%loop_file%%" fast="%%fast%%" />
    </MOVERS>
    <PROTOCOLS>
        <Add mover_name="modeler"/>
    </PROTOCOLS>
</ROSETTASCRIPTS>
```

Let the home directory of Rosetta be */path/to/rosetta/main*; the input structure be my_structure.pdb; the loop file be *my_structure.loop*; the Rosetta script XML file be loopmodel.xml; and the path to the loophash database be *path_to_loophash_db*. Then the command line to run one simulation is:

```
/path/to/rosetta/main/source/bin/rosetta_scripts.mysql.linuxgccrelease -database
/path/to/rosetta/main/main/database -in:file:s my_structure.pdb -parser:protocol loopmodel.xml -
parser:script_vars loop_file=my_structure.loop fast=no -out:prefix prefix -overwrite -lh:loopsizes
6 8 10 -lh:db_path path_to_loophash_db
```

By default, LHKIC does not mutate the sequence of the loop. When the *loophash_perturb_sequence* option is set to true, LHKIC applies the sequence of the returned loop to the pivot residues and the residues between the pivots.

Rosetta scripts & command line for simulations with native bond lengths and angles

For control simulations that use native bond lengths and angles as input, we replaced the Rosetta script XML file with the following:

```
<ROSETTASCRIPTS>
    <TASKOPERATIONS>
        <RestrictToLoops name="loop" loops_file="%%loop_file%%"/>
    </TASKOPERATIONS>
    <MOVERS>
        <LoopModeler
```

```
            name="modeler"
            config="kic_with_frags"
            loops_file="%%loop_file%%"
            fast="%%fast%%">
            <Build skip=”True”
/>
        </LoopModeler>
    </MOVERS>
    <PROTOCOLS>
        <Add mover_name=”modeler”/>
    </PROTOCOLS>
</ROSETTASCRIPTS>
```

12-Residue “*Standard*” benchmark dataset

The “Standard” set is a 12-residue loop benchmark dataset used in previous work [17], [26], [42]. This benchmark dataset consists of 45 protein structures from the PDB containing non-redundant 12-residue target segments without regular secondary structure, curated from two previously described datasets [23], [25]. We used this dataset even though it is not ideal (for example, the conformation of several segments might be influenced by crystal contacts, see **Appendix II: Supplementary Note 1** and **Table 2.6**) to facilitate comparison of FKIC with previous protocols. For each loop, we retained the N and Cα atoms of the N-terminal residue, as well as the Cα, C and O atoms of the C-terminal residue, which serve as loop anchor points for kinematic closure (as in ref. [17]).

16-Residue "Mixed Segment" benchmark dataset

The *Mixed Segment* benchmark dataset consists of 30 structures from the PDB containing 16-residue target segments. The target segments were derived from structures in the *Standard* benchmark dataset above using the following criteria:

- The crystallographic resolution of the experimentally determined structure is equal to or better than 2Å.
- Each segment has 5 to 11 residues that contain alpha helices or beta strands defined using DSSP[54] and the remainder of the segment is designated as loop.
- Residues in the segment are at least 4Å away from any chains or copies of the molecule in other asymmetric units, to avoid crystal contacts.
- The segment is at least 5 residues away from the chain termini.

The segment that satisfied all criteria with the lowest distance from the protein surface was selected.

10-Residue "*Multiple Segments*" benchmark dataset

The *Multiple Segments* benchmark dataset consists of 30 structures from the PDB each containing a pair of 10-residue interacting target segments. Structures were derived from the top8000 dataset [55] using the following criteria:

- The crystallographic resolution of the experimentally determined structure is equal to or better than 2Å.
- Each segment has less than 3 residues that have regular secondary structure, defined as above.

- Residues in the segment are at least 4Å away from any chains or copies of the molecule in other asymmetric units, to avoid crystal contacts.
- Each segment is at least 5 residues away from the chain termini.
- Segments in each pair are within 4Å and are separated by at least 5 residues in primary sequence.

The pair of segments that satisfied all criteria with the lowest distance from the protein surface was selected. We also constructed two analogous sets that contain either two 8-residue segments or two 12-residue segments (**Appendix II: Supplementary Note 2; Table 2.7** and **Table 2.8**).

Generation of fragment libraries

We generated libraries of 9-mer and 3-mer fragments for all benchmark cases using the fragment picking method described in ref. [19] (9-and 3-mers are established fragment sizes tested in a variety of Rosetta applications). The method selects fragments from a representative database of 16,801 protein chains extracted from the PDB and culled such that any two chains have at most 60% sequence identity. The fragment database is part of the Rosetta software and located in rosetta/tools/fragment_tools/vall.jul19.2011.gz. The fragments selected for each segment sequence position span a 3- or 9-residue frame, which overlaps with neighboring frames. Moreover, we allow sampling of 1-mer fragments, which consist of a single triplet of $\phi/\psi/\omega$ torsions that are generated on the fly by the respective modeling protocol (see below) based on the 3-mer fragment library for the given position. The make_fragments.pl script in the rosetta/tools/fragment_tools/ directory integrates several data sources to maximize fragment quality, including sequence similarity and the detection of homologs using PsiBLAST [56], predicted secondary structure similarity using PsiPred [57] and prediction of preferred ϕ/ψ torsions and solvent accessibility using SPARKS-X [58]. Importantly, for benchmarking purposes, we ran simulations using fragment libraries that excluded homologs to the given query sequence, by providing the –nohoms flag to the make_fragments.pl script, which excluded all protein chains with a PsiBLAST E-value < 0.05 [56] from the fragment picking process.

2.6.2 Computational design of new conformations (PIP protocol)

Overview

Input files.

KSI designs were based on PDB structure 1QJG [59]. KSI is an obligate dimer, so we included both monomers in our initial structure. To design different loop lengths, we created several versions of the initial structure: one for each deletion of up to 6 residues, and one with wild-type length. We replaced N38 with a glutamate and relaxed the resulting models 100 times in the talaris2013 (version 1) or ref2015 (version 2) score function using FastRelax, with all atom coordinates restrained to their starting positions. The best (lowest-scoring) relaxed structure was then repacked 100 times, and the lowest-scoring model was used as a template for design.

The desired position of the E38 sidechain was defined in a restraint file (see below). Each atom in the E38 carboxylate group was restrained to the position of the corresponding atom in the N38 amide group in the starting structure (1QJG). We visually confirmed that N38 in 1QJG had the same rotameric conformation as D38 in wild-type KSI (PDB structure 8CHO). When working with the version 1 designs, we noticed that F54 sometimes changed its rotamer conformation, causing subsequent designed mutations to stabilize the altered conformation. For version 2, we addressed this issue by placing restraints on the zeta-carbon of F54 in a manner similar to the catalytic residue.

The residues being remodeled were defined in loop files (see below). We chose which residues to remodel based on proximity to secondary structure elements and intuition. Our goals were (i) to allow sufficient remodeling on either side of E38 to stabilize its new conformation, (ii) to anchor the loop in secondary structural elements, and (iii) to minimize loop length. With these considerations in mind, we remodeled segments that were 7-13 (version 1) or 12-13 (version 2) residues long. (While in principle other segment lengths could be tried, 12-13 residue loops resulted in the best designs from round 1, and allowed us to limit the considerable computational expense exploring large ensemble of potential conformations and sequences at each length). For version 2, we also remodeled a second 4-residue loop on the dimer interface (residues 199-202 using Rosetta numbering, residues 74-77 on chain B using PDB numbering), hoping to maintain favorable contacts between those residues and the catalytic loop.

The residues that were allowed to design (change amino acid identity) and repack (only change rotamer conformation) were specified in a resfile (see below). For version 1, any residue that had a sidechain atom within 4Å or 6Å of any loop atom in any model generated in PIP step 1 was allowed to design or repack, respectively. For version 2, we only allowed Rosetta to design residues on the catalytic loop, as well as four residues on the short dimerization loop which directly interacts with the catalytic loop (described above). Repackable residues were selected using the Rosetta clash-based repack shell selector. F54, A114, and F116 were not allowed to design in either version because they are known to be important for positioning the catalytic residue [60]. For version 1, each designed residue was allowed to change to any of the 20 canonical amino acids except cysteine (due to the potential for disulfide bonds) and histidine (due to the potential for pH-dependent behavior). For version 2, we used the LayerDesign task operation in Rosetta to determine which residue identities were allowed at each position. Since version 2 introduced backbone degrees of freedom during the design step, we specified a fold tree to keep conformational changes as local as possible (see below).

PIP Step 1: Build Models.

We created models positioning the E38 carboxylate group by running approximately 20,000 NGK[26] (version 1) or LHKIC (version 2) simulations with restraints as described above. Backbone remodeling was limited to the loop defined in the appropriate loop file and design was allowed according to the appropriate resfile (see below). In version 1 of PIP, the initial coordinates of the loop being remodeled were discarded and rebuilt from scratch. This step was skipped for version 2. Only models that put all three restrained atoms within 0.6 Å (version 1) or 0.7 Å (version 2) of their intended positions were carried on to the next step.

PIP Step 2: Design Models.

To stabilize models that correctly positioned E38, we ran 50 fixed-backbone (version 1) or 10 FastDesign (version 2) simulations per model, or more if there were relatively few models. Design was allowed according to the appropriate resfile (see below). For version 1, we picked 200 designs for PIP step 3 with probability proportional to their Boltzmann-weighted talaris2013 scores (in Rosetta energy units, REU). For version 2, we used a combination of Pareto fronts and thresholds to pick designs so that we could supplement information from the Rosetta score function with additional metrics. The exact parameters used to pick designs are included in a "picks" file, described below. The metrics for which we applied thresholds were the E38 restraint distance (defined as the maximum distance of the restrained atoms in E38 to their ideal position), the number of hydrogen bonds to the E38 sidechain, and the number of oversaturated hydrogen bonds. Models that passed these thresholds were included in the Pareto front calculation, whose metrics consisted of the total solvent-accessible surface area of the model, two "foldability" metrics that perform 60 brief forward-folding simulations on pieces of the loop and report the fraction of results that placed the segment's N-terminus within 4 Å of the concomitant residue in the design structure, the E38 restraint distance, a fragment quality metric (see below), and the Rosetta fa_attr score. The Foldability metrics remove a portion of the design's backbone, then rebuild it starting from the N-terminus of the deleted segment using fragment-based assembly. This is repeated 100 times, and the average distance of the C-terminus of the rebuilt segment to its position in the design is reported.

PIP Step 3: Structure Prediction.

We computationally assessed our designs by running between 100 and 500 NGK (version 1) or FKIC (version 2) structure prediction simulations for each design; for version 2, we opted to perform fewer structure prediction simulations on a larger number of designs during the early rounds of design. Backbone movement was limited to the loop defined in the appropriate loop file (see below). The initial coordinates for that loop were discarded and rebuilt from scratch. Any design for which the lowest scoring decoy put all three carboxylate atoms within 1.2 Å of their intended positions was carried on to the design selection step. Furthermore, any decoy (regardless of score) that put all three carboxylate atoms within 0.6 Å of their intended positions was used as input for a second round of design simulations.

Design Selection.

We picked designs to experimentally test by comparing quality metrics and visually inspecting models. The quality metrics are described in **Table 2.9**. We paid particular attention to the score gap, which measures the difference in the score between the lowest-scoring model with under 1 Å restraint satisfaction and the lowest-scoring model with over 2 Å restraint satisfaction. Several designs were selected despite having a score gap of 0 REU, as they had multiple low-energy conformations. We also made an effort to pick designs from different sequence and structure clusters. Design sequences were clustered hierarchically such that inter-cluster distance was no greater than the mean sequence distance (calculated according to the BLOSUM80 substitution matrix) across all designs. Structure clusters were formed hierarchically such that the RMSD between any two designs in the same cluster was no greater than 1.2 Å. We visually inspected the lowest scoring model for each design to eliminate those with irregular backbone or strained sidechain conformations.

Wildtype Reversions.

For each design from PIP version 1 selected for experimental validation, we reran the structure prediction simulations (PIP Step 3) for each single wildtype reversion mutation. We then combined any reversions that had no apparent detrimental effect on our quality metrics and again ran the structure prediction simulations. In cases where the combination of all the individually acceptable reversions had a deleterious effect, we selected more conservative combinations of reversions for additional structure prediction simulations. If no acceptable combination of reversions could be found, no reversions were made (**Table 2.9**). For PIP version 2, positions where the backbone was in a similar position to wildtype were reverted, or in some cases mutated to a residue picked by visual inspection (**Table 2.10**), and designs with and without those reversions were ordered for experimental validation.

Input files for PIP

E38 constraints definition.

We used the following energetic constraints file for PIP Version 1 and Version 2:

```
CoordinateConstraint CG  38 CA 1 12.159  64.031 -28.170 HARMONIC 0.0 1.0
CoordinateConstraint OE1 38 CA 1 10.881  63.345 -30.090 HARMONIC 0.0 1.0
CoordinateConstraint OE2 38 CA 1 11.759  65.409 -30.106 HARMONIC 0.0 1.0
```

Additional constraint on F54 for PIP version 2.

For PIP Version 2, we included the following additional constraint to prevent a conformational change in the sidechain of F54 that was frequently observed in PIP Version 1:

```
CoordinateConstraint CZ  54 CA 1 15.196  64.334 -29.952 HARMONIC 0.0 1.0
```

Loop definitions for design V1D8r.

We used the following loop definition for PIP Version 1 models with a single deletion in the loop region:

```
LOOP 34 45 45 0 1
```

The loop definition for input models with no deletions included residue 46.

Loop definitions for design V2D9r.

For PIP Version 2, we used the following loop definitions:

```
LOOP  26  51  40 0 0
LOOP 198 203 200 0 0
```

Design step resfile for PIP Version 1.

We used the following resfile for PIP Version 1 (for input models with a single deletion):

```
NATRO
START
# Design residues in the loop itself. Don't move the catalytic residue,
# because we want to find designs which stabilize that rotamer.
34 - 37 A NOTAA HC
38 A NATRO
39 - 45 A NOTAA HC
# Design any residue that has a sidechain atom within 4A of any loop atom in
# any input model. Phe53, Ala113, and Phe115 are excluded because they are
# known to be important for positioning the catalytic residue.

29 A NOTAA HC

30 A NOTAA HC

31 A NOTAA HC

32 A NOTAA HC

33 A NOTAA HC

46 A NOTAA HC

48 A NOTAA HC

49 A NOTAA HC

50 A NOTAA HC

52 A NOTAA HC

54 A NOTAA HC

56 A NOTAA HC

57 A NOTAA HC

108 A NOTAA HC

109 A NOTAA HC

110 A NOTAA HC

111 A NOTAA HC

112 A NOTAA HC

114 A NOTAA HC

116 A NOTAA HC

117 A NOTAA HC

120 A NOTAA HC

198 B NOTAA HC

199 B NOTAA HC

200 B NOTAA HC

201 B NOTAA HC

# Repack any residue that has a sidechain atom within 6A of any loop atom in
# any input model.
10 A NATAA
11 A NATAA
13 A NATAA
14 A NATAA
15 A NATAA
16 A NATAA
17 A NATAA
18 A NATAA
23 A NATAA
26 A NATAA
27 A NATAA
28 A NATAA
47 A NATAA
51 A NATAA
53 A NATAA
55 A NATAA
58 A NATAA
59 A NATAA
60 A NATAA
62 A NATAA
```

For inputs with no deletions, we included the additional loop residue with the tag NOTAA HC and adjusted the residue numbers of post-loop positions accordingly.

Design step resfile for PIP Version 2.

We used the following resfile for PIP version 2:

```
NATRO
START
34      A NOTAA CH
35      A NOTAA CH
36      A NOTAA CH
37      A NOTAA CH
38      A PIKAA E
39      A NOTAA CH
40      A NOTAA CH
41      A NOTAA CH
42      A NOTAA CH
43      A NOTAA CH
44      A NOTAA CH
45      A NOTAA CH
46      A NOTAA CH
199     B NOTAA CH
200     B NOTAA CH
201     B NOTAA CH
202     B NOTAA CH

# Repack positions
# ================
# The following repack positions were chosen by the clash-based repack
# shell creator (excluding the ligand).
```

```
14      A NATAA

30      A NATAA

50      A NATAA

51      A NATAA

54      A NATAA

55      A NATAA

95      A NATAA

109     A NATAA

111     A NATAA

112     A NATAA

113     A NATAA

114     A NATAA

115     A NATAA

116     A NATAA

121     A NATAA

127     B NATAA

204     B NATAA

225     B NATAA

227     B NATAA


# The following repack positions were added after visual inspection of

# clash-based repack shell.


10      A NATAA

13      A NATAA

17      A NATAA

25      A NATAA
```

```
52      A NATAA

53      A NATAA

56      A NATAA

57      A NATAA

58      A NATAA

108     A NATAA

110     A NATAA

117     A NATAA

118     A NATAA

126     B NATAA

128     B NATAA

228     B NATAA
```

Picks file.

Thresholds and Pareto front metrics for PIP version 2 were defined using a YAML-formatted file. We used the following YAML-formatted file to define the metrics to be used as thresholds and in Pareto fronts for picking designs for step 3 in PIP Version 2. Designs which meet the criteria defined under “threshold” are passed into the Pareto front. Metrics listed under “pareto” are used to define the Pareto front. The “depth” value describes how many times the Pareto front is applied, and the “epsilon” value defines how close two points can be in all Pareto dimensions being considered before they are considered to be the same and one is excluded. Its units are the percent of the range of points between the 10th and 90th percentiles. Let *restraint_dist_e38* be the E38 restraint distance, *h_bonds_to_e38_sidechain* be the number of hydrogen bonds to E38, *oversaturated_h_bonds* be the number of oversaturated hydrogen bonds, *fa_attr* be the attractive Van der Waals term in the Rosetta energy function, *max_9_residue_fragment_rmsd_c_alpha* be the Fragment Quality score (see below), *total_sasa* be the total solvent-accessible surface area, *foldability_37_44* be the Foldability score (described above) for residues 37 through 44, and *foldability_35_41* be the Foldability score for residues 35 through 41.

```
threshold:
- restraint_dist_e38 < 0.8
- h_bonds_to_e38_sidechain == 0
- oversaturated_h_bonds == 0

pareto:
- fa_attr
- restraint_dist_e38
- max_9_residue_fragment_rmsd_c_alpha
- total_sasa
- foldability_37_44
- foldability_35_41

depth: 1
epsilon: 0.5
```

Fragment Quality Analysis for Designed Regions

The fragment quality metric assesses the geometric similarity between 9-residue fragments in the designed segments and fragments of natural proteins in the PDB, as described in ref. [44]. Specifically, we picked protein fragments from the PDB based on their similarity in sequence, predicted solvent exposure, and predicted secondary structure to the post-simulation design sequence and determined the RMSD of the backbone atoms in each fragment to the final structure. We then determined the lowest RMSD at each position being evaluated (assigning the RMSD to position at the center of each nine residue fragment). The fragment score filter uses the highest value of these lowest fragment RMSDs at each position.

PIP Step 1: Build Models

We used the following command line for the Build Models step in PIP Version 1. Let the "main" directory in Rosetta be */path/to/rosetta/main/*; the (relaxed) input PDB be $INPUT_PDB; the (unrelaxed) native PDB be $NATIVE_PDB; the full-atom and centroid scorefunction parameters for the talaris2013 scorefunction be "EQU.fa.params" and "EQU.cen.params", respectively; the resfile be $RESFILE; the constriants file be $CONSTRAINTS and the loops file be $LOOP.

```
/path/to/rosetta/main/source/bin/loopmodel.linuxgccrelease \
-in:file:s $INPUT_PDB \
-in:file:native $NATIVE_PDB \
-in:file:extra_res_fa "EQU.fa.params" \
-in:file:extra_res_cen "EQU.cen.params" \
-in:file:fullatom \
-out:overwrite \
-out:pdb_gz \
-packing:ex1 \
-packing:ex2 \
-packing:extrachi_cutoff 0 \
-packing:resfile $RESFILE \
-constraints:cst_fa_weight 1.0 \
-constraints:cst_fa_file $CONSTRAINTS \
-loops:loop_file $LOOP \
-loops:remodel "perturb_kic" \
-loops:refine "refine_kic" \
-loops:kic_rama2b \
-loops:kic_omega_sampling \
-loops:allow_omega_move "true" \
-loops:ramp_fa_rep \
-loops:ramp_rama \
```

For PIP Version 2, we used the following Rosetta script to build new backbone geometries. Let the loops file be $LOOPS_PATH. Definitions common to all steps are found in “shared_defs.xml”, described below. For all Rosetta scripts, variables are filled in by the PIP package.

```
<ROSETTASCRIPTS>

  {% include "shared_defs.xml" %}

  <TASKOPERATIONS>
    <RestrictToRepacking name="repackonly"/>
  </TASKOPERATIONS>
  <MOVERS>
    <LoopModeler name="modeler"
      config="loophash_kic"
      scorefxn_fa="scorefxn_cst"
      task_operations="resfile,repackonly,ex,aro,curr"
      loops_file="$LOOPS_PATH"
      loophash_perturb_sequence="yes"
      loophash_seqposes_no_mutate="38"
      fast="no"
    />
  </MOVERS>

  <PROTOCOLS>
    <!-- Constraints read from command line -->
    <Add mover_name="modeler"/>
    <Add mover_name="writer"/>
  </PROTOCOLS>

  <OUTPUT scorefxn="scorefxn"/>

</ROSETTASCRIPTS>
```

The PIP package also builds the command line, but a representative example is shown below. Let the path to the "main" folder in Rosetta be */path/to/rosetta/main/*; the (relaxed) input PDB be $INPUT_PDB; the (unrelaxed) native PDB be $NATIVE_PDB; the folder where models are to be saved be $OUPUT_FOLDER; the name of the particular design be

$OUTPUT_NAME; the “build models” Rosetta script be *build_models.xml*; the resfile path be $RESFILE_PATH; the path to the constraints file be $CONSTRAINTS, and the path to the loophash database be *path_to_loophash_db*.

```
/path/to/rosetta/main/source/bin/rosetta_scripts.linuxgccrelease \
-database /path/to/rosetta/main/database/ \
-in:file:s $INPUT_PDB \
-in:file:native $NATIVE_PDB \
-out:prefix $OUTPUT_FOLDER \
-out:suffix $OUTPUT_NAME \
-out:no_nstruct_label -out:overwrite -out:pdb_gz \
-out:mute protocols.loops.loops_main \
-parser:protocol build_models.xml \
-packing:resfile $RESFILE_PATH \
-constraints:cst_fa_file $CONSTRAINTS \
-lh:loopsizes 6 7 8 9 10 11 12 13 14 \
-lh:db_path path_to_loophash_db
```

PIP Step 2: Design Models

We used the following command line to design models in step 2 of PIP Version 1. Let the “main” directory in Rosetta be */path/to/rosetta/main/*; the (relaxed) input PDB be $INPUT_PDB; the full-atom and centroid scorefunction parameters be “EQU.fa.params” and “EQU.cen.params”, respectively, and the resfile be $RESFILE.

```
/path/to/rosetta/main/source/bin/fixbb.linuxgccrelease \
-in:file:s $INPUT_PDB \
-in:file:extra_res_fa "EQU.fa.params" \
-in:file:extra_res_cen "EQU.cen.params" \
-out:overwrite \
-out:pdb_gz \
-packing:ex1 \
-packing:ex2 \
-packing:extrachi_cutoff 0 \
-packing:use_input_sc \
-packing:resfile $RESFILE \
```

For PIP Version 2, we defined a custom fold tree for the design step:

```
FOLD_TREE
EDGE   1  39 -1
EDGE 100   1  3
EDGE 100  40 -1
EDGE 100 125 -1
EDGE 100 225  1
EDGE 100 251  2
EDGE 126 200 -1
EDGE 225 126  4
EDGE 225 201 -1
EDGE 225 250 -1
```

We then used the following Rosetta script to design sequences for the new backbone geometries. Let the path to the fold tree file be $FOLDTREE. Definitions common to all steps are found in “shared_defs.xml”, described below.

```
<ROSETTASCRIPTS>

  {% include "shared_defs.xml" %}

  <RESIDUE_SELECTORS>
    <Index name="turn" resnums="200-201"/>
  </RESIDUE_SELECTORS>

  <TASKOPERATIONS>
    <LayerDesign name="layer"
        ignore_pikaa_natro="yes"/>
    <ConsensusLoopDesign name="abego"
        residue_selector="turn"
        include_adjacent_residues="no"/>
  </TASKOPERATIONS>

  <MOVERS>
    <AtomTree name="foldtree" fold_tree_file="$FOLDTREE"/>
    <AtomTree name="unfoldtree" simple_ft="yes"/>
    <AddChainBreak name="break_loop" resnum="39" change_foldtree="no"/>
    <AddChainBreak name="break_turn" resnum="200" change_foldtree="no"/>
    <FastDesign name="fastdesign"
        task_operations="resfile,layer,abego,ex,aro,curr"
```

```
          scorefxn="scorefxn_cst" >
        <MoveMap bb="no" chi="yes" jump="no">
           <Span begin="26"  end="51"  chi="yes" bb="yes"/>
           <Span begin="198" end="203" chi="yes" bb="yes"/>
        </MoveMap>
      </FastDesign>
    </MOVERS>

    <PROTOCOLS>
      <Add mover_name="nativebonus"/>
      <Add mover_name="cst"/> <!-- Added via mover b/c command-line ignored. --
>
      <Add mover_name="foldtree"/>
      <Add mover_name="break_loop"/>
      <Add mover_name="break_turn"/>
      <Add mover_name="fastdesign"/>
      <Add mover_name="unfoldtree"/> <!-- Otherwise Foldability segfaults. -->
      <Add mover_name="writer"/>
    </PROTOCOLS>

    <OUTPUT scorefxn="scorefxn"/>

</ROSETTASCRIPTS>
```

A representative command line is shown below. Let the path to the “main” folder in Rosetta be */path/to/rosetta/main/*; the (relaxed) input PDB be $INPUT_PDB; the (unrelaxed) native PDB be $NATIVE_PDB; the folder where models are to be saved be $OUPUT_FOLDER;

the name of the particular design be $OUTPUT_NAME; the “design models” Rosetta script be *design_models.xml*, and the resfile path be $RESFILE_PATH.

```
/path/to/rosetta/main/source/bin/rosetta_scripts.linuxgccrelease \
-database /path/to/rosetta/main/database/ \
-in:file:s $INPUT_PDB \
-in:file:native $NATIVE_PDB \
-out:prefix $OUTPUT_FOLDER \
-out:suffix $OUTPUT_NAME \
-out:no_nstruct_label -out:overwrite -out:pdb_gz \
-out:mute protocols.loops.loops_main \
-parser:protocol design_models.xml \
-packing:resfile $RESFILE_PATH \
```

PIP Step 3: Structure Prediction

We used the following command line to design models in step 2 of PIP Version 1. Let the “main” directory in Rosetta be */path/to/rosetta/main/*; the (relaxed) input PDB be $INPUT_PDB; the full-atom and centroid scorefunction parameters be “EQU.fa.params” and “EQU.cen.params”, respectively, and the loops file be $LOOPS.

```
/path/to/rosetta/main/source/bin/loopmodel.linuxgccrelease \
-in:file:s $INPUT_PDB \
-in:file:native $NATIVE_PDB \
-in:file:extra_res_fa "EQU.fa.params" \
-in:file:extra_res_cen "EQU.cen.params" \
-in:file:fullatom \
-out:pdb_gz \
-out:overwrite \
-packing:ex1 \
-packing:ex2 \
-packing:extrachi_cutoff 0 \
-loops:loop_file $LOOPS \
-loops:remodel "perturb_kic" \
-loops:refine "refine_kic" \
-loops:kic_rama2b \
-loops:kic_omega_sampling \
-loops:ramp_fa_rep \
-loops:ramp_rama \
```

For PIP Version 2, we used the following Rosetta script to predict the structures of picked designs. Let the loops file be $LOOPS_PATH. Definitions common to all steps are found in “shared_defs.xml”, described below.

```
<ROSETTASCRIPTS>

  {% include "shared_defs.xml" %}

  <MOVERS>
    <LoopModeler name="modeler"
      config="kic_with_frags"
      scorefxn_fa="scorefxn"
      loops_file="$LOOPS_PATH"
      fast="no">
        <Build skip="yes"/>
    </LoopModeler>
  </MOVERS>

  <PROTOCOLS>
    <Add mover_name="modeler"/>
    <Add mover_name="writer"/>
  </PROTOCOLS>

  <OUTPUT scorefxn="scorefxn"/>

</ROSETTASCRIPTS>
```

A representative command line for PIP Version 2 is shown below. Let the path to the "main" folder in Rosetta be */path/to/rosetta/main/*; the (relaxed) input PDB be $INPUT_PDB; the (unrelaxed) native PDB be $NATIVE_PDB; the folder where models are to be saved be $OUPUT_FOLDER; the name of the particular design be $OUTPUT_NAME; the "predict models" Rosetta script be *predict_models.xml*; the paths to the 9-mer fragments for the first and second loop be *path_to_9mers_A* and *path_to_9mers_B*, respectively, and the paths to 3-mer fragments for the first and second loop be *path_to_3mers_A* and *path_to_3mers_B*, respectively.

```
/path/to/rosetta/main/source/bin/rosetta_scripts.linuxgccrelease \
-database /path/to/rosetta/main/database/ \
-in:file:s $INPUT_PDB \
-in:file:native $NATIVE_PDB \
-out:prefix $OUTPUT_FOLDER \
-out:suffix $OUTPUT_NAME \
-out:no_nstruct_label -out:overwrite -out:pdb_gz \
-out:mute protocols.loops.loops_main \
-parser:protocol predict_models.xml \
-loops:frag_sizes 9 9 3 3 \
-loops:frag_files path_to_9mers_A path_to_9mers_B path_to_3mers_A
path_to_3mers_B
```

PIP Version 2 Shared Definitions

Filter, scorefunction, residue selector, and certain Rosetta mover definitions were used during every step of the PIP Version 2 protocol. These were stored as separate RosettaScripts files and imported into each main step template. Let the path to the scorefunction weights file be $SCOREFXN_WEIGHTS and the path to the constraints file be $CONSTRAINTS. The weights file was identical to the default weights for the ref2015 scorefunction.

```
  {% include "filters.xml" %}

  <SCOREFXNS>
    <ScoreFunction name="scorefxn" weights="$SCOREFXN_WEIGHTS"/>
    <ScoreFunction name="scorefxn_cst" weights="$SCOREFXN_WEIGHTS">
      <Reweight scoretype="coordinate_constraint" weight="1.0"/>
      <Reweight scoretype="atom_pair_constraint" weight="1.0"/>
      <Reweight scoretype="angle_constraint" weight="1.0"/>
      <Reweight scoretype="dihedral_constraint" weight="1.0"/>
      <Reweight scoretype="res_type_constraint" weight="1.0"/>
      <Reweight scoretype="chainbreak" weight="100.0"/>
    </ScoreFunction>
  </SCOREFXNS>

  <RESIDUE_SELECTORS>
    <Chain name="chA" chains="A"/>
    <Index name="E38" resnums="38"/>
  </RESIDUE_SELECTORS>

  <TASKOPERATIONS>
    <ReadResfile name="resfile"/>
    <ExtraRotamersGeneric name="ex" ex1="yes" ex2="yes" extrachi_cutoff="0"/>
    <LimitAromaChi2 name="aro" include_trp="yes"/>
    <IncludeCurrent name="curr"/>
  </TASKOPERATIONS>

  <MOVERS>
    <FavorNativeResidue name="nativebonus" />
    <ConstraintSetMover name="cst" cst_fa_file="$CONSTRAINTS"/>
    <WriteFiltersToPose name="writer" prefix="EXTRA_METRIC "/>
  </MOVERS>
```

Filters for PIP Version 2

We used the following Rosetta script to run filters for all three steps in PIP Version 2, with the exceptions of the fragment quality metric (FragmentScoreFilter) and the Foldability metric, which were not included in the Structure Prediction step. For fragment picking, several variables are defined. Let the folder where fragment picking files are stored be $OUTPUT_DIR; the job-specific name of these files be $OUTPUT_NAME; the path to the required CSBLAST, BLAST, PSIPRED, and SPARKS-X programs be */path/to/csblast-2.2.3_linux64*, */path/to/blast-2.2.26/bin/blastpgp*, */path/to/psipred/runpsipred_single*, and */path/to/sparks-x*, respectively; the path to the BLAST database consisting of FASTA-formatted sequence information for proteins in the PDB be */path/to/BLAST/sequences*; the weights file for scoring fragments be $FRAMGNET_WEIGHTS, and the path to the vall database */path/to/Rosetta/database/sampling/vall.jul19.2011.torsions.gz*.

```
<FILTERS>
  <PackStat
    name="PackStat Score [+]"
    threshold="0"
    chain="0"
    repeats="1"
  />
  <ResidueIE
    name="E38 Interaction Energy [-]"
    scorefxn="scorefxn_cst"
    score_type="total_score"
    energy_cutoff="-10"
    restype3="GLU"
    interface="0"
    whole_pose="0"
    selector="E38"
    jump_number="1"
    interface_distance_cutoff="8.0"
    max_penalty="1000.0"
    penalty_factor="1.0"
  />
  <PreProline
    name="Pre-Proline Potential [-]"
    use_statistical_potential="true"
  />
  <TotalSasa
    name="Total SASA [-]"
    threshold="0"
    upper_threshold="1000000000000000"
    hydrophobic="0"
    polar="0"
  />
  <ExposedHydrophobics
    name="Exposed Hydrophobic Residue SASA [-]"
    sasa_cutoff="20"
    threshold="-1"
  />
  <HbondsToResidue
    name="H-bonds to E38 [+]"
    scorefxn="scorefxn_cst"
    partners="0"
    energy_cutoff="-0.5"
    backbone="true"
    bb_bb="true"

    sidechain="true"

    residue="38"

    from_other_chains="true"

    from_same_chain="true"

  />

  <HbondsToResidue

    name="H-bonds to E38 (Backbone) [+]"

    scorefxn="scorefxn_cst"

    partners="0"

    energy_cutoff="-0.5"

    backbone="true"

    bb_bb="true"

    sidechain="false"

    residue="38"

    from_other_chains="true"

    from_same_chain="true"

  />

  <HbondsToResidue

    name="H-bonds to E38 (Sidechain) [+]"

    scorefxn="scorefxn_cst"

    partners="0"

    energy_cutoff="-0.5"

    backbone="false"

    bb_bb="false"

    sidechain="true"

    residue="38"

    from_other_chains="true"
    from_same_chain="true"
  />
  <BuriedUnsatHbonds
    name="Buried Unsatisfied H-Bonds [-]"
    scorefxn="scorefxn"
    print_out_info_to_pdb="true"
    task_operations="resfile"
  />
  <OversaturatedHbondAcceptorFilter
    name="Oversaturated H-bonds [-]"
    scorefxn="scorefxn_cst"
    max_allowed_oversaturated="0"
    hbond_energy_cutoff="-0.5"
    consider_mainchain_only="false"
  />
  <RepackWithoutLigand
    name="Repack Without Ligand (delta REU) [-]"
    scorefxn="scorefxn_cst"
    target_res="all_repacked"
    rms_threshold="100"
  />
  {% if w.focus_name != 'validate_designs' %}
  <Foldability
    name="Foldability (35-41)"
    tries="60"
    start_res="35" {# Unaffected by loop length. #}
    end_res="41"   {# Unaffected by loop length. #}
  />
```

```
<Foldability
  name="Foldability (37-44)"
  tries="60"
  start_res="37" {# Unaffected by loop length. #}
  end_res="44"   {# Unaffected by loop length. #}
/>
<FragmentScoreFilter
  name="Max 9-Residue Fragment RMSD (C alpha) [-]"
  scoretype="FragmentCrmsd"
  sort_by="FragmentCrmsd"
  threshold="9999"
  direction="-"
  start_res="26 "
  end_res="51"
  compute="maximum"
  outputs_folder="$OUTPUT_DIR"
  outputs_name="$OUTPUT_NAME"
  csblast="/path/to/csblast-2.2.3_linux64"
  blast_pgp="/path/to/blast-2.2.26/bin/blastpgp"
  placeholder_seqs="/path/to/BLAST/sequences"
  psipred="/path/to/psipred/runpsipred_single"
  sparks-x="/path/to/sparks-x"
  sparks-x_query="/path/to/sparks-x/bin/buildinp_query.sh"
  frags_scoring_config="$FRAGMENTS_WEIGHTS"
  n_frags="200"
  n_candidates="1000"
  fragment_size="9"
```

```
vall_path="/path/to/Rosetta/main/database/sampling/vall.jul19.2011.torsions.g
z"
    print_to_pdb="true"
  />
  {% endif %}
</FILTERS>
```

The fragment quality filter required the following weights file describing which scores to use when picking fragments[8] during the structure prediction step:

```
# score name          priority  wght   min_allowed  extras
ProfileScoreL1           700     1.0        -
ProfileScoreStructL1     100     4.0        -
SolventAccessibility     500     1.5        -
Phi                      300     1.0        -
Psi                      200     0.6        -
SecondarySimilarity      600     1.0        -       predA
RamaScore                400     0.8        -       predA
FragmentCrmsd            0       0.0        -
```

2.6.3 Experimental characterization of designs

Purification of KSI designs

Cells were lysed in 40 mM potassium phosphate, 2 mM DTT, 1 mM EDTA, and 6 U/mL DNAse I, pH 7.2 using a Microfluidics M-110L microfluidizer. Clarified lysate was then passed through a 10 mL sodium deoxycholate gravity affinity column, prepared as described in reference [46]. The column was washed with 400 mM phosphate, 2 mM DTT, 1 mM EDTA, pH 7.2 followed by lysis buffer (minus DNAse), then eluted with 40 mM phosphate, 2 mM DTT, 1 mM EDTA, and 50% ethanol, pH 7.2. Proteins were then either further purified using a HiLoad 16/600 Superdex 75 pg gel filtration column or dialyzed twice in 1L lysis buffer to remove the ethanol. Most other designed proteins expressed in the insoluble fraction, so inclusion bodies were first purified from the cell lysate: Cells were grown in 1 L LB broth to an optical density of 0.6 at 37 °C, followed by overnight expression at 18 °C. Cells were then harvested by centrifugation at 3500 rpm for 20 minutes at 4 °C, then resuspended in lysis buffer (40 mM Tris-HCl, 1 mM EDTA, 25% sucrose w/v, pH 8.5). Suspensions were lysed in a M-110L microfluidizer and centrifuged at 20,000 rpm for 20 minutes at 4 °C. The resulting inclusion body pellet was washed once in 25 mL of 20 mM Tris-HCl, 1% sodium deoxycholate, 200 mM NaCl, and 2 mM EGTA, followed by at least 3 washes of 25 mL 10 mM Tris-HCl, 0.25% sodium doxycholate, pH 8.5, followed by at least 3 washes of 25 mL 20 mM Na-HEPES, 500 mM NaCl, 1 mM EDTA, pH 8.5. Inclusion bodies were centrifuged at 8,000 xg for 10 minutes at 4 °C between washes. Proteins in inclusion bodies were solubilized by shaking for 30 minutes with 10 mL 8 M urea, 20 mM Na-HEPES, 500 mM NaCl, 10 mM DTT, and 1 mM EDTA at pH 8.5, then centrifuged at 20,000 rpm for 20 minutes at 4 °C to remove cell debris. Solubilized protein was then refolded by stirring for 2 hours at 4 °C in 200 mL of 40 mM KPi, 1 mM EDTA, 2 mM DTT. Proteins were then sterile-filtered using 0.4 μm filter paper and further purified via deoxycholate column as described above.

Size exclusion chromatography

Six uM of purified wild-type KSI, V1D8r, or V2D9r were loaded onto a Superdex 75 10/300 GL column from Cytiva which was pre-equilibrated with running buffer (40 mM phosphate, 2 mM DTT, and 1 mM EDTA). Samples were run isocratically in an Agilent Technologies 1200 Series HPLC for 150 minutes and absorbance was monitored at 280 nm.

CD spectroscopy

Samples for CD analysis were prepared at approximately 6 μM enzyme in 40 mM phosphate pH 8.5, 2 mM DTT, and 1 mM EDTA. CD spectra were recorded at 25 °C using 2 mm cuvettes (Starna, 21-Q-2) in a JASCO J-710 CD spectrometer (Serial #9079119). The bandwidth was 2 nm, rate of scanning 20 nm/min, data pitch 0.2 nm, and response time 8 s. Each spectrum represents the average of 5 scans. Buffer spectra were subtracted from the sample spectra using the Spectra Manager software Version 1.53.01 from JASCO Corporation. Melting temperatures were assessed by measuring molar ellipticity at 222 nm and increasing the temperature from 25 °C to 95 °C at 1 °C per minute, using a data pitch of either 0.1, 0.5, or 1.0 °C.

X-ray data collection and processing

Prior to X-ray data collection, crystals were cryoprotected and flash-cooled by rapid plunging into liquid nitrogen. Crystals that yielded the V1D8r structure were cryoprotected using a mixture of 50% glycerol and 50% crystallization mother liquor, and crystals that yielded the V2D9r structure were cryoprotected using a mixture of 25% glycerol and 75% crystallization mother liquor. We collected single-crystal X-ray diffraction data on beamline 8.3.1 at the Advanced Light Source. Data collection for V1D8r was performed while the beamline was equipped with a Quantum 315r CCD detector (ADSC), while data collection for the V2D9r structure utilized a newer Pilatus3 S 6M photon-counting detector (Dectris). Both data sets were collected using an X-ray energy of 11111 keV, and the crystals were maintained at a cryogenic temperature (100 K) throughout the course of data collection.

We processed the X-ray data using the Xia2 system[61], which performed indexing, integration, and scaling with XDS and XSCALE[62], followed by merging with Pointless[63]. For the [6UAE] structure, a resolution cutoff (1.93 Å) was taken where the signal-to-noise ratio (<I/σI>) of the data fell to a value of approximately 1.0. In the case of the V1D8r structure, the data were collected on an older, smaller detector, and the resolution was limited by the detector edge and the geometric requirements of the experiment. Although other metrics of data quality (such as CC1/2 and <I/σI>) suggest that a more aggressive resolution cutoff would be acceptable, we were limited by the data completeness that could be obtained with the minimum accessible sample-to-detector distance. Further information regarding data collection and processing is presented in **Table 2.14**. The reduced diffraction data were analyzed with phenix.xtriage to check for common crystal pathologies, none of which were identified.

2.7 Appendix II: Supplementary Notes

2.7.1 Supplementary Note I: Analysis of failure cases

Despite the performance improvements with FKIC, in particular for the challenging *Mixed Segment* and *Multiple Segments* datasets, FKIC failed to accurately model 12 of the 45 segments in the *Standard* 12-residue benchmark dataset. In four cases (1cs6, 1msc, 2tgi and 4i1b) no sub-Å model was generated. In the other eight cases (1arb, 1bhe, 1cyo, 1m3s, 1onc, 1qlw, 1t1d and 1thg), sub-Å models were generated but could not be identified by energy (the RMSDs of lowest energy structures were larger than 1.1Å). These failures could result from deficiencies in sampling near-native conformations, from inaccuracies in the energy function and/or from problems with the crystal structure conformation such as effects of crystal packing. Sampling and energy function errors are often coupled, as the energy function guides sampling during the simulations. To gain insights into potential reasons for the failures we observed, we ran simulations of the failed proteins starting from their native structures as inputs. In these simulations, we skipped the first build stage (yellow in **Fig. 2.5**) so that the native bond angles and bond lengths were kept.

The results of these simulations allowed us to classify failure cases into 4 categories (**Fig. 2.6**):

(1) Only the native-input simulations generate sub-Å models, which are correctly identified by energy. This occurred in two of the 12 failure cases (1cs6 and 2tgi). As the energies of native-like models are much lower than the non-native decoys (**Fig. 2.6a**), failures in these cases are most likely due to the insufficient sampling.

(2) Both standard and native-input FKIC generate sub-Å models, but these models are only correctly identified by lowest energy in the native-input simulations (three of the 12 failure cases: 1bhe, 1onc and 1t1d). As the native-input simulations generated a larger number of correct models with lower energies (**Fig. 2.6b**), the failures are likely caused by the failure of Rosetta to efficiently sample near-native energy minima. One of the possible explanations is that the standard simulation idealizes bond lengths and bond angles in standard FKIC. Because of the rugged energy landscape, small conformational changes can result in significant energy differences.

(3) As in (2), both simulations generate sub-Å models and native-input simulations correctly identify these models by energy, but standard FKIC generates incorrect models with

lower energies (**Fig. 2.6c,** two of the 12 failure cases: 1arb and 1qlw). This behavior indicates linked scoring and sampling deficiencies.

(4) Neither standard nor native input simulations generate sub-Å models (**Fig. 2.6d**; five of 12 failure cases: 1cyo, 1m3s, 1msc, 1thg and 4i1b). While these simulations start from the native backbones, they do not include crystal contacts. Because crystal packing affecting loop conformations is well known[64], [65], there is a formal possibility that the failures of 1cyo, 1m3s, 1msc and 4i1b are due to crystal packing. For the lowest energy models for 1cyo and 4i1b, the incorrectly modeled loops would have unfavorable contacts in the crystal lattice. For 1m3s and 1msc, the native loop conformations make contacts with another monomer in the crystal. For 1thg, the RMSD of the lowest energy structure improved from 1.86Å to 1.12Å when including native bond lengths and angles, so there might be both sampling and energy function problems.

In sum, in particular for categories (1) and (2), it could be beneficial to incorporate sampling of bond lengths and bond angles, which we kept to their idealized values to reduce the conformational space to be sampled. Category (3) is indicative of energy function failures although we note that sampling and scoring are coupled in our simulations that accept or reject models based on their energies. Category (4) identifies a number of cases where crystal packing may influence the conformation of the modeled segment in the experimentally determined structure.

2.7.2 Supplementary Note 2: 8-Residue and 12-residue Multiple Segments datasets

To further benchmark the performance of FKIC on multiple interacting segments, we constructed a dataset of 8-residue interacting segments and a dataset of 12-residue interacting segments in a similar manner to the construction of the 10-residue *Multiple Segments* dataset (see **Appendix I: Supplementary Methods**). On the 8-residue interacting segment dataset, FKIC has 0.65Å median accuracy and 59.9% median fraction of sub-Å prediction; NGK has 0.79Å median accuracy and 35.6% median fraction of sub-Å prediction (median accuracy and median fraction of sub-Å prediction are described in the main text). On the 12-residue interacting dataset, FKIC has 1.53Å median accuracy and 0.21% median fraction of sub-Å prediction; NGK has 1.94Å median accuracy and 0% median fraction of sub-Å prediction. Thus, FKIC improves the prediction accuracy for multiple interacting segments consistently on different segment lengths and is able to find correct solutions for large conformational search problems, such as the set with two interacting 12-residue segments where previous methods such as NGK and CCD frequently fail (**Table 2.7** and **Table 2.8**).

2.7.3 Supplementary Note 3: Benchmark LHKIC on structure prediction

The LHKIC method was developed for loop design, but it can also be used for structure prediction. We benchmarked the prediction performance of LHKIC on all three benchmark datasets (**Table 2.2**). The performance of LHKIC is comparable to NGK and FKIC in terms of median RMSD of lowest scoring models. For the *Standard* and *Mixed Segment* datasets, the median sub-Å fraction of LHKIC is 24.35% and 15.40%, better than NGK but worse than FKIC. This result indicates that sequence-independent fragments (as in LHKIC) can improve sampling in structure predictions over non fragment-based methods such as NGK, but the improvement is smaller than when using fragments picked with sequence information as in FKIC (and for the *Multiple Segments* dataset, LHKIC had a median sub-Å fraction of 4.01%, which is lower than for both NGK and FKIC.) Note that the absolute energies for LHKIC cannot be directly compared to the other methods since LHKIC was developed in a newer version of Rosetta (revision 60022, see **Methods**).

2.7.4 Supplementary Note 4: Catalytic activity of KSI designs

V1D8r and V2D9r were overall less active than both wild-type and D38E KSI (**Table 2.1**). This result might be expected for several reasons: First, while we took steps to avoid mutations in residues known to be important for catalysis, we still made extensive changes (19 and 12 mutations in design V1D8r and V2D9r, respectively) in and around the active site, which could change the electrostatic environment as well as affect functional or non-productive dynamics that impact catalysis. Second, while wild-type KSI is a dimer, our designs (while modeled as a dimer) are monomeric at the concentrations of the enzyme assay (**Fig. 2.9**) although dimeric in the crystal. These differences could affect functional group positioning in solution[66]. Third, even though the glutamate side chain placement in V2D9r was close to ideal, it was not perfect; even small perturbations towards nonproductive conformations can be significant to catalysis and the designed glutamate may only sample catalytic conformations a fraction of the time. Finally, there is evidence that the catalytic residue in the homologous *Pseudomonas putida* KSI accesses multiple specific productive conformations to enable its participation throughout the catalytic cycle[67], a property which was not considered by our protocol. Despite these difficulties, our designed enzymes still enhanced the catalysis of their substrate by 4 to 5 orders of magnitude when compared to the water-catalyzed isomerization of the similar 5-androstene-3,17-dione[36].

2.7.5 Supplementary Note 5: Analysis of sequence differences between wildtype KSI and V2D9r

The mutational effects of alanine and wildtype reversion mutations on catalytic activity of V2D9r are all rather small, except for mutations at the catalytic E38 to either aspartate or alanine. Most of the other changes are within error (**Fig. 2.4f**, shaded bar). Exceptions are Q41A that has slightly decreased activity and T39S that has a slightly increased activity. Q41 is the residue with the largest shift in Cα position between aligned wildtype and V2D9r crystal structures (**Table 2.15**). T39S is a manually designed mutation to avoid a potential issue with burial of polar groups. Nevertheless, none of the individual side chain mutations (apart from mutations to the catalytic glutamate) show more than 2-fold effects on kinetics. This observation suggests that there are not individual key side chain-mediated interactions that determine the new loop conformation, but that many interactions each contribute to a smaller extent.

We also analyzed the local sequence-structure compatibility of the sequences of wildtype and V2D9r with the two structures in the reshaped segment using the Rosetta fragment quality metric (**Appendix I: Supplementary Methods**). This metric does not consider specific side chain interactions but instead evaluates a tertiary-structure-independent compatibility of the primary sequence for a given structure. **Figure 2.10f** shows that the wild-type sequence has a lower (better) fragment RMSD for the wild-type structure at all positions in the reshaped region, whereas the sequence of V2D9r has a lower fragment RMSD for the designed structure at most positions in the designed regions. These observations are consistent with the idea that the structure of V2D9r in the reshaped region is at least partially dependent on local sequence-structure compatibility in addition to tertiary interactions.

2.7.6 Supplementary Note 6: Loop modeling on template-based models and perturbed datasets

To test the performance of FKIC in contexts where the environment of the remodeled loop is non-native, we benchmarked FKIC on several datasets from ref [27]: a benchmark set from template-based modeling, and three sidechain/backbone perturbed loop datasets (**Table 2.12**). On the side chain perturbed datasets, FKIC performs similarly to NGK, and outperforms the other methods. This behavior is likely due to the fact that surrounding residues are repacked during FKIC or NGK simulations, indicating that these methods can account for slight imperfections in the environment that can be resolved by altering side chain conformations. On the backbone-perturbed dataset, the performance of FKIC is comparable to the reported results of GalaxyLoop-PS2 [27], with a median RMSD of lowest-scoring models of 1.68Å and 1.65Å for FKIC and Galaxy-PS2, respectively. On the template-based model dataset, none of the methods performs well, with the median RMSD of lowest scoring models above 3 Å for all methods.

It should be noted that the prediction and evaluation approach described here is not suited to appropriately assess the accuracy of loop modeling methods in environments where the surrounding backbone is perturbed, since the backbone in the environment is not allowed to change during the simulation. In our study, we compare a predicted loop structure to the native loop structure after aligning the surrounding environment of the loop. The native loop is the correct answer when the surrounding environment is unperturbed. However, the native loop conformation may not be compatible with the perturbed backbone and can therefore not be identified as the lowest scoring model in simulations where the backbone of the environment remains in its (unchanged) perturbed conformation. Our analysis on the dataset containing backbone perturbations after MD simulations from ref[27] supports this argument. For each protein in this dataset, we defined the residues within 10 Å from the native loop as surrounding residues. We then superimposed the native structure and the perturbed structure by the backbone heavy atoms of the surrounding residues and calculated heavy atom steric clashes between the native loop and the perturbed surrounding backbone atoms. Two heavy atoms were defined as clashing if their interatomic distance was within 2.5 Å. This analysis revealed that twelve out of the twenty native loops in the dataset contained clashes with their perturbed environments that cannot be resolved with simulations that do not relax the surrounding backbone. When excluding these cases, the FKIC and GalaxyLoop-PS2 median RMSDs improved to 1.3 Å and 1.4 Å, respectively (**Table 2.12**).

These considerations highlight that modeling loops in perturbed environments such as homology models remains an important unsolved problem. To adapt the design-centric loop modeling methods presented here to the problem of homology modeling should include simultaneous or iterative refinement of both loop structures and the environment.

2.8 Appendix III: Supplementary Tables

Table 2.2. Datasets and performance summary

a) Comparison of methods

Dataset	Sampling method	Rosetta energy function	Median RMSD of lowest energy model (Å)	Median RMSD of lowest RMSD model (Å)	Median RMSD all models (Å)	Median sub-A fraction	Median lowest energy (REU)	Median time (s)
Standard	KIC*	score12	1.05	NA	NA	4.30%	NA	NA
Standard	CCD	ref2015	1.26	0.47	3.27	2.00%	-709.18	2299
Standard	NGK	ref2015	0.64	0.37	2.70	13.00%	-712.65	3642
Standard	FKIC	ref2015	0.62	**0.32**	**1.16**	**47.80%**	**-716.78**	3456
Standard	LHKIC	ref2015	**0.55****	0.34	2.66	24.35%	-655.18***	3057
Mixed	CCD	ref2015	1.29	0.67	3.46	0.50%	-719.42	4309
Mixed	NGK	ref2015	1.07	0.45	4.65	1.15%	-728.18	7341
Mixed	FKIC	ref2015	0.53	**0.34**	**1.46**	**52.30%**	**-739.38**	7196
Mixed	LHKIC	ref2015	**0.48**	0.35	4.14	15.40%	-693.95***	5322
Multiple	CCD	ref2015	1.97	0.90	2.95	0.20%	-557.06	5204
Multiple	NGK	ref2015	1.29	0.52	2.35	5.50%	-573.60	9472
Multiple	FKIC	ref2015	**1.00**	**0.41**	**1.82**	**28.50%**	**-581.42**	8834
Multiple	LHKIC****	ref2015	1.31	0.55	2.50	4.01%	-517.69***	7831

REU, Rosetta energy units
* taken from ref. [17]; all other simulations were run using the Rosetta energy function "ref2015" as described in ref.[25]
** bold numbers denote best performance for given dataset
*** REU value not directly comparable to other methods as LHKIC was benchmarked using a more recent Rosetta version (see **Methods**).
**** PDB 1FO9 was excluded from the *Multiple Segments* benchmark set for LHKIC because for this case simulations failed to converge during minimization for the majority of simulations.

b) Inclusion of fragments from homologous structures

Dataset	Sampling method	Rosetta energy function	Median RMSD of lowest scoring model (Å)	Median RMSD of lowest RMSD model (Å)	Median RMSD all models (Å)	Median sub-A fraction	Median lowest energy (REU)	Median time (s)
Standard	FKIC	talaris2013	0.70	0.36	1.19	44.89%	-274.50	1646
Standard	FKIC (+ homologs)	talaris2013	**0.59**	**0.33**	**0.85**	**66.80%**	-273.29	1919

Table 2.3. Mixed Segment dataset detailed performance

		CCD					NGK					FKIC				
PDB name	Target segment residues	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	Fraction sub-Å models	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	Fraction sub-Å models	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	Fraction sub-Å models
1a8d	275-290	3.04	-984.24	1.94	-944.26	0.00	2.94	-992.55	0.77	-984.31	0.01	0.42	-1000.57	0.38	-974.06	0.04
1arb	165-180	4.80	-510.37	3.21	-493.67	0.00	6.23	-528.00	1.17	-507.84	0.00	6.48	-531.35	2.31	-509.84	0.00
1bhe	340-355	2.10	-820.58	0.72	-815.20	0.00	0.66	-843.95	0.53	-824.24	0.36	0.76	-835.54	0.65	-825.45	0.07
1bn8	38-53	3.08	-937.27	1.37	-928.40	0.00	0.55	-946.28	0.44	-941.02	0.03	0.52	-948.32	0.45	-943.16	0.03
1c5e	57-72	2.77	-691.08	1.33	-524.80	0.00	3.57	-706.51	0.50	-699.10	0.03	0.29	-709.42	0.29	-709.42	0.67
1cb0	91-106	0.63	-662.32	0.47	-651.31	0.28	0.68	-668.11	0.30	-663.53	0.97	0.61	-683.38	0.34	-663.29	0.96
1cs6	46-61	5.27	-779.57	0.87	-766.75	0.00	1.34	-789.74	0.74	-781.50	0.00	1.27	-796.08	0.51	-770.26	0.16
1dqz	103-118	0.50	-1243.04	0.36	-1207.68	0.33	2.40	-1255.72	0.45	-1222.43	0.01	0.35	-1261.68	0.24	-1198.37	0.84
1ede	76-91	0.38	-694.76	0.38	-694.76	0.03	0.26	-710.85	0.19	-689.15	0.37	0.28	-719.71	0.21	-706.81	0.72
1exm	252-267	1.45	-977.38	0.66	-969.81	0.01	4.62	-983.36	1.38	-962.42	0.00	4.38	-991.09	0.38	-983.50	0.21
1ezm	24-39	6.88	-654.25	3.31	-556.10	0.00	0.50	-684.52	0.40	-668.73	0.00	6.00	-681.88	1.37	-611.59	0.00
1f46	45-60	5.48	-705.82	1.07	-701.81	0.00	7.10	-723.09	0.99	-708.18	0.00	1.51	-735.38	1.08	-704.28	0.00
1i7p	155-170	0.68	-647.33	0.68	-647.33	0.00	1.59	-654.72	1.51	-648.48	0.00	0.56	-657.64	0.37	-652.65	0.46
1ms9	431-446	10.98	-2721.15	4.21	-2392.51	0.00	11.83	-2744.95	3.10	-2427.70	0.00	6.42	-2749.58	0.70	-2079.87	0.01
1oth	272-287	1.31	-612.54	0.74	-510.97	0.01	2.24	-685.23	0.45	-555.42	0.02	0.71	-673.30	0.38	-559.34	0.80
1oyc	330-345	1.35	-557.96	0.48	-137.82	0.08	5.66	-536.60	0.59	126.42	0.00	0.54	-593.99	0.39	-471.11	0.65
1pbe	200-215	0.35	-803.31	0.35	-803.31	0.17	0.59	-814.05	0.31	-807.25	0.25	0.57	-816.86	0.25	-813.62	0.58
1qlw	284-299	0.46	-1557.59	0.39	-1547.37	0.09	0.52	-1566.76	0.34	-1561.36	0.06	0.35	-1573.51	0.18	-1569.30	0.73
1srp	418-433	0.53	-733.70	0.53	-733.70	0.00	0.67	-733.27	0.65	-729.35	0.01	0.42	-743.39	0.29	-739.06	0.32
1tca	163-178	2.33	-829.93	1.05	-815.78	0.00	0.28	-850.81	0.28	-850.81	0.08	0.36	-852.46	0.27	-850.07	0.17
1thg	359-374	3.78	-1290.91	0.98	-798.31	0.00	7.31	-836.82	2.20	-782.91	0.00	0.87	-820.01	0.76	-793.95	0.28
1thw	97-112	0.50	-365.34	0.50	-365.34	0.00	0.33	-404.36	0.28	-361.20	0.35	0.34	-406.53	0.27	-361.89	0.61
1tib	166-181	0.49	-469.26	0.49	-469.26	0.07	0.30	-513.50	0.25	-488.46	0.71	0.29	-512.87	0.25	-499.93	0.81
1tml	26-41	0.83	-733.03	0.69	-729.62	0.02	0.35	-756.99	0.34	-745.83	0.04	0.35	-764.53	0.26	-756.61	0.80
1xif	50-65	1.28	-815.74	0.47	-813.47	0.02	0.45	-820.65	0.37	-819.84	0.01	0.59	-829.72	0.32	-820.15	0.35
2ebn	11-26	0.50	-686.47	0.50	-686.47	0.03	0.31	-715.52	0.25	-708.04	0.09	0.29	-719.70	0.24	-711.08	0.74
2exo	48-63	3.51	-615.43	1.17	-602.03	0.00	4.79	-654.47	2.30	-602.47	0.00	4.88	-680.12	0.43	-626.40	0.07
2pia	43-58	1.08	-666.90	0.37	-594.66	0.33	1.52	-683.56	0.39	-581.39	0.43	0.45	-682.60	0.34	-600.69	0.96
2sil	209-224	0.65	-440.99	0.35	-366.67	0.13	6.82	-523.38	0.37	-370.74	0.01	0.48	-551.19	0.29	-365.12	0.67
3hsc	125-140	0.33	-874.75	0.31	-869.11	0.99	0.81	-868.57	0.81	-868.57	0.01	0.30	-878.17	0.27	-873.40	0.80

Table 2.4. Multiple Segments dataset detailed performance

		CCD					NGK					FKIC				
PDB name	Target segment residues	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	Fraction sub-Å models	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	Fraction sub-Å models	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	Fraction sub-Å models
1a3a	11-20 87-96	5.65	-340.13	0.83	-284.99	0.00	0.84	-316.80	0.45	-246.79	0.13	0.84	-325.15	0.48	-254.18	0.58
1deu	170-179 225-234	6.49	16511.10	1.81	16591.50	0.00	2.21	16114.00	1.56	16135.00	0.00	2.57	16115.50	0.97	16143.30	0.00
1dqz	148-157 252-261	2.62	-623.02	1.45	-555.00	0.00	1.93	-671.27	0.41	-637.28	0.12	2.07	-667.08	0.55	-626.29	0.06
1euv	529-538 570-579	2.17	-500.01	1.56	-474.99	0.00	1.88	-521.07	0.96	-513.85	0.00	1.10	-523.17	0.93	-517.57	0.01
1fo9	239-248 261-270	1.09	-774.02	1.09	-774.02	0.00	0.84	-804.34	0.43	-795.79	0.67	0.81	-811.44	0.40	-797.06	0.83
1ftr	211-220 278-287	9.14	-700.66	2.23	-670.62	0.00	8.51	-719.42	2.41	-696.58	0.00	3.02	-714.95	0.77	-712.83	0.00
1h1n	116-125 153-162	0.49	-651.04	0.43	-637.42	0.17	1.29	-670.91	0.25	-633.85	0.36	0.36	-671.26	0.23	-668.07	0.60
1h6u	170-179 202-211	0.37	-728.01	0.36	-724.78	0.45	0.37	-732.42	0.32	-729.53	0.28	0.42	-734.81	0.28	-730.74	0.38
1i7k	85-94 118-127	3.33	-308.26	1.79	-295.82	0.00	3.11	-326.13	1.51	-317.13	0.00	1.95	-325.51	1.06	-308.24	0.00
1idp	71-80 111-120	0.65	-289.74	0.44	-286.54	0.11	4.43	-302.14	0.60	-296.58	0.02	0.77	-306.48	0.28	-296.60	0.80
1inl	251-260 266-275	2.65	-520.07	0.67	-432.87	0.00	0.52	-579.68	0.43	-562.67	0.07	0.57	-586.58	0.40	-558.11	0.55
1j7d	35-44 69-78	0.61	-332.51	0.47	-324.83	0.04	1.30	-337.14	1.01	-328.33	0.00	2.01	-341.38	0.93	-329.21	0.02
1jfr	44-53 15-24	3.61	-554.55	0.71	-434.64	0.01	1.28	-567.51	0.38	-469.28	0.42	7.05	-576.27	0.36	-494.83	0.17
1jfu	34-43 136-145	1.38	-407.70	0.91	-398.11	0.00	5.58	-414.98	1.04	-403.42	0.00	1.19	-415.17	0.91	-397.13	0.02
1ku1	628-637 645-654	0.67	550.86	0.58	551.87	0.22	0.35	445.64	0.35	542.16	0.22	0.40	445.59	0.27	455.27	0.83
1kzq	44-53 83-92	3.58	-444.31	1.87	-407.29	0.00	2.61	-466.18	1.72	-458.54	0.00	1.99	-463.25	0.85	-450.18	0.00
1m0z	71-80 49-58	0.72	-559.57	0.57	-550.88	0.15	0.73	-564.57	0.51	-558.06	0.05	0.66	-571.10	0.40	-560.56	0.76
1nxm	12-21 98-107	0.40	-465.63	0.24	-453.77	0.47	0.32	-482.36	0.23	-468.24	0.92	0.24	-476.32	0.22	-468.21	0.98
1qwd	18-27 93-102	2.97	-376.26	1.84	-368.08	0.00	1.17	-393.42	0.78	-388.26	0.00	0.90	-389.35	0.36	-383.74	0.01
1t6g	140-149 197-206	1.00	-753.66	0.81	-737.17	0.01	1.17	-777.29	0.58	-753.35	0.17	1.10	-777.69	0.56	-763.90	0.44
1u09	240-249 293-302	1.85	-916.14	1.23	-893.76	0.00	2.13	-1110.23	1.00	-902.43	0.00	3.31	-933.51	0.39	-926.61	0.01
1w0d	275-284 315-324	1.84	-663.35	0.86	-662.27	0.00	1.61	-670.13	0.54	-664.68	0.09	0.75	-669.27	0.41	-662.14	0.66

Table 2.4. Multiple Segments dataset detailed performance (continued)

		CCD					NGK					FKIC				
PDB name	Target segment residues	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	Fraction sub-Å models	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	Fraction sub-Å models	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	Fraction sub-Å models
1xdw	49-58 27-36	0.89	-783.62	0.89	-783.62	0.00	0.68	-805.64	0.59	-797.40	0.42	0.68	-811.82	0.42	-802.75	0.62
1xg2	184-193 218-227	3.59	-594.97	0.93	-554.27	0.00	2.33	-637.06	0.49	-614.76	0.01	0.52	-638.36	0.41	-621.65	0.56
1xsz	77-86 119-128	0.91	-815.59	0.41	-774.90	0.57	0.49	-816.42	0.32	-793.45	0.35	0.58	-823.76	0.32	-795.46	0.95
1xwt	333-342 369-378	2.10	-941.99	0.92	-923.93	0.00	1.57	-976.37	0.35	-953.47	0.44	1.82	-977.69	0.40	-950.40	0.34
1yif	176-185 144-153	3.06	-748.46	2.18	-682.37	0.00	2.50	-761.17	1.81	-759.79	0.00	2.92	-765.53	1.88	-755.31	0.00
1zvt	659-668 716-725	3.72	-508.86	2.52	-505.57	0.00	0.32	-542.65	0.32	-542.65	0.07	1.49	-535.43	1.28	-531.60	0.00
2a4a	37-46 18-27	2.44	-598.61	1.60	-450.13	0.00	2.01	-620.40	0.71	-594.80	0.00	1.84	-631.78	0.93	-610.00	0.00
2b0a	102-111 123-132	0.45	-395.53	0.45	-395.53	0.00	0.42	-404.93	0.42	-404.93	0.04	0.38	-408.45	0.31	-395.35	0.24

Table 2.5. *Standard* dataset detailed performance

		CCD					NGK					FKIC				
PDB name	Target segment residues	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	Fraction sub-Å models	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	Fraction sub-Å models	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	Fraction sub-Å models
1a8d	155-166	2.83	-1008.38	2.14	-973.52	0.00	0.38	-1024.27	0.38	-1024.27	0.03	0.42	-1022.05	0.31	-1019.72	0.01
1arb	182-193	2.37	-554.35	0.23	-530.49	0.01	0.54	-562.94	0.37	-539.98	0.38	2.06	-563.70	0.39	-526.90	0.17
1bhe	121-132	2.09	-923.69	1.84	-913.87	0.00	0.68	-908.66	0.30	-903.45	0.18	1.91	-915.76	0.31	-906.99	0.13
1bn8	298-309	0.70	-987.69	0.33	-982.93	0.02	0.75	-993.21	0.43	-986.78	0.09	0.48	-1000.64	0.30	-991.29	0.44
1c5e	82-93	0.40	-742.69	0.30	-742.50	0.09	0.36	-745.93	0.28	-742.69	0.41	0.38	-748.17	0.31	-744.34	0.90
1cb0	33-44	0.41	-679.12	0.30	-671.31	0.10	0.25	-679.95	0.16	-651.78	0.36	0.36	-680.65	0.14	-675.44	0.84
1cnv	188-199	3.09	-691.00	1.24	-665.03	0.00	1.06	-701.43	0.47	-685.34	0.01	1.02	-700.87	0.39	-699.85	0.03
1cs6	145-156	3.60	-731.51	1.71	-704.86	0.00	4.26	-777.82	0.99	-760.81	0.00	4.24	-773.94	1.55	-745.58	0.00
1cyo	12-23	0.91	-192.93	0.58	-191.35	0.03	4.98	-197.51	0.53	-189.01	0.02	5.09	-196.90	0.51	-191.38	0.01
1dqz	209-220	0.40	-1471.41	0.40	-1471.41	0.00	0.49	-1476.24	0.30	-1456.86	0.33	0.46	-1480.22	0.32	-1325.32	0.32
1dts	41-52	7.66	-445.87	0.95	-436.78	0.00	6.63	-470.55	0.94	-436.96	0.01	1.07	-472.01	0.75	-417.87	0.16
1eco	35-46	0.36	-283.70	0.29	-267.09	0.63	0.40	-284.48	0.32	-277.13	0.49	0.37	-285.75	0.30	-269.98	0.99
1ede	150-161	1.26	-598.32	0.76	-592.01	0.01	0.72	-615.96	0.59	-612.93	0.05	0.77	-631.71	0.54	-613.56	0.21
1exm	291-302	0.52	-1062.06	0.47	-1051.67	0.35	0.47	-1048.50	0.27	-1040.46	0.62	0.63	-1048.74	0.27	-1039.80	0.89
1ezm	122-133	2.53	-733.12	1.25	-719.78	0.00	0.37	-758.95	0.37	-758.95	0.04	0.62	-753.62	0.44	-747.20	0.03
1f46	64-75	2.27	-709.18	0.80	-697.38	0.03	3.09	-712.65	0.78	-705.47	0.14	0.44	-716.78	0.32	-710.87	0.30
1i7p	63-74	0.51	-669.97	0.36	-661.74	0.11	0.42	-674.13	0.31	-651.77	0.53	0.38	-672.19	0.33	-669.48	0.90
1m3s	68-79	5.40	-672.68	0.68	-666.48	0.01	5.05	-674.66	0.44	-664.02	0.31	5.66	-673.18	0.42	-664.04	0.51
1ms9	529-540	2.48	-3018.90	0.77	-2998.79	0.04	0.27	-3041.97	0.22	-3041.75	0.91	0.35	-3044.20	0.23	-3030.97	0.69
1msc	9-20	7.68	151.93	2.93	188.30	0.00	6.54	148.08	1.27	162.88	0.00	7.85	147.53	1.08	155.56	0.00
1my7	254-265	0.49	-504.92	0.35	-496.96	0.11	0.61	-511.55	0.40	-507.90	0.75	0.53	-512.36	0.45	-507.57	0.81
1onc	23-34	2.88	-200.66	1.70	-190.59	0.00	3.63	-209.76	0.46	-208.03	0.04	3.62	-208.46	0.93	-200.42	0.00
1oth	69-80	0.52	-837.57	0.38	-836.26	0.18	0.45	-837.27	0.31	-834.46	0.34	0.50	-843.18	0.27	-833.69	0.97
1oyc	203-214	2.99	-776.66	0.89	-691.71	0.00	0.30	-818.69	0.28	-719.38	0.13	0.32	-819.70	0.25	-727.24	0.54
1pbe	129-140	2.60	-833.07	0.43	-790.79	0.06	0.45	-820.65	0.32	-801.51	0.32	0.38	-824.99	0.33	-804.46	0.72
1qlw	31-42	4.34	-1611.33	2.07	-1604.34	0.00	4.87	-1624.91	0.30	-1605.29	0.13	2.83	-1626.69	0.33	-1620.54	0.06
1rro	17-28	0.44	-272.26	0.34	-270.03	0.08	0.64	-277.16	0.39	-270.25	0.10	0.63	-279.08	0.32	-269.39	0.48
1srp	311-322	0.48	-782.41	0.31	-773.57	0.05	0.31	-877.44	0.25	-772.31	0.99	0.37	-879.03	0.25	-759.90	0.98
1t1d	127-138	0.42	-206.93	0.34	-204.69	0.14	2.79	-217.61	0.32	-207.60	0.09	2.94	-215.11	0.30	-209.60	0.39
1tca	305-316	9.52	-846.00	2.37	-806.46	0.00	0.66	-859.34	0.26	-854.54	0.12	0.29	-864.98	0.24	-848.65	0.24
1thg	127-138	2.88	-1419.28	1.64	-1385.22	0.00	1.05	-1434.24	0.71	-1411.26	0.07	1.86	-1434.72	0.28	-1423.84	0.07
1thw	178-189	1.06	-394.74	1.06	-394.74	0.00	0.65	-413.28	0.49	-411.02	0.05	0.61	-414.92	0.46	-409.31	0.23
1tib	99-110	0.64	-527.93	0.39	-524.55	0.10	0.75	-529.58	0.40	-526.29	0.07	0.63	-535.99	0.39	-529.46	0.57
1tml	243-254	0.46	-766.55	0.38	-765.86	0.05	0.50	-776.42	0.25	-773.23	0.51	0.52	-777.92	0.28	-774.05	0.79
1xif	203-214	1.90	-821.11	1.26	-814.10	0.00	0.41	-838.28	0.18	-828.66	0.94	0.37	-839.04	0.22	-833.59	0.85
2cpl	145-156	0.60	-457.29	0.26	-453.37	0.26	0.32	-459.00	0.21	-451.22	0.26	0.30	-458.11	0.24	-454.07	0.99
2ebn	136-147	0.35	-722.21	0.35	-722.21	0.02	0.44	-719.92	0.43	-719.89	0.01	0.60	-721.73	0.43	-718.80	0.09
2exo	293-304	0.61	-737.11	0.41	-728.13	0.12	0.49	-731.90	0.42	-722.45	0.18	0.47	-734.53	0.35	-725.14	0.99
2pia	30-41	1.00	-686.94	0.38	-662.42	0.55	0.82	-688.54	0.48	-680.08	0.99	0.86	-688.99	0.56	-673.23	1.00
2rn2	90-101	1.41	-283.79	1.41	-283.79	0.00	0.64	-297.34	0.32	-289.54	0.40	0.68	-298.25	0.29	-290.57	0.69
2sil	255-266	0.69	-744.26	0.41	-732.81	0.55	1.26	-745.15	0.60	-734.86	0.00	0.93	-747.04	0.46	-740.32	0.57
2tgi	48-59	2.69	-140.59	1.73	-121.99	0.00	3.57	-162.15	0.36	-89.29	0.00	3.14	-163.39	1.36	-87.48	0.00
3cla	176-187	1.32	-475.15	0.41	-472.78	0.21	1.21	-483.29	0.39	-478.16	0.04	0.64	-484.91	0.28	-477.79	0.63
3hsc	72-83	0.51	-894.79	0.46	-891.28	0.04	0.32	-896.76	0.25	-890.33	0.72	0.56	-899.54	0.25	-893.37	0.94
4i1b	46-57	3.63	-134.68	2.14	-130.51	0.00	6.69	-140.23	0.82	-135.44	0.00	6.63	-140.98	1.04	-97.29	0.00

Table 2.6. Failure cases, *Standard* dataset

		FKIC				FKIC native input				Reason for failure
PDB name	Target segment residues	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	
1arb	182-193	2.06	-563.70	0.39	-526.90	0.40	-557.64	0.21	-537.67	insufficient sampling
1bhe	121-132	1.91	-915.76	0.31	-906.99	0.24	-940.55	0.24	-928.05	insufficient sampling
1cs6	145-156	4.24	-773.94	1.55	-745.58	0.39	-809.13	0.25	-804.74	insufficient sampling
1cyo	12-23	5.09	-196.90	0.51	-191.38	**1.06***	-194.88	0.32	-188.95	crystal packing
1m3s	68-79	5.66	-673.18	0.42	-664.04	**5.39**	-680.83	0.43	-669.76	crystal packing
1msc	9-20	7.85	147.53	1.08	155.56	**8.74**	146.72	0.86	163.39	crystal packing
1onc	23-34	3.62	-208.46	0.93	-200.42	0.51	-208.74	0.39	-200.49	insufficient sampling
1qlw	31-42	2.83	-1626.69	0.33	-1620.54	0.48	-1623.97	0.29	-1595.85	insufficient sampling / energy function deficiency
1t1d	127-138	2.94	-215.11	0.30	-209.60	0.33	-220.36	0.22	-217.79	insufficient sampling
1thg	127-138	1.86	-1434.72	0.28	-1423.84	**1.12**	-1451.32	0.23	-1434.68	insufficient sampling / energy function deficiency
2tgi	48-59	3.14	-163.39	1.36	-87.48	0.41	-189.26	0.31	-182.44	insufficient sampling
4i1b	46-57	6.63	-140.98	1.04	-97.29	**6.64**	-140.95	0.81	-137.17	crystal packing

Table 2.7. *Multiple Segments* (12 residues) dataset detailed performance

		NGK					FKIC				
PDB name	Target segment residues	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	Fraction sub-Å models	RMSD of lowest energy Model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	Fraction sub-Å models
1ceo	10-21, 53-64	3.74	-786.28	2.57	-763.87	0.00	2.47	-801.47	1.60	-774.43	0.00
1deu	47-58, 117-128	2.07	16572.20	1.27	16605.60	0.00	1.05	16537.10	0.53	16568.70	0.06
1dqz	146-157, 250-261	2.78	-657.72	0.89	-627.68	0.00	0.80	-656.04	0.80	-656.04	0.00
1euv	553-564, 527-538	1.35	-497.87	1.16	-480.99	0.00	2.59	-502.24	1.58	-495.34	0.00
1ftr	209-220, 276-287	9.25	-717.41	2.57	-698.25	0.00	7.33	-723.89	2.19	-708.61	0.00
1h6u	168-179, 190-201	1.01	-715.67	1.01	-715.67	0.00	0.40	-745.35	0.26	-733.33	0.64
1i7k	138-149, 66-77	1.80	-322.00	0.78	-312.85	0.00	0.47	-327.72	0.36	-321.18	0.30
1j7d	33-44, 67-78	2.30	-329.13	1.35	-323.97	0.00	1.98	-343.43	0.92	-333.10	0.01
1jfu	32-43, 134-145	5.49	-409.27	1.49	-391.57	0.00	1.29	-415.60	0.54	-400.58	0.05
1ku1	677-688, 729-740	4.37	567.65	1.65	576.01	0.00	1.11	560.84	0.67	563.89	0.02
1m0z	199-210, 244-255	3.35	-576.11	1.60	-126.88	0.00	2.17	-583.89	1.12	27.73	0.00
1qs1	264-275, 333-344	1.98	-957.12	1.77	-923.44	0.00	1.67	-970.92	0.90	-957.65	0.00
1ryl	54-65, 118-129	2.19	-363.80	1.73	-334.16	0.00	2.23	-362.49	1.75	-348.55	0.00
1suu	741-752, 780-791	6.21	-711.24	2.66	-698.90	0.00	3.11	-708.22	2.53	-692.72	0.00
1t6g	195-206, 220-231	1.10	-765.81	0.85	-747.86	0.01	1.87	-771.86	0.62	-756.32	0.17
1u09	238-249, 291-302	2.59	-945.68	1.25	-916.78	0.00	2.91	-952.12	0.60	-929.50	0.00
1v5d	282-293, 308-319	6.08	-938.87	4.75	-895.23	0.00	7.20	-945.93	5.15	-920.17	0.00
1w0d	65-76, 274-285	5.47	-661.63	1.25	-534.61	0.00	0.65	-676.38	0.31	-650.39	0.44
1wko	38-49, 109-120	1.52	-353.30	0.97	-321.36	0.00	1.19	-357.76	0.46	-321.97	0.08
1xwt	367-378, 38-49	5.47	-974.75	1.52	-954.86	0.00	2.62	-995.65	1.07	-950.23	0.00
1xyz	792-803, 813-824	7.55	-786.12	3.18	-737.98	0.00	1.99	-800.04	0.65	-767.73	0.00
1yif	142-153, 174-185	5.15	-773.41	2.34	-712.48	0.00	3.84	-781.24	1.91	-702.07	0.00
1zvt	657-668, 714-725	0.35	-545.67	0.32	-545.44	0.03	1.44	-542.72	1.41	-526.26	0.00
2ahf	182-193, 208-219	6.52	-707.08	3.03	-269.19	0.00	4.51	-775.93	0.80	-371.41	0.00
2b49	674-685, 652-663	0.55	-574.78	0.55	-574.78	0.04	2.88	-579.10	0.81	-572.12	0.00
2c61	410-421, 164-175	3.31	-981.07	0.74	-973.54	0.00	5.14	-987.68	1.29	-970.39	0.00
2cyg	248-259, 294-305	0.72	-728.09	0.72	-728.09	0.02	1.22	-753.92	0.58	-723.71	0.55
2e01	87-98, 256-267	2.02	-909.67	1.61	-894.78	0.00	0.44	-946.66	0.31	-922.67	0.75
2g30	790-801, 822-833	1.45	-500.35	1.18	-486.01	0.00	3.35	-486.53	1.61	-483.88	0.00
2hjv	299-310, 326-337	1.78	-390.05	1.08	-377.63	0.00	1.44	-393.43	1.35	-385.84	0.00

Table 2.8. *Multiple Segments* (8 residues) dataset detailed performance

		NGK					**FKIC**				
PDB name	Target segment residues	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	Fraction sub-Å models	RMSD of lowest energy model (Å)	Lowest energy (REU)	Lowest RMSD (Å)	Energy of lowest RMSD model (REU)	Fraction sub-Å models
1a8d	278-285,303-310	1.15	-998.03	0.35	-993.09	0.10	0.94	-983.98	0.73	-978.66	0.01
1a8u	225-232,251-258	0.94	-1073.51	0.94	-1073.51	0.00	0.65	-1081.86	0.58	-1036.74	0.42
1ako	150-157,173-180	0.45	-668.54	0.26	-666.92	0.66	0.43	-670.50	0.29	-662.28	0.68
1bhe	282-289,344-351	0.51	-877.73	0.28	-837.52	0.93	0.68	-875.49	0.38	-865.22	0.24
1bn8	250-257,338-345	1.98	-963.94	0.62	-959.38	0.05	2.45	-967.40	0.68	-949.27	0.00
1brt	28-35,205-212	0.76	-510.31	0.73	-500.12	0.21	0.81	-609.85	0.33	-488.96	0.94
1c5e	49-56,97-104	0.45	-702.21	0.29	-695.39	0.66	0.44	-703.36	0.26	-697.43	0.67
1cil	39-46,80-87	0.57	178.15	0.46	199.82	0.96	0.64	187.24	0.38	217.96	1.00
1cs6	126-133,158-165	1.10	-741.75	0.33	-719.69	0.57	0.72	-742.82	0.43	-741.61	0.73
1dqz	135-142,111-118	0.23	-1264.84	0.21	-1221.96	0.98	0.23	-1262.80	0.21	-1149.68	0.99
1ede	52-59,283-290	0.79	-689.00	0.69	-680.74	0.46	0.47	-694.65	0.32	-684.81	0.84
1exm	305-312,255-262	0.95	-998.75	0.30	-980.32	0.22	0.38	-994.71	0.31	-970.07	0.57
1gai	282-289,401-408	0.65	-306.61	0.26	-292.27	0.25	0.33	-294.19	0.28	-288.39	0.63
1gof	7-14,31-38	0.80	-1253.43	0.72	-1216.17	0.22	0.87	-1228.70	0.66	-1174.50	0.41
1jev	451-458,147-154	0.79	-1258.90	0.43	-1242.82	0.86	0.88	-1247.44	0.37	-1239.03	0.97
1ms9	433-440,468-475	2.27	-2629.19	0.32	-2392.50	0.14	2.74	-2411.81	0.29	-2332.46	0.02
1nif	64-71,221-228	1.49	-722.30	0.93	-643.85	0.00	1.50	-726.32	0.51	-651.66	0.14
1oth	219-226,280-287	0.30	-671.55	0.22	-640.91	1.00	0.27	-654.61	0.22	-649.22	1.00
1qlw	130-137,67-74	0.40	-1600.69	0.25	-1560.94	1.00	0.37	-1596.93	0.26	-1592.87	1.00
1srp	260-267,294-301	0.26	-760.01	0.19	-758.04	0.55	0.25	-760.73	0.19	-758.14	0.45
1tad	109-116,159-166	0.65	-1951.28	0.36	-1945.22	0.98	0.64	-1964.78	0.33	-1947.35	0.76
1thg	351-358,307-314	2.63	-979.60	2.07	-928.81	0.00	3.48	-977.31	1.80	-935.77	0.00
1tib	171-178,211-218	1.08	-466.52	0.67	-440.39	0.37	0.54	-471.50	0.29	-460.77	0.53
1udc	32-39,78-85	0.62	-668.79	0.54	-660.04	0.06	0.59	-667.57	0.46	-660.73	0.19
3bto	256-263,280-287	1.15	-3377.16	0.29	-3110.83	0.19	1.07	-3405.40	0.34	-3274.16	0.06
3grs	131-138,292-299	1.06	-998.99	0.29	-916.54	0.34	1.52	-1000.24	0.30	-929.86	0.67
4pga	112-119,216-223	0.60	-1315.25	0.24	-1309.43	0.97	0.57	-1380.06	0.26	-1313.87	1.00
6cel	132-139,367-374	1.36	-588.32	0.97	-568.66	0.00	1.70	-584.39	0.81	-573.68	0.00

Table 2.9. Parameters of designs selected for experimental testing.
Boxes on the left side indicate whether designs were based off of a full-length input structure or contained a 1-residue deletion (Del1) in the catalytic loop. Sequence cluster: Designs were clustered hierarchically (see **Appendix I: Supplementary Methods**) according to sequence distance, determined using the BLOSUM80 substitution matrix. Struct Cluster: Designs were clustered (see **Appendix I: Supplementary Methods**) according to the C/Cα/N/O RMSD for the positions where the backbone was remodeled. Largest Restraint Dist (Å) in lowest scoring model: The furthest distance between any of the atoms in the E38 carboxylate (Cδ, Oε1, or Oε2) and their target positions, for the lowest scoring loop modeling decoy. Loop RMSD (Å): The average RMSD in Å between loop modeling decoys and the input design structure. Score Gap: The difference in fa_attr score between the lowest scoring decoy that puts all of the atoms of the E38 carboxylate less that 1Å from their target positions, and the lowest scoring decoy that puts at least one atom of the E38 carboxylate more than 2Å from its target position. A score gap of 0 indicates that the lowest-scoring decoy was more than 2Å from its target position. % Sub-Å Restraints: The fraction of the loop modeling decoys that are predicted to position the all atoms of the Glu carboxylate less than 1Å from their target positions. % Sub-Å Loops: The fraction of loop modeling decoys that are predicted to position all backbone atoms (C/Cα/N/O) within 1Å RMSD of the input design structure.

	Design Name	*Sequence Cluster*	*Struct Cluster*	*Largest Restraint Dist (Å) in lowest scoring model*	*Loop RMSD (Å)*	*Score Gap (REU)*	*% Sub-Å Restraints*	*% Sub-Å Loops*
Full-length	V1D1r	1	1	0.83	0.20	3.16	7.60	13.00
	V1D2r	1	1	1.05	0.38	2.05	0.20	12.20
	V1D3r	2	1	0.55	0.20	6.50	7.20	4.60
	V1D4	2	1	0.68	0.24	6.56	6.60	5.20
	V1D5r	2	1	0.92	0.27	11.88	2.20	2.00
	V1D6	3	2	2.37	0.33	0.00	0.00	17.20
	V1D7	3	3	2.63	0.25	0.00	0.00	3.20
Del1	V1D8r	1	2	0.57	0.09	4.72	11.20	31.80
	V1D9r	1	2	0.80	0.08	5.09	11.40	20.00
	V1D10r	1	2	0.87	0.18	4.51	13.20	19.40
	V1D11	2	1	2.25	0.21	0.00	0.60	10.60
	V1D12r	2	1	2.35	0.32	0.00	3.20	20.40
	V1D13	2	1	2.36	1.22	0.00	1.40	6.80
	V1D14	3	3	2.31	0.23	0.00	0.00	4.60
Full-length	V2D1	3	2	0.67	0.32	8.16	5.80	7.25
	V2D2	3	2	0.83	0.73	8.97	4.23	7.04
	V2D3	3	2	0.91	0.38	5.54	8.45	14.08
	V2D4	3	2	0.71	0.23	5.35	5.88	5.88
	V2D5	3	2	0.72	0.24	10.31	1.41	1.41
	V2D6	1	4	0.95	0.16	0.90	19.12	63.24
	V2D7	5	1	1.03	0.61	5.52	5.71	32.86
	V2D8	5	1	1.05	0.81	2.92	5.88	35.29
	V2D9	2	3	1.15	0.13	0.22	4.42	71.46
	V2D10	1	1	0.86	0.71	0.09	8.57	61.10
	V2D11	1	1	0.87	0.66	12.02	26.70	54.92

Table 2.10a. List of mutations for PIP version 1 designs.
Designs appended with “r” indicate that mutations were reverted to the wild-type residue based on visual inspection; 8/14 designs from version 1 contained reversion mutations.

Design Name	*Mutations*
V1D1r	T35K, E37T, D38E, P39D, V40A, S42L, E43G, P44G, R45Y, S46Q, A50W, V74N, L115T
V1D2r	D33N, T35Y, D38E, P39S, V40A, S42Q, E43P, P44K, R45Y, S46W, A49D, E53K, S58Q, V74A, M112A, L115I, E118D
V1D3r	T35K, E37I, D38E, P39T, V40Q, G41Y, S42P, S46K, V74A, M112A
V1D4	F30Y, D32P, D33N, T35K, E37I, D38E, P39T, V40Q, G41Y, S42P, S46K, T48R, A49D, A50N, N57E, S58A, K60R, V74A, A75N, N76G, V109I, V110N, S111Y, M112A, R113Q, L115V, E118P
V1D5r	D33N, T35R, D38E, P39T, V40K, G41Y, S42P, P44D, S46K, S58Q, K60A, V74A, M112A, R113Q
V1D6	F30Y, A31D, D33T, T35R, V36R, E37N, D38E, P39I, V40G, S42P, E43P, R45L, S46P, T48R, A49D, A50N, E53K, N57E, S58D, K60A, V74T, A75N, N76G, V109I, V110S, S111Y, M112A, R113Q, L115V, E118D
V1D7	F30Y, A31S, D32P, T35I, V36R, E37R, D38E, P39R, V40Y, G41A, S42K, E43A, P44N, R45P, S46R, T48R, A49D, E53Q, N57E, S58D, K60A, V74T, A75N, N76G, V109I, V110A, S111E, M112A, R113Q, L115I, E118D
V1D8r	F30Y, D32S, E37T, D38E, P39S, V40F, G41-, S42R, E43P, R45F, S46T, A49E, V74A, A75N, N76G, V109I, S111Y, M112A, R113Q
V1D9r	E37R, D38E, P39S, V40F, G41-, S42R, E43P, R45F, S46T, S58N, V74S, A75N, S111E, M112A, R113Q
V1D10r	T35S, E37T, D38E, P39S, V40F, G41-, S42R, E43P, R45F, S46T, S58N, V74S, A75N, M112A
V1D11	F30Y, D32P, D33N, T35R, V36Y, E37D, D38E, P39I, V40G, G41-, S42F, E43P, P44D, R45T, S46G, A49D, A50N, E53A, N57K, S58Q, V74T, A75N, N76G, V109I, V110A, S111E, M112A, L115I, E118D
V1D12r	T35Q, V36Y, E37N, D38E, P39I, V40G, G41-, S42F, E43R, P44G, R45D, S58Q, A75N, N76G, S111Y, M112A
V1D13	F30Y, A31D, D32S, D33T, T35Q, V36Y, E37D, D38E, P39I, V40G, G41-, S42F, E43D, P44G, R45G, A49E, A50N, E53R, N57K, S58Q, V74T, A75N, N76G, V109I, V110A, S111E, M112A, L115I, E118D
V1D14	F30Y, A31D, D32S, D33T, T35I, V36R, E37Y, D38E, P39Q, V40Y, G41-, S42Y, P44G, R45G, S46K, A49D, A50N, E53K, N57M, S58D, V74T, A75N, N76G, V109I, V110A, S111E, M112A, L115I, E118D

Table 2.10b. List of mutations for PIP V2 designs.
Designs appended with "r" indicate that mutations were reverted to the wild-type residue based on visual inspection; 11/22 designs from version 2 contained reversion mutations.

Design Name	*Mutations*
V2D1	V27A, A28G, L29F, F30L, D33G, A34I, T35K, V36I, E37D, D38E, P39D, V40Q, G41N, S42R, E43K, P44Q, R45V, S46T, G47D, T48A, A50Q, I51K, A73S, A75S
V2D1r	D33G, A34I, V36I, E37D, D38E, P39D, G41N, S42R, E43K, P44Q, R45V, G47D, A50Q, A73S, A75S
V2D2	V27A, A28G, L29F, F30L, D33G, A34V, T35K, E37D, D38E, P39D, V40Q, G41K, S42K, E43T, P44T, R45V, S46T, G47D, T48A, A50Q, I51K, A73S, A75D
V2D2r	D33G, A34V, E37D, D38E, P39D, V40Q, G41K, S42K, E43T, P44T, R45V, G47D, A50Q, A75D
V2D3	V27A, A28G, L29F, F30L, D33G, A34V, T35Q, V36I, E37D, D38E, P39D, V40Q, G41N, S42K, E43T, P44T, R45V, S46T, G47D, T48A, A50Q, I51K, A73S, A75D
V2D3r	D33G, A34V, V36I, E37D, D38E, P39D, V40Q, G41N, S42K, E43T, P44T, R45V, G47D, A50Q, A75D
V2D4	V27A, A28G, L29F, F30L, D33G, A34V, T35Q, V36I, E37D, D38E, P39D, V40Q, G41Q, S42T, E43S, P44T, R45V, S46T, G47D, T48A, A50Q, I51K, A73S, A75K
V2D4r	D33G, A34V, V36I, E37D, D38E, P39D, V40Q, G41Q, S42T, E43S, P44T, R45V, G47D, A50Q, A75K
V2D5	V27A, A28G, L29F, F30L, D33G, A34V, V36I, E37L, D38E, P39D, V40Q, G41Q, S42K, E43S, P44T, R45V, S46T, G47D, T48A, A50Q, I51K, A73S, A75D
V2D5r	D33G, A34V, V36I, E37D, D38E, P39D, V40Q, G41Q, S42K, E43S, P44T, R45V, G47D, A50Q, A75D
V2D6	F30L, D32S, A34V, V36L, E37W, D38E, P39T, V40S, G41Q, S42D, E43R, P44T, R45Y, S46T, T48N, A49S, A73S, V74N, A75R
V2D6r	V36L, E37W, D38E, P39T, V40S, G41Q, S42D, E43R, P44T, R45Y, V74N, A75R
V2D7	A28G, L29Q, A31G, D32P, D33Q, A34V, V36I, D38E, P39S, V40K, G41F, S42P, E43P, P44A, R45D, S46P, G47D, T48L, A49S, A73V, V74Y, A75N, N76Y, E77T, A78T
V2D7r	A31G, D32P, A34V, V36I, D38E, P39S, V40K, G41F, S42P, E43P, P44A, R45D, S46P, G47D, V74Y, A75N
V2D8	A28G, L29Q, A31G, D32P, D33Q, A34V, T35V, V36I, D38E, P39S, V40K, G41Q, S42P, E43P, P44T, R45D, S46P, G47D, T48L, A49S, A73V, V74Y, A75N, E77T, A78T
V2D8r	A31G, D32P, V36I, D38E, P39S, V40K, G41Q, S42P, E43P, P44T, R45D, S46P, G47D, V74Y, A75N
V2D9	F30L, D32S, A34V, V36L, E37Y, D38E, P39T, V40S, G41Q, S42D, E43R, P44T, R45Y, S46T, T48N, A49S, A73S, V74N, A75R, E77T
V2D9r	E37Y, D38E, P39T, V40S, G41Q, S42D, E43R, P44T, R45Y, S46T, V74N, A75R
V2D10	A28G, L29Q, A31G, D32P, D33Q, A34V, V36L, E37V, D38E, P39S, V40K, A(G)41S, S42P, E43P, P44A, R45D, S46P, A(G)47D, T48L, A49S, A73V, V74S, A75Q, E77T, A78T
V2D11	A28G, L29Q, A31G, D32P, D33Q, A34V, V36L, E37W, D38E, P39S, V40K, G41Y, S42P, E43P, P44A, R45D, S46P, G47D, T48L, A49S, A73V, V74S, A75G, A78T
V2D11r(1)	A31G, D32P, A34V, V36L, E37W, D38E, P39S, V40K, G41Y, S42P, E43P, P44A, R45D, S46P, G47D, T448L, A49S, V74S, A75G
V2D11r(2)	A31G, D32P, A34V, E37W, D38E, P39S, V40K, G41Y, S42P, E43P, P44A, R45D, S46P, G47D, T448L, A49S, V74S, A75G

Table 2.11. Experimental characterization of designs from PIP version 1.
Mutations: Number of mutations from wild-type KSI, excluding deletions. Expression: Whether the design expressed in inclusion bodies (“Insoluble”) or not at all (“None”). Purified: Whether or not the design was successfully purified. Solubility: Whether the purified design was soluble after re-folding from inclusion bodies. Activity: Whether the design had any observable enzymatic activity (“+”) or not (“-”); N/A: not applicable as protein could not be purified or was not soluble after purification.

Design	Mutations	Expression	Purified	Solubility	Activity
V1D1r	14	Insoluble	Yes	Soluble	-
V1D2r	18	Insoluble	Yes	Insoluble	N/A
V1D3r	11	Insoluble	Yes	Soluble	-
V1D4	28	Insoluble	Yes	Insoluble	N/A
V1D5r	15	Insoluble	Yes	Insoluble	N/A
V1D6	31	None	No	N/A	N/A
V1D7	32	Insoluble	No	N/A	N/A
V1D8r	19	Insoluble	Yes	Soluble	+
V1D9r	15	Insoluble	Yes	Soluble	+
V1D10r	14	Insoluble	Yes	Soluble	+
V1D11	29	Insoluble	Yes	Insoluble	N/A
V1D12r	16	None	No	N/A	N/A
V1D13	29	Insoluble	Yes	Insoluble	N/A
V1D14	29	None	No	N/A	N/A

Table 2.12. Comparison of median RMSD of lowest energy models on perturbed structures. References for the different methods are indicated. Bold numbers denote best performance for given dataset (excluding FKIC with homologous fragments).

Method	8 residue side chain perturbed (Å)	12 residue side chain perturbed (Å)	template based models (Å)	12 residue backbone perturbed (Å)	12 residue backbone perturbed no unavoidable clashes (Å)	12 residue backbone perturbed unavoidable clashes (Å)
HLP[20] *	2.2	2.25	-	-	-	-
HLP-SS[4] *	0.85	1.15	-	-	-	-
NGK[2] *	**0.4**	0.75	3.9	-	-	-
Galaxy-PS1[26] *	1.45	3.05	3.5	-	-	-
Galaxy-PS2[24] *	1.05	1.55	**3.3**	**1.65**	1.4	**1.8**
FKIC	0.45	**0.64**	3.9	1.68	**1.28**	2.59
FKIC with homologous fragments	0.42	0.54	-	-	-	-

* Values reported by ref [27].

Table 2.13. Summary of energy function comparisons

Dataset	Sampling method	Rosetta energy function	Median RMSD of lowest energy model (Å)	Median RMSD of lowest RMSD model (Å)	Median RMSD all models (Å)	Median sub-Å fraction	Median time (s)
Standard	NGK	talaris2013	0.74	0.37	2.71	11.40%	2281
Standard	NGK	ref2015	0.64	0.37	2.70	13.00%	3642
Standard	FKIC	talaris2013	0.70	0.36	1.19	44.89%	1646
Standard	FKIC	talaris2014	0.64	0.35	1.27	46.20%	1813
Standard	FKIC	ref2015	0.62	0.32	1.16	47.80%	3456
Mixed	NGK	talaris2013	1.94	0.45	4.66	1.90%	3788
Mixed	NGK	ref2015	1.07	0.45	4.65	1.15%	7341
Mixed	FKIC	talaris2013	0.61	0.34	1.74	34.60%	3654
Mixed	FKIC	ref2015	0.53	0.34	1.46	52.30%	7196

Table 2.14. X-ray crystallography information

Structure	*V1D8r (6UAD)*	*V2D9r (6UAE)*
Wavelength	1.116Å	1.116Å
Resolution Range	46.03-1.75 (1.80-1.75)	105.00-1.93 (1.96-1.93)
Unit Cell	a=b=53.15Å , c=178.03Å **α**=**β**=90° **γ**=120°	a=73.01Å , b=210.00Å , c=39.64Å **α**=**β**=**γ**=90°
Space Group	$P6_522$	$P2_12_12$
Unique Reflections	15833 (1048)	46239 (2182)
Multiplicity	9.7 (3.9)	19.1 (17.7)
Completeness	99.2% (92.0%)	98.1% (94.2%)
<I/σI>	34.2 (4.0)	9.3 (1.0)
$CC_{1/2}$[68]	1.000 (0.939)	0.995 (0.652)
R_{pim}[61]	0.013 (0.159)	0.059 (0.818)
R_{work} [69]	0.1742 (0.2083)	0.1752 (0.3041)
R_{free}[69]	0.2095 (0.2852)	0.2122 (0.3916)
Total Refined Atoms	1244	4662
Protein Residues	121	495
Solvent Molecules	163	372
Refined Ligand Atoms	66	204
Average B-factor	24.8Å^2	37.7Å^2
$RMSD_{bonds}$	0.014Å	0.006Å
$RMSD_{angles}$	1.23°	0.91°

Table 2.15. Cα distances between positions of designed residues in the crystal structures of V2D9r and WT.

Residue #	WT amino acid	V2D9r amino acid	Cα Distance (Å)
34	ALA	ALA	0.17
35	THR	THR	0.26
36	VAL	VAL	0.32
37	GLU	TYR	0.48
38	ASN	GLU	1.31
39	PRO	THR	2.47
40	VAL	SER	5.96
41	GLY	GLN	6.55
42	SER	ASP	0.79
43	GLU	ARG	2.47
44	PRO	THR	0.91
45	ARG	TYR	0.50
46	SER	THR	0.22
74	VAL	ASN	0.63
75	ALA	ARG	1.02

2.9 Appendix IV: Supplementary figures

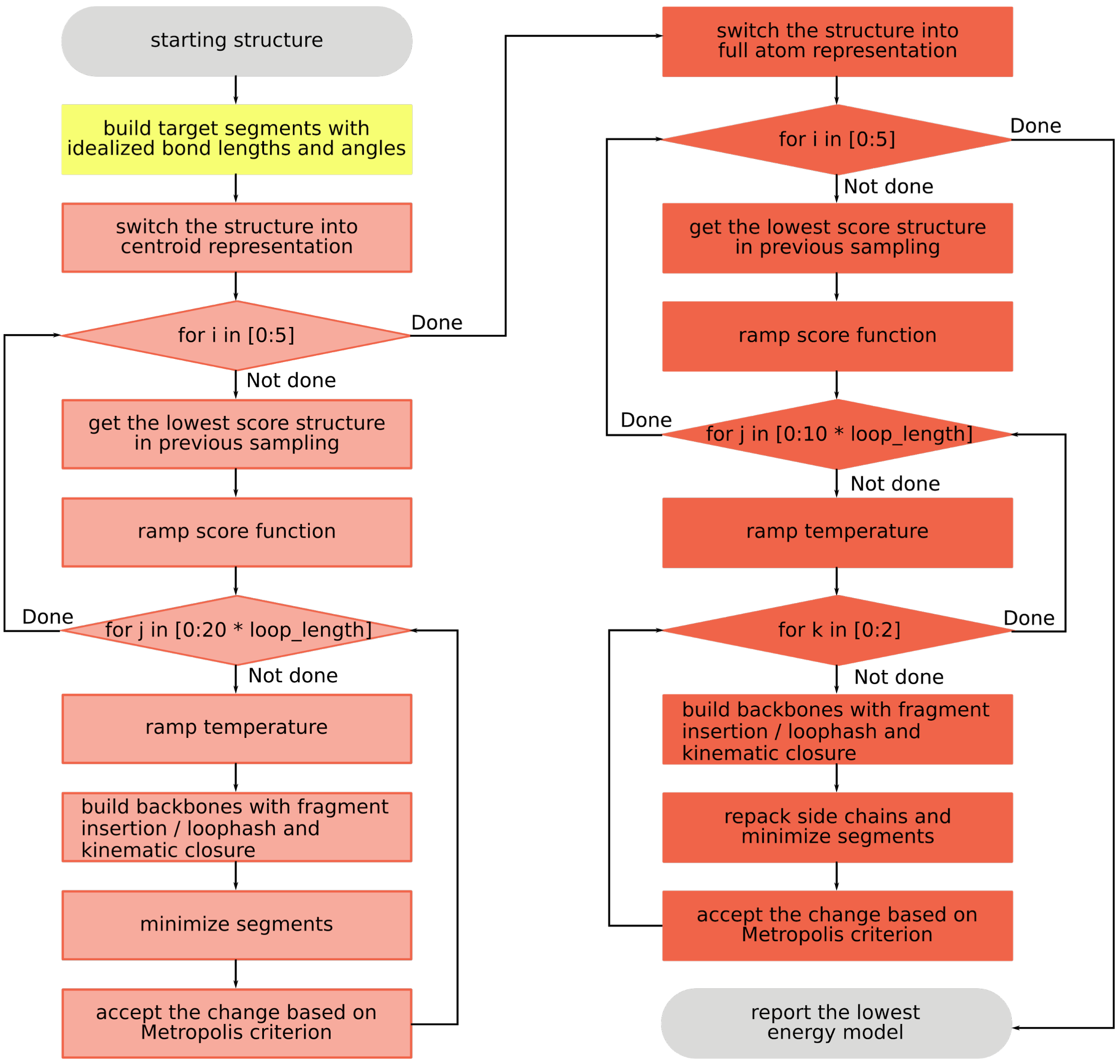

starting structure
build target segments with idealized bond lengths and angles
switch the structure into centroid representation
for i in [0:5]
Done
Not done
get the lowest score structure in previous sampling
ramp score function
for j in [0:20 * loop_length]
Done
Not done
ramp temperature
build backbones with fragment insertion / loophash and kinematic closure
minimize segments
accept the change based on Metropolis criterion
switch the structure into full atom representation
for i in [0:5]
Done
Not done
get the lowest score structure in previous sampling
ramp score function
for j in [0:10 * loop_length]
Done
Not done
ramp temperature
for k in [0:2]
Done
Not done
build backbones with fragment insertion / loophash and kinematic closure
repack side chains and minimize segments
accept the change based on Metropolis criterion
report the lowest energy model

Figure 2.5. Detailed FKIC protocol.
The FKIC / LHKIC modeling protocol has a build stage (yellow), a centroid sampling stage (light red) and a full atom sampling stage (red). Both the centroid stage and the full atom stage perform simulated annealing which ramp the *rama* and *fa_rep* terms of the Rosetta energy function [21], [22] in outer cycles and ramp the temperature in inner cycles.

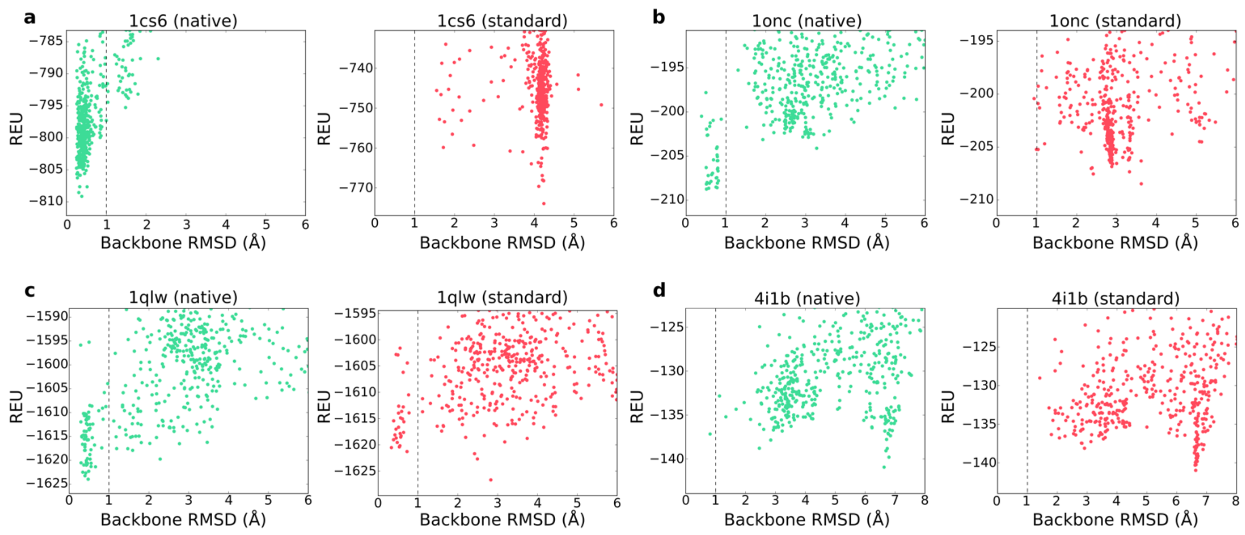


Figure 2.6. Examples of failures of FKIC.
Results of standard FKIC are shown in red (right in each panel) and results of FKIC with native input information and native bond lengths and angles are shown in green (left in each panel). Each point represents a Rosetta generated model. REU, Rosetta energy units. (**a**) Sub-Å models are generated only with native inputs. (**b**) Standard FKIC generates a few sub-Å models but they are not identified by energy. Using native inputs generates a larger number of near-native solutions that can be correctly identified by energy. (**c**) The simulation with native inputs correctly identifies native-like models, but a model generated by standard FKIC has lower energy. (**d**) Neither standard nor native-input simulations correctly identify sub-Å models.

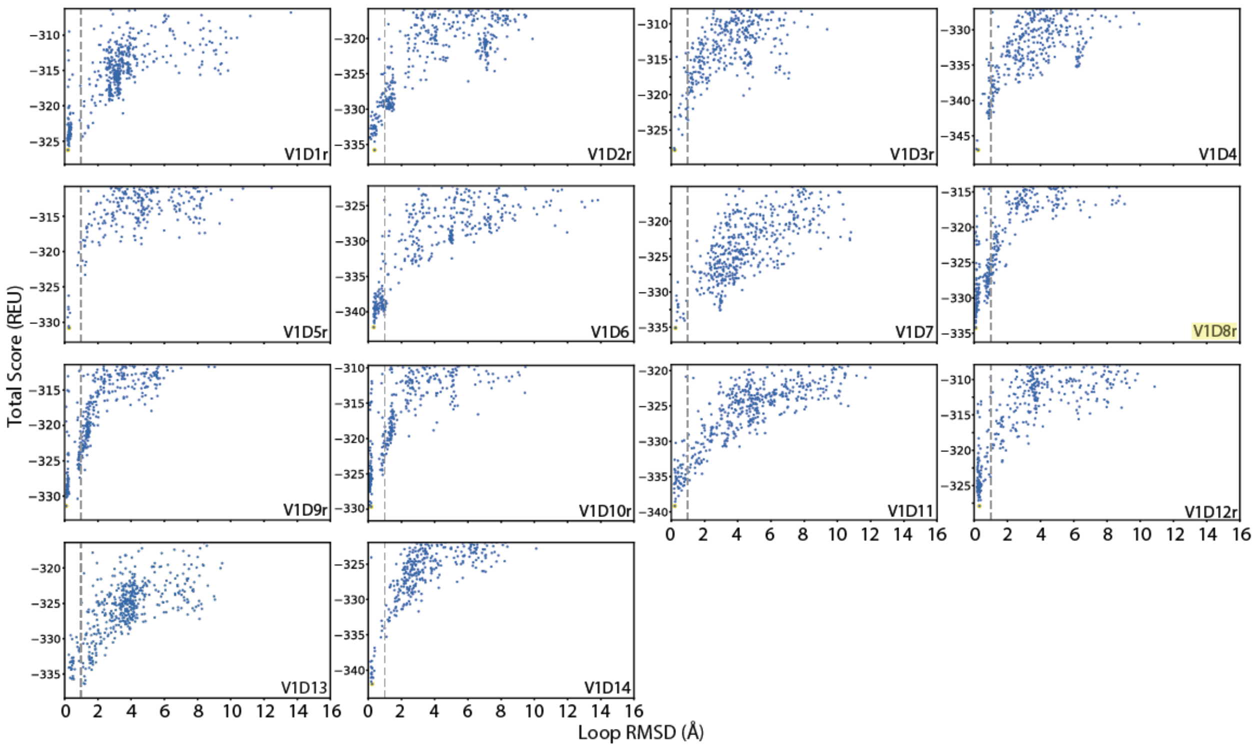


Figure 2.7. Computational structure prediction of designs from PIP V1.
Rosetta total score (in REU) versus loop backbone RMSD (in Å) for designs selected for experimental testing from PIP version 1. Vertical dashed lines indicate 1Å loop RMSD. The experimentally characterized V1D8r design is highlighted in yellow. Structure prediction was performed on both the initial designs and the reversion mutants.

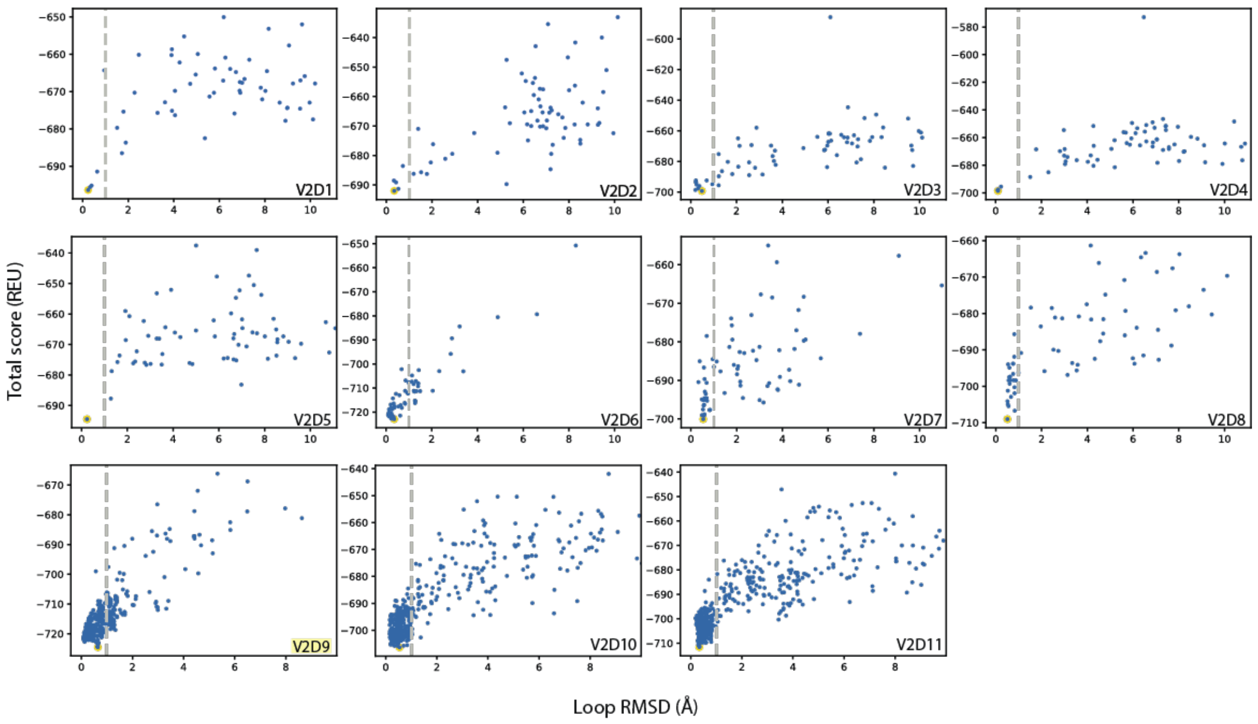


Figure 2.8. Computational structure predictions of designs from PIP V2.
Rosetta total score (in REU) versus loop backbone RMSD (in Å) for designs selected for experimental testing from PIP version 2. Plots are shown for the designs excluding the reversion mutants. Vertical dashed lines indicate 1Å loop RMSD. V2D9, the design corresponding to the experimentally characterized V2D9r reversion mutant, is highlighted in yellow.

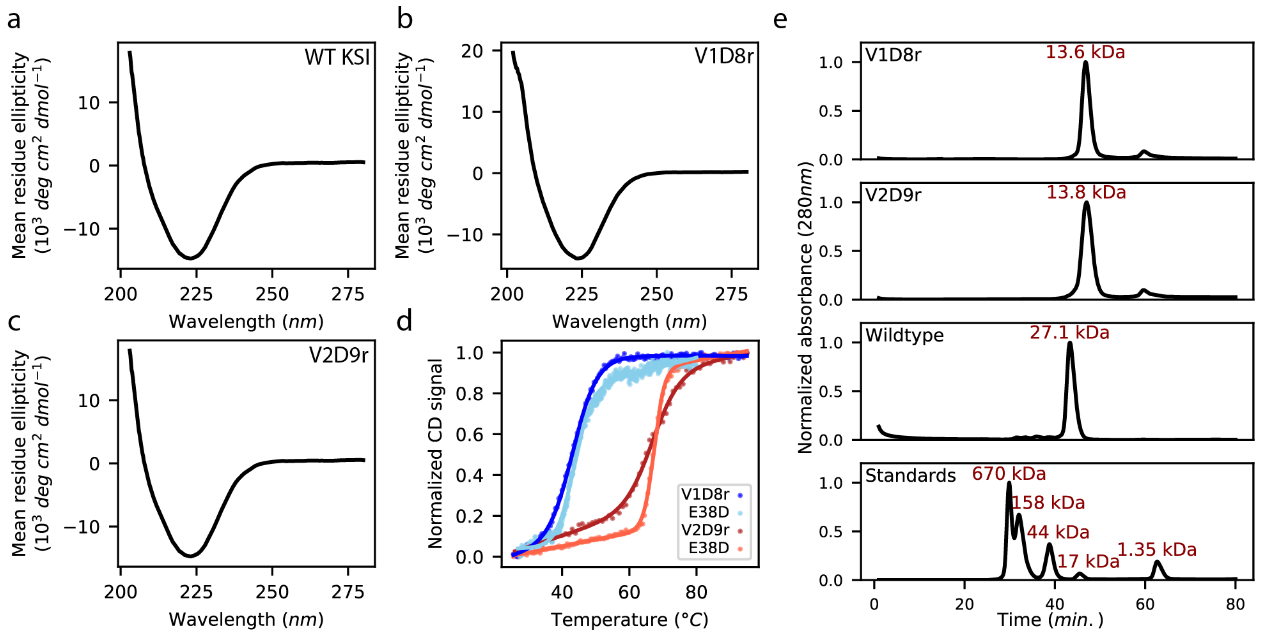


Figure 2.9. Biophysical and biochemical characterization of designs.
Circular dichroism (CD) spectra for wild-type (**a**), V1D8r (**b**), or V2D9r (**c**). (**d**) Normalized temperature melting curves, measured via CD at 222 nm, for V1D8r (dark blue) and V2D9r (dark red) and their corresponding E38D reversion mutants (light blue and orange, respectively). The temperature melts are not reversible. Apparent melting temperatures are as follows: V1D8r: 44 °C, V1D8r E38D: 43 °C, V2D9r: 67 °C, V2D9r E38D: 67 °C. (**e**) Normalized absorbance (280 nm) from analytical size exclusion chromatography of V1D8r, V2D9r, wild-type KSI or a standards mixture with molecular weights as labeled.

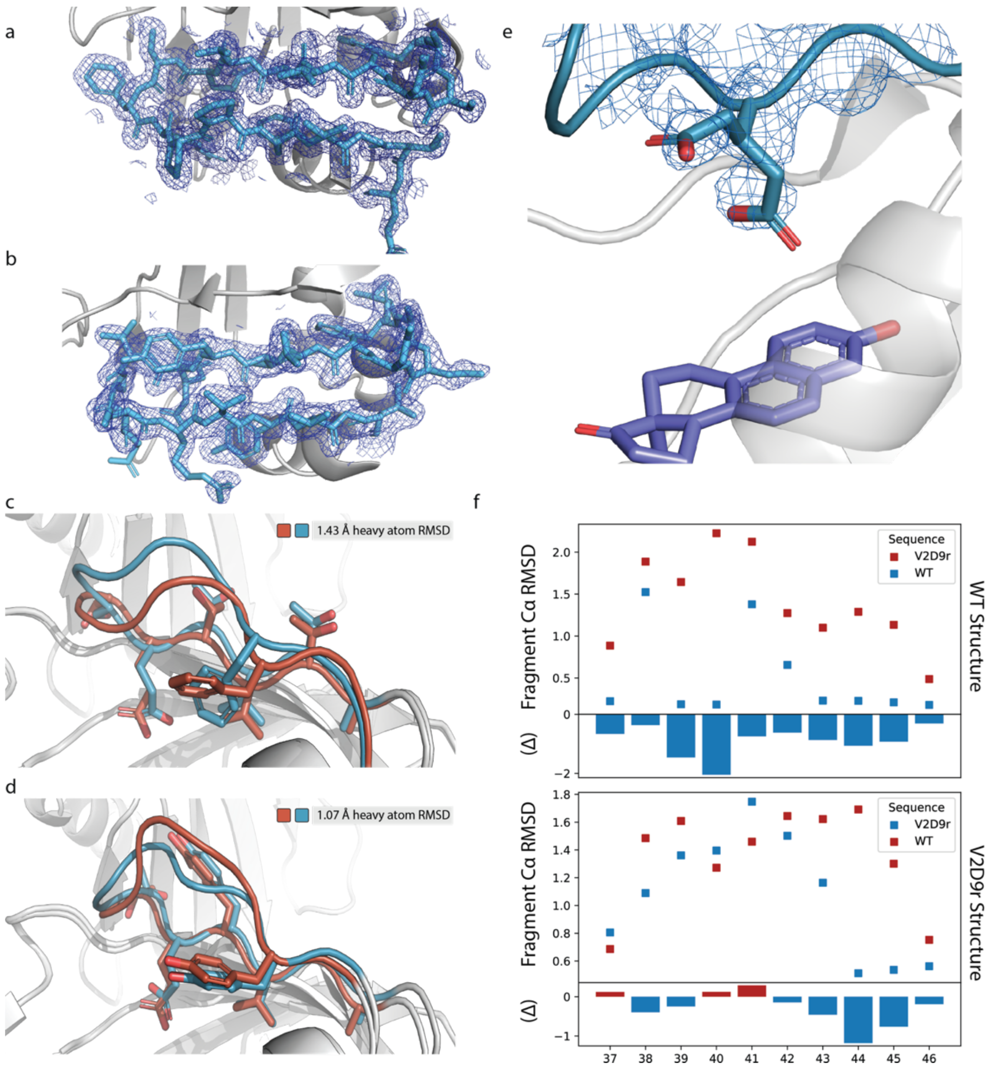


Figure 2.10. Structural analysis of designs.
Electron density of reshaped backbone region for V1D8r **(a)** and V2D9r **(b)** at 1.0 sigma in mesh representation. Comparison of buried (<40 Å^2 solvent-accessible surface area) sidechain positioning between lowest-energy design model (orange) and crystal structure (blue) for **(c)** V1D8r or **(d)** V2D9r. Heavy-atom RMSDs for the displayed residues are shown in each panel. **(e)** Electron density of possible alternate conformations of E38 observed in design 2. Density is contoured at 0.5 sigma for residues 37-39 in chain B. Residue E38 (teal) and equilenin (purple) are shown as sticks. **(f)** Cα RMSD of the closest 9-residue fragment whose midpoint (5th residue) corresponds to the x-axis position. Top: Fragments picked using the WT sequence (blue) or the incorrect V2D9r sequence (red) aligned to the corresponding position in the WT crystal structure. Bottom: Fragments picked using the V2D9r sequence (blue) or the incorrect WT sequence (red) aligned to the corresponding position in the V2D9r crystal structure. The difference in fragment Cα RMSD between the correct and incorrect sequences are shown at the bottom of each graph. Blue bars (negative change in fragment Cα RMSD) indicate the correct structure is favored.

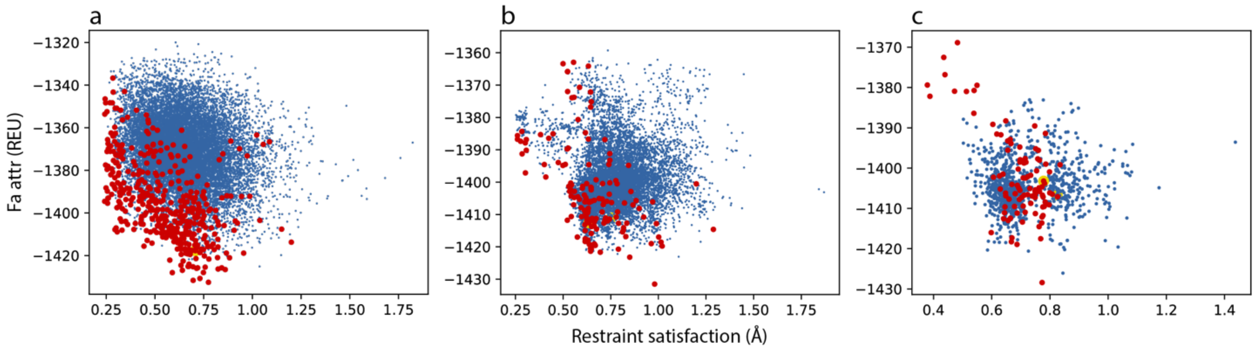


Figure 2.11. Selection of designs for via Pareto fronts in PIP version 2.
Designs picked for computational structure prediction. Plots of design Lennard-Jones attractive (fa_attr) Rosetta score, in REU, vs. restraint satisfaction (longest distance of any restrained atom to its ideal position, in Å (**Appendix I: Supplementary Methods**)) for the first (**a**), second (**b**), and third (**c**) iterations of design for PIP version 2. Designs chosen via Pareto fronts for structure prediction are shown in red, other designs in blue.

2.10 References

[1] B. Kuhlman and P. Bradley, “Advances in protein structure prediction and design,” *Nat. Rev. Mol. Cell Biol.*, vol. 20, no. 11, Art. no. 11, Nov. 2019, doi: 10.1038/s41580-019-0163-x.

[2] X. Pan and T. Kortemme, “Recent advances in de novo protein design: Principles, methods, and applications,” *J. Biol. Chem.*, vol. 296, p. 100558, Mar. 2021, doi: 10.1016/j.jbc.2021.100558.

[3] A. A. Glasgow *et al.*, “Computational design of a modular protein sense-response system,” *Science*, vol. 366, no. 6468, pp. 1024–1028, Nov. 2019, doi: 10.1126/science.aax8780.

[4] J. Marcandalli *et al.*, “Induction of Potent Neutralizing Antibody Responses by a Designed Protein Nanoparticle Vaccine for Respiratory Syncytial Virus,” *Cell*, vol. 176, no. 6, pp. 1420-1431.e17, Mar. 2019, doi: 10.1016/j.cell.2019.01.046.

[5] D.-A. Silva *et al.*, “De novo design of potent and selective mimics of IL-2 and IL-15,” *Nature*, vol. 565, no. 7738, Art. no. 7738, Jan. 2019, doi: 10.1038/s41586-018-0830-7.

[6] P.-S. Huang, S. E. Boyken, and D. Baker, “The coming of age of de novo protein design,” *Nature*, vol. 537, no. 7620, Art. no. 7620, Sep. 2016, doi: 10.1038/nature19946.

[7] D. Baker, “What has de novo protein design taught us about protein folding and biophysics?,” *Protein Sci. Publ. Protein Soc.*, vol. 28, no. 4, pp. 678–683, Apr. 2019, doi: 10.1002/pro.3588.

[8] N. Koga *et al.*, “Principles for designing ideal protein structures,” *Nature*, vol. 491, no. 7423, Art. no. 7423, Nov. 2012, doi: 10.1038/nature11600.

[9] N. F. Polizzi *et al.*, “De novo design of a hyperstable non-natural protein–ligand complex with sub-Å accuracy,” *Nat. Chem.*, vol. 9, no. 12, Art. no. 12, Dec. 2017, doi: 10.1038/nchem.2846.

[10] S. J. Fleishman and D. Baker, “Role of the Biomolecular Energy Gap in Protein Design, Structure, and Evolution,” *Cell*, vol. 149, no. 2, pp. 262–273, Apr. 2012, doi: 10.1016/j.cell.2012.03.016.

[11] K. Kundert and T. Kortemme, “Computational design of structured loops for new protein functions,” *Biol. Chem.*, vol. 400, no. 3, pp. 275–288, Feb. 2019, doi: 10.1515/hsz-2018-0348.

[12] B. Borgo and J. J. Havranek, “Motif-directed redesign of enzyme specificity,” *Protein Sci.*, vol. 23, no. 3, pp. 312–320, 2014, doi: 10.1002/pro.2417.

[13] P. M. Murphy, J. M. Bolduc, J. L. Gallaher, B. L. Stoddard, and D. Baker, “Alteration of enzyme specificity by computational loop remodeling and design,” *Proc. Natl. Acad. Sci.*, vol. 106, no. 23, pp. 9215–9220, Jun. 2009, doi: 10.1073/pnas.0811070106.

[14] M. Baek *et al.*, “Accurate prediction of protein structures and interactions using a 3-track neural network,” *Science*, vol. 373, no. 6557, pp. 871–876, Aug. 2021, doi: 10.1126/science.abj8754.

[15] J. Jumper *et al.*, “Highly accurate protein structure prediction with AlphaFold,” *Nature*, vol. 596, no. 7873, Art. no. 7873, Aug. 2021, doi: 10.1038/s41586-021-03819-2.

[16] K. T. Simons, C. Kooperberg, E. Huang, and D. Baker, “Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions,” *J. Mol. Biol.*, vol. 268, no. 1, pp. 209–225, Apr. 1997, doi: 10.1006/jmbi.1997.0959.

[17] D. J. Mandell, E. A. Coutsias, and T. Kortemme, “Sub-angstrom accuracy in protein loop reconstruction by robotics-inspired conformational sampling,” *Nat. Methods*, vol. 6, no. 8, pp. 551–552, Aug. 2009, doi: 10.1038/nmeth0809-551.

[18] E. A. Coutsias, C. Seok, M. P. Jacobson, and K. A. Dill, “A kinematic view of loop closure,” *J. Comput. Chem.*, vol. 25, no. 4, pp. 510–528, Mar. 2004, doi: 10.1002/jcc.10416.

[19] D. Gront, D. W. Kulp, R. M. Vernon, C. E. M. Strauss, and D. Baker, “Generalized Fragment Picking in Rosetta: Design, Protocols and Applications,” *PLOS ONE*, vol. 6, no. 8, p. e23294, Aug. 2011, doi: 10.1371/journal.pone.0023294.

[20] M. Tyka, K. Jung, and D. Baker, “Efficient sampling of protein conformational space using fast loop building and batch minimization on highly parallel computers,” *J Comput Chem*, vol. 33, pp. 2483–2491, 2012.

[21] H. Park *et al.*, “Simultaneous Optimization of Biomolecular Energy Functions on Features from Small Molecules and Macromolecules,” *J. Chem. Theory Comput.*, vol. 12, no. 12, pp. 6201–6212, Dec. 2016, doi: 10.1021/acs.jctc.6b00819.

[22] R. F. Alford *et al.*, “The Rosetta All-Atom Energy Function for Macromolecular Modeling and Design,” *J. Chem. Theory Comput.*, vol. 13, no. 6, pp. 3031–3048, Jun. 2017, doi: 10.1021/acs.jctc.7b00125.

[23] B. D. Sellers, K. Zhu, S. Zhao, R. A. Friesner, and M. P. Jacobson, “Toward better refinement of comparative models: Predicting loops in inexact environments,” *Proteins*, vol. 72, no. 3, pp. 959–971, Aug. 2008, doi: 10.1002/prot.21990.

[24] A. A. Canutescu and R. L. Dunbrack, “Cyclic coordinate descent: A robotics algorithm for protein loop closure,” *Protein Sci. Publ. Protein Soc.*, vol. 12, no. 5, pp. 963–972, May 2003.

[25] C. Wang, P. Bradley, and D. Baker, “Protein-protein docking with backbone flexibility,” *J. Mol. Biol.*, vol. 373, no. 2, pp. 503–519, Oct. 2007, doi: 10.1016/j.jmb.2007.07.050.

[26] A. Stein and T. Kortemme, “Improvements to Robotics-Inspired Conformational Sampling in Rosetta,” *PLOS ONE*, vol. 8, no. 5, p. e63090, May 2013, doi: 10.1371/journal.pone.0063090.

[27] H. Park, G. R. Lee, L. Heo, and C. Seok, “Protein Loop Modeling Using a New Hybrid Energy Function and Its Application to Modeling in Inaccurate Structural Environments,” *PLOS ONE*, vol. 9, no. 11, p. e113811, Nov. 2014, doi: 10.1371/journal.pone.0113811.

[28] J. R. López-Blanco, A. J. Canosa-Valls, Y. Li, and P. Chacón, “RCD+: Fast loop modeling server,” *Nucleic Acids Res.*, vol. 44, no. Web Server issue, pp. W395–W400, Jul. 2016, doi: 10.1093/nar/gkw395.

[29] C. Marks *et al.*, “Sphinx: merging knowledge-based and ab initio approaches to improve protein loop prediction,” *Bioinformatics*, vol. 33, no. 9, pp. 1346–1353, May 2017, doi: 10.1093/bioinformatics/btw823.

[30] S. Liang, C. Zhang, and Y. Zhou, “LEAP: highly accurate prediction of protein loop conformations by integrating coarse-grained sampling and optimized energy scores with all-atom refinement of backbone and side chains,” *J. Comput. Chem.*, vol. 35, no. 4, pp. 335–341, Feb. 2014, doi: 10.1002/jcc.23509.

[31] Y. Choi and C. M. Deane, “FREAD revisited: Accurate loop structure prediction using a database search algorithm,” *Proteins*, vol. 78, no. 6, pp. 1431–1440, May 2010, doi: 10.1002/prot.22658.

[32] C. Marks, J. Shi, and C. M. Deane, “Predicting loop conformational ensembles,” *Bioinformatics*, vol. 34, no. 6, pp. 949–956, Mar. 2018, doi: 10.1093/bioinformatics/btx718.

[33] J. P. Schwans *et al.*, “Experimental and Computational Mutagenesis To Investigate the Positioning of a General Base within an Enzyme Active Site,” *Biochemistry*, vol. 53, no. 15, pp. 2541–2555, Apr. 2014, doi: 10.1021/bi401671t.

[34] H. J. Cha *et al.*, “Role of conserved Met112 residue in the catalytic activity and stability of ketosteroid isomerase,” *Biochim. Biophys. Acta BBA - Proteins Proteomics*, vol. 1864, no. 10, pp. 1322–1327, Oct. 2016, doi: 10.1016/j.bbapap.2016.06.016.

[35] D. A. Kraut, P. A. Sigala, T. D. Fenn, and D. Herschlag, “Dissecting the paradoxical effects of hydrogen bond mutations in the ketosteroid isomerase oxyanion hole,” *Proc. Natl. Acad. Sci.*, vol. 107, no. 5, pp. 1960–1965, Feb. 2010, doi: 10.1073/pnas.0911168107.

[36] D. C. Hawkinson, T. C. Eames, and R. M. Pollack, “Energetics of 3-oxo-delta 5-steroid isomerase: source of the catalytic power of the enzyme,” *Biochemistry*, vol. 30, no. 45, pp. 10849–10858, Nov. 1991, doi: 10.1021/bi00109a007.

[37] Y. Wu, S. D. Fried, and S. G. Boxer, “A Preorganized Electric Field Leads to Minimal Geometrical Reorientation in the Catalytic Reaction of Ketosteroid Isomerase,” *J. Am. Chem. Soc.*, vol. 142, no. 22, pp. 9993–9998, Jun. 2020, doi: 10.1021/jacs.0c00383.

[38] C. Yang *et al.*, “Bottom-up de novo design of functional proteins with complex structural features,” *Nat. Chem. Biol.*, vol. 17, no. 4, pp. 492–500, Apr. 2021, doi: 10.1038/s41589-020-00699-x.

[39] X. Pan *et al.*, “Expanding the space of protein geometries by computational design of de novo fold families,” *Science*, vol. 369, no. 6507, pp. 1132–1136, Aug. 2020, doi: 10.1126/science.abc0881.

[40] J. A. Davey, A. M. Damry, N. K. Goto, and R. A. Chica, “Rational design of proteins that exchange on functional timescales,” *Nat. Chem. Biol.*, vol. 13, no. 12, Art. no. 12, Dec. 2017, doi: 10.1038/nchembio.2503.

[41] M. J. O’Meara *et al.*, “Combined Covalent-Electrostatic Model of Hydrogen Bonding Improves Structure Prediction with Rosetta,” *J. Chem. Theory Comput.*, vol. 11, no. 2, pp. 609–622, Feb. 2015, doi: 10.1021/ct500864r.

[42] S. Ó Conchúir *et al.*, “A Web Resource for Standardized Benchmark Datasets, Metrics, and Rosetta Protocols for Macromolecular Modeling and Design,” *PloS One*, vol. 10, no. 9, p. e0130433, 2015, doi: 10.1371/journal.pone.0130433.

[43] B. North, A. Lehmann, and R. L. Dunbrack, “A new clustering of antibody CDR loop conformations,” *J. Mol. Biol.*, vol. 406, no. 2, pp. 228–256, Feb. 2011, doi: 10.1016/j.jmb.2010.10.030.

[44] G. J. Rocklin *et al.*, “Global analysis of protein folding using massively parallel design, synthesis, and testing,” *Science*, vol. 357, pp. 168–175, 2017.

[45] M. E. Lee, W. C. DeLoache, B. Cervantes, and J. E. Dueber, “A Highly Characterized Yeast Toolkit for Modular, Multipart Assembly,” *ACS Synth. Biol.*, vol. 4, no. 9, pp. 975–986, Sep. 2015, doi: 10.1021/sb500366v.

[46] S. W. Kim and K. Y. Choi, “Identification of active site residues by site-directed mutagenesis of delta 5-3-ketosteroid isomerase from Pseudomonas putida biotype B,” *J. Bacteriol.*, vol. 177, no. 9, pp. 2602–2605, May 1995, doi: 10.1128/jb.177.9.2602-2605.1995.

[47] A. J. McCoy, R. W. Grosse-Kunstleve, P. D. Adams, M. D. Winn, L. C. Storoni, and R. J. Read, “Phaser crystallographic software,” *J. Appl. Crystallogr.*, vol. 40, no. Pt 4, pp. 658–674, Aug. 2007, doi: 10.1107/S0021889807021206.

[48] P. D. Adams *et al.*, “PHENIX: a comprehensive Python-based system for macromolecular structure solution,” *Acta Crystallogr. D Biol. Crystallogr.*, vol. 66, no. Pt 2, pp. 213–221, Feb. 2010, doi: 10.1107/S0907444909052925.

[49] K. A. Kantardjieff and B. Rupp, “Matthews coefficient probabilities: Improved estimates for unit cell contents of proteins, DNA, and protein–nucleic acid complex crystals,” *Protein Sci. Publ. Protein Soc.*, vol. 12, no. 9, pp. 1865–1871, Sep. 2003.

[50] C. X. Weichenberger and B. Rupp, “Ten years of probabilistic estimates of biocrystal solvent content: new insights via nonparametric kernel density estimate,” *Acta Crystallogr. D Biol. Crystallogr.*, vol. 70, no. Pt 6, pp. 1579–1588, Jun. 2014, doi: 10.1107/S1399004714005550.

[51] P. V. Afonine *et al.*, “Towards automated crystallographic structure refinement with phenix.refine,” *Acta Crystallogr. D Biol. Crystallogr.*, vol. 68, no. 4, Art. no. 4, Apr. 2012, doi: 10.1107/S0907444912001308.

[52] N. W. Moriarty, R. W. Grosse-Kunstleve, and P. D. Adams, “electronic Ligand Builder and Optimization Workbench (eLBOW): a tool for ligand coordinate and restraint generation,” *Acta*

Crystallogr. D Biol. Crystallogr., vol. 65, no. Pt 10, pp. 1074–1080, Oct. 2009, doi: 10.1107/S0907444909029436.

[53] S. M. Le Grand and K. M. Merz Jr., “Rapid approximation to molecular surface area via the use of Boolean logic and look-up tables,” *J. Comput. Chem.*, vol. 14, no. 3, pp. 349–352, 1993, doi: 10.1002/jcc.540140309.

[54] W. G. Touw *et al.*, “A series of PDB-related databanks for everyday needs,” *Nucleic Acids Res.*, vol. 43, no. Database issue, pp. D364–D368, Jan. 2015, doi: 10.1093/nar/gku1028.

[55] V. B. Chen *et al.*, “MolProbity: all-atom structure validation for macromolecular crystallography,” *Acta Crystallogr. D Biol. Crystallogr.*, vol. 66, no. Pt 1, pp. 12–21, Jan. 2010, doi: 10.1107/S0907444909042073.

[56] S. F. Altschul *et al.*, “Gapped BLAST and PSI-BLAST: a new generation of protein database search programs,” *Nucleic Acids Res.*, vol. 25, no. 17, pp. 3389–3402, Sep. 1997, doi: 10.1093/nar/25.17.3389.

[57] J. J. Ward, L. J. McGuffin, B. F. Buxton, and D. T. Jones, “Secondary structure prediction with support vector machines,” *Bioinforma. Oxf. Engl.*, vol. 19, no. 13, pp. 1650–1655, Sep. 2003, doi: 10.1093/bioinformatics/btg223.

[58] Y. Yang, E. Faraggi, H. Zhao, and Y. Zhou, “Improving protein fold recognition and template-based modeling by employing probabilistic-based matching between predicted one-dimensional structural properties of query and corresponding native properties of templates,” *Bioinforma. Oxf. Engl.*, vol. 27, no. 15, pp. 2076–2082, Aug. 2011, doi: 10.1093/bioinformatics/btr350.

[59] H. S. Cho *et al.*, “Crystal structure of delta(5)-3-ketosteroid isomerase from Pseudomonas testosteroni in complex with equilenin settles the correct hydrogen bonding scheme for transition state stabilization,” *J. Biol. Chem.*, vol. 274, no. 46, pp. 32863–32868, Nov. 1999, doi: 10.1074/jbc.274.46.32863.

[60] J. P. Schwans, F. Sunden, J. K. Lassila, A. Gonzalez, Y. Tsai, and D. Herschlag, “Use of anion–aromatic interactions to position the general base in the ketosteroid isomerase active site,” *Proc. Natl. Acad. Sci.*, vol. 110, no. 28, pp. 11308–11313, Jul. 2013, doi: 10.1073/pnas.1206710110.

[61] G. Winter, “xia2: an expert system for macromolecular crystallography data reduction,” *J. Appl. Crystallogr.*, vol. 43, no. 1, Art. no. 1, Feb. 2010, doi: 10.1107/S0021889809045701.

[62] W. Kabsch, “XDS,” *Acta Crystallogr. D Biol. Crystallogr.*, vol. 66, no. Pt 2, pp. 125–132, Feb. 2010, doi: 10.1107/S0907444909047337.

[63] P. Evans, “Scaling and assessment of data quality,” *Acta Crystallogr. D Biol. Crystallogr.*, vol. 62, no. Pt 1, pp. 72–82, Jan. 2006, doi: 10.1107/S0907444905036693.

[64] C. S. Rapp and R. M. Pollack, “Crystal packing effects on protein loops,” *Proteins*, vol. 60, no. 1, pp. 103–109, Jul. 2005, doi: 10.1002/prot.20492.

[65] M. P. Jacobson *et al.*, “A hierarchical approach to all-atom protein loop prediction,” *Proteins*, vol. 55, no. 2, pp. 351–367, May 2004, doi: 10.1002/prot.10613.

[66] D.-H. Kim *et al.*, “Equilibrium and Kinetic Analysis of Folding of Ketosteroid Isomerase from Comamonas testosteroni,” *Biochemistry*, vol. 39, no. 42, pp. 13084–13092, Oct. 2000, doi: 10.1021/bi000872d.

[67] F. Yabukarski *et al.*, “Assessment of enzyme active site positioning and tests of catalytic mechanisms through X-ray–derived conformational ensembles,” *Proc. Natl. Acad. Sci.*, vol. 117, no. 52, pp. 33204–33215, Dec. 2020, doi: 10.1073/pnas.2011350117.

[68] “Linking Crystallographic Model and Data Quality.” https://www.science.org/doi/10.1126/science.1218231 (accessed May 25, 2022).

[69] A. T. Brünger, “Free R value: cross-validation in crystallography,” *Methods Enzymol.*, vol. 277, pp. 366–396, 1997, doi: 10.1016/s0076-6879(97)77021-6.

Chapter 3

Generalizable design of protein-protein interfaces using helical structural elements

3.1 Abstract

Computationally designed protein interfaces have many applications in medicine and synthetic biology, but typically require extensive experimental optimization to achieve affinities needed for robust *in vivo* usage. High-affinity protein interfaces require precise interactions with high shape complementarity, facilitated by backbone geometries that support favorable sidechain orientations. Here we present an approach, in line with these requirements, for the design of *de novo* proteins that bind a specific target protein: Helical ELements Interface eXplorer (HELIX) is a bottom-up method for designing protein binders that positions helical fragments that are geometrically compatible with the target interface, then thoroughly optimizes the sequences of these small helical structures, and finally searches for scaffolds that support as many of these structural elements as possible. When combined with systematic sampling of helical geometries using methods previously developed in our lab, HELIX produces *de novo* protein interfaces with a high level of shape complementarity at multiple sites along the target protein's surface, including polar regions. To computationally validate the method, we ran HELIX on several proteins that have been targeted by other *de novo* interface design methods and compare their

performance. HELIX produces designs with affinity-related metrics that are on par with or better than designs created using other interface design methods, while simultaneously creating a higher number of well-satisfied interface hydrogen bonds. With computational and experimental validation ongoing, we envision HELIX and similar approaches becoming important tools in the design of new biologics and protein circuits.

3.2 Introduction

The computational design of protein-protein interactions has enabled many applications, including the creation of binders to medically-relevant targets [1]–[3], small molecule-sensing heterodimers [4], and synthetic protein circuits [5], [6]. While many examples of successfully designed protein interfaces exist, it is common that initial binders need to be optimized through a combination of affinity maturation and deep mutational scanning to achieve the desired affinities. Thus, a generalizable computational method for reliably designing high-affinity *de novo* interfaces would be extremely valuable.

Difficulties arise due to the strict geometric requirements of high-affinity protein-protein interfaces: Highly favorable sidechain interactions must be accommodated by compatible backbone segments, and often several noncontiguous backbone segments are necessary to form a large high-affinity interface with a protein target (**Fig. 3.1a**). These cascading geometric requirements for protein-protein recognition are congruent with bottom-up design approaches, where first residue-level interactions are determined that are then matched or grafted into a backbone “scaffold” of a potential binding protein, as opposed to top-down design where an existing protein is first docked against the target followed by design of the interface residues for affinity with the target protein.

Recent methods have partially addressed the protein interface design problem: Yang and colleagues developed a method to graft known binding motifs into stable scaffolds to generate protein binders and used it to design binders for respiratory syndrome virus (RSV) F and G proteins [7]. The same method was later used to design a PD-1 agonist [8], demonstrating a degree of generalizability, but only in cases where strong binding motifs are known, and even then requiring experimental optimization to reach the desired affinities. More recently, Cao. and colleagues designed protein binders from target structure alone by docking a scaffold library to a rotamer interaction field (RIF) [9], then focusing design on well-scoring motifs while resampling backbones from the scaffold library [10]. Combining this method with combinatorial sequence optimization via site-saturation mutagenesis yielded binders with nanomolar affinity for several targets, though it remains an unsolved problem to reach these affinities without the need for experimental optimization.

Here, we present a new protein interface design method, Helical ELements Interface eXplorer (HELIX, **Fig. 3.1b**), which specifically aims to optimize the affinity of individual secondary structural elements before matching those to a scaffold that already supports the correct secondary structure geometry for binding. These secondary structural fragments are obtained using information from the geometry, but not sequence, of real structures, reducing reliance on known binding motifs while still taking advantage of the wealth of data provided by the Protein Databank (PDB). Our approach also splits the problems of sequence and structure optimization into two distinct steps, allowing each one to be more thoroughly optimized by reducing the extent of the combinatorial explosion caused by the vastness of sequence and structure space.

3.3 Results

3.3.1 HELIX overview

To design high-affinity protein-protein interfaces, we created a method, HELIX, that builds interfaces starting from short protein fragments (**Fig. 3.1b**). HELIX treats the design of interactions with the target protein and the design of the potential binding protein's backbone geometry as two separate challenges, and then looks for the overlap between the two solution spaces. The method begins by docking fragments of α-helices such that they enable favorable interactions with the target protein. Next, we design the sequences of these protein fragments, placing an emphasis on favorable intermolecular energies and shape complementarity. The fragmented nature of the budding interface allows us to thoroughly optimize the sequence of the protein fragments during this step. The docked protein fragments are then matched against a library of scaffolds such that as many fragments as possible are present in the proper orientations in the matched backbone. While it would be possible to dock protein fragments with any secondary structure in the first step, here we focus on α-helical units for several reasons: (1) The geometries of (regular) α-helices can be expressed as a vector, making it possible to match their relative orientations to a scaffold library with minimal computational cost; (2) methods exist [11] to thoroughly sample helical geometries within a topology, allowing one to match against a library of *de novo* scaffolds with a variety of helical orientations; and (3) approximately 40% of buried interface residues are part of an alpha helix (**Appendix Fig. 3.5, Methods**), making it the most common secondary structure for energetically important positions.

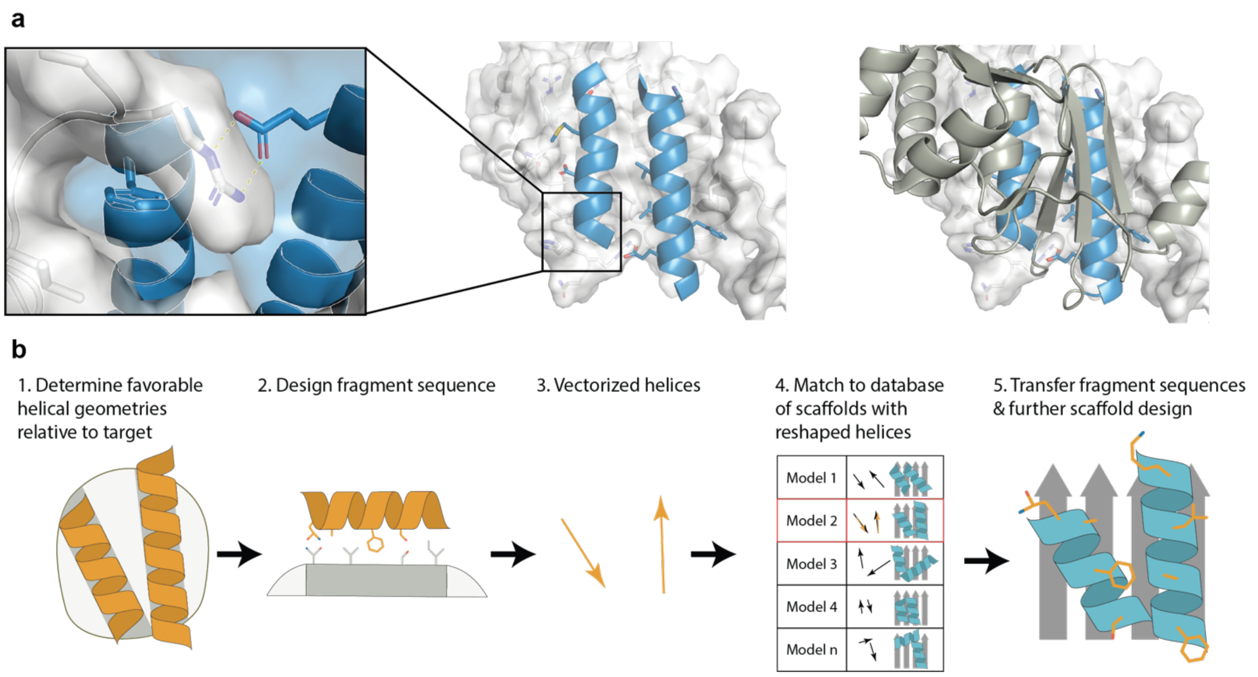

a
b
1. Determine favorable helical geometries relative to target
2. Design fragment sequence
3. Vectorized helices
4. Match to database of scaffolds with reshaped helices
5. Transfer fragment sequences & further scaffold design
Model 1
Model 2
Model 3
Model 4
Model n

Figure 3.1. HELIX motivation and method overview. (a) Protein interfaces (blue & white) contain highly specific residue-level interactions such as tight hydrophobic packing and hydrogen bonding (left), facilitated by several noncontiguous structural units (center) which are held together by a larger backbone scaffold (gray, right). **(b)** HELIX protocol conceptual overview. First, geometries of α-helices that are capable of making favorable residue-level interactions with the target protein are determined and the sequences of these geometries are optimized. Next, the helical geometries described as vectors, and their relative orientations are matched against a database of scaffolds with reshaped helical units. Finally, the optimized sequence is transferred from the original helical geometries to the matched scaffold, and the surrounding residues are designed.

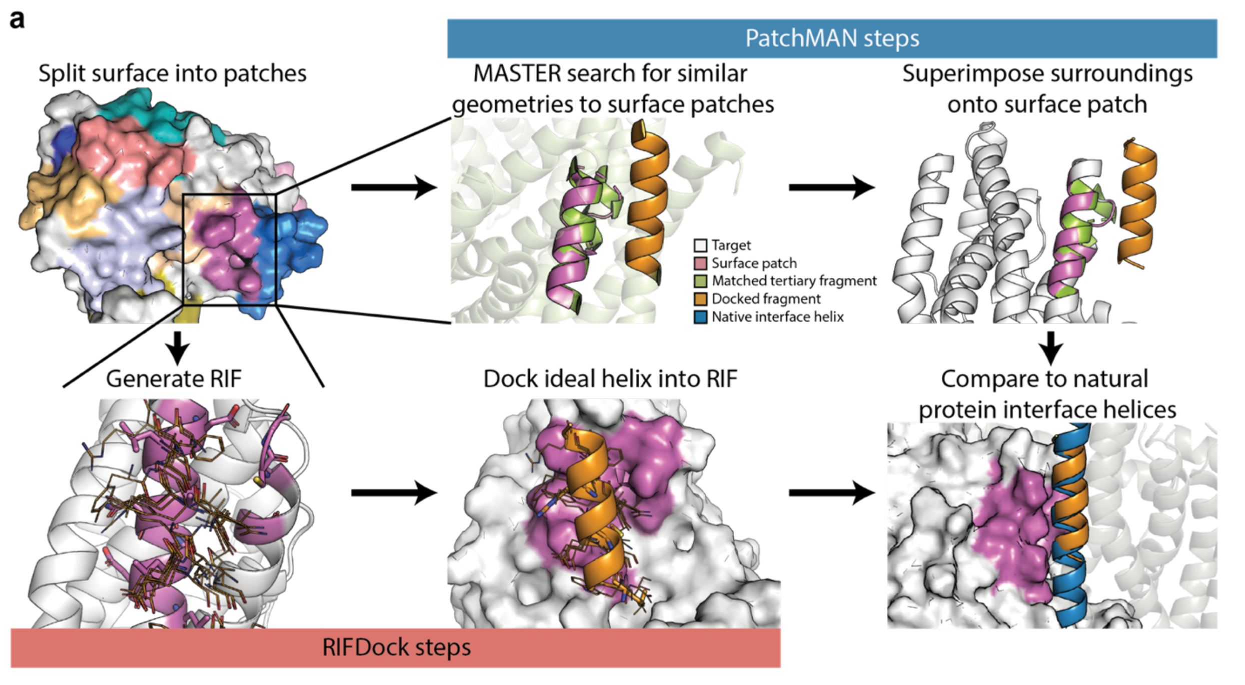
3.3.2 Docking helices

We examined two methods for finding fragment geometries of α-helices that are compatible with the target interface. First, we used RIFDock [9]. RIFDock creates a rotamer interaction field (RIF), a field of discrete rotamers that make interactions that are predicted to be favorable with the target. It does this by placing polar rotamers based on their hydrogen bonding geometries and docking hydrophobic residues, which are then filtered based on Rosetta energy. In a typical application, RIFDock is used to dock a library of protein scaffolds to a small molecule [9] or another protein [10] to find binding geometries that enable a large number of interactions with the ligand. By contrast, here we are docking short, ideal α-helices, with the goal of then matching those helices to a larger protein scaffold. To bias RIFDock towards extensive coverage of the protein surface, we split the protein into surface patches, defined as groups of surface residues that are close in space but not necessarily congruent, and allow RIFDock to create up to 80 docked poses for each patch (**Methods**).

Second, we examined the peptide docking protocol PatchMAN [12]. PatchMAN was developed for blind peptide docking, which is achieved by leveraging the wealth of tertiary interaction data in the PDB [13], [14]. Like our RIFDock protocol, PatchMAN begins by splitting the receptor protein's surface in to patches with a roughly 10 Å radius. For each patch, a nonredundant subset of the PDB is searched (using Method of Accelerated Search for Tertiary Ensemble Representatives, MASTER) [15] for up to 50 tertiary protein fragments that are geometrically similar (<1.5 Å root-mean-square deviation, RMSD) to the receptor protein's surface patch, referred to here as tertiary motifs. PatchMAN then takes a nearby contiguous stretch of residues surrounding these tertiary fragments and superimposes it back onto the receptor protein surface, resulting in possible peptide docking geometries. For peptide docking applications, further refinement is carried out to determine likely docking orientations for a given peptide sequence. For our purposes, it is convenient that PatchMAN results in protein fragments with known geometric compatibility with the target protein surface, and that it does so in a way that is agnostic to the sequence of the docked peptide, since that sequence will be subject to change during later design steps. Because many of these fragments are helical, and because PatchMAN exploits geometric data from protein structures, we hoped this would result in more realistic helix geometries.

To compare the ability of RIFDock and PatchMAN to identify favorable helical interactions in protein interfaces, we created a benchmark set of helices found in the interfaces of protein-protein complexes from the PDB (**Methods**). To minimize bias against RIFDock, which relies on Rosetta-scored rotamers, we restricted the benchmark set such that each interface had at least 2 helices that, when minimized and scored by Rosetta, had more than 4 residues making energetically favorable intermolecular contributions to the estimated binding energy (better than 1.5 Rosetta energy units, REU). We hoped this approach would increase the likelihood that RIFDock would find near-native rotamers when generating the RIF, resulting in docking poses that better match the natural interface helices. The final benchmark set consisted of 34 of interfaces, with a total of 107 of helices (**Table 3.2**).

We ran both RIFDock and PatchMAN on all target proteins in the benchmark set and evaluated the ability of each method to recapitulate the natural interface geometries by calculating the RMSD between their outputs and each benchmark helix (**Fig. 3.2a**). For RIFDock simulations, we evaluated its performance by docking two backbones, each consisting of a single


a
PatchMAN steps
Split surface into patches
MASTER search for similar geometries to surface patches
Superimpose surroundings onto surface patch
☐ Target
🟥 Surface patch
🟩 Matched tertiary fragment
🟧 Docked fragment
🟦 Native interface helix
Generate RIF
Dock ideal helix into RIF
Compare to natural protein interface helices
RIFDock steps

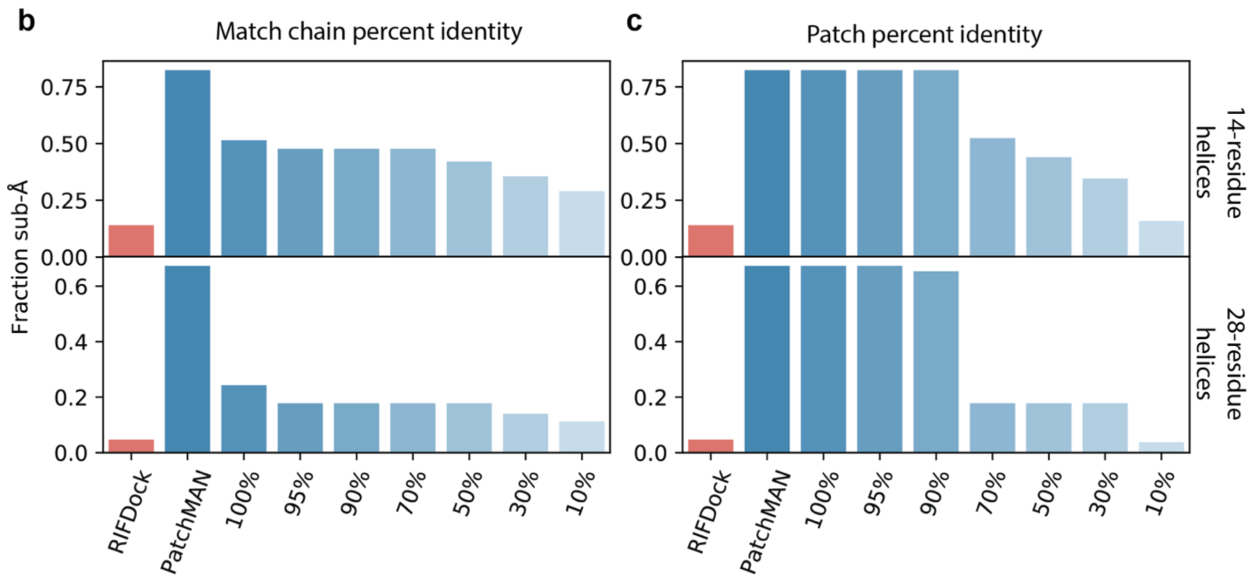


Figure 3.2. Benchmarking α-helix docking. (a) Target protein surfaces are split into patches, each of which is fed into two helix docking protocols. For PatchMAN (top middle & right), the surface patch (pink) is queried (top middle) against a database of tertiary motifs (green: matched tertiary motif; transparent green: scaffold of tertiary motif). A fragment surrounding the matched motif (orange) is then superimposed back onto the target structure (top right, white). For RIFDock (bottom left & middle), a rotamer interaction field (RIF, bottom left, orange lines) is generated to make favorable interactions with the surface patch (pink). Next, ideal α-helices (orange cartoon) are docked into the RIF (bottom middle). For both protocols, the RMSD is computed (bottom

right) between the docked fragment and a helix that is part of the target's native interface (blue). **(b, c)** Results of docking benchmark and impact of homology on PatchMAN interface recapitulation. Bars represent the fraction of benchmark helices for which the protocol (RIFDock, red, or PatchMAN, blue) found a sub-Å helix. Additional PatchMAN results are shown for various levels of sequence identity cutoffs between the **(b)** target and any chain in the protein from which the tertiary fragment was found or **(c)** the surface patch and the matched tertiary motif.

regular alpha helix: a 4-turn helix (14 residues) and an 8-turn helix (28 residues). Similarly, PatchMAN was allowed to return peptides of two lengths, 14 and 28 residues, matching the lengths of the helices used by RIFDock. Since the sizes of the benchmark helices and docked helices were not necessarily the same, these RMSDs were evaluated for the portion of the longer helix that gave the lowest possible value (**Methods**). RIFDock was able to place a helix within 1 Å RMSD of 14% of benchmark helices when docking with a 4-turn helix, and 5% of benchmark helices when docking with an 8-turn helix (**Fig. 3.2b**). Despite not being restricted to returning alpha helices, PatchMAN performed better, finding sub-Å fragments 82% of the time for the 14-residue fragments and 67% of the time for the 28-residue fragments.

Next, we evaluated the ability of PatchMAN to recapitulate natural interface positions without the aid of homologous structures. Since PatchMAN uses existing structures to find candidate peptide geometries, it is possible for it to return helical geometries based on the target or its homologs. While this is desirable behavior for any design applications where a natural interface is being at least partially mimicked, many designable positions might not benefit from homology as they are not at known interface positions. We reasoned that if PatchMAN is able to place helices at native interface positions without the aid of homology, it is also more likely to be capable of placing helices in realistic geometries near positions that are not known to be an interface. We evaluated PatchMAN’s reliance on sequence identity in two ways: first, we excluded results where the tertiary motif came from proteins that contained a chain with a sequence identity above various cutoffs (**Fig. 3.2b**). Second, we excluded results whose matched fragments had above a certain sequence identity with the target patch (**Fig. 3.2c**). As expected, PatchMAN performs best when it can query homologous proteins. We found that it had the sharpest drop in accuracy when it was restricted from using structural information from the target protein (100% sequence identity cutoff), and that this effect was particularly pronounced for the 28-residue fragments (**Fig. 3.2b**). When sequence identity cutoffs were applied to the tertiary motifs, however, PatchMAN withstood much more aggressive cutoffs (**Fig. 3.2c**), suggesting that global similarities can compensate for local sequence differences.

3.3.3 Design of docked helices

The second step in the HELIX protocol is to optimize the amino acid sequences (design) of the identified complementary helical backbones. Since PatchMAN produces geometries based on known protein structures, we saw an opportunity to utilize the sequence information provided by the PatchMAN fragments in addition to their backbone conformation. We tested several methods of maintaining this sequence information, each involving a comparison of the intermolecular interface residue scores of the docked fragments to those of natural proteins. We first minimized the docked fragments with backbone constraints, then compared each docked residue to residues from natural protein interfaces with the same amino acid identity, burial, and secondary structure (**Methods**). If the intermolecular Rosetta score of the docked residue was better than the median value for natural proteins and was not in contact with any buried unsatisfied polar atoms, this interaction was determined to be “native-like” and we applied one of several different types of modifications to our design protocol: (1) We applied a residue type constraint to that position, favoring retention of the amino acid type, but not the specific rotamer (termed *special residue*). (2) We disabled packing for each native-like residue (termed *residue lock*). (3) We applied a “special rotamer” score bonus [16] to the fragment’s minimized rotamer (termed *special residue*) to bias Rosetta towards picking the input rotamer. We also tested these methods in combination with a pairwise decomposable energy term that penalizes buried unsatisfied hydrogen bonds, called 3-Body Oversaturation Penalty (3BOP) [17], and with 3BOP in addition to ramping down constraint terms during the course of the simulation.

We applied each of these design protocols to all docked PatchMAN fragments that were within 1.0 Å RMSD of a benchmark helix so that we could evaluate their ability to reproduce natural protein sequences. In general, methods that favored the preservation of native-like input rotamers (*residue lock* and *special rotamer*) performed better at recovering the native sequence of the interface helix than those that did not, including methods that attempt to retain sequence information, but not rotamer information, via residue type constraints (**Fig. 3.3a**). Interestingly, these methods that preserve rotamer-level information also resulted in a slightly higher number of interface hydrogen bonds and a lower number of buried unsatisfied hydrogen bonds, indicating that preserving polar interactions from protein fragments is more likely to produce well-satisfied hydrogen bonding pairs than Rosetta sampling alone.

Disabling packing for positions which passed the residue-level filter (*residue lock*) resulted in the highest sequence recovery for docked helices that were within 1 Å of a benchmark helix, even compared to other methods which bias Rosetta towards picking that rotamer (**Fig. 3.3a**, *special rotamer*). When combined with 3BOP and constraint ramping, we found that *residue lock* also produced the highest number of cross-interface hydrogen bonds. This is particularly useful in the context of interface design, as buried protein interface positions are typically more polar than protein cores because both partners must be soluble when unbound. *Special rotamer*, while resulting in slightly lower sequence recovery and interface hydrogen bonds than *residue lock*, also had the lowest number of buried unsatisfied hydrogen bonds when the energetic bonus for the input rotamer was set to -3.0 and combined with 3BOP and constraint ramping. As buried unsatisfied polar atoms can incur large energetic costs, we placed particular emphasis on this metric. Moreover, we reasoned that the advantages held by *residue lock* were less likely to be applicable to positions that are not part of a native interface, so we chose to use the special rotamer bonus for designing PatchMAN fragments, giving Rosetta the opportunity to sample solutions with potentially better Rosetta scores than the input rotamer while still encouraging the use of sidechain information from the docked fragments.

We next filtered the docked helices, hoping to increase the likelihood of matched scaffolds having high affinity for the target protein. We used the benchmark dataset to optimize the filtering parameters; first, the *special rotamer* design protocol was run on all PatchMAN results (not just sub-Å results) with the 3BOP score term, constraint ramping, and a weight of -3.0. Next, several filtering metrics were analyzed for their impact on the coverage of the benchmark set; we wanted to maintain recapitulation of natural protein interfaces while reducing the

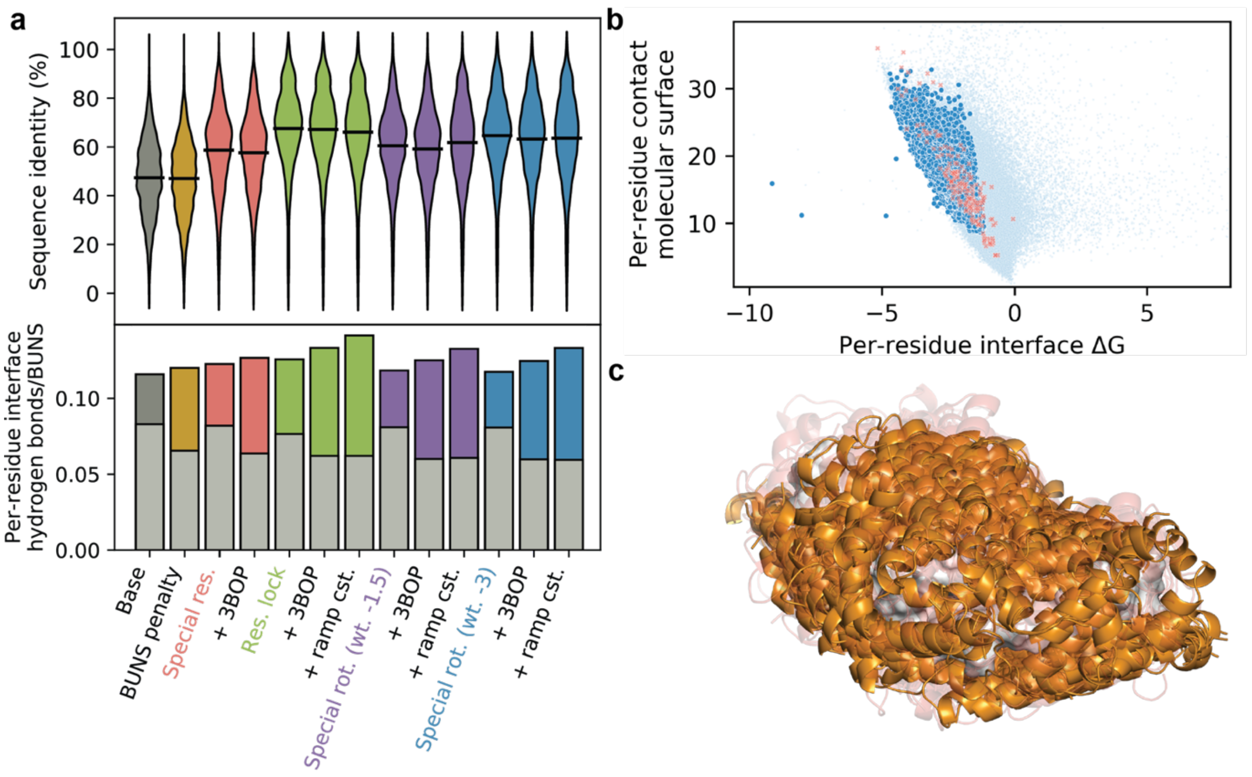


Figure 3.3. Benchmarking α-helix design. (a) Interface metrics for designed PatchMAN fragments. Top: Sequence identity distribution between the designed helices and the closest benchmark helix segment for various design protocols. Bottom: Average number of interface hydrogen bonds (colored bars) or buried unsatisfied hydrogen bonds (gray bars) between the PatchMAN fragment and the target interface for each design protocol. **(b)** Filtered PatchMAN fragments (dark blue) show distributions of interface ΔG and CMS scores more similar to native proteins (red) than the unfiltered fragments (light blue). **(c)** PatchMAN helix distributions. PatchMAN helices that have been filtered (orange cartoon) cover much of the target surface (white). Transparent red: helices eliminated by filtering.

number of helices in the dataset. Specifically, we examined the change in Rosetta score upon binding (interface ΔG), a metric that measures shape complementarity while penalizing poor packing (contact molecular surface, CMS) [10], the percentage of helical residues, and the number of buried unsatisfied hydrogen bond donors and acceptors. We found that necessitating that all fragments be at least 70% helical, have at most one buried unsatisfied hydrogen bond, were among the best (lowest) 40% in terms of their interface ΔG, and were among the best (highest) 50% in terms of their CMS score resulted in a large decrease in the number of helices without significant impact on the number of benchmark helices with a PatchMAN fragment within 1.0 Å (**Appendix Fig. 3.6**). We also analyzed the latter two metrics (interface ΔG and CMS) for all benchmark helices and found that after filtering, the docked fragments had scores similar to those of natural interface helices (**Fig. 3.3b**) while still covering the majority of the protein surface (**Fig. 3.3c**).

3.3.4 Designing de novo protein binders

We tested our method by designing protein binders for 9 signaling and pathogen-derived proteins for which miniprotein binders were recently designed by Cao et. al [10]: fibroblast growth factor receptor 2 (FGFR2, PDBID 1DJS), epidermal growth factor receptor (EGFR, PDBID 1MOX), CD3 delta chain (CD3δ, PDBID 1XIW), angiopoietin-1 receptor (TIE2, PDBID 2GY7), tropomycin receptor kinase A (TrkA, PDBID 2IFG), platelet-derived growth factor receptor (PDGFR, PDBID 3MJG), Rickettsia typhi VirB8-like protein (VirB8), insulin receptor (InsulinR, PDBID 4OGA), and insulin-like growth factor 1 receptor (IGF1R, PDBID 5U8R). While it is possible to target certain regions of the target protein by only running PatchMAN on patches that contain a specified set of residues, we chose to design binders for the entire protein to evaluate our protocol more fully. We designed binders for each of these targets using the protocol outlined above (**Figs. 3.1b, 3.4a**). For each target, we generated an average of 130 surface patches, and for each patch a median of 65 fragments were found using PatchMAN (**Appendix Table 3.1**). We designed these fragments using the *special rotamer* protocol with a weight of -3.0, the 3BOP score term, and ramping down constraints. After filtering, between 114 and 1302 fragments were left for matching (**Fig. 3.4b, Appendix Table 3.1**).

We next matched the docked fragments to a library of scaffolds created by combinatorial sampling of loop-helix-loop (LHL, loop-helix-loop unit combinatorial sampling: LUCS) elements within a *de novo* Rossmann fold topology [11]. This library offered several features that made it well-suited for this task: First, the library contains a diverse set of reshaped backbone geometries for two of its helices (**Fig. 3.4c**, right), making it more likely to achieve good alignment with the docked fragments. Second, the remainder of the protein consists of a four-stranded β-sheet separating the reshaped region from two additional α-helices, giving the scaffold a sizeable stable region, which remains largely unchanged during design. Finally, the sequence of each library member had previously been optimized for stability (**Methods**), so only interface regions would need to be redesigned. Matching was performed by first defining the binned relative orientations of the reshaped helices in both the scaffold library and the set of docked helices (**Methods**), then searching for pairs of helices in the scaffold library that shared the same relative orientation as any pair of docked helices. Successful matches were then superimposed into the docked helices and passed through a low-resolution clash filter (**Methods**). Out of 48,187 scaffolds, between 18,129 and 45,174 (**Fig. 3.4b , Table 3.1**) were matched to a pair of docked helices, and between 1,384 and 30,135 passed the initial clash filter. A final filter that included the clash score and the RMSD between the docked helices and their match was tuned for each target such that between 365 and 835 complexes passed, which were subsequently designed to optimize their interactions with the target (**Methods**). Despite the low percentage of scaffolds left at the end of this filtering,

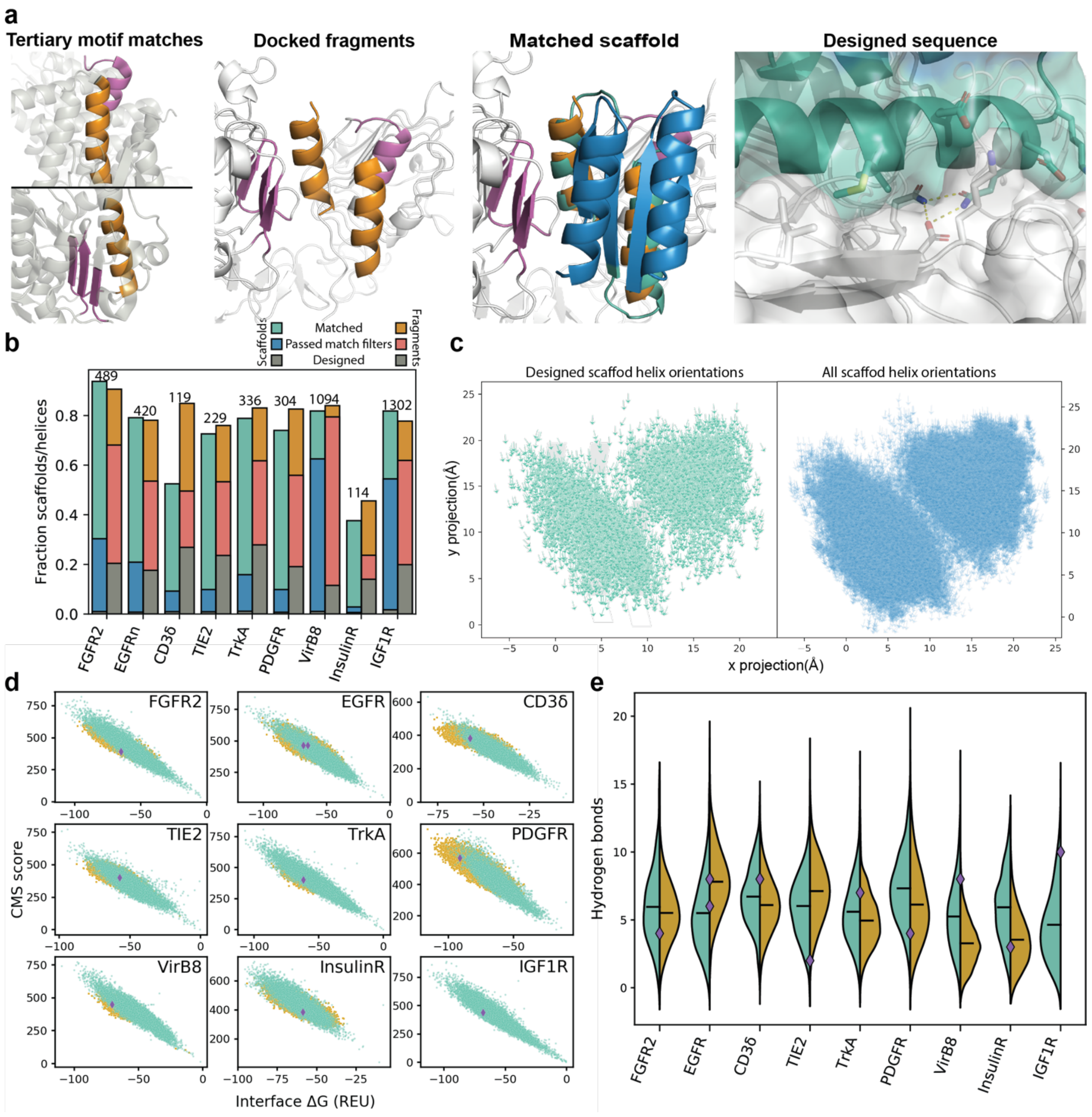


Figure 3.4. Design of protein-interacting proteins. (a) Demonstration of design process. Left: PatchMAN fragments (orange) in their original context (gray), with their interacting tertiary motif (pink). Mid-left: PatchMAN fragments in their new context with the target protein (white) and the surface patches (pink). Mid-right: A matched scaffold (blue) with two helices (teal) that have similar relative orientations to the PatchMAN fragments (orange). Right: The final sequence of the designed scaffold (blue & teal) with target (white) showing complementary packing and polar

interactions. **(b)** Retention of scaffolds and PatchMAN fragments throughout the design process for each target. Shown are the fractions of scaffolds (teal, blue & gray bars) or PatchMAN fragments (orange, red & gray bars) that were involved in a match (teal, orange), that were involved in a match that passed the initial clash filter (blue, red), and that were involved in a match that was eventually designed (gray). Numbers above the bars represent the number of PatchMAN fragments at the start of matching; for all targets, there were 48,187 scaffolds at the start of matching. **(c)** Distribution of scaffold helical geometries in the designs (left, teal) versus the entire library (right, blue). Geometries are represented as vectors, where each arrow represents the centroid position and direction of the helix projected onto the β-sheet backbone. **(d)** Interface ΔG and CMS score plots for HELIX designs (teal), experimentally tested miniprotein binder designs (yellow), and experimentally optimized miniprotein binder designs (purple diamonds) for each target. **(e)** Distributions of the number of well-satisfied interface hydrogen bonds present in the HELIX designs (teal) or miniprotein binder designs (yellow) for each target. Bars represent the mean value. Values for the experimentally optimized miniprotein binders are shown as diamonds (purple).

the final models still had a wide range of loop-helix-loop geometries (**Fig. 3.4c**, left; **Appendix Fig. 3.7**) and binding modes (**Appendix Fig. 3.8a**).

A major advantage of HELIX is that optimizing the sequences of smaller tertiary fragments is easier than optimizing the sequences of entire scaffolds. To maintain this advantage, we transferred the interface residues of each docked fragment onto the matched scaffold. Because we found rotamer-level information to be beneficial when designing the PatchMAN fragments, we attempted to preserve the sidechain conformations of the transferred residues by applying constraints based on their original conformation, and then designing the surrounding residues while minimizing the transferred residues (**Methods**). In keeping with the lessons learned from our design benchmark (**Fig. 3.3a**), after optimization of the transferred residue positions, we either prevented them from repacking or applied a special rotamer bonus with a weight of -3.0.

Each match complex was designed 20 times, half with the *residue lock* protocol and half with the *special rotamer* protocol.

We next compared our designs to all available models of the experimentally tested miniprotein binders, including published models of the final designs that were optimized by *in vitro* site-saturation mutagenesis. We note that because this is the set of all *tested* designs, they presumably represent a well-optimized fraction out of a much larger set of computational models, whereas our dataset contained all designs except for the few that had positive interface ΔG scores (0.17% of all designs). We also note that design models of miniprotein binders were not available for IGF1R except for the final experimentally optimized model. We calculated several metrics for both the HELIX designs and the miniprotein binders, including the CMS and interface ΔG scores, the number of interface hydrogen bonds, and the number of buried unsatisfied hydrogen bonds in the interface. HELIX resulted in designs with similar interface ΔG and CMS score distributions compared to the filtered miniprotein binders for most targets (**Fig. 3.4d**). For CD3δ and PDGFR, the interface ΔG and CMS scores of the miniprotein binders were more favorable (lower interface ΔG and higher CMS) on average compared to those of the HELIX outputs; however, given that some HELIX proteins still had scores similar to the best of the miniprotein binders, it is possible that this difference could be resolved by scaling up our design process and filtering the outputs. For all targets, HELIX produced designs whose interface ΔG and CMS scores overlapped with the scores computed for the final optimized miniprotein binders.

HELIX thoroughly optimizes the interactions between its docked fragments and the target protein, including polar interactions, so a possible advantage is that it may be more likely to create interfaces with well-satisfied hydrogen-bonding networks. To test this, we sought to examine the prevalence of interfacial hydrogen bonds in our designs compared to the designed miniprotein binders. Designs with large numbers of hydrogen bonds are more likely to have buried unsatisfied polar groups, which can carry a heavy energetic cost [18], so we first filtered each dataset down to designs that had zero buried unsatisfied hydrogen bonds in the interface (**Methods**, *Designing matched scaffolds*). Additionally, since we are not interested in highly polar interfaces that have low predicted affinities, we further filtered the dataset such that only designs with an interface ΔG of less than -30 REU were analyzed. When we examined the distribution of interface hydrogen bonds in the remaining designs (**Fig. 3.4e**), we saw that for 6 out of the 8 targets for which miniprotein binder data was available, HELIX indeed produced a higher average number of well-satisfied interface hydrogen bonds. In line with this observation, visual inspection of our designs frequently shows well-satisfied hydrogen-bonding networks (**Fig. 3.4a, right**). Two targets, EGFR and TIE2, showed the opposite trend. Interestingly, these two targets had sparse coverage of the regions targeted by the miniprotein binders by matched scaffolds, despite significant overlap of the targeted region with PatchMAN fragments (**Appendix Fig. 3.8b**). It is possible that these targeted sites are well suited to form hydrogen-bonding interactions, but were missed by HELIX due to limited scaffold diversity or structural information about the region from which PatchMAN could draw; indeed, these sites were chosen for their proximity to ligand-binding sites, and hydrogen-bonding networks are involved in native ligand binding for both EGFR and TIE2 [19], [20].

3.4 Discussion

Design of *de novo* protein binders is challenging because of the linked requirements between the scaffold and residue-level interactions with the target. Here, we show that bottom-up interface design starting from protein fragments docked in a sequence-agnostic manner can address this challenge. Conceptually, the search for compatible fragment geometries shares many similarities with previous approaches to ligand binding design [21], [22]; where these methods split a target ligand into fragments so that motif interactions can be mined from the PDB, the PatchMAN portion of our protocol does the same for protein surface interactions. This mining of sub-structure data is likely an important determinant of HELIX's ability to create designs with favorable metrics, and its success in our docking benchmark (**Fig. 3.2**) may occur in part as a result of structural degeneracy in protein-protein interfaces [23].

Scaffold placement via HELIX is limited by both the placement of docked helices and the variation of the scaffold library, both in terms of relative helix orientations and overall topology. In this work, we only matched PatchMAN fragments to a single *de novo* Rossmann fold with two reshaped LHL units generated via LUCS. While this resulted in a wide variety in binding modes, it is possible that any given topology will be fundamentally incompatible with certain binding sites in our targets, as suggested by the somewhat clustered placement of binding scaffolds seen for some targets (**Appendix Fig. 3.8b**). It is also possible, however, that this clustering represents real binding site preferences for these proteins, as is often seen in nature [24]. Nonetheless, the LUCS protocol was critical in enabling HELIX's ability to match scaffolds, and it would be interesting to see how other topologies with reshaped LHL units would change the coverage of the target protein.

HELIX benefits from separating the problem of designing favorable interactions with their target into many smaller optimization tasks, reducing the combinatorics of the sequence space that must be explored to produce binding modes with many high-affinity interactions. To illustrate, we designed on average just 1.0% of our 48,187-scaffold library (**Table 3.1**). In contrast, a naïve top-down approach, even if no docking step were required, might involve exhaustively sampling our scaffold library, meaning 96,374 helices (two per scaffold) would need to be designed. HELIX, on the other hand, designed only 5,371 fragments on average, while still sampling native-like geometries (**Fig. 3.2**). Docking-based approaches such as [10] similarly reduce the size of the search space, but do so by sacrificing resolution in the early stages. Here, all fragments are designed at full resolution from the start, helping to facilitate favorable interface metrics in the final designs (**Fig. 3.4d-e**). We believe that this full-resolution optimization applied to small fragments helps HELIX to design more polar interactions with the target interface (**Fig. 3.4e**), which is an important consideration owing to the relative scarcity of large hydrophobic patches in protein-protein interfaces [25].

Several further advantages arise because of HELIX's fragmented design process: For example, by thoroughly optimizing interactions between individual α-helices, it is possible to specifically filter for desired interface properties, such as the number of hydrogen bonds, before a scaffold is even selected. Such tuning of interface properties could be useful for objectives such as avoiding aggregation or minimizing immune response. It may also be possible to combine

HELIX with other methods that benefit from having many binding geometries to choose from, such as matching of functional sites [21], [26], [27], where combining multiple fragments that each meet one of several geometric criteria (e.g., matching a single functional site residue per helix) could be more tractable than attempting to find or create a scaffold that meets them all at once.

While we matched PatchMAN fragments to a pre-generated LUCS library, it may be desirable to forego scaffold matching altogether, opting instead to utilize recent advances in machine learning for protein structure prediction [28], [29] to hallucinate structures [30]–[32] that meet the geometric requirements defined by the best-scoring docked fragments. This strategy would additionally circumvent the requirement that docked fragments be helical, as arbitrary geometries, including loops, can be expressed via the lossfunction. However, it is unclear at this time whether hallucinated structures are more likely to be stable, or whether their accuracy is sufficient to recapitulate the specific geometries required to facilitate the interactions designed into our docked fragments.

We show here that HELIX is capable of creating protein-binding proteins with computational interface metrics in line with or better than designs made using similar methods. However, further computational and experimental validation will be important in demonstrating its applicability to real design challenges. Nonetheless, we believe HELIX has the potential to be a powerful tool for the design of protein-protein interfaces, and its fragmented design workflow lends itself to many interesting design applications.

3.5 Methods

3.5.1 Software versions

All design was carried out using PyRosetta version 2021.11+release.e9f4797. RIFGen and RIFDock outputs were generated using commit 0a30a0971692746359593df0e9674d24a3751254 from https://github.com/rifdock/rifdock.

HELIX

HELIX is available as a Python package at https://github.com/ckrivacic/helix_matcher/tree/publication. Once installed, HELIX works by running a series of commands which outlined below and in the software documentation.

3.5.2 Generation of helical geometries: PatchMAN

We used a slightly modified version of the PatchMAN protocol as described in [12] to determine geometries of protein fragments, including helical fragments, that were compatible with the target geometry. Generating surface patches, searching for local structural motif matches using MASTER, and generating initial complexes were carried out as described in [12]. The protocol was modified only in the naming convention for the initial complex output files to indicate which MASTER hits these complexes originated from. Generation of PatchMAN matches was carried out with HELIX using the following command (let "<workspace>" be the directory of the HELIX workspace):

```
helix 01_prep_patchman <workspace>

helix 02_patchman <workspace>
```

3.5.3 Generation of helical geometries: RIFDock

Rotamer interaction fields (RIFs) were generated only for the residues in each surface patch (defined identically to the PatchMAN protocol). Parametrically generated ideal polyvaline alpha helices with lengths of either 14 or 28 residues were then docked into the rotamer interaction field. For each combination of patch and scaffold, up to 80 RIFDocked outputs were generated. We used the following flags to generate the RIFs for each surface patch (let "{target}" be the path to the target, and "{db}" be the path to the Rosetta database):

```
-rifgen:data_cache_dir .
-database {db}
-rifgen::rif_type RotScore
-rifgen:target      {target}
-rifgen:target_res residues.txt
-rifgen:outdir     .
-rifgen:outfile    test_out.rif.gz
-rifgen:score_cut_adjust 0.8
-rifgen::rosetta_field_resl 0.25
-rifgen::search_resolutions 3.0 1.5 0.75
rif gen
-rifgen:beam_size_M 1000.0
-rifgen:rif_hbond_dump_fraction 0.01
-rifgen:rif_apo_dump_fraction 0.01
-rifgen:extra_rotamers false
```

```
-rifgen:extra_rif_rotamers true
-renumber_pdb
-add_orbitals
-hbond_cart_sample_hack_range 0.33
-hbond_cart_sample_hack_resl  0.33
-rifgen:tip_tol_deg        60.0 # for now, do either 60 or 36
-rifgen:rot_samp_resl       6.0
-rifgen:hash_preallocate_mult 0.125
-rifgen:max_rf_bounding_ratio 4.0
-rifgen:hash_cart_resls   16.0   8.0   4.0   2.0   1.0
-rifgen:hash_cart_bounds   512   512   512   512   512
-rifgen:lever_bounds      16.0   8.0   4.0   2.0   1.0
-rifgen:hash_ang_resls     38.8  24.4  17.2  13.6  11.8 # yes worky worky
-rifgen:lever_radii        23.6 18.785501 13.324600  8.425850  4.855575
```

We used the following flags to dock ideal helices into the RIF (let “{target}” be the path to the target protein, “{cache}” be the path to the cache generated by RIFDock, and “{scaffold}” be the path to the ideal helix to be docked):

```
-rif_dock:target_pdb            ./{target}.rif.gz_target.pdb.gz
-rif_dock:target_rf_resl        0.25
-rif_dock:target_rf_cache       ./{cache}
-rif_dock:target_bounding_xmaps ./{target}.rif.gz_BOUNDING_RIF_16.xmap.gz
-rif_dock:target_bounding_xmaps ./{target}.rif.gz_BOUNDING_RIF_08.xmap.gz
-rif_dock:target_bounding_xmaps ./{target}.rif.gz_BOUNDING_RIF_04.xmap.gz
-rif_dock:target_bounding_xmaps ./{target}.rif.gz_BOUNDING_RIF_02.xmap.gz
```

```
-rif_dock:target_bounding_xmaps ./{target}.rif.gz_BOUNDING_RIF_01.xmap.gz
-rif_dock:target_rif            ./{target}.rif.gz
-rif_dock:extra_rotamers        0
-rif_dock:extra_rif_rotamers    1
-rif_dock:rot_spec_fname        ./rotamer_index_spec.txt
-database {db}
-rif_dock:rotrf_cache_dir cache/
-rif_dock:data_cache_dir  .
-rif_dock:cache_scaffold_data true
-rif_dock:scaffolds {scaffold}
-rif_dock:scaffold_res
-rif_dock:outdir docked_full/
-rif_dock:dokfile all.dok
-rif_dock:n_pdb_out 80
-rif_dock:align_output_to_scaffold false
-rif_dock:pdb_info_pikaa false
-require_satisfaction 0
```

Generation of RIFDock outputs can be carried out via HELIX using the following commands (let “<workspace>” be the directory of the HELIX workspace):

```
helix prep_rifdock <workspace>
helix rifdock <workspace>
```

3.5.4 Creation of interfacial helix benchmark set and RMSD calculations

Proteins from the PDB were analyzed for interfaces containing helices with large energy contributions to binding. For each interface residue, we summed the intermolecular Rosetta energy terms to estimate its contribution to the binding energy. We then took complexes that contained at least two helices, each containing at least 4 residues with at least a -1.5 REU intermolecular energy score. We additionally required that all benchmark proteins be X-ray structures with less than 2.0 Å resolution. Complexes were then inspected visually, and only interfaces that had helices that appeared mostly ideal were included in the final benchmark set. We selected a total of 34 protein complexes to include in the benchmark, consisting of 107 interface helices (**Appendix Table 3.2**).

RMSDs between each benchmark helix and the RIFDock or PatchMAN outputs were calculated by first determining whether the docked fragment or the benchmark helix contained more residues. For the shorter peptide, all residues were used in the RMSD calculation. For a shorter peptide *n* residues long, we calculated the RMSD between each possible stretch of *n* residues of the longer peptide and all residues of the shorter peptide. We used the lowest of these values as the final RMSD.

3.5.5 Summarizing PDB interface residue scores

31,828 multimeric protein structures from the PDB were minimized with backbone and sidechain constraints using PyRosetta. Interface residues were classified based on whether their Cβ-Cβ distance was within 8 Å. For each interface residue, we calculated the total cross-chain energy using the Ref2015 scorefunction in Rosetta, and saved this information along with the residue's secondary structure (determined using the dynamic secondary structure prediction algorithm, DSSP [33]) and burial (determined by the LayerResidueSelector in Rosetta). To summarize the data, we grouped each amino acid by its secondary structure and burial, then took the median value to avoid disproportionate influence from unrealistically low or high scores from nonphysical structural features (such as multiple alternate conformations occupying the same space or interactions with malformed residue types).

3.5.6 Helix refinement

The sequences of geometrically compatible helices found by PatchMAN were refined in PyRosetta. For all design protocols, we only allowed design of interface positions on the docked helix and allowed repacking for all positions on the helix as well as any residues on the target protein that could possibly clash with all allowed rotamers for the docked helix. For design positions, all residue types were allowed except glycine. PatchMAN outputs were minimized with backbone constraints prior to design. For the design of binders of the 9 targets, we also biased the simulations towards favorable interface contacts by upweighting cross-chain energy terms by a factor of 1.5. On top of these base parameters, we tested several variations of our design protocol, described below. Helix refinement was carried out by running the following command, appended with options listed for the different protocols, where <workspace> is the path to the HELIX workspace.

```
helix 03_design_patchman <workspace>
```

Special residue: We compared the amino acid sequence of each target surface patch to the tertiary motif found via MASTER to generate each PatchMAN output. If the sequence identity between the patch and its match was above 70%, we applied a score bonus of -1.5 to all rotamers that matched the input residue type. This protocol can be enabled by passing the "--special-res" and "--keep-good-rotamers" options.

Residue lock: For each interface residue in the minimized PatchMAN output, we calculated its cross-chain score and compared it to the summarized interface residue scores (described above). If the cross-chain score was lower than the median cross-chain score for residues with the same amino acid type, burial, and secondary structure (helix, loop, or beta sheet), and it did not contain or come in contact with any buried unsatisfied polar atoms, we considered that residue to be "native-like" and disallowed packing for that position (**Appendix Fig. 3.9**). This protocol can be enabled by passing the "`--keep-good-rotamers`" option.

Special rotamer bonus: Native-like residues were determined as described in Residue Lock. Instead of disabling packing entirely, we applied a score bonus to the input rotamer of either -1.5 or -3.0. We note that whereas other design protocols used the FastDesign class in PyRosetta to carry out design, here we had to build our own Python version of the mover which calls the same Rosetta functions as FastDesign, but enables the addition of special rotamers to the PackerTask each time it is created. This protocol can be enabled by passing the "`--keep-good-rotamers`", "`--special-rot`" and "`--special-rot-weight=-1.5`" or "`--special-rot-weight=-3.0`" options.

Buried unsatisfied hydrogen-bond penalty: We added the buried unsatisfied hydrogen-bond penalty (3BOP) [17] to the Ref2015 scorefunction [34], using a penalty value of 5.0, a hydrogen-bonding threshold of -0.5 REU, a burial depth of 4.0, and assuming a constant backbone composition. This protocol can be enabled by passing the "`--buns-penalty`" option.

In addition to these options, ramping down constraints can be enabled by passing the "`--ramp-cst`" option.

After refining the docked helix sequences, we calculated a number of metrics. We used the InterfaceAnalyzerMover in Rosetta to calculate interface ΔG, defined as the difference in Rosetta score between the sum of the separated chains and the complex score. We also calculated the contact molecular surface between the two chains as described in [10]. Interface hydrogen bonds were defined as a pair of intermolecular residues that had a hydrogen-bonding score in Rosetta of less than -0.5 REU. Interface buried unsatisfied hydrogen bonds were counted using the default burial cutoff of 4.4 Å distance from the surface.

3.5.7 Filtering helices

For a given target protein, docked fragments passed filters if they had 1 or fewer buried unsatisfied hydrogen bonds for all interface residues and were at least 70% helical. Additionally, we only kept fragments that had a change in Rosetta score upon separating the helix from the target below the 40th percentile for fragments of the same length and that were above the 50th percentile in contact molecular surface area for fragments of the same length. Filtering can be run via the following command, where <workspace> is the path to the HELIX workspace:

```
helix 04_filter <workspace>
```

3.5.8 Scaffold library

Proteins with altered loop-helix-loop (LHL) geometries were created started from a *de novo* designed Rossmann fold [35] as described previously [11]. Briefly, for each reshaped region, loops at either end of the LHL unit were systematically sampled from a library of protein loop fragments. Helices were then grown from the ends of each loop to meet in the middle, and LHL units with suboptimal geometries were discarded. Next, combinations of LHL units were tested for clashes. The sequences of the reshaped regions and their surroundings were then designed using Rosetta and filtered such that only designs that had Rosetta holes scores [36] of less than 0 and fragment qualities lower than 2 Å [37], [38]. The resulting scaffolds underwent further sequence optimization to produce the 48,187 structures used for matching.

3.5.9 Matching helical geometries to scaffolds

All helices, both docked from PatchMAN and from within the scaffold database, were vectorized to facilitate matching. To vectorize each helix, we first calculated the direction of each consecutive set of C, N, and Ca atoms as their normalized axis of rotation with respect to the next set of atoms. The resulting vector was averaged for all residues, scaled up to the length of the helix (defined as the distance between the first and last Ca), and translated to the centroid of the helix backbone C, N, and Ca atoms.

We defined the relative orientation between two helix vectors using two angles, one dihedral, and one distance measurement. For two helix vectors AB and CD, we calculated angles ABC and BCD, the dihedral ABCD, and the distance between the centroids of AB and CD. These values were calculated for each pair of helices within a given scaffold database protein, or each pair of helices within the filtered docked helices, and binned with 30 degree/5 Å bin sizes. Bins were defined twice for each parameter describing the relative helical orientations, with the second bin being offset from the first by half of the bin size (15 degrees or 2.5 Å). This way, any pair of helices that had similar relative orientations to another pair of helices would be guaranteed to share at least one of the 16 possible bins. Binned values were then placed in a hash table (one for query helices and one for database helices), where both an identifier for the protein and the bin itself was hashed. Docked helices were only binned if they had a centroid distance of greater than 2.0 Å and less than 20 Å to reduce the size of the query database, as helices outside these restrictions would likely be either clashing or farther than any pair of helices in our scaffold database.

To match pairs of docked helices to a database protein, the database was queried for the binned relative orientation of the pair of docked helices, and the identifier for the database protein was saved. This was carried out for all pairs of query proteins. We then constructed a graph where the nodes consisted of a tuple representing two helices, one docked helix and one database helix, that could be superimposed on one another. An edge was created between nodes where the docked helices of each node had the same binned relative orientation as the database helices. The maximum number of helices from a database protein that could be superimposed onto docked helices at once can therefore be defined as the maximum dense subgraph, where all superpositions are compatible with one another (have the same binned relative orientation). For each dense subgraph, we calculated the rotation and translation matrices between the matched database helices and the docked helices and applied them to the database protein coordinates.

The scaffold database was prepared by running the following commands, where <workspace> is the path to the HELIX workspace, and <pdb_folder> is a folder containing all of the scaffold PDBs:

```
helix scan_pdb_folder <workspace> <pdb_folder>
helix bin_database <workspace> --angstroms 2.5 --degrees 15
```

Matching was carried out using the following command, where <workspace> is the path to the HELIX workspace:

```
helix 05_match <workspace> --angstroms 2.5 --degrees 15
```

3.5.10 Scoring matches

We implemented several low-resolution scoring metrics to filter out matches with egregious clashes or whose helices were translated compared to the docked helices by half a helix turn, which would cause optimized sidechain interactions in the docked helices to be incompatible with the match. First, scaffolds were superimposed onto the docked helices to which they were matched using the transformation defined by their helix vectors. To filter out clashing proteins, we defined a convex hull around the target chain [39], [40], then counted the number of backbone atoms in the matched protein that fell inside the convex hull. Matches with more than 20 atoms inside the convex hull were discarded.

For fragments that had a clash score of less than 20, we next calculated the RMSD between each docked helix and its match. Since docked and matched helices are not necessarily the same length and could be staggered along their long axis, we first calculated overlapping sets of atoms to use in the RMSD calculation. For helices A and B, CA atoms from helix A were included if they fell between two planes, both perpendicular to the vector describing B, that intersected the start and end points of B. The reverse operation was performed to determine CA atoms to include from helix B. If the two resulting atom sets differed in length, we calculated RMSDs for all possible starting points of the longer set and chose the lowest of these values.

For most targets, matches that had a clash score of less than 10 and an RMSD of less than 1.2 were exported for design. We adjusted these parameters slightly for the following targets: For TIE2, we increased the RMSD threshold to 1.25. For VirB8, we decreased the RMSD threshold to 1.0 and added an additional constraint that the helical fragments that the scaffold was matched to could not have any buried unsatisfied hydrogen bonds in their interface. For PDGFR, we increased the RMSD threshold to 1.3. For InsulinR, we increased the RMSD threshold to 1.5. For IGF1R, we decreased the RMSD threshold to 1.0.

Match scoring was carried out using the following command, where <workspace> is the path to the HELIX workspace:

```
helix 06 score_matches <workspace>
```

Exported complexes were created using the following command, where <workspace> is the path to the HELIX workspace:

```
helix 07 export_matches <workspace>
```

3.5.11 Designing matched scaffolds

The design process consists of several stages, with the goal of keeping any interface residues already optimized during the PatchMAN design protocol that could be accommodated by the scaffold (**Appendix Fig. 3.10**). First, residues from the designed PatchMAN fragments were transferred to the matched scaffold. Second, we attempted to optimize the local environment of these transferred residues so that we could determine if they were compatible with the scaffold geometry. We performed several rounds of design, keeping compatible transferred residues constant or applying a SpecialRotamer score bonus to them. Finally, we relaxed the structures and calculated metrics.

To transfer the residues, we first calculated distance maps between the Cαs in each PatchMAN fragment and their corresponding scaffold helix. Interface residues here were defined via the InterfaceByVector residue selector in Rosetta: residues on opposite chains whose Cβs were within 5.5 Å of each other or whose Cβs were within 11 Å and whose Cα-Cβ vectors pointed toward each other within a cone of 75° were considered to be part of the interface. For each interface scaffold helix residue whose Cα fell within 1.5 Å of a PatchMAN fragment residue, we mutated the scaffold residue to the amino acid type found during the earlier PatchMAN design step using the PackRotamersMover in Rosetta. After all interface residues that were proximal to a PatchMAN residue were mutated, we next defined coordinate constraints for the transferred residues' sidechains based on their position in the PatchMAN fragment, and then performed another round of repacking, this time allowing repacking of all mutated residues as well as their neighbors (defined using the ClashBasedRepackShell residue selector with num_shells=2).

We next attempted to relieve any remaining clashes between transferred residues and the scaffold by performing a single round of FastDesign. Here, the interface definition was expanded to include all residues that were within 10 Å of the opposite chain. During this and subsequent design steps, backbone constraints were generated, and their weight was ramped down during the course of each design run. We also used the 3BOP energy term as described for helix refinement, though here we did not assume a constant backbone and had a stricter burial cutoff of 3.5 Å. Design was allowed for all interface positions except for the transferred residues, giving Rosetta the opportunity to find mutations that stabilize the previously optimized positions. After this step, a final list of "native-like" residues that were still favorable in their new context was obtained using the logic described above (**Methods**, Helix refinement).

Two rounds of FastDesign were then carried out on the scaffold-target complex using the InterfaceDesign2019 relax script [41]. Native-like residues were either assigned a SpecialRotamer bonus of -3.0 or prevented from repacking as described for helix refinement. Refined sequences were then relaxed using a single repeat of Rosetta FastRelax, and metrics were calculated as described above with the exception that we used a stricter burial cutoff of 0.1 Å when calculating buried unsatisfied hydrogen bonds.

Design with the *residue lock* protocol was carried out using the following command, where <workspace> is the path to the HELIX workspace:

```
helix 08_design_scaffolds <workspace> --suffix _base
```

Design with the *special rotamer* protocol was carried out using the following command, where <workspace> is the path to the HELIX workspace:

```
helix 08_design_scaffolds <workspace> --special-rot --suffix _special_rot
```

3.6 Appendix

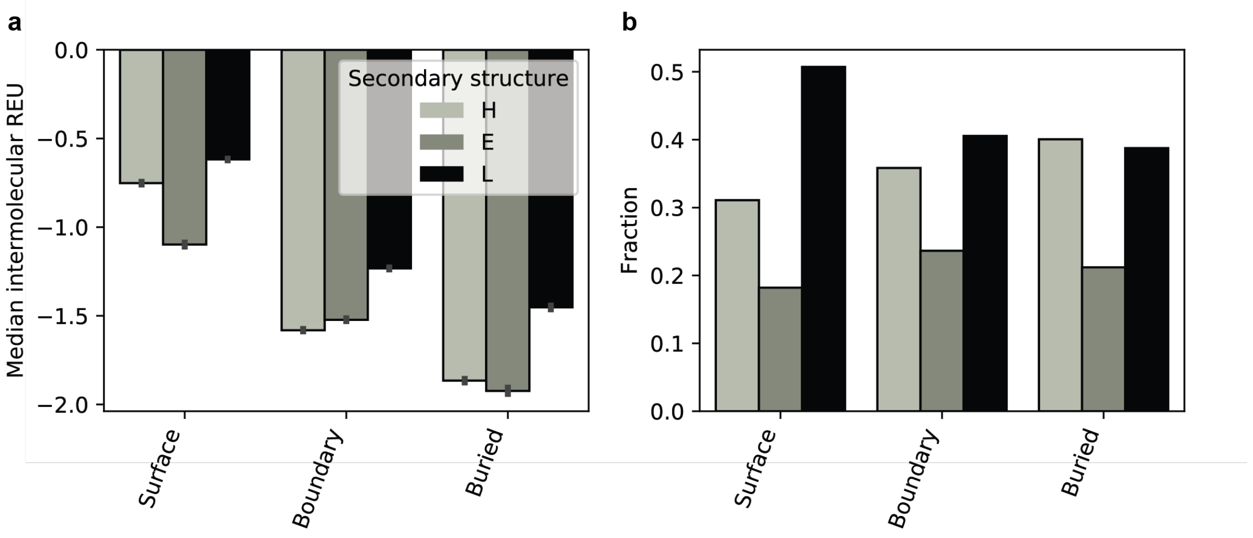


Figure 3.5. Analysis of interface positions. (a) Median interface scores for all surface, boundary, or buried residues, split by α-helices (light gray), β-sheets (dark gray), or loops (black). Error bars represent the standard error. **(b)** Fraction of surface, boundary, or buried interface residues that belong to α-helices (light gray), β-sheets (dark gray), or loops (black).

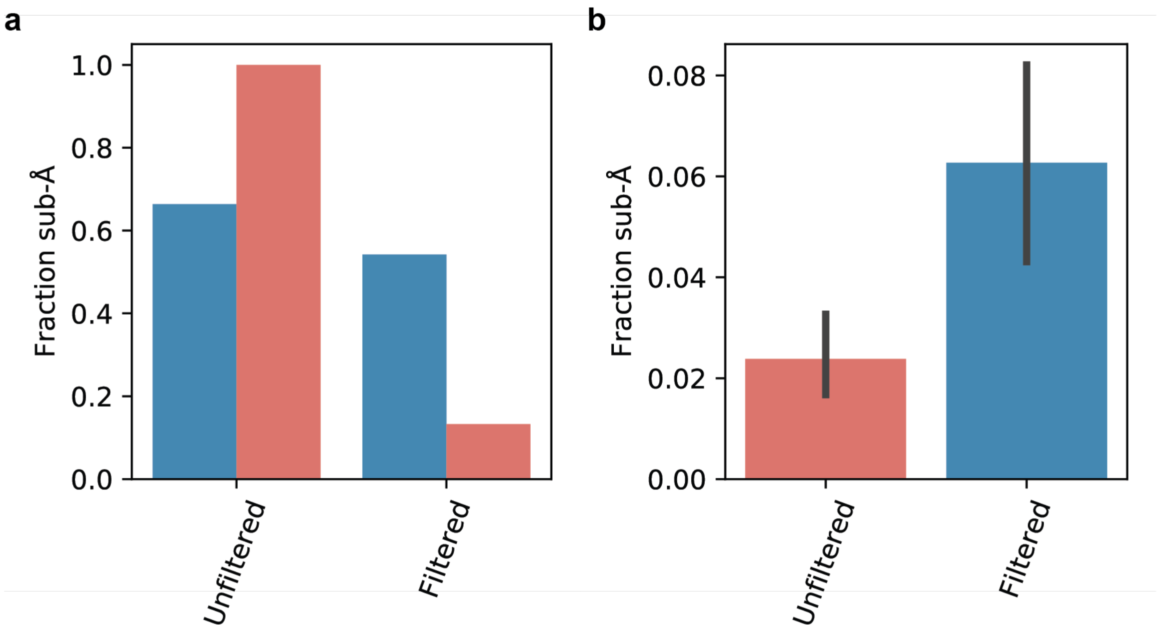


Figure 3.6. Impact of filtering on interface recapitulation. (a) Percentage of benchmark helices with a sub-Å PatchMAN fragment (blue) and the fraction of total docked helices (red) in the base dataset (left) or the filtered dataset (right). **(b)** Fraction of docked helices that are within 1 Å of a benchmark helix in the full dataset (red) or the filtered dataset (blue). Error bars represent the standard among the different benchmark targets.

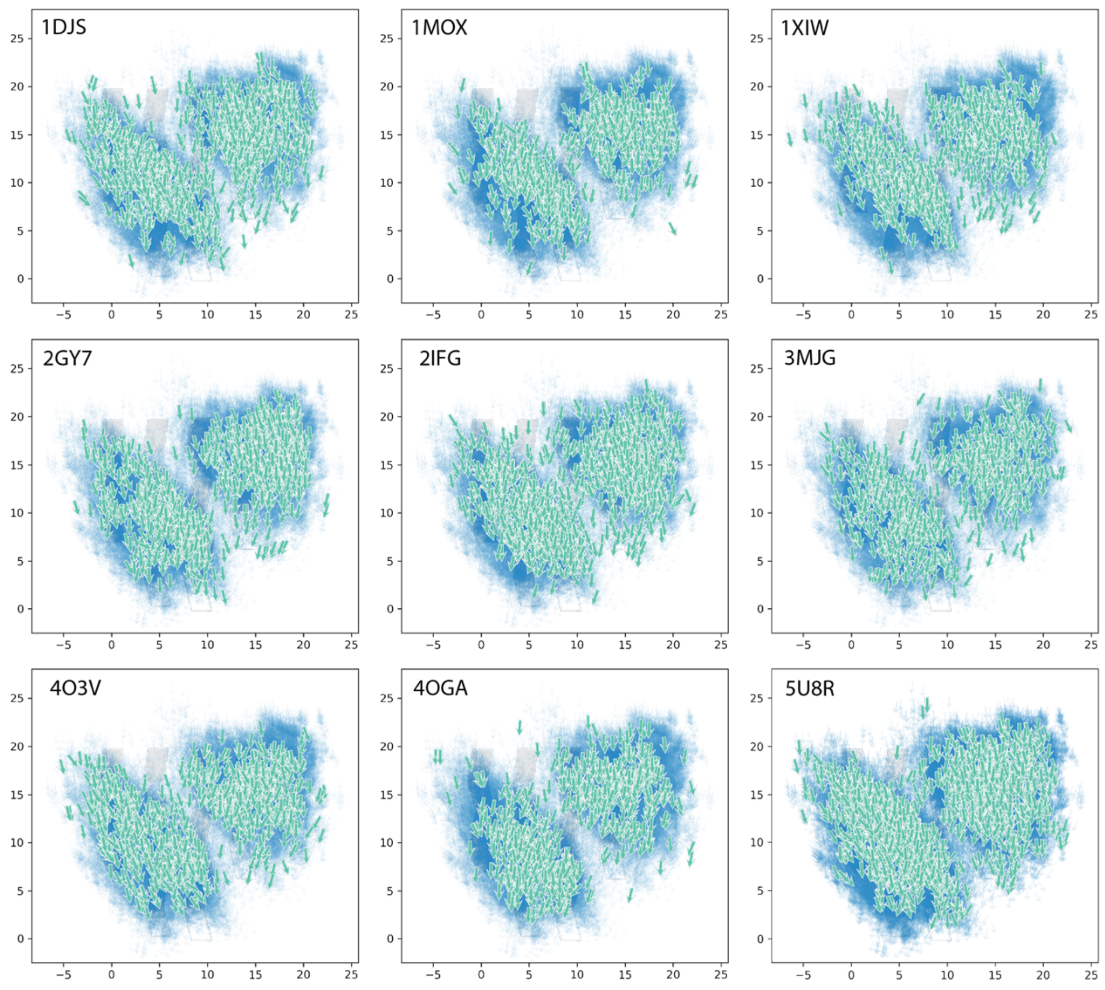


Figure 3.7. Matched scaffold geometry distributions. Helical geometry distributions for scaffolds that were matched and designed (teal) versus the entire scaffold library (blue). As described in **Fig. 3.4c**, but separated by target.

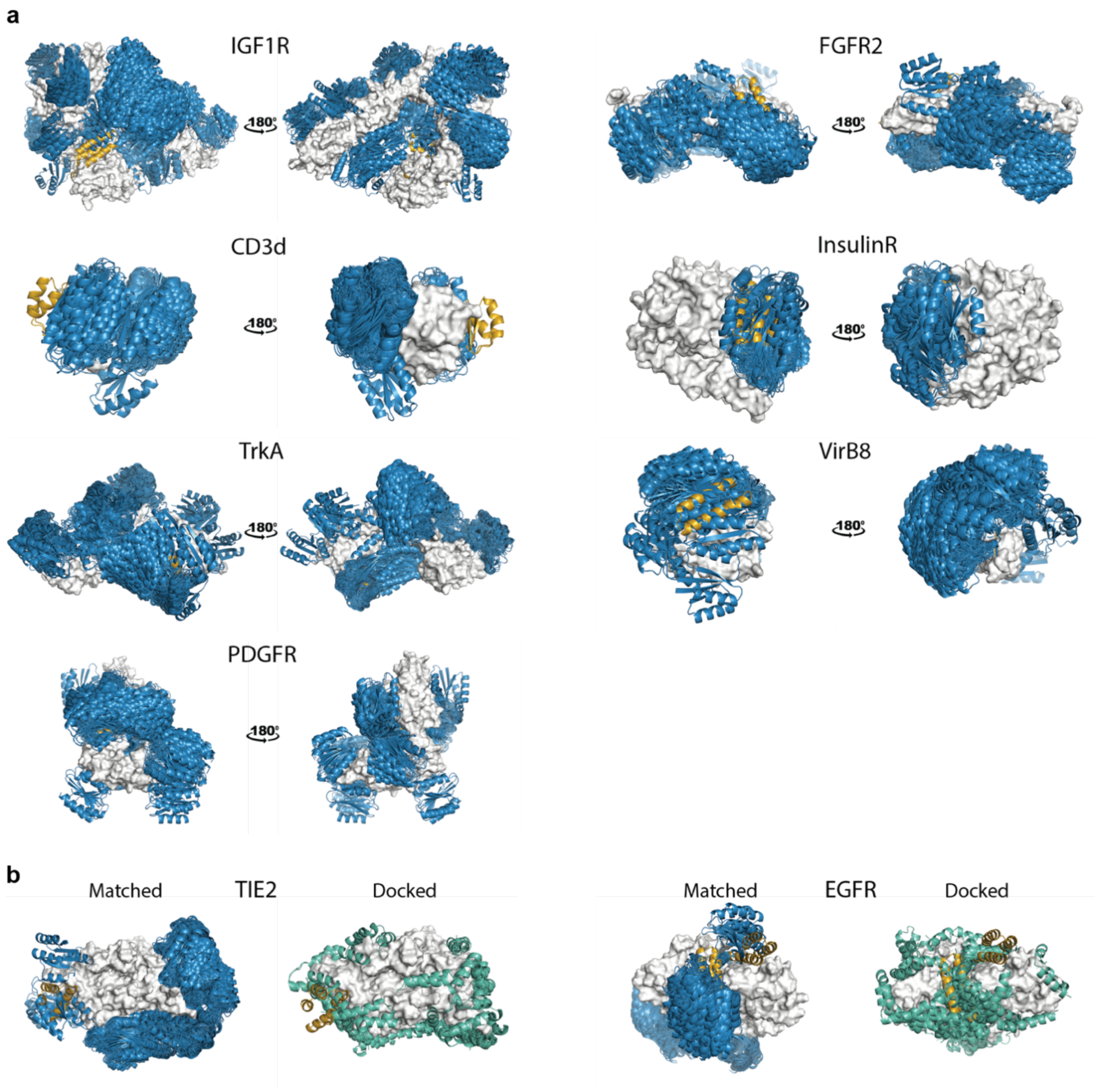


Figure 3.8. Target surface coverage. (a) Designed scaffolds (blue) and the optimized miniprotein binder (yellow) for 7 of the 9 targeted proteins (white). **(b)** Designed scaffolds (blue), optimized miniprotein binder (yellow), and filtered PatchMAN fragments (teal) for the two remaining targets (white) that had a lower average number of hydrogen bonds in the HELIX designs compared to the miniprotein binders.

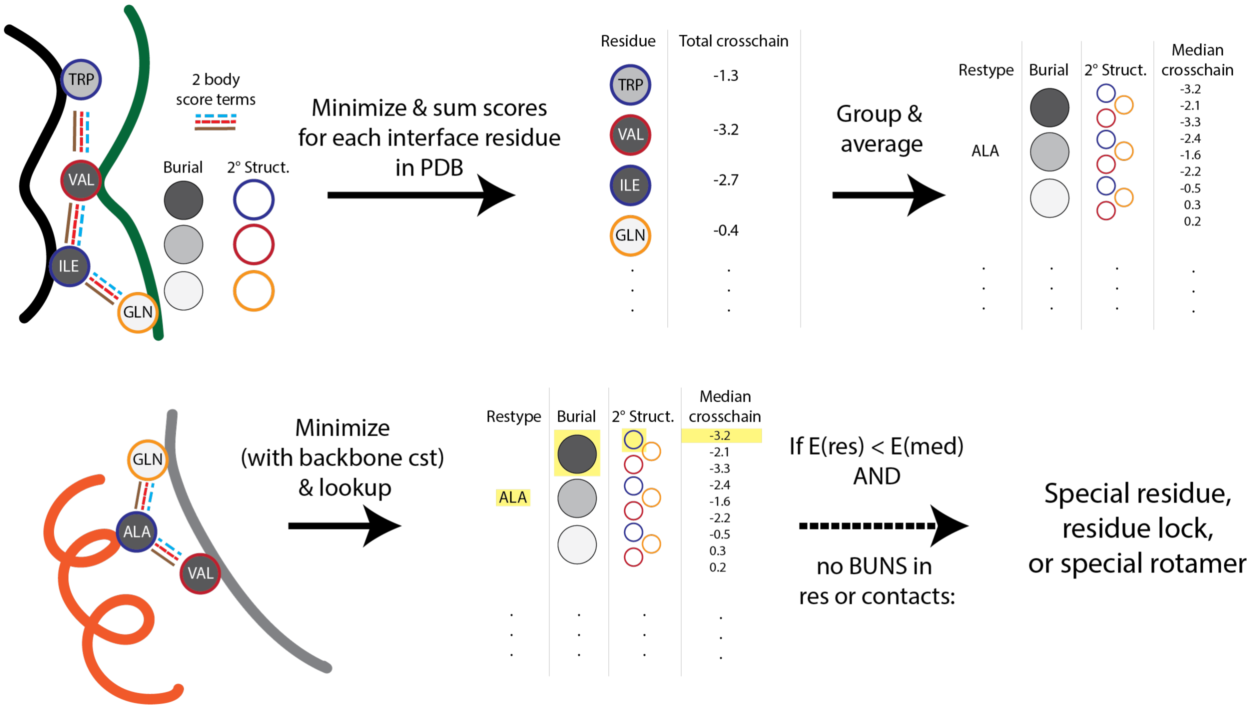


Figure 3.9. Workflow for determining "native-like" residues. Top: Minimized interface residues are scored, binned by secondary structure and burial, and then averaged. Bottom: Residues on PatchMAN fragments are minimized, then compared to residues with similar burial and secondary structure from the previously created table. Residues whose intermolecular (cross-chain) scores are better than the median are treated with one of the design methods that emphasize retention of native-like residues.

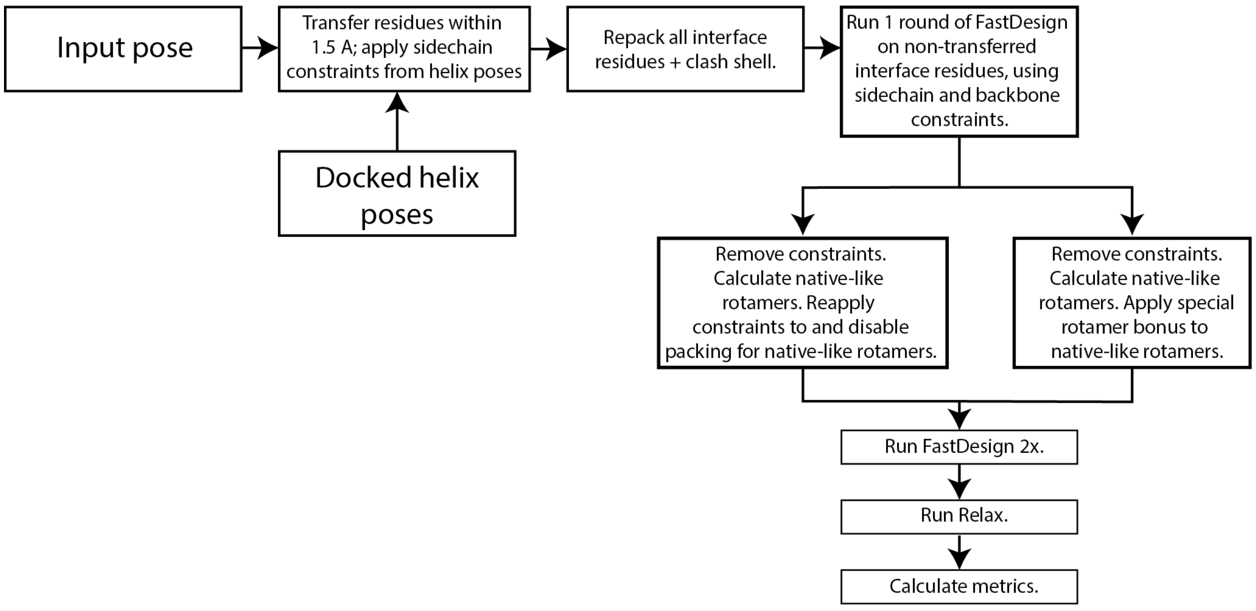

Input pose
Transfer residues within 1.5 A; apply sidechain constraints from helix poses
Docked helix poses
Repack all interface residues + clash shell.
Run 1 round of FastDesign on non-transferred interface residues, using sidechain and backbone constraints.
Remove constraints. Calculate native-like rotamers. Reapply constraints to and disable packing for native-like rotamers.
Remove constraints. Calculate native-like rotamers. Apply special rotamer bonus to native-like rotamers.
Run FastDesign 2x.
Run Relax.
Calculate metrics.

Figure 3.10. Flowchart of scaffold design protocol. Design protocol flowchart as described in **Methods**. Residues are transferred from the optimized fragment sequence to the matched scaffold. These are then repacked, and design is run on the remaining interface residues. Next, either the *residue lock* or the *special rotamer* protocol is applied as described in **Methods**, followed by two rounds of FastDesign with the InterfaceDesign2019 script and one round of FastRelax.

Table 3.1. Retention of scaffolds and PatchMAN fragments. *Patches:* Total number of patches generated. Depends heavily on protein surface area. *Total scaffolds:* Number of proteins in the scaffold library available for matching. *Scaffolds matched:* Number of scaffolds that matched at least one pair of PatchMAN fragments. *Scaffolds passed clash filter:* Number of scaffolds that had a clash score of less than 20. *Scaffolds designed:* Number of scaffolds that passed the clash and RMSD filters and were fed into the design step. *Total fragments:* Total number of outputs generated by PatchMAN. *Filtered fragments:* Number of PatchMAN fragments after filtering. *Fragments matched:* Number of fragments involved in a at least one match with a scaffold. *Fragments passed clash filter:* Number of fragments that were involved in at least one match with a scaffold that had a clash score of less than 20. *Fragments designed:* Number of fragments involved in a match that led to a designed protein.

Target	*Patches*	*Total scaffolds*	*Scaffolds matched*	*Scaffolds passed clash filter*	*Scaffolds designed*	*Total fragments*	*Filtered fragments*	*Fragments matched*	*Fragments passed clash filter*	*Fragments designed*
FGFR2	102	48,187	45,174	14,618	486	3,827	489	443	333	100
EGFR	247	48,187	38,132	10,062	387	5,393	420	328	225	74
CD3δ	45	48,187	25,286e	4,462	471	1,142	119	101	59	32
TIE2	211	48,187	34,977	4,761	453	3,782	229	174	122	54
TrkA	173	48,187	38,001	7,635	559	4,276	366	304	226	102
PDGFR	136	48,187	35,657	4,748	365	5,111	304	251	170	58
VirB8	68	48,187	39,424	30,135	509	7,117	1,094	919	869	126
Insulin R	142	48,187	18,129	1,384	321	2,798	114	52	27	16
IGF1R	400	48,187	39,421	26,247	835	14,900	1302	1,012	806	259

Table 3.2. Benchmark dataset. PDBID: PDB identifier for the complex. Helix chain: Chain of the interface helix being recapitulated. Target chain: Chain targeted by PatchMAN or RIFDock. Helix residues: Residue numbers of the target helix. Number hotspot residues: Number of residues in the helix with an intermolecular Rosetta score of less than -1.5. Helix interface score: Total intermolecular Rosetta score of the helix with the target chain.

PDBID	Helix chain	Target chain	Helix residues	Number hotspot residues
1d9c	B	A	['204', '205', '206', '207', '208', '209', '210', '211', '212', '213', '214']	4
1d9c	B	A	['242', '243', '244', '245', '246', '247', '248', '249', '250', '251', '252', '253', '254', '255', '256', '257', '258', '259']	10
1d9c	B	A	['303', '304', '305', '306', '307', '308', '309', '310', '311']	6
1d9c	B	A	['313', '314', '315', '316', '317', '318', '319']	4
1e7l	B	A	['59', '60', '61', '62', '63', '64', '65', '66', '67', '68', '69', '70', '71', '72', '73']	5
1e7l	B	A	['84', '85', '86', '87', '88', '89', '90', '91', '92', '93', '94', '95']	8
1e7l	B	A	['106', '107', '108', '109', '110', '111', '112', '113', '114', '115', '116', '117', '118']	4
1i36	A	B	['154', '155', '156', '157', '158', '159', '160', '161', '162', '163', '164', '165', '166', '167', '168', '169', '170', '171', '172', '173', '174', '175', '176', '177', '178', '179', '180', '181', '182']	13
1i36	A	B	['186', '187', '188', '189', '190', '191', '192', '193', '194']	4
1i36	A	B	['238', '239', '240', '241', '242', '243', '244', '245', '246', '247', '248', '249']	4
1kqp	A	B	['18', '19', '20', '21', '22', '23', '24', '25', '26', '27', '28', '29', '30', '31', '32', '33', '34', '35', '36']	4
1kqp	A	B	['108', '109', '110', '111', '112', '113', '114', '115', '116', '117', '118', '119', '120', '121', '122']	7
1kqp	A	B	['128', '129', '130', '131', '132', '133', '134', '135', '136', '137', '138', '139', '140', '141', '142', '143', '144', '145', '146', '147', '148', '149', '150']	6
1mty	C	E	['14', '15', '16', '17', '18', '19', '20', '21', '22', '23']	4
1mty	C	E	['106', '107', '108', '109', '110', '111', '112', '113', '114', '115', '116', '117', '118', '119', '120', '121', '122', '123', '124', '125', '126', '127', '128', '129']	8
1mty	C	E	['138', '139', '140', '141', '142', '143', '144', '145', '146', '147', '148', '149', '150', '151', '152', '153', '154', '155', '156', '157', '158', '159', '160', '161', '162', '163', '164', '165', '166', '167', '168', '169']	5
1mty	C	E	['173', '174', '175', '176', '177', '178', '179', '180', '181', '182', '183', '184', '185', '186', '187', '188', '189', '190', '191', '192', '193', '194', '195', '196', '197', '198', '199', '200', '201', '202', '203']	9
1nh2	D	B	['14', '15', '16', '17', '18', '19', '20', '21', '22', '23', '24', '25', '26', '27', '28']	6
1nh2	D	B	['34', '35', '36', '37', '38', '39', '40', '41', '42', '43', '44', '45', '46', '47', '48', '49', '50', '51', '52', '53', '54']	7
1p6x	A	B	['91', '92', '93', '94', '95', '96', '97', '98', '99', '100', '101', '102', '103', '104', '105', '106', '107', '108', '109', '110', '111', '112', '113', '114', '115', '116', '117']	10
1p6x	A	B	['160', '161', '162', '163', '164', '165', '166', '167', '168', '169']	4
1p6x	A	B	['285', '286', '287', '288', '289', '290', '291', '292', '293', '294', '295', '296', '297', '298']	4
1um0	A	D	['106', '107', '108', '109', '110', '111', '112', '113']	6
1um0	A	D	['229', '230', '231', '232', '233', '234', '235', '236', '237', '238']	4
1um0	A	D	['251', '252', '253', '254', '255', '256']	4
1yg6	G	A	['37', '38', '39', '40', '41', '42', '43', '44', '45', '46', '47', '48', '49', '50', '51', '52', '53']	6
1yg6	G	A	['70', '71', '72', '73', '74', '75', '76', '77', '78', '79', '80', '81', '82']	5
1yg6	G	A	['132', '133', '134', '135', '136', '137', '138', '139', '140', '141', '142', '143', '144', '145', '146', '147', '148', '149', '150', '151', '152', '153', '154', '155', '156', '157']	4

PDBID	Helix chain	Target chain	Helix residues	Number hotspot residues
1ykd	B	A	['61', '62', '63', '64', '65', '66', '67', '68', '69', '70', '71', '72', '73', '74', '75']	13
1ykd	B	A	['80', '81', '82', '83', '84', '85', '86', '87', '88', '89', '90', '91', '92', '93', '94', '95', '96', '97', '98', '99']	8
1ykd	B	A	['232', '233', '234', '235', '236', '237', '238', '239', '240', '241', '242', '243', '244', '245', '246', '247', '248', '249', '250', '251', '252', '253', '254', '255', '256', '257', '258', '259', '260', '261', '262', '263', '264', '265', '266']	14
1ykd	B	A	['414', '415', '416', '417', '418', '419', '420', '421', '422', '423', '424', '425', '426', '427', '428', '429', '430', '431', '432', '433', '434', '435', '436', '437', '438', '439', '440']	5
2dh4	B	A	['3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14']	5
2dh4	B	A	['101', '102', '103', '104', '105', '106', '107', '108', '109', '110', '111', '112', '113', '114', '115', '116', '117', '118', '119', '120', '121']	5
2dh4	B	A	['126', '127', '128', '129', '130', '131', '132', '133', '134', '135', '136', '137', '138', '139', '140', '141', '142', '143', '144', '145', '146', '147', '148', '149', '150', '151', '152', '153', '154', '155', '156']	7
2e52	A	B	['227', '228', '229', '230', '231', '232', '233', '234', '235', '236', '237', '238', '239', '240', '241', '242', '243', '244', '245', '246', '247', '248', '249', '250', '251', '252', '253', '254', '255', '256', '257', '258', '259', '260', '261']	10
2e52	A	B	['264', '265', '266', '267', '268', '269', '270', '271', '272', '273', '274']	6
2e52	A	B	['277', '278', '279', '280', '281', '282', '283', '284', '285', '286', '287', '288', '289', '290', '291', '292']	10
2f4m	B	A	['283', '284', '285', '286', '287', '288', '289', '290', '291', '292', '293', '294']	5
2f4m	B	A	['296', '297', '298', '299', '300', '301', '302', '303', '304', '305', '306', '307', '308', '309']	4
2ftx	B	A	['157', '158', '159', '160', '161', '162', '163', '164', '165', '166', '167', '168']	6
2ftx	B	A	['200', '201', '202', '203', '204', '205', '206', '207', '208', '209', '210', '211']	5
2hp3	B	A	['44', '45', '46', '47', '48', '49', '50', '51', '52', '53', '54', '55', '56']	5
2hp3	B	A	['172', '173', '174', '175', '176', '177', '178', '179', '180', '181', '182', '183', '184', '185']	4
2hp3	B	A	['197', '198', '199', '200', '201', '202', '203', '204', '205', '206', '207', '208', '209', '210', '211', '212', '213', '214', '215', '216', '217', '218']	6
2hzl	B	A	['120', '121', '122', '123', '124', '125', '126', '127', '128', '129']	4
2hzl	B	A	['259', '260', '261', '262', '263', '264', '265', '266', '267', '268', '269', '270', '271', '272', '273', '274', '275', '276', '277', '278', '279', '280', '281', '282', '283', '284', '285', '286', '287', '288', '289', '290', '291', '292', '293']	13
2hzl	B	A	['325', '326', '327', '328', '329', '330', '331', '332', '333', '334', '335', '336', '337', '338', '339', '340', '341', '342', '343', '344', '345', '346']	6
2hzl	B	A	['348', '349', '350', '351', '352', '353', '354', '355', '356', '357', '358', '359', '360', '361']	8
2j91	B	A	['168', '169', '170', '171', '172', '173', '174', '175', '176', '177', '178', '179', '180', '181', '182', '183', '184', '185', '186', '187', '188', '189', '190', '191', '192']	4
2j91	B	A	['246', '247', '248', '249', '250', '251', '252', '253', '254', '255', '256', '257', '258', '259', '260', '261', '262', '263', '264', '265', '266', '267', '268', '269', '270', '271', '272', '273', '274']	6
2j91	B	A	['299', '300', '301', '302', '303', '304', '305', '306', '307', '308', '309', '310', '311', '312', '313', '314', '315', '316', '317', '318', '319', '320', '321', '322', '323', '324']	4
2ov9	B	A	['26', '27', '28', '29', '30', '31', '32', '33', '34', '35', '36', '37', '38', '39', '40', '41', '42', '43']	8
2ov9	B	A	['49', '50', '51', '52', '53', '54', '55', '56', '57', '58', '59', '60', '61', '62', '63', '64', '65', '66', '67']	7
2ov9	B	A	['131', '132', '133', '134', '135', '136', '137', '138', '139', '140', '141', '142', '143', '144', '145', '146', '147', '148']	5
2pa8	D	L	['22', '23', '24', '25', '26', '27', '28', '29', '30', '31', '32', '33', '34']	4
2pa8	D	L	['238', '239', '240', '241', '242', '243', '244', '245', '246', '247', '248', '249', '250', '251', '252', '253', '254', '255', '256', '257', '258', '259', '260', '261', '262', '263']	7

PDBID	Helix chain	Target chain	Helix residues	Number hotspot residues
2q73	B	A	['16', '17', '18', '19', '20', '21', '22', '23', '24', '25', '26', '27', '28', '29', '30', '31', '32', '33', '34', '35', '36', '37', '38', '39']	4
2q73	B	A	['55', '56', '57', '58', '59', '60', '61', '62', '63', '64', '65', '66', '67', '68', '69', '70', '71', '72', '73']	9
2q73	B	A	['77', '78', '79', '80', '81', '82', '83', '84', '85', '86', '87', '88', '89', '90']	6
2qib	B	A	['147', '148', '149', '150', '151', '152', '153', '154', '155', '156', '157', '158', '159', '160', '161', '162', '163', '164', '165', '166', '167', '168']	8
2qib	B	A	['175', '176', '177', '178', '179', '180', '181', '182', '183', '184', '185', '186', '187', '188', '189', '190', '191', '192', '193', '194']	6
2qib	B	A	['216', '217', '218', '219', '220', '221', '222', '223', '224', '225', '226', '227']	6
2qtq	D	C	['110', '111', '112', '113', '114', '115', '116', '117', '118', '119', '120']	4
2qtq	D	C	['159', '160', '161', '162', '163', '164', '165', '166', '167', '168', '169', '170', '171', '172', '173', '174', '175']	6
2qtq	D	C	['177', '178', '179', '180', '181', '182', '183', '184']	5
2qtq	D	C	['191', '192', '193', '194', '195', '196', '197', '198', '199', '200', '201', '202', '203', '204', '205', '206', '207', '208', '209']	5
2ycl	A	B	['237', '238', '239', '240', '241', '242', '243', '244', '245', '246', '247', '248', '249', '250', '251', '252', '253']	6
2ycl	A	B	['272', '273', '274', '275', '276', '277', '278', '279', '280', '281', '282', '283', '284']	5
2ycl	A	B	['297', '298', '299', '300', '301', '302', '303', '304', '305', '306', '307', '308', '309', '310', '311']	8
2yf3	B	A	['10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20']	5
2yf3	B	A	['34', '35', '36', '37', '38', '39', '40', '41', '42', '43', '44', '45', '46', '47', '48', '49', '50', '51', '52', '53', '54', '55', '56', '57', '58', '59', '60', '61', '62', '63']	11
2yf3	B	A	['71', '72', '73', '74', '75', '76', '77', '78', '79', '80', '81', '82', '83', '84', '85', '86', '87', '88', '89', '90', '91', '92', '93']	13
2yf3	B	A	['98', '99', '100', '101', '102', '103', '104', '105', '106', '107', '108', '109', '110', '111', '112', '113', '114']	7
2yxh	B	A	['3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15']	5
2yxh	B	A	['24', '25', '26', '27', '28', '29', '30', '31', '32', '33', '34', '35', '36', '37', '38', '39', '40', '41', '42', '43']	4
2yxh	B	A	['47', '48', '49', '50', '51', '52', '53', '54', '55', '56', '57', '58', '59', '60', '61', '62', '63', '64', '65', '66', '67', '68', '69', '70', '71', '72']	12
2yxh	B	A	['76', '77', '78', '79', '80', '81', '82', '83', '84', '85', '86', '87', '88', '89', '90']	8
3bpj	C	A	['153', '154', '155', '156', '157', '158', '159', '160', '161', '162', '163', '164', '165', '166', '167', '168', '169', '170']	8
3bpj	C	A	['176', '177', '178', '179', '180', '181', '182', '183', '184', '185', '186', '187', '188']	7
3bpj	C	A	['193', '194', '195', '196', '197', '198', '199', '200', '201', '202', '203', '204', '205', '206', '207', '208', '209', '210', '211', '212', '213', '214', '215']	8
3dhi	B	A	['80', '81', '82', '83', '84', '85', '86', '87', '88', '89', '90', '91', '92', '93', '94', '95', '96', '97', '98', '99', '100', '101', '102', '103']	9
3dhi	B	A	['123', '124', '125', '126', '127', '128', '129', '130', '131', '132', '133', '134', '135', '136', '137', '138', '139', '140', '141', '142']	6
3dhi	B	A	['146', '147', '148', '149', '150', '151', '152', '153', '154', '155', '156', '157', '158', '159', '160', '161', '162', '163', '164', '165', '166', '167', '168', '169', '170', '171', '172', '173', '174', '175', '176']	10
3dhi	B	A	['194', '195', '196', '197', '198', '199', '200', '201', '202', '203', '204', '205', '206', '207', '208']	5
3g5o	A	B	['49', '50', '51', '52', '53', '54', '55', '56', '57', '58', '59', '60', '61', '62', '63']	5
3g5o	A	B	['67', '68', '69', '70', '71', '72', '73', '74', '75', '76', '77', '78', '79']	4
3g5o	A	B	['86', '87', '88', '89', '90', '91', '92', '93']	4

PDBID	Helix chain	Target chain	Helix residues	Number hotspot residues
3ilx	B	A	['131', '132', '133', '134', '135', '136', '137', '138', '139', '140', '141']	5
3ilx	B	A	['156', '157', '158', '159', '160', '161', '162', '163', '164', '165', '166', '167', '168', '169', '170', '171', '172', '173', '174', '175', '176', '177']	7
3ilx	B	A	['182', '183', '184', '185', '186', '187', '188', '189', '190', '191', '192', '193']	4
3kkb	B	A	['40', '41', '42', '43', '44', '45', '46', '47', '48', '49', '50', '51', '52', '53', '54', '55', '56', '57', '58', '59', '60', '61', '62', '63', '64', '65', '66', '67', '68', '69', '70', '71']	8
3kkb	B	A	['101', '102', '103', '104', '105', '106', '107', '108', '109', '110', '111', '112', '113', '114', '115', '116', '117', '118', '119']	4
3kkb	B	A	['132', '133', '134', '135', '136', '137', '138', '139', '140', '141', '142', '143', '144', '145', '146', '147', '148', '149', '150', '151', '152', '153', '154', '155', '156', '157', '158', '159', '160', '161', '162', '163', '164']	14
3kra	A	B	['106', '107', '108', '109', '110', '111', '112', '113', '114', '115', '116', '117', '118', '119', '120', '121', '122', '123', '124', '125', '126', '127']	9
3kra	A	B	['133', '134', '135', '136', '137', '138', '139', '140', '141', '142', '143', '144', '145', '146', '147']	5
3kra	A	B	['152', '153', '154', '155', '156', '157', '158', '159', '160']	6
4g92	B	C	['50', '51', '52', '53', '54', '55', '56', '57', '58', '59', '60']	5
4g92	B	C	['69', '70', '71', '72', '73', '74', '75', '76', '77', '78', '79', '80', '81', '82', '83', '84', '85', '86', '87', '88', '89', '90', '91', '92', '93', '94', '95', '96']	12
4g92	B	C	['50', '51', '52', '53', '54', '55', '56', '57', '58', '59', '60']	5
4g92	B	C	['69', '70', '71', '72', '73', '74', '75', '76', '77', '78', '79', '80', '81', '82', '83', '84', '85', '86', '87', '88', '89', '90', '91', '92', '93', '94', '95', '96']	12
4g92	B	C	['117', '118', '119', '120', '121', '122', '123', '124', '125', '126', '127', '128', '129', '130', '131']	8
4i0x	B	A	['11', '12', '13', '14', '15', '16', '17', '18', '19', '20', '21', '22', '23', '24', '25', '26', '27', '28', '29', '30', '31', '32', '33', '34', '35', '36', '37', '38', '39', '40', '41', '42', '43', '44', '45', '46', '47']	14
4i0x	B	A	['53', '54', '55', '56', '57', '58', '59', '60', '61', '62', '63', '64', '65', '66', '67', '68', '69', '70', '71', '72', '73', '74', '75', '76', '77', '78', '79', '80', '81', '82', '83', '84', '85', '86']	11
4lfg	B	A	['101', '102', '103', '104', '105', '106', '107', '108', '109', '110', '111', '112', '113', '114', '115', '116', '117', '118', '119', '120', '121']	9
4lfg	B	A	['126', '127', '128', '129', '130', '131', '132', '133', '134', '135', '136', '137', '138', '139', '140']	6
4lfg	B	A	['145', '146', '147', '148', '149', '150', '151', '152', '153', '154']	5

3.7 References

[1] C. M. Bryan *et al.*, “Computational design of a synthetic PD-1 agonist,” *Proc. Natl. Acad. Sci.*, vol. 118, no. 29, p. e2102164118, Jul. 2021, doi: 10.1073/pnas.2102164118.

[2] L. Cao *et al.*, “De novo design of picomolar SARS-CoV-2 miniprotein inhibitors,” *Science*, vol. 370, no. 6515, pp. 426–431, Oct. 2020, doi: 10.1126/science.abd9909.

[3] A. Glasgow *et al.*, “Engineered ACE2 receptor traps potently neutralize SARS-CoV-2,” *Proc. Natl. Acad. Sci.*, vol. 117, no. 45, pp. 28046–28055, Nov. 2020, doi: 10.1073/pnas.2016093117.

[4] A. A. Glasgow *et al.*, “Computational design of a modular protein sense-response system,” *Science*, vol. 366, no. 6468, pp. 1024–1028, Nov. 2019, doi: 10.1126/science.aax8780.

[5] R. A. Langan *et al.*, “De novo design of bioactive protein switches,” *Nature*, vol. 572, no. 7768, Art. no. 7768, Aug. 2019, doi: 10.1038/s41586-019-1432-8.

[6] A. H. Ng *et al.*, “Modular and tunable biological feedback control using a de novo protein switch,” *Nature*, vol. 572, no. 7768, pp. 265–269, Aug. 2019, doi: 10.1038/s41586-019-1425-7.

[7] C. Yang *et al.*, “Bottom-up de novo design of functional proteins with complex structural features,” *Nat. Chem. Biol.*, vol. 17, no. 4, pp. 492–500, Apr. 2021, doi: 10.1038/s41589-020-00699-x.

[8] C. M. Bryan *et al.*, “Computational design of a synthetic PD-1 agonist,” *Proc. Natl. Acad. Sci.*, vol. 118, no. 29, p. e2102164118, Jul. 2021, doi: 10.1073/pnas.2102164118.

[9] J. Dou *et al.*, “De novo design of a fluorescence-activating β-barrel,” *Nature*, vol. 561, no. 7724, Art. no. 7724, Sep. 2018, doi: 10.1038/s41586-018-0509-0.

[10] L. Cao *et al.*, “Design of protein-binding proteins from the target structure alone,” *Nature*, vol. 605, no. 7910, pp. 1–10, May 2022, doi: 10.1038/s41586-022-04654-9.

[11] X. Pan *et al.*, “Expanding the space of protein geometries by computational design of de novo fold families,” *Science*, vol. 369, no. 6507, pp. 1132–1136, Aug. 2020, doi: 10.1126/science.abc0881.

[12] A. Khramushin, Z. Ben-Aharon, T. Tsaban, J. K. Varga, O. Avraham, and O. Schueler-Furman, “Matching protein surface structural patches for high-resolution blind peptide docking,” *Proc. Natl. Acad. Sci.*, vol. 119, no. 18, p. e2121153119, May 2022, doi: 10.1073/pnas.2121153119.

[13] F. Zheng, J. Zhang, and G. Grigoryan, “Tertiary structural propensities reveal fundamental sequence/structure relationships,” *Struct. Lond. Engl. 1993*, vol. 23, no. 5, pp. 961–971, May 2015, doi: 10.1016/j.str.2015.03.015.

[14] C. O. Mackenzie, J. Zhou, and G. Grigoryan, “Tertiary alphabet for the observable protein structural universe,” *Proc. Natl. Acad. Sci.*, vol. 113, no. 47, pp. E7438–E7447, Nov. 2016, doi: 10.1073/pnas.1607178113.

[15] J. Zhou and G. Grigoryan, “Rapid search for tertiary fragments reveals protein sequence–structure relationships,” *Protein Sci. Publ. Protein Soc.*, vol. 24, no. 4, pp. 508–524, Apr. 2015, doi: 10.1002/pro.2610.

[16] S. B. Thyme, D. Baker, and P. Bradley, “Improved Modeling of Side-Chain–Base Interactions and Plasticity in Protein–DNA Interface Design,” *J. Mol. Biol.*, vol. 419, no. 3, pp. 255–274, Jun. 2012, doi: 10.1016/j.jmb.2012.03.005.

[17] B. Coventry and D. Baker, “Protein sequence optimization with a pairwise decomposable penalty for buried unsatisfied hydrogen bonds,” *PLoS Comput. Biol.*, vol. 17, no. 3, p. e1008061, Mar. 2021, doi: 10.1371/journal.pcbi.1008061.

[18] P. J. Fleming and G. D. Rose, “Do all backbone polar groups in proteins form hydrogen bonds?,” *Protein Sci.*, vol. 14, no. 7, pp. 1911–1917, 2005, doi: 10.1110/ps.051454805.

[19] J. M. Sanders, M. E. Wampole, M. L. Thakur, and E. Wickstrom, “Molecular Determinants of Epidermal Growth Factor Binding: A Molecular Dynamics Study,” *PLOS ONE*, vol. 8, no. 1, p. e54136, Jan. 2013, doi: 10.1371/journal.pone.0054136.

[20] W. A. Barton *et al.*, “Crystal structures of the Tie2 receptor ectodomain and the angiopoietin-2–Tie2 complex,” *Nat. Struct. Mol. Biol.*, vol. 13, no. 6, Art. no. 6, Jun. 2006, doi: 10.1038/nsmb1101.

[21] J. E. Lucas and T. Kortemme, “New computational protein design methods for de novo small molecule binding sites,” *PLOS Comput. Biol.*, vol. 16, no. 10, p. e1008178, Oct. 2020, doi: 10.1371/journal.pcbi.1008178.

[22] N. F. Polizzi and W. F. DeGrado, “A defined structural unit enables de novo design of small-molecule–binding proteins,” *Science*, vol. 369, no. 6508, pp. 1227–1233, Sep. 2020, doi: 10.1126/science.abb8330.

[23] M. Gao and J. Skolnick, “Structural space of protein–protein interfaces is degenerate, close to complete, and highly connected,” *Proc. Natl. Acad. Sci.*, vol. 107, no. 52, pp. 22517–22522, Dec. 2010, doi: 10.1073/pnas.1012820107.

[24] R. B. Russell, P. D. Sasieni, and M. J. E. Sternberg, “Supersites within superfolds. Binding site similarity in the absence of homology11Edited by J. Thornton,” *J. Mol. Biol.*, vol. 282, no. 4, pp. 903–918, Oct. 1998, doi: 10.1006/jmbi.1998.2043.

[25] C. J. Tsai and R. Nussinov, “Hydrophobic folding units at protein-protein interfaces: implications to protein folding and to protein-protein association.,” *Protein Sci. Publ. Protein Soc.*, vol. 6, no. 7, pp. 1426–1437, Jul. 1997.

[26] A. Zanghellini *et al.*, “New algorithms and an in silico benchmark for computational enzyme design,” *Protein Sci.*, vol. 15, no. 12, pp. 2785–2794, Dec. 2006, doi: 10.1110/ps.062353106.

[27] X. Pan and T. Kortemme, “De novo protein fold families expand the designable ligand binding site space,” *PLOS Comput. Biol.*, vol. 17, no. 11, p. e1009620, Nov. 2021, doi: 10.1371/journal.pcbi.1009620.

[28] M. Baek *et al.*, “Accurate prediction of protein structures and interactions using a 3-track neural network,” *Science*, vol. 373, no. 6557, pp. 871–876, Aug. 2021, doi: 10.1126/science.abj8754.

[29] J. Jumper *et al.*, “Highly accurate protein structure prediction with AlphaFold,” *Nature*, vol. 596, no. 7873, Art. no. 7873, Aug. 2021, doi: 10.1038/s41586-021-03819-2.

[30] I. Anishchenko *et al.*, “De novo protein design by deep network hallucination,” *Nature*, vol. 600, no. 7889, Art. no. 7889, Dec. 2021, doi: 10.1038/s41586-021-04184-w.

[31] C. Norn *et al.*, “Protein sequence design by conformational landscape optimization,” *Proc. Natl. Acad. Sci.*, vol. 118, no. 11, p. e2017228118, Mar. 2021, doi: 10.1073/pnas.2017228118.

[32] M. Jendrusch, J. O. Korbel, and S. K. Sadiq, “AlphaDesign: A de novo protein design framework based on AlphaFold,” Oct. 2021. doi: 10.1101/2021.10.11.463937.

[33] W. Kabsch and C. Sander, “Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features,” *Biopolymers*, vol. 22, no. 12, pp. 2577–2637, 1983, doi: 10.1002/bip.360221211.

[34] H. Park *et al.*, “Simultaneous Optimization of Biomolecular Energy Functions on Features from Small Molecules and Macromolecules,” *J. Chem. Theory Comput.*, vol. 12, no. 12, pp. 6201–6212, Dec. 2016, doi: 10.1021/acs.jctc.6b00819.

[35] N. Koga *et al.*, “Principles for designing ideal protein structures,” *Nature*, vol. 491, no. 7423, Art. no. 7423, Nov. 2012, doi: 10.1038/nature11600.

[36] W. Sheffler and D. Baker, “RosettaHoles2: A volumetric packing measure for protein structure refinement and validation,” *Protein Sci. Publ. Protein Soc.*, vol. 19, no. 10, pp. 1991–1995, Oct. 2010, doi: 10.1002/pro.458.

[37] E. Marcos *et al.*, “Principles for designing proteins with cavities formed by curved β sheets,” *Science*, vol. 355, no. 6321, pp. 201–206, Jan. 2017, doi: 10.1126/science.aah7389.

[38] G. J. Rocklin *et al.*, “Global analysis of protein folding using massively parallel design, synthesis, and testing,” *Science*, vol. 357, pp. 168–175, 2017.

[39] J. Liang and K. A. Dill, “Are Proteins Well-Packed?,” *Biophys. J.*, vol. 81, no. 2, pp. 751–766, Aug. 2001, doi: 10.1016/S0006-3495(01)75739-6.

[40] H. Edelsbrunner and E. P. Mück, “Three-Dimensional Alpha Shapes,” *ACM Trans. Graph.*, vol. 13, no. 1, 1994.

[41] J. B. Maguire *et al.*, “Perturbing the energy landscape for improved packing during computational protein design,” *Proteins*, vol. 89, no. 4, pp. 436–449, Apr. 2021, doi: 10.1002/prot.26030.

Chapter 4

Discussion and future directions

Here, I have presented two new methods for designing functional proteins, both of which place an emphasis on enabling sidechain functional interactions by generating suitable backbone geometries. Where these methods differ is in terms of the number of interactions that can be accommodated versus the relative irregularity of the backbones generated. We show that PIP is capable of placing a single functional residue at a time, and while it may be possible to place several, it is unlikely to be able to accommodate many such residues; however, it does so without being constrained by any sort of secondary structure requirements. This approach is best paired with situations where a few residues have a disproportionate contribution to the designed function, such as binding of a small ligand or catalysis. As a result, improvements would be most impactful if they addressed scalability: for instance, one could create an algorithm that uses residue-residue or tertiary motifs to determine whether a different local backbone geometry could accommodate a more favorable functional interaction, then generate that geometry, followed by some optimization of the local environment. Currently, such a method would be limited by the time it takes to sample both new backbone conformations and sequences, but with the rise of machine learning in protein design, it might be possible to quickly generate designable backbones [1], or even generate local backbones and surrounding sequences simultaneously. The latter could be achieved by masking both the structure and sequence of a local protein environment and using a deep neural net to fill in both the geometry and sequence.

The strength of HELIX, on the other hand, is in facilitating many functional interactions at once, but it is constrained to doing so mainly via α-helices (though other geometries from the scaffold can of course interact with the target protein). Thus, enhancements should focus on making HELIX compatible with other types of secondary structure. It may be possible to define β-sheets in a way that facilitates fast matching, as HELIX has accomplished with α-helices, but the more generalizable approach is to combine the placement of functional tertiary motifs with constrained hallucination [2]–[4]. Additionally, while HELIX was designed as a means of one-sided interface design, the information gained from using PatchMAN [5] to place secondary structural units lends itself quite well to the design of orthogonal binding pairs. This could be accomplished by modifying the sequence of the target protein to match the sequence of the tertiary motif found by MASTER [6], causing both sides of the interface to have known compatibility in terms of both their sequences and geometries.

While on-demand design of protein function remains a distant aspiration, the methods herein represent a modest waypoint on the long road towards that goal. It is my heartfelt hope as I depart UCSF that they provide some utility and inspiration to the scientific community.

4.1 References

[1] J. Sun and B. Wu, “Protein design with a machine-learned potential about backbone designability,” *Trends Biochem. Sci.*, Apr. 2022, doi: 10.1016/j.tibs.2022.04.004.

[2] I. Anishchenko *et al.*, “De novo protein design by deep network hallucination,” *Nature*, vol. 600, no. 7889, Art. no. 7889, Dec. 2021, doi: 10.1038/s41586-021-04184-w.

[3] M. Jendrusch, J. O. Korbel, and S. K. Sadiq, “AlphaDesign: A de novo protein design framework based on AlphaFold,” Oct. 2021. doi: 10.1101/2021.10.11.463937.

[4] J. Wang *et al.*, “Deep learning methods for designing proteins scaffolding functional sites,” Nov. 2021. doi: 10.1101/2021.11.10.468128.

[5] A. Khramushin, T. Tsaban, J. Varga, O. Avraham, and O. Schueler-Furman, “PatchMAN docking: Modeling peptide-protein interactions in the context of the receptor surface.” bioRxiv, p. 2021.09.02.458699, Sep. 03, 2021. doi: 10.1101/2021.09.02.458699.

[6] J. Zhou and G. Grigoryan, “Rapid search for tertiary fragments reveals protein sequence–structure relationships,” *Protein Sci. Publ. Protein Soc.*, vol. 24, no. 4, pp. 508–524, Apr. 2015, doi: 10.1002/pro.2610.