Learning How to Make Mini-Proteins that Bind to Specific Target Proteins

Brian Coventry

A dissertation

submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

University of Washington

2021

Reading Committee:

David Baker, Chair

Ning Zheng

Frank Dimaio

Phil Bradley

Program authorized to Offer Degree:

Molecular Engineering and Sciences

University of Washington

Abstract

Learning How to Make Mini-Proteins That Bind to Specific Target Proteins

Brian Coventry

Chair of the Supervisory Committee:
David Baker
Department of Biochemistry

Antibodies seem to have no trouble targeting and binding to specific natural proteins. But how do they do it? Human attempts to copy this ability often directly copy antibodies; either by borrowing their loops or their hotspots. While humans can analyze interactions between antibodies and their targets or interactions between native protein complexes, the proof of understanding requires one to make and test predictions. In this work, we set out to learn the principles that allow for specific protein binding. Leveraging the modern biochemist's toolkit, we tested more than one million *de novo* mini-protein binders (< 65aa) to more than twenty natural targets. With the goal of learning, not copying, we threw out the native complex information and discovered our own hotspots and interactions. Some of the target sites did not even have natural partners. By iterating on design-test-learn, we were able to uncover secrets about protein folding and protein binding. We found that the hydrophobic effect and hydrophobic packing dominate the correlations. While future work may find ways to bind to polar patches, for now, we have a sure-fire way to make binders to hydrophobic patches.

Table of contents:

Ctrl F: 8.2.1

List of Figures:

Ctrl F: Fig 8.4.1.A

1. Background:

Proteins are molecules with distinct 3-dimensional shapes that form the foundation for life. A structure-function relationship exists where the specific shape of a protein determines its function. The specific shape is encoded by the amino acid sequence and for the most part, proteins always fold into the same 3-dimensional shape[1].

The ability to specifically pick out a given protein from a larger pool of proteins is an important ability for any entity interested in modifying biology. Viruses use this methodology to bind to specific cellular receptors that trigger cells to engulf the virus[2]. Meanwhile, the immune system uses antibodies to specifically identify virus proteins[3]. Modern antibody-based vaccines use this same idea to target human proteins to achieve therapeutic effect[4].

In all of these cases, the critical piece of the puzzle is that a protein exists that specifically sticks to another protein. It is not enough to produce a universal binding protein that sticks to everything. In all cases above, this would lead to so many non-specific interactions that the protein would never be able to accomplish its intended goal. Instead, what is needed is a protein that binds specifically to another protein.

The question of "how to make a protein that sticks specifically to another protein" has answers that range from hard to very hard depending on how much control one wants. If one simply wants a protein sticks to any part of the target protein, humans have convinced animal immune systems to do this for us[5]. If, however, one wants to design the interaction on the computer in advance and produce a protein that binds exactly, there is a great challenge.

Historically, the way to make specific binding proteins was to borrow native interactions. Only a small portion of a native protein-complex is in contact, and by taking these interactions and putting them on a new "scaffold protein", one may make a new protein sequence that binds exactly correctly[6,7,8]. The other method is to borrow smaller protein interactions like single amino acids hotspots and graft them onto new scaffolds[9]. This works, but does not achieve the same level of success.

2. Motivation:

While designing binders by borrowing interactions from nature is likely to lead to high success rates, it has a fundamental limitation in that there must be an interaction to borrow from. Moreover, although a native interaction may be known, sometimes it is impossible to incorporate these interactions into a different scaffold; difficult-to-stabilize loops and motifs that incorporate several adjacent, but disconnected elements, preclude many grafting projects. It should in-principle always be possible to steal individual amino acids from an interface, but it is often difficult to find a scaffold suitable to hold all of the interactions. Additionally, when collecting

these hotspots, one might be tempted to modify the hotspots slightly, but doing so is difficult without an example in nature.

If instead, we set out to understand the fundamental principles that make interfaces work, these challenges all disappear. We no longer must question if a modified hotspot will work, because we know the answer. Grafting complicated secondary structures isn't necessary, because we bring our own and discover our own interactions. Finally, a lack of known interactions does not slow us down at all; all we need to know is the structure of the target protein.

The totally *de novo* method just described has yet another benefit that cannot be captured by a grafting method; we understand a bit more about the physics with each iteration. Since we start each project with no native information, each interaction we produce is a sort of hypothesis. When we get the results back, we can see what worked, what didn't, and try to determine why. The native grafting method doesn't uncover principles like this. Perhaps we could learn more about what interactions are possible to graft, and better strategies to graft them. But the knowledge of hotspot modification will come very slowly.

With the goal of learning what makes interfaces work, we set out to perform round after round of design-test-learn cycles with mini-protein binders. By incorporating lessons from the previous round into the next, we can test hypothesis and increase our success rate. The goal at the end would be to order a single protein sequence that binds to the target.

3. Challenges:

While the goal of producing mini-protein binders to natural targets sounds straightforward, there are several challenges that stand in the way:

3.1. Scaffold quality / protein folding:

Although advances in protein folding and protein design have shown that we can predict protein structures and design protein structures, the success rate of these methods is dependent on one's definition of "success". If we limit "success" to be getting the fold-topology of a protein correct, then both protein folding and protein design have very high success rates. However, if we limit "success" to be structures accurate to 0.5Å atomic placement, the success rate falls dramatically[10]. (The radius of a carbon atom is about 2Å.)

The resolution of our designed structures is incredibly important because as indicated in *Fig 8.4.1.A*, the required resolution for our binders may be as low as 0.3Å. With this in mind, it is a great challenge to ensure that our computational model matches the solution structure of our binder exactly. It doesn't matter how good our designed interface is if the binder does not fold into a shape that can realize it.

3.2. Computational metric cutoffs

With enough time and effort, computational design protocols can be produced to optimize just about any metric one wishes. At the start of this project, the popular metrics were: Rosetta ddG, Shape Complementarity, Interface Buried SASA, and Buried Unsatisfied Polar Atoms. As time went on, flaws in these metrics began to emerge and by the end, it was clear that with the exception of Rosetta ddG, these are not the right metrics to optimize on.

The challenge of this project was a bootstrapping problem; where identification of the right metrics required designing successful binders, but designing successful binders required finding the right metrics. Pair this with the fact that the best metrics we have now did not exist at the outset, identifying the best metrics was very challenging. It is entirely likely that we still do not have the best set of metrics, and only with more iterations and more clever observations will we find the truth.

3.3. Experimental noise

A further complication to identifying the best metrics is the high degree of experimental noise in these procedures. Testing 100,000 proteins at once is a blessing and a curse. On one hand, you get lots of data, on the other hand, your data has a lot of noise. The primary piece of noise that plagues this project is false positive binders: protein sequences that appear to bind, but do so for the wrong reason. There are three primary ways by which this can happen: 1) A pure experimental artifact whereby their DNA “hitch-hiked” inside a yeast-cell containing a strong binder. 2) A poorly behaved design that folded into an alternate conformation that bound to the target. 3) A correctly folding design that bound to the target in an alternate conformation. All of these sources of noise lead to erroneous examples of binding where any metric computed on the computational model does not correspond with what happened in reality. See *8.3.2. SSM validation rates* for more information.

3.4. Computational method

In the design of mini-protein binders, knowing which metrics to optimize is only half the battle, you must also actually make mini-protein binders that have these features. There are an infinite number of ways to design a binder, and choosing the best one can be very different. Worse is the situation where all available options are not good enough. The latter situation was the case for this project and nearly every step of the existing design approach needed to be revamped in order to hit our metrics. Just like the metrics however, it is almost certainly the case that a better method exists than what we have produced here.

4. A brief overview of the technologies behind this work:

Several techniques within the modern biochemist's toolkit are so powerful that they defy one's expectation. This work uses several of these pieces of technology which allows enormous amounts of data but causes several key considerations.

Chip synthesized oligo-nucleotide arrays allow hundreds of thousands of protein sequences to be synthesized at once[11]. This allowed this work to test millions of proteins across the more than twenty individual experiments. However, at the time of writing, the maximum length nucleotide that can be synthesized in this format is 230 base pairs. After adding the necessary adaptors, this results in proteins that are 65 amino acids in length. That limited this study to proteins of 65 amino acids which we call "mini-proteins".

Fluorescence Activated Cell Sorters (FACS) machines exist that can sort 5,000 yeast cells per second. Running this machine for an hour results in nearly 20 million sorted cells. This is the technology used to make decisions in the wet-lab. Each of the different yeast cells may contain a different DNA strand, and using this technique, we can sort binders from non-binders.

Next Generation Sequencing (NGS) is the final piece of the puzzle that makes this work. A technology that came from the Human Genome Project, NGS machines can sequence around 1 billion DNA sequences in 24 hours. This is the final step in the process where we can identify which yeast cells got sorted.

Without these tools, this project would not have been possible. We would have been testing around one hundred designs at once, and there's no way we could have produced correlations like the ones here with such low throughput.

However, the higher scale does come with a tradeoff, data quality. Analysis at this scale requires looking through extreme noise.

The original idea of using the techniques to screen thousands of proteins was originally devised by Gabe Rocklin. The idea then was to use protease treatment to determine the principles that allowed proteins to fold[12]. This protease assay proved to be useful in the early stages of this project. Now we rely primarily on the technology which allowed tens of thousands of designs to be tested at once.

5. Primary lessons and takeaways

This dissertation seeks to cover every last detail of what was tried and learned. If you wish to design and research mini-protein binders of your own, it is recommended to read the entire thing. If, however, you only want to learn the major points, this section is for you.

5.1. Totally *de novo* protein binder design is possible:

A first-principles approach using the Rosetta forcefield[13] along with simple docking and Rosetta Design[14] is enough to make protein binders, assuming one has the right filters.

5.2. Larger and smarter sampling provides better results:

An unfortunate but true reality of this project is that spending more CPU time making binders invariably leads to better binders. Whether you choose to look at 5x more docks, or perform design with 5x more replicates, your top binders will be better. Scaling up by a factor of 1000 is better than scaling up by a factor of 100.

Perhaps a glimmer of hope here is that smarter sampling can shortcut the large sampling problem. The promising candidates can usually be identified quickly before intensive CPU time is applied. In this way, one may sample very large and perform a quick calculation to find the best. This method provides nearly the same level of success as moving forward with all members. (See *6.5.1. Rosetta Design pre-filtering.*)

5.3. Hydrophobicity and packing are the critical metrics:

Out of all the metrics that we looked at, hydrophobicity and packing remain the metrics with the most predictive power. Specifically, what we found is that maximizing the buried hydrophobicity of the target with tight packing is what leads to success. On the binder side, there is a careful balance between too little and too much hydrophobicity. The Sap Score[developability] was determined to be the most accurate measure of hydrophobicity. (See *8.2.1. Hydrophobicity metrics*, *8.2.2. Packing measures*, and *6.1. SAP score*.)

For packing, although it is tempting to use Shape Complementarity[16], we found that this method is too easily fooled by small interfaces. Instead, a method called Contact Molecular Surface was created based on Shape Complementarity that instead calculates the distance-weighted surface area of contact between the two proteins. By selecting a few specific hydrophobic residues on the target, and maximizing the Contact Molecular Surface to them, you will arrive at proteins that pack very tightly to your buried hydrophobic residues. (See *6.2. Contact Molecular Surface*).

5.4. Scaffold quality is paramount

If your scaffold does not fold correctly, it does not matter how good your designed interface is. While there are trends dictating how to make scaffolds of a particular fold more accurate, the larger trend is between scaffold folds. In our experiments, we looked at 3-helical bundles, 4-helical bundles, ferredoxins, and proteins that look like protein G. We found that the more helical character and the fewer the number of loops, the higher the success rate. With this in mind, the

3-helical bundles were our shining stars and had a 3x success rate over the 4-helical bundles. The protein G type scaffolds with only 1 helix virtually never worked, and the ferredoxins had mixed success. If one can get away with using only 3-helical bundles, the success rate will be maximized. (See *8.2.6. Other metrics* and *8.5. Scaffold topology correlations*.)

6. Protocol optimization:

This section is devoted to explaining all of the ideas we tried with regards to the final protocol that we use. Nearly every design choice was benchmarked, and hopefully the following sections will prevent future researches from following avenues that have already been explored.

6.1. SAP score:

The Developability Index[15] is a metric used to identify the aggregation propensity of an antibody. The idea being that antibodies with a high value should not be pursued because they will not be shelf stable. It is calculated by calculating a SAP score and then adding to it 0.05 times the net charge squared. While this metric is useful, it turns out that the SAP score itself is potentially even more useful. This specific implementation seems to capture the hydrophobicity of proteins exceptionally well.

The SAP score is a 3D representation of the hydrophobicity of a protein, and taking the sum results in the overall SAP score. The method is SASA based and relies on the hydrophobicities calculated for each amino acid here[17]. The method first starts by calculating the SASA of each atom and applying the following formula to get the score of each atom:

Atom_score = residue_hydrophobic * atom_sasa / max_residue_sasa

where the max_residue_sasa is a precomputed value for each amino acid type.

To calculate the actual SAP score, each atom looks at all atoms within 5Å and sums their scores. If the resulting value is less than 0 it is set to 0. This number is the atom_sap_score, and summing it either by residue or by structure results in the overall SAP score.

The utility of the SAP score as an aggregation predictor worked well. However, modifications to it provide an incredible insight into protein binding.

6.1.1. Delta SAP score:

By calculating the SAP score of a complex, and then of each monomer, one may arrive at a delta SAP score. By doing careful accounting, one may also arrive at the delta SAP score for both the binder and the target. By looking at specifically the binder and the target side, one can determine where the hydrophobic driving force for the reaction is coming from. As the correlations show

(*8.2.1. Hydrophobicity metrics*), maximizing the hydrophobicity of the target, and putting the binder in the sweet spot are key.

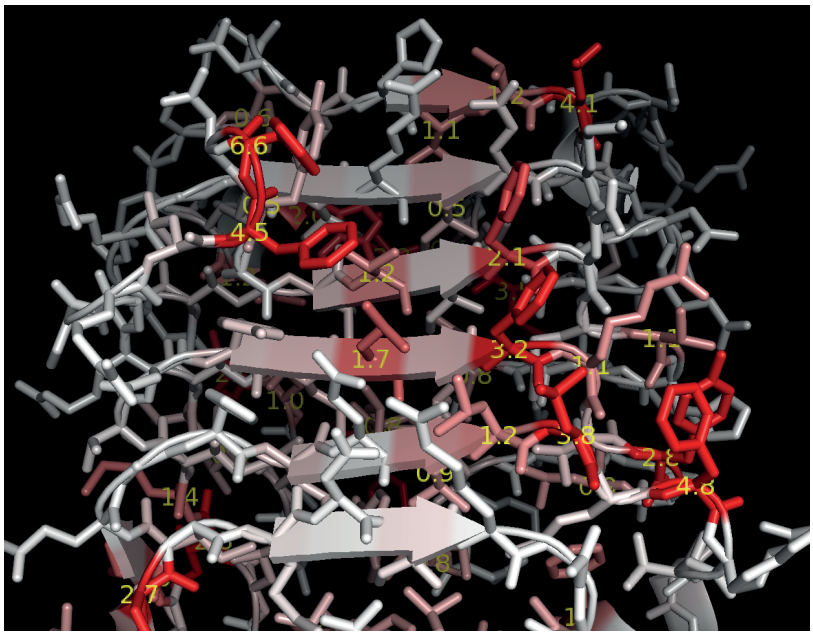

Fig 6.1.1.A: Target protein colored by SAP Score: This rendering shows residues with high SAP scores (red) in comparison to those with lower or 0 sap scores (white). The labels indicate the SAP score of the residue. Target protein is Insulin Receptor PDB: 4ZXB[18].

Additionally, knowing that maximizing the delta sap is critical, one can color a target pdb by SAP score in PyMOL before making binders to determine feasibility.

6.1.2. SAP score in Rosetta:

It was desirable to get SAP score into Rosetta as both a filter and a scoreterm. In this way, files coming from Rosetta would already have SAP calculated, and adding it as a scoreterm give the ability to optimize it during design. Adding the filter was easy as the SAP score is straightforward to calculate, but the scoreterm was very challenging because Rosetta likes to work with pairwise-decomposable scoreterms. This term is inherently not pairwise decomposable and as such, must be calculated on the fly.

The naïve implementation of this scoreterm using a simple distance-based SASA measure turned out to be too slow by making Rosetta take 10x longer to design a protein. A second approximation where the SASA values of all rotamers were precomputed and did not change during design was better at only a 50% slowdown. Finally, an approximation where all rotamers of a given aa-type were treated the same and the SASA values precomputed was fast enough to give only a 5% slowdown. Surprisingly, these methods all performed about the same with regards to SAP accuracy.

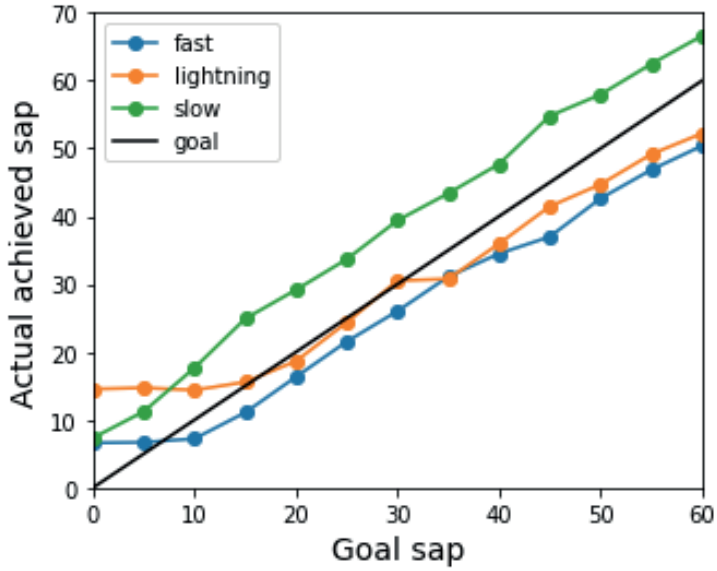

Fig 6.1.2.A: Accuracy of SAP design methodologies: Slow, fast, and lightning are the 10x, 50% and 5% slowdown variants in the text. 20 65-aa 3helical bundles had their surfaces redesigned with the SAP_goal in Rosetta set to the x-axis value. The y-axis value shows the actual achieved SAP score. There is a setting in the code (packing_correction) that allows one to correct for the constant-offsets seen in the "slow" setting.

6.2. Contact Molecular Surface:

Contact Molecular Surface is a measure of interface area developed by Longxing Cao during the course of this investigation. Its primary goal is to provide a measure of the interfacial area while penalizing poor packing. This measure replaces both Shape Complementarity and SASA by combining the benefits of both and eliminating the downsides.

6.2.1. Problems with Shape Complementarity and Delta SASA:

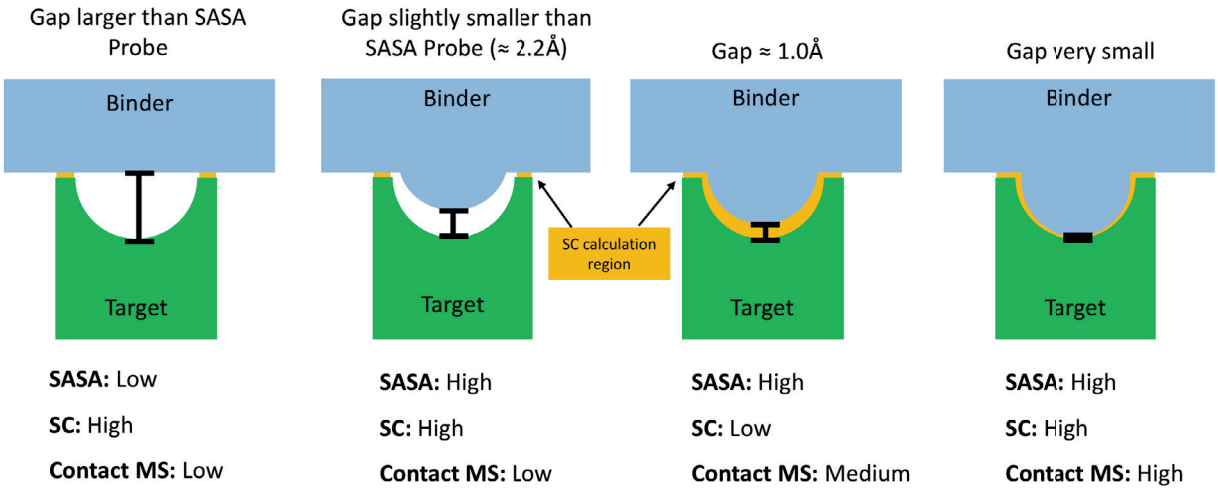

Fig 6.2.1.A: Illustration of the issues with delta SASA and SC: The quality of the interfaces increases from left to right; however, if one uses SC and Delta SASA as their measure of quality, the 2nd and 4th are indistinguishable. On the second, SC decides to not look at the region with the gap even though Delta SASA counted that as interface.

First, we'll look at the problems with Shape Complementarity (SC) as defined Lawrence et al[16]. SC strives to be a metric that corresponds to the quality of atomic contacts between two proteins. When applied to crystal structures, this method works great and is able to distinguish natural complexes from antibody structures. The method works by enumerating the molecular surfaces between two proteins. It then steps through each point on the surface, finds the closest point on the other surface, and multiplies a distance penalty by the dot-product to arrive at a number between -1 and 1; where 1 represents touching surfaces with a perfectly complementary dot product. The overall SC value is the median of all of these individual values.

The fundamental problem with SC is that once two surfaces go beyond a certain max distance, they are no longer included in the calculation. This results in a strong preference towards small interfaces. The larger the interface, the more places for mistakes, and generally the lower the score. Presumably one could devise a SC-SASA table to identify good SC values for different Delta SASA values, but Delta SASA has its own problems that make this not possible.

Delta SASA or Delta Solvent Accessible Surface Area represents the difference in the amount of surface before and after complexation as seen by a water molecule. Typically, a 1.4Å sphere is rolled across the surface of the proteins to determine this value (with the ability to enter closed cavities). In well packed native crystal structures, this method works very well and gives a good estimate of the actual buried surface area. However, in a poorly packed protein, a new issue arises.

The fundamental issue with Delta SASA comes from poorly packed interfaces. The method is designed to enter water-sized holes, and in a native structure, any hole probably has a water in it. In our *de novo* designs, we often have holes in our interfaces smaller than a water molecule. This means that the SASA probe will not be able to enter cavities and as such they will count towards the overall Delta SASA value. We've seen entire interfaces where only a few atoms are in contact and the rest is vacuum. However, Delta SASA cannot tell they are not in contact.

The flaws of Delta SASA and SC combine when we have the poorly packed interface just described with only a few atoms in contact. SC will only look at the few contacting atoms and arrive at a very high SC value. Meanwhile Delta SASA will say we have a large interface. The result is that by the metrics, we appear to have a great interface, but on closer inspection, the interface is terrible.

6.2.2. Contact Molecular Surface description:

To solve the issue of a mismatch between SC and Delta SASA, Contact Molecular Surface (CMS) instead merges the two concepts into one term. Instead of looking at the change in total surface area, it instead just looks at the interfacial surface area. And instead of calculating the surface area and complementarity on different surfaces, it combines them into one value.

CMS begins by enumerating the molecular surfaces between the two proteins. Computationally, this surface is composed of a pattern of edge-to-edge triangles. For each triangle CMS finds the distance to the nearest triangle across the interface. Then, the following equation is used to penalize the triangle's area based on gaps between the proteins:

$$\text{penalized\_area} = \text{area} * \exp(-0.5 * \text{distance}^2)$$

The overall CMS is then the sum of the penalized_area. For historical reasons, CMS is typically calculated on only one side of the interface. For this reason, CMS values are at most only half of the typical SASA value.

6.2.3. Contact Molecular Surface variants:

Many variants to CMS have been produced that provide better correlations in different circumstances.

By counting only the hydrophobic residues and hydrophobic atoms, one often arrives at terms that correlate to data better than raw CMS.

An attempt was made to give CMS a bonus in areas of high density. One can imagine two interfaces, one very large and poorly packed, and another small but tightly packed, that have the same CMS value. Presumably, the small and tightly packed interface is the better of the two. A metric like this was accomplished by squaring the value of all CMS triangles within a 5Å radius.

The penalized_area of each triangle was instead replaced by the square of the sum of all nearby triangles. Densely packed areas benefit more from the square than sparse areas, and this metric adequately captured densely packed regions. A similar method whereby instead of squaring the area, only counting areas above a certain threshold, would likely work just as well of better. But a suitable threshold would need to be identified.

6.3. Penalizing buried unsaturated hydrogen bonds:

Polar atoms in proteins should ideally be making hydrogen bonds to other polar atoms in order to optimize the energy. The reasoning is that in an unbound or unfolded state, all polar atoms make hydrogen bonds to water. Therefore, if in your protein structure, a polar atom is not making a hydrogen bond, it is unsatisfied because in another state it would be making a hydrogen bond. We will refer to such atoms as buried unsats.

Rosetta has a very hard time penalizing buried unsats for a good reason, buried unsats are not a two-body interaction, instead, they require knowing about all interactions in the area. The historical Rosetta definition for a buried unsat is a polar heavy-atom (non-hydrogen) that is making no hydrogen bonds and has no available SASA. This calculation is performed after a design trajectory by calculating the SASA of every polar atom and determining whether or not it is participating in a h-bond. For interface design this can be made a little more accurate by identifying buried unsats in the apo and complex forms and looking for buried unsats caused by complexation.

Rosetta has methods to try to penalize buried unsats, but they suffer from the critical issue that they are purely additive. Fa_sol[13] and lk_ball[19] penalize polar atoms when other atoms get too close. In this way, a polar atom should be penalized for being buried. When a hydrogen bond is made, the h-bond strength should overpower the burial weight. In practice though, what happens is that these solvation terms create small pockets around polar atoms. These small pockets lessen the penalty and allow the polar atoms to be buried without much penalty.

Rosetta designs used to be plagued with buried unsats where Rosetta would either totally ignore a buried unsat on the target, or place a glutamine into a totally apolar pocket. A way to explicitly penalize them was needed and the following algorithm was created.

6.3.1. The approximate_buried_unsat_penalty:

The approximate_buried_unsat_penalty[20] also known as the 3-body oversaturation penalty or 3BOP was developed to give Rosetta a way to explicitly penalize buried unsats. It uses two tricks to turn buried unsats into a pair-wise decomposable energy. The first trick is to define a burial region so that burial can be assessed by xyz position alone. The second trick is to perform a 3-

body calculation in advance using all rotamers and to assign a special set of 2-body interactions that penalize buried unsats correctly.

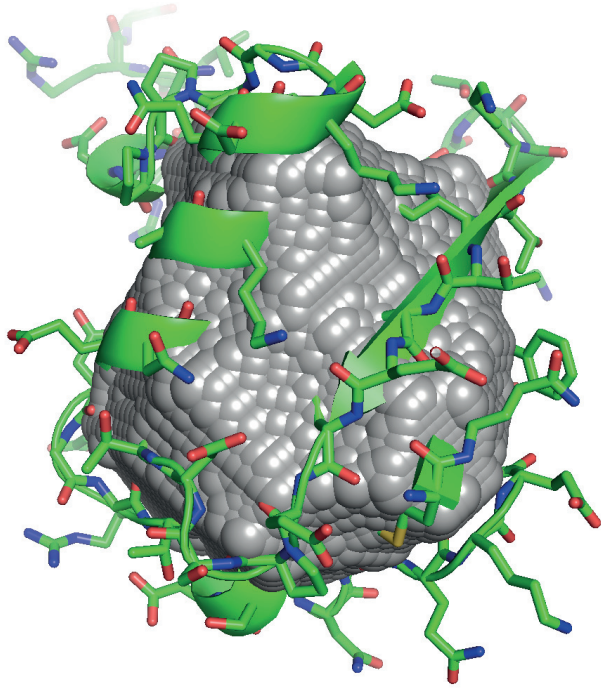

Fig 6.3.1.A: The burial region at 3.5Å of Ubiquitin: A model of Ubiquitin (PDB: 1UBQ)[21] was subjected to the EDTSurf method and all points deeper than 3.5Å from the surface shaded with spheres.

The burial method is based on the EDTSurf method[22,23] which determines the depth below the molecular surface for any point on a protein. In order to make this surface sequence independent, every position on the protein is replaced with Leucine before the depth is calculated. Then, by setting a depth cutoff (usually between 3.5Å and 5.5Å), one can determine a burial region.

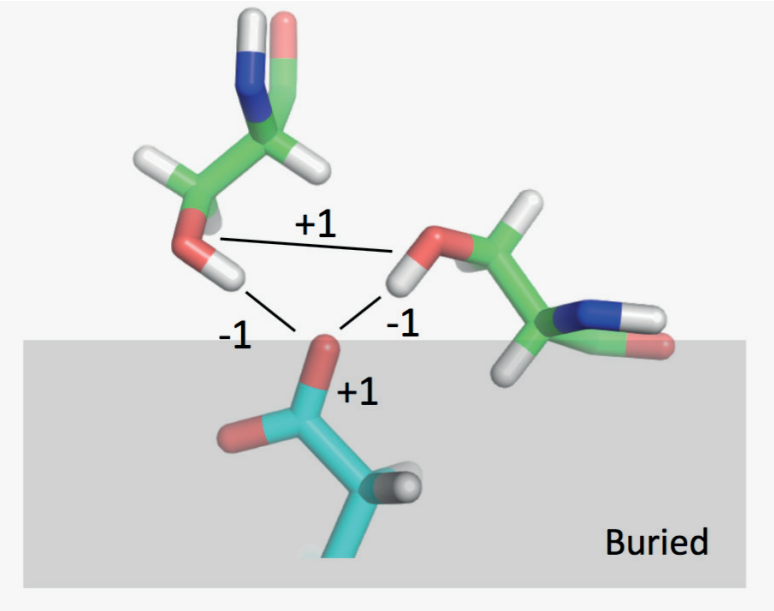

Fig 6.3.1.B: Visual Illustration of the 3BOP rules: The calculation for the upper oxygen atom of the buried ASP on the bottom proceeds as follows. The oxygen is buried: +1. The right SER satisfies it: -1. The left SER satisfies it: -1. Since both SER satisfy the same atom, they are penalized: +1.

In preparation for the 3-body calculation, one needs to know the rotamers, which atoms are buried, and which rotamers can make hydrogen bonds. At the start of the packing procedure in Rosetta, the rotamers are known, and so the polar atoms inside the burial region can be calculated. Hydrogen bonds between rotamers can also easily be calculated.

The procedure to create the penalty rules is as follows:

1. Assign a penalty to any buried polar atom
2. Assign a pairwise bonus to any atom that h-bonds to that polar atom
3. Assign a pairwise penalty to any two atoms that satisfy 2.

By assigning these penalties, a set of pair-wise interactions are created that nearly-perfectly accomplish the goal of penalizing buried unsats. If for instance, all the penalties and bonuses are set to a value of 1: Burying a polar atom provides a penalty of 1. A satisfying rotamer cancels this penalty by providing a bonus of -1 along their interaction edge. Now when another satisfying rotamer appears, it first double counts the satisfaction with another -1 to the buried polar atom. However, the oversaturation rule (3.) causes a penalty between the two satisfiers bringing the overall score back up to 0.

By modifying the weights of these bonuses and penalties, any quadratic relationship between score and number of h-bonds to a buried polar atom may be created. In the above example where all numbers are 1, the system prefers 1 or 2 h-bonds and deviates quadratically beyond that. By

settings the numbers to 3, 2, and 1 for 1., 2. and 3. respectively, a system can be made to prefer 2 or 3 h-bonds.

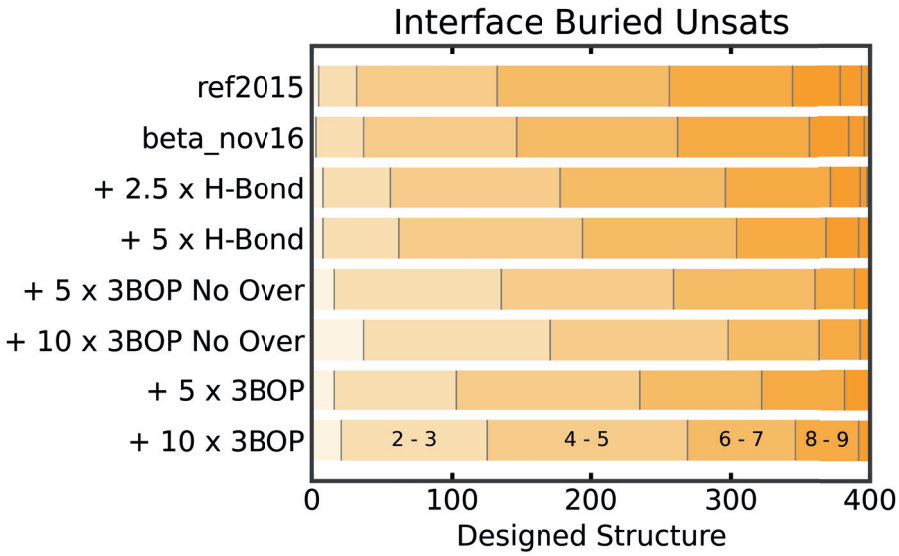

Fig 6.3.1.C: 3BOP interface validation: 400 docks between mini-proteins and barnase (PDB: 1BRS)[24] were designed using the approximate_buried_unsat_penalty in Rosetta along with various other conditions. Ref2015 and beta_nov16 were the two default scorefunctions. + 2.5 x H-Bond was ref2015 with the h-bond weights set to 3.5 and the 5.0 version set to 6.0. The methods with 3BOP in the name use the approximate_buried_unsat_penalty set to the weight specified, however, the No Over categories removed the oversaturation penalty. Although subtle, the 3BOP algorithm produced fewer buried unsats that the default scorefunctions or simply upweighting h-bonds.

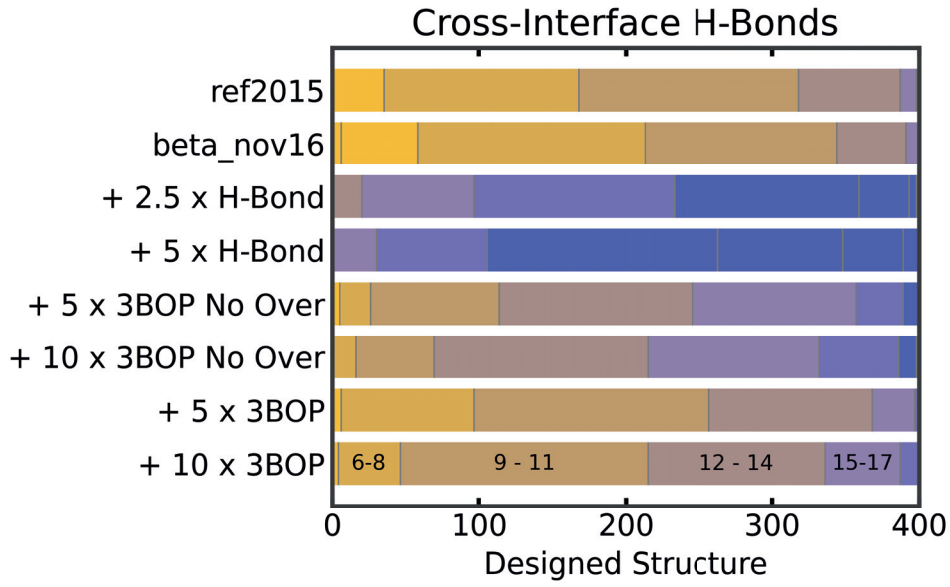

Fig 6.3.1.D: 3BOP interface validation h-bonds: A continuation of the previous figure. We see that although previously the 3BOP achieved the same number of buried unsats with and without the oversaturation penalty enabled, the difference is clear in the number of h-bonds. The oversaturation penalty was able to achieve the same number of buried unsats while making fewer h-bonds across the interface. For more detailed information, see[20].

As an approximation to a 3-body, there is of course a tradeoff. The critical flaw is that the penalty 3. above gets applied even when the buried polar atom is not present. The extent to which this causes problems was investigated and only causes minor deviations in amino acid recovery for residues that have a lot of polar atoms (like ARG). However, even though the recovery wasn't great, the designed structures still ended up with totally satisfied networks.

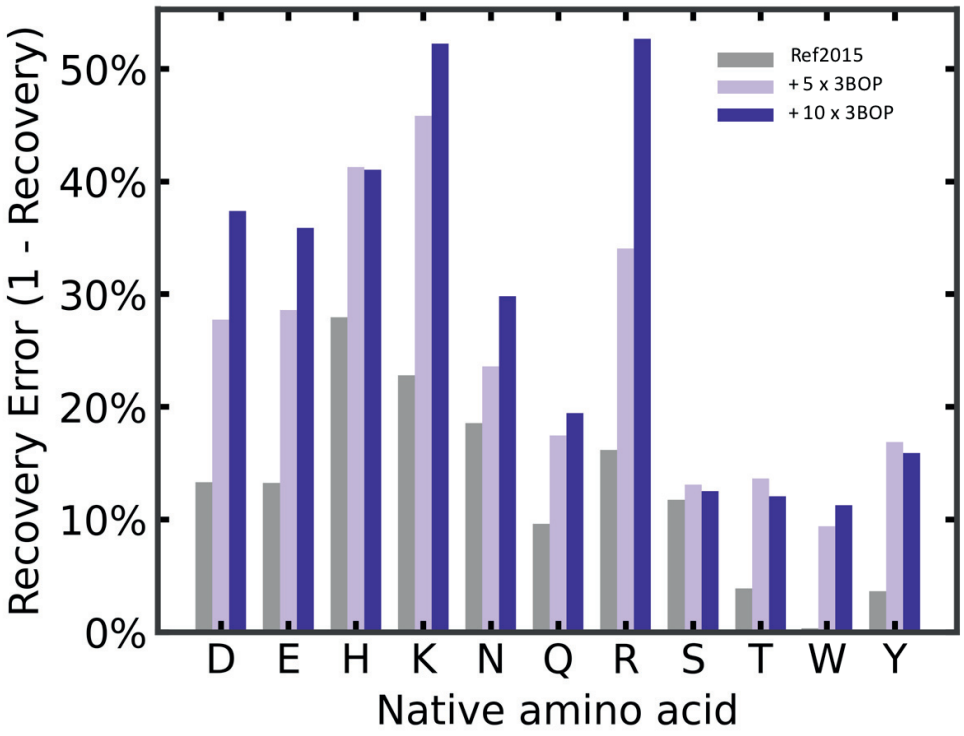

Fig 6.3.1.E: Native recovery with 3BOP enabled: As mentioned in the text, the extraneous oversaturation penalties of 3BOP may cause issues with residues being penalized unnecessarily. Ninety-seven native proteins containing buried h-bond networks had their networks redesigned. Plotted is the error in recovery when 3BOP was enabled. The general trend is that the more h-bonds that the amino acid can make, the harder of a time 3BOP had to place it. See[20].

For *de novo* design, we'd rather have a method that sometimes throws out good structures rather than one that never produces good structures. For this reason, the 3BOP, even with its flaws, is worth using.

6.4. The docking method:

The following section describes the method we created for docking proteins and all the various tweaks we applied.

6.4.1. Docking method description:

6.4.1.1. RifGen:

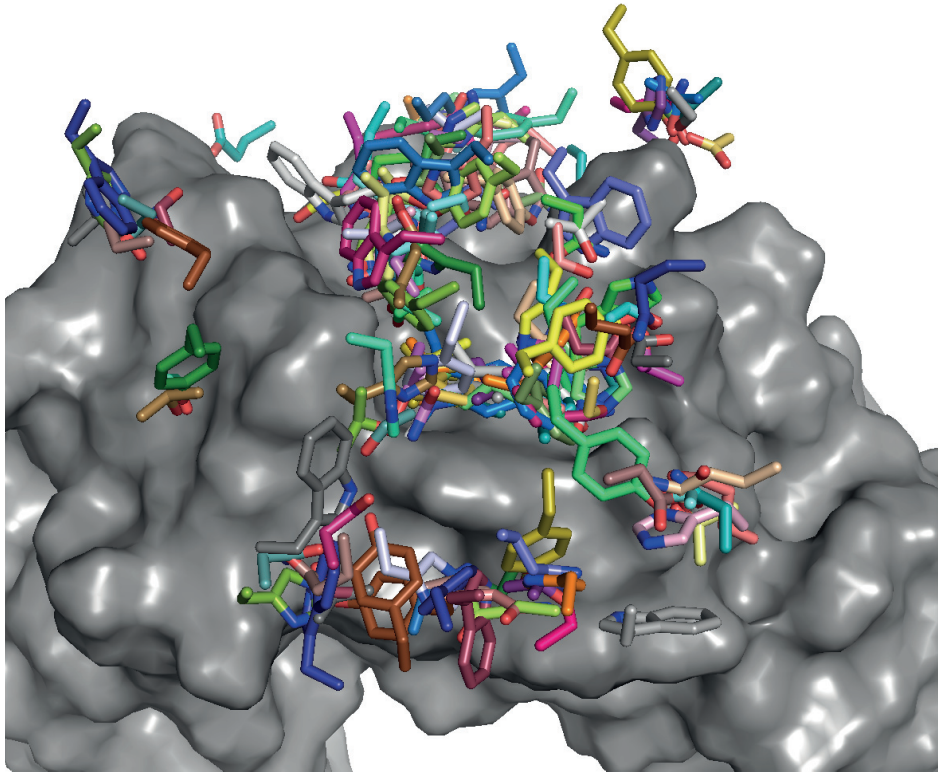

Fig 6.4.1.1.A: Example RifGen output: A tiny fraction of the rotamers placed by RifGen against IL7 Receptor alpha (PBB: 3DI3)[25] are shown. In reality, billions of rotamers are placed.

RifGen is the program that creates the Rotamer Interaction Field (RIF)[26]. While this program does not directly deal with docking proteins, it is the first step towards that goal. The RIF was envisioned as a way to pre-calculate all possible hotspot residues with the target surface. Using a 6-dimensional hash-table covering the 3 cartesian degrees of freedom and the 3 orientational degrees of freedom, the RIF allows one to dock proteins without expensive pair-interaction calculations. The idea is to fill the RIF with rotamers ahead of time and to store their backbone positions into the hash table, then when you bring a protein into the field, you can check the hash table at the specific backbone position to see if a good interaction exists.

The first thing RifGen does is to create a voxel-grid around the target in order to quickly calculate atom-target interactions. Each Rosetta atom type gets its own grid, and the atom is placed at each grid location and the energy recorded. In this way, later, rotamers can be scored by linear interpolation between the voxels for each atom.

The polar rotamers are inserted into the RIF by enumeratively sampling around each donor or acceptor atom on the target and calculating the h-bond energy as well as the overall score of the rotamer based on the voxel energies. All inverse rotamers (all rotamers where the polar head-group are superimposed) are stored into the RIF along with the energies.

The hydrophobic rotamers are inserted into the RIF by using a branch and bound algorithm. The idea is to take the grid energies and reduce their resolution several times while remembering the best scoring voxel. If the grid energies start at 0.5Å, they are approximated to 1.0Å and then 2.0Å in a nested fashion where the score of the 2.0Å voxel is the lowest score of the 64 contained 0.5Å voxels. The hydrophobic rotamers are then docked into these hierarchies of voxels using a suitably matched tree of 6D orientations. By starting with a 2.0Å grid of backbone positions and a suitably matched angular step, RifGen can estimate the lowest possible score that a given rotamer could achieve. The nested resolutions also allow one to say that none of the slightly perturbed versions of this rotamer at higher resolutions could score better. In this way, RifGen can throw away unpromising regions and space and focus only on areas where rotamers score below a certain threshold value. By performing this procedure at finer and finer resolutions, RifGen arrives at all possible hydrophobic hotspot residues which it inserts into the RIF.

6.4.1.2. RifDock hierarchical search:

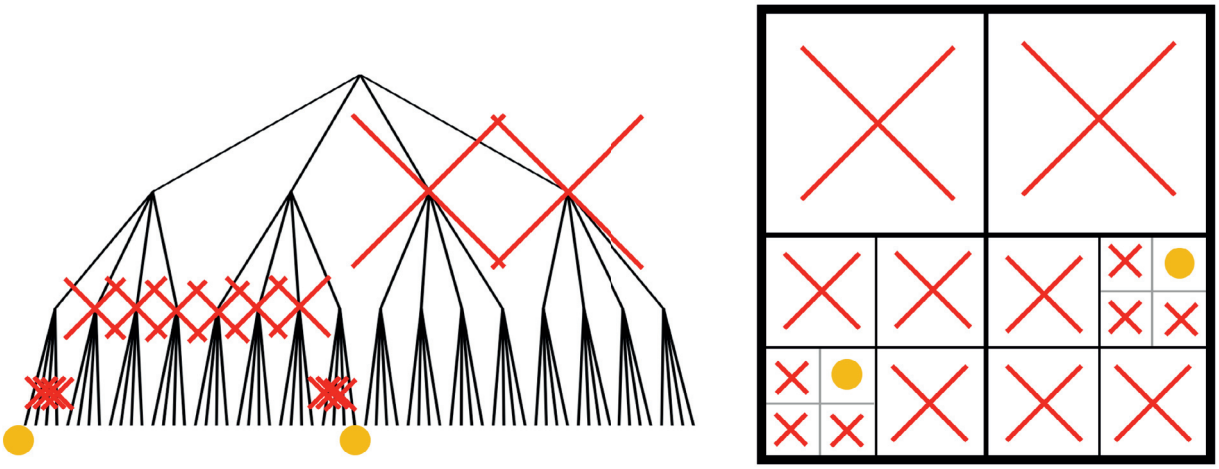

Fig 6.4.1.2.A: Schematic diagram of the hierarchical search procedure: The branch and bound algorithm allows us to skip searching branches that can be proven to be unsuccessful. If the goal here is to find the orange circles, when there is no orange circle in the current box, the box can be safely eliminated without need to "zoom in".

The hierarchical search was the original formulation of RifDock and allows one to in-principle find every conceivable dock between a binder protein and the target protein below a certain threshold. In practice, it turns out to be extremely susceptible to pool-poisoning (where false positives overwhelm the good docks), but with careful setup, it can work very well.

In a method similar to the hydrophobic hotspot placement, RifDock uses a series of nested docks at finer and finer resolutions to perform a branch and bound search. The resolutions range from 16Å to 0.5Å and 37° to 14°. The RIF itself is decreased in resolution by keeping the best hotspots found in the RIF as the resolution decreases. By precomputing the rotamer-scaffold energy for each rotamer at each position, RifDock is then in position to perform very fast energy evaluations at various resolutions.

For each resolution of docking, RifDock proceeds like this. For each of the scaffold positions to try, RifDock moves the scaffold to the right position and identifies the location of all backbone residue positions. Then, RifDock looks up each backbone position into the RIF and pulls out all the hotspots it found. RifDock then chooses the best amino acid at each position keeping in mind the rotamer-scaffold energy it found earlier. The sum of these best rotamers is the score for the dock. And in principle, this score is the best score that any of the higher resolution docks could achieve.

Knowing the best score of any child dock at each position, RifDock can safely prune away docks. Pruning a single dock has a massive effect because all of the children are pruned too. RifDock then expands each of the remaining docks into 64 new docks at the next resolution and repeats until the final resolution.

6.4.1.3. RifDock packing:

So far, RifDock has not considered whether or not the hotspots it has chosen are actually compatible with one another but has rather chosen the best one at each position. In the final steps of RifDock, it performs a packing operation to determine the best combinations of hotspots and to weed out impossible combinations.

RifDock does this again with clever pre-calculation. Since the backbone is the same for each packing trajectory, RifDock can pre-calculate all rotamer-rotamer interactions for each position. Then when it’s time to pack, RifDock uses the RIF score, the rotamer-scaffold score, and the rotamer-rotamer score in a rapid monte-carlo simulated annealing trajectory to produce the best combination.

After the packing is complete, RifDock knows the final score of each design. From here it performs an all-by-all clustering to remove redundant docks and outputs the best scoring docks for subsequent steps.

6.4.1.4. PatchDock + RifDock:

As mentioned previously, the hierarchical search in RifDock is prone to being overwhelmed by docks that look good at low resolution, but don’t pan out at high resolution. Before a fix for this

was identified, another approach was developed to remove the hierarchical search completely. The basic idea was to skip straight to the finest resolution and exhaustively sample around good starting points.

These good starting points were provided by PatchDock[27]. PatchDock uses the SASA surface of two proteins to identify good docks between them. It identifies the initial docks by comparing the angles between adjacent triangles on the surfaces and then scores by examining the shape compatibility of the SASA surfaces. At this stage, the binder interface sequence is not known, and it was found the mutating the binder to poly Valine produced the best results. The only goal of this step is to find locations where the binder fits well to the target structure without knowledge of the sidechains.

These PatchDock outputs are then fed into RifDock for exhaustive sampling. Using the finest resolution of the RIF, the docks are enumeratively moved through 0.5Å sampling with 3° angular resolution in the local area around each dock. The workflow from here is exactly like the standard RifDock with the best docks moving on to packing, then clustering, and then output.

6.4.1.5. Rosetta Design:

The Rosetta Design procedure is discussed in *6.5. Rosetta Design optimizations*. However, the best docks from RifDock have the interfaces optimized using Rosetta to identify the best combination of amino acids.

6.4.1.6. *De novo* motif graft:

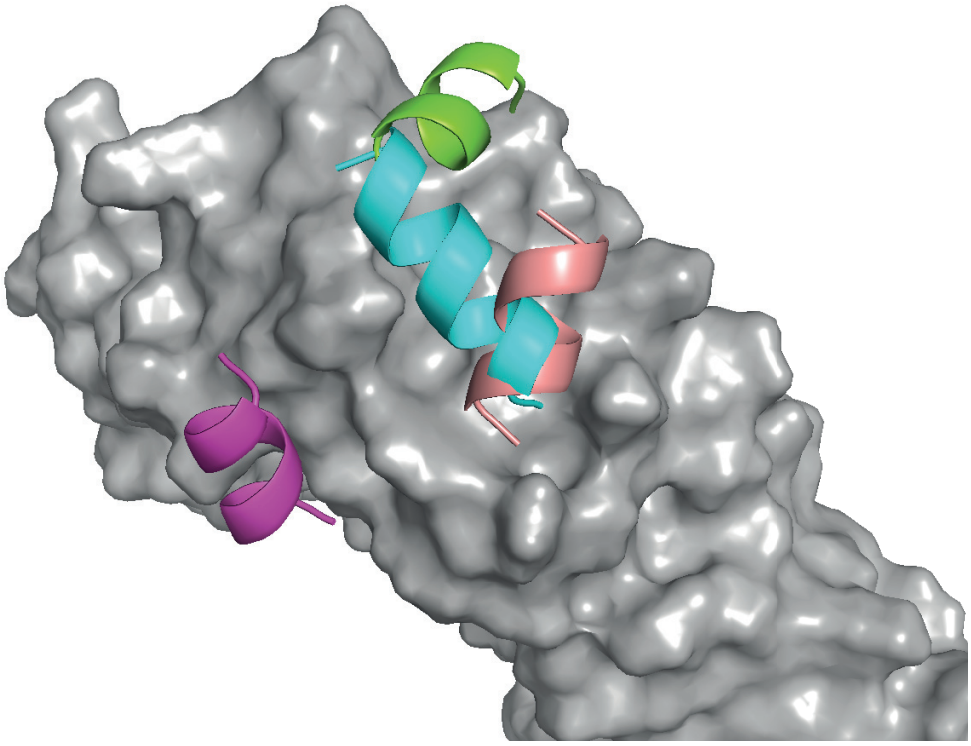

Fig 6.4.1.6.A: Examples of de novo motifs: Here we see a few de novo motifs placed against IL7 Receptor alpha (PDB: 3DI3)[25]. In reality, hundreds of motifs are used.

While RifDock is very powerful, for a variety of reasons it does not always find the best docks for a given scaffold. With this in mind, a protocol was developed to redock the scaffolds in such a way that half of the interface was already known to be excellent. The general idea is to take all of the designs after RifDock and Rosetta Design, to extract and cluster secondary structure elements that make incredible contacts with the target, and to superimpose all of the scaffolds back onto these great secondary structure elements. While most of the superimpositions don't result in great docks, occasionally, the rest of the scaffold fits perfect and one ends up with a fully incredible dock.

The motif extraction process is relatively straightforward and consists of identifying secondary structural elements by DSSP[28], scoring them, and saving them for later. The clustering is equally straightforward by using TM-Align[29] to perform a greedy clustering strategy with a given cutoff value.

Selection of the best motif in each cluster involves identifying critical positions, and then identifying motifs with the best score at the critical positions. When looking at a cluster of motifs,

some positions always have favorable interactions while others are more sporadic. The sporadic interactions are usually an errant tryptophan making a poor interaction. For this reason, each position is given a relative weight based on the proportion of times it makes good interactions. The weights are then multiplied by the per-position ddG values to arrive at the weighted ddG for the motif. The best motif within each cluster then become the representative for the next step.

The final step in the motif selection process is to pick the best motif clusters. Since the clusters have been reduced to one member, the top X motifs are chosen. The number of motifs we used varied from 10 to 10,000, with numbers around 1,000 being optimal.

For the actual motif grafting, each of the scaffolds is compared to each of the motifs using all possible alignments. Any alignment with an all-backbone-atom RMSD of less than 0.5Å was kept. The scaffold is then superimposed upon the motif and clash checked against the target. If no severe clashes are detected, the hotspot amino acids from the motif are copied to the scaffold protein and the output treated the same as a RifDock output.

6.4.2. Improvements to the docking procedure

At a higher level, we can consider the replacement of the hierarchical search with PatchDock and the addition of the de novo motif grafting stage to be the biggest improvements. More subtle improvements to each part have also made a difference though.

6.4.2.1. RifDock improvements:

The biggest improvements made to RifDock all come from better scoring. RifDock's base score is based on the van-der-Waals forces and solvation forces found in Rosetta plus the scaffold energies. Perhaps the single largest improvement was to ignore small clashes between the scaffold and its rotamers. Whereas with full accounting, a typical dock would find 2 to 3 "Rifres", with small clashes allowed, a typical dock could have more than 7.

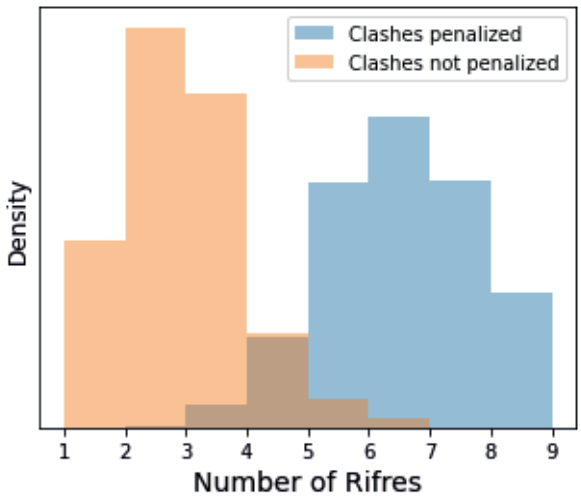

Fig 6.4.2.1.A: Number of Rifres with regard to penalized clashes: A quick test against IL7 Receptor Alpha (PDB: 3DI3)[25] docking 100 mini-proteins showed that by setting -favorable_1body_multiplier to 0.2 and -favorable_1body_multiplier_cutoff to 4, that the number of Rifres per output greatly improved.

Longxing Cao also introduced an important system that gives the user more control over what RifDock will output. By enforcing certain "requirements", like a specific bidentate or specific hydrophobic interaction, one can tailor a dock to their liking.

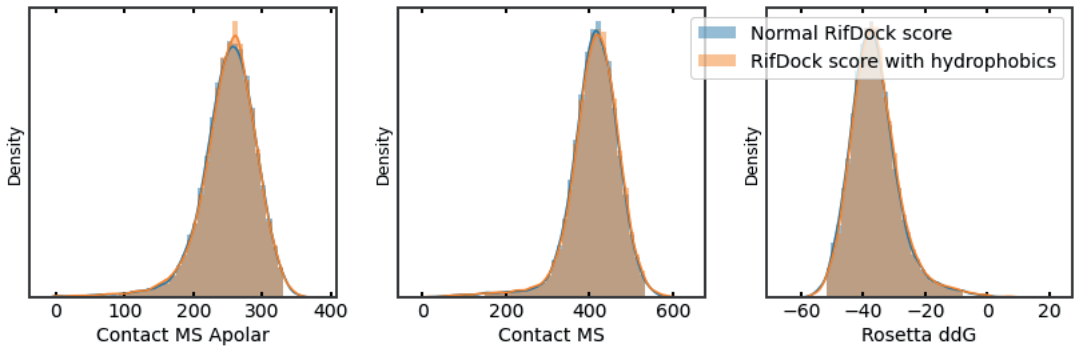

Fig 6.4.2.1.B: Effect of RifDock hydrophobic contacts: A quick test against IL7 Receptor Alpha (PDB: 3DI3)[25] using RifDock with and without the -require_hydrophobic_residue_contacts flag using the PatchDock + RifDock method, followed by prediction and Rosetta Design. While this flag greatly affects what the RifDock outputs look like, evidently, it doesn't show through after Rosetta Design.

Another improvement was the incorporation of a hydrophobic-centric energy system to RifDock. This system allowed the number of hydrophobic residues on the target that were contacted to be tallied. With this system, it was possible to tell RifDock to only output docks that contact many hydrophobic residues on the target.

Two more noteworthy features were added to RifDock that didn't help as much as one would expect: Delta SASA and buried unsats. While the Delta SASA allowed for docks with more overall shape matching, it didn't necessarily require that the docks actually place sidechains to make the

contact. As to the buried unsats, the “burial” of a polar atom was very difficult to measure and used the same voxel-based approach as for SASA. Even with the burial working correctly, the bigger issue was that RifDock was perhaps too early in the protocol to worry about buried unsats. Inside RifDock, the polar sidechains cannot repack. It was found that decreasing the number of buried unsats at the RifDock stage didn’t not actually decrease the number of buried unsats after Rosetta Design.

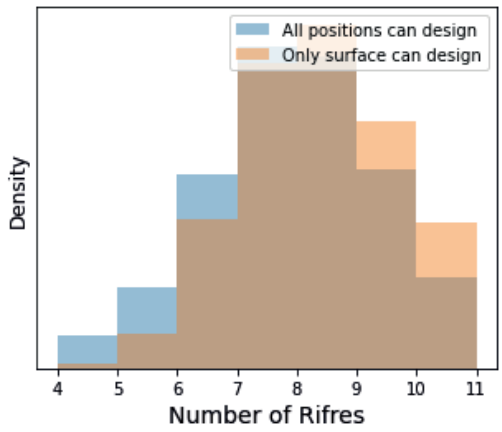

Fig 6.4.2.1.C: Effect of design selection for RifDock: A quick test against IL7 Receptor Alpha (PDB: 3DI3)[25] showcasing the effect of the -scaffold_res_use_best_guess flag. While in this example the effect is not large, on other targets and with different flags, it can be significant.

Finally, it’s worth pointing out the critical change that saved the hierarchical search. The biggest flaw of the hierarchical search is that a few great scoring hotspots on the target can be placed at nearly every position on the scaffold at lower resolutions. This is a totally non-physical situation; however, it cannot be properly accounted for when rotamers are allowed to move 4Å from their docked position. The solution here was to limit the number of positions on the scaffold protein that are allowed to accept rotamers. Limiting the positions to simply the surface was enough to fix the problem as evidently the core of the scaffold was full of “good interactions”.

One last trick for the hierarchical search if using the requirements system is to actually perform hundreds of simultaneous searches at the same time, but all slightly offset from one another. The hierarchical search has problems with rotamers that are very near bin edges. Some of these bin edges end up in different 16Å bins even though they are only 0.2Å away from each other (because the 16Å bins have to cut *somewhere*). This leads to the hotspots appearing and disappear as a scaffold works down the resolutions. Instead, if one sets up several misaligned hierarchical searches, the hope is that in one search, the hotspot will be present through the resolutions.

6.4.2.2. Improvements to the *de novo* grafting:

The *de novo* grafting technique saw improvements in both motif selection and grafting method.

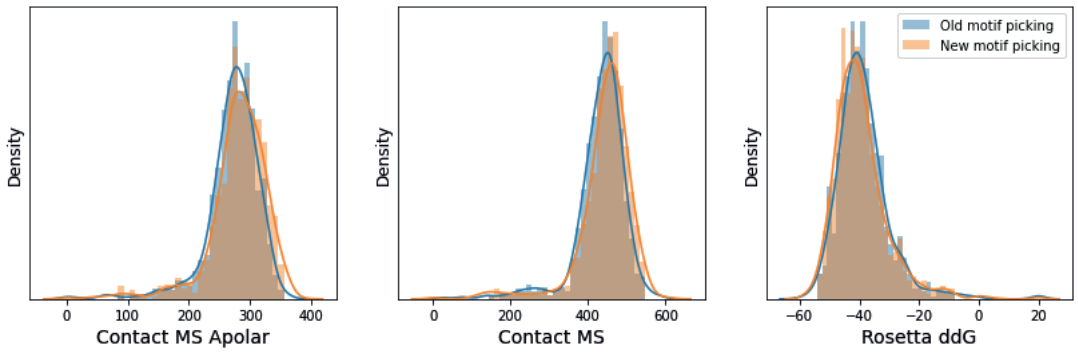

Fig 6.4.2.2.A: Effect of new motif selection method: Here we see a slight shift in the metrics when a small design run was performed against IL7 Receptor alpha (PDB: 3DI3)[25] using either the old method of picking motifs based purely on ddG and h-bonds vs the new weighted method.

For motif selection, the biggest improvement was the position-weighted ddG method from Longxing where only positions that always score well are counted. This removed motifs with "random tryptophans" and resulted in higher quality motifs. There was also an attempt to capture buried unsats at this stage and to count the number of h-bonds. While both of these methods worked to decrease buried unsats and increase the number of h-bonds, as the correlations in *8.2.4. Buried unsats* show, optimizing these metrics does not actually lead to an improved success rate.

A fundamental question exists in motif grafting which is: "What is the best motif?" Is it the motif that makes the best contacts, or the motif that "shares the interface" and allows the rest of the motif to make a great contact. This question has been briefly examined and it appears that the best motif is the best scoring motif. If one looks at the fraction of grafts that end up passing the filters used for ordering, there is a huge discrepancy between number of successes for each motif. The motifs with the highest success rates, however, seem to be the best scoring motifs. This fits well with the hypothesis that a motif is an extraordinary part of an interface waiting for the other half. Whereas a poor scoring motif will always lead to an interface where half is poor scoring.

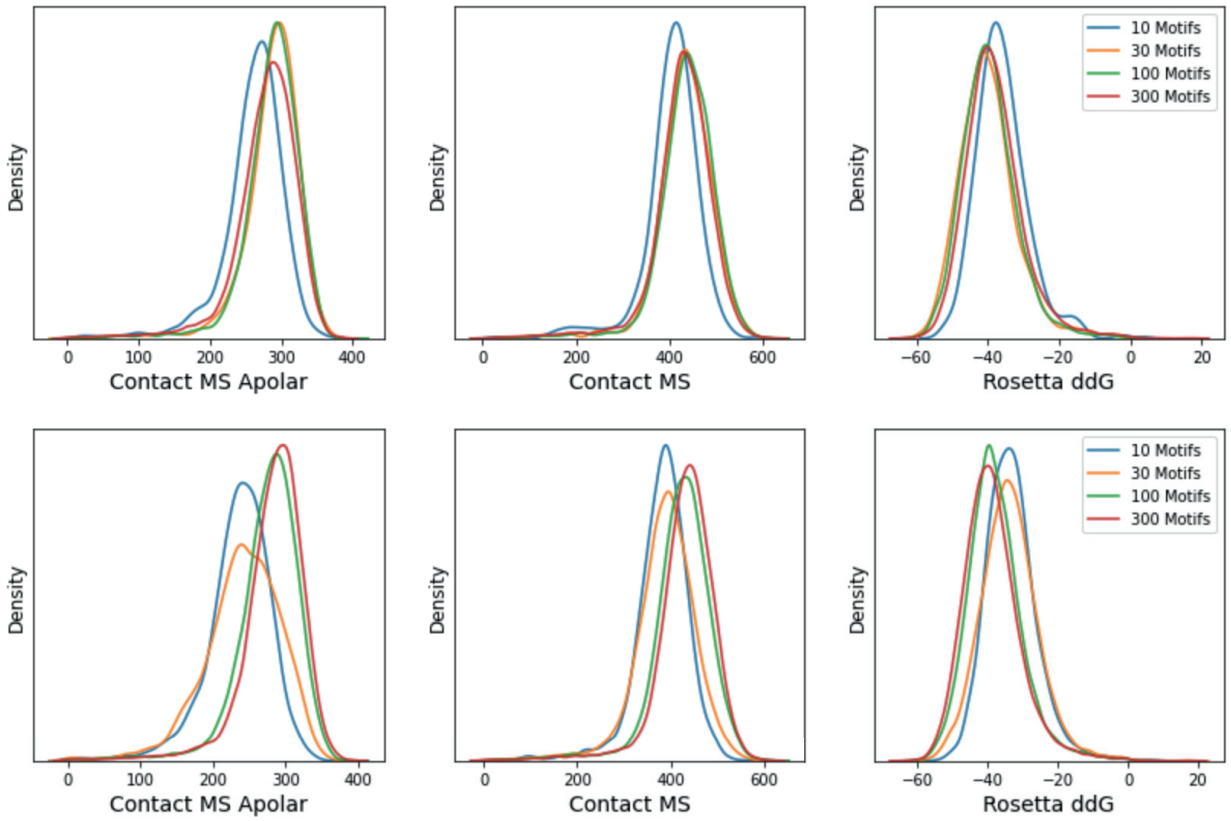

Fig 6.4.2.2.B: Number of motifs vs metrics: Here we see the results of picking different numbers of motifs against IL7 Receptor alpha (PDB: 3DI3)[25]. 30,000 docks were used to generate motifs and after motif grafting, either the best 10% of grafts (upper graphs) or best 15,000 grafts (lower graphs) were chosen. When looking at the best 10% of outputs, 30 to 100 motifs seems to be best. However, when taking the top 15,000 grafts, more motifs seems to always do a better job, simply because one gets more outputs.

There is a question of how many motifs should one move forward with. The optimal number depends on whether one looks at the top 10% or the top 100 docks. The reason for the discrepancy is that using more motifs leads to more docks. And since this area of research almost invariably benefits from more docks, using more motifs will lead to a stronger best 100 docks. However, looking at the top 10% looks more at the average quality in which case there is an optimal number. In a test case for an average sized interface, 30 motifs beat out 300 motifs by the 1% metric. However, if only the top 100 are considered (which is the same criteria one uses when placing an order), then 300 motifs is the right choice.

Finally, the actual grafting procedure must be considered. It was quickly noted that the Delta SASA of a graft does not change much after Rosetta Design, and so filtering on Delta SASA was the first obvious improvement. Next, there is the question of whether to replace part of the scaffold with the motif, or to superimpose the scaffold onto the motif and copy amino acids. The superposition method eventually won out because it leaves the core of the protein intact. Creating a well packed core takes a large amount of backbone sampling, we found that most of

the cores of the proteins could not be salvaged after a helix was replaced. (See *6.6.2. Faster/smarter scaffold design*).

The last parameter for motif graft is what RMSD threshold should be used to consider a graft acceptable. While the choice is not critical as the designs will still be filtered later, the grafting experiment (*Fig 8.4.1.A: Success of grafted designs versus RMSD of graft*) indicates that a threshold of somewhere between 0.5Å and 0.7Å CA RMSD is likely to be the best.

6.5. Rosetta Design optimizations:

After the docking procedures above provide their output. The rough docks must have their sequences optimized by Rosetta in order to arrive at the final binder. A monte-carlo simulated annealing procedure picks the amino acids at each position and a gradient-decent minimizer optimizes the energy slightly after the amino acids have been placed. During the minimization, the scaffold backbone and sidechains are allowed to move; however, on the target, only the sidechains may move. The reasoning for this discrepancy is that Rosetta is not very good at scoring backbone conformations. It has been found that allowing the target backbone to minimize allows scores to arbitrarily improved while the target enters conformations it would never adopt.

The specific design procedure used involves running the monte-carlo simulated annealer (the packer) followed by the minimizer several times at varying repulsive weights. The original method was to set the repulsive part of the scorefunction to: 0.02, 0.25, 0.55, and 1.0 its original weight and perform a packing and minimization procedure[30]. In this way, large amino acids can force their way into cavities and push everything else out of the way. As a later improvement will show however, there is a better set of weights. (See *Fig 6.5.2.B: Effect of better Rosetta Design ramping schedule*).

6.5.1. Rosetta Design pre-filtering:

A serious runtime mismatch exists between the docking methods and Rosetta Design. While Rosetta Design takes 400 seconds per dock, RifDock only takes about 3 seconds. Visual inspection of some RifDock outputs makes it clear that they could never form a good interface, and indeed, after Rosetta Design, they do not. Instead, it was hypothesized that performing a rapid Rosetta Design would provide enough insight into a dock to allow for an early decision to be made.

A protocol was eventually developed that allowed for exactly this. Using only 15 seconds to poorly design an interface, a protocol was able to accurately rank designs on their likelihood to pass a set of filters. The rapid design procedure uses the Rosetta packer, but uses a limited number of amino-acid choices and a limited score function. Nearly all rotamer-rotamer interactions in the score function are turned off leaving only the van-der-Waals forces, solvation

forces, and hydrogen bonds. This allows the packer to run in as little as 4 seconds. While the outputs from this packer trajectory are sufficient for calculating Contact Molecular Surface, if one wishes to calculate Rosetta ddG, it is necessary to minimize to avoid clashes. A gentle minimization followed by a more aggressive minimization are used to remove clashes. The resulting structure is adequate for estimating Rosetta ddG.

In parallel to the rapid protocol, a small subset of the docks is subjected to the normal design procedure (typically 1,000). These docks then serve as the training data for the machine learning model.

A variety of models were examined in their ability to predict the results of the full design procedure based on the fast design procedure. Specifically, the goal was to predict which designs would pass a set of filters that would be used to place the order. An issue arose very quickly, however, which is that of the 1,000 docks used for training, only about 6 would pass all of the filters. This precluded the use of any model that looked only at the final result, and instead, the individual filters would need to be investigated.

Through lots of trial and error with a Random Forest, it was eventually discovered that the best predictor of each filter was that same metric in the rapid design trajectory. For instance, the best predictor of whether or not a design would have sufficient Contact Molecular Surface, was the Contact Molecular Surface value in the rapid design trajectory. With such a clear 1-1 relationship, the Random Forest was scrapped in favor of a simpler model that would not be subject to overfitting/memorization.

The final model was a Maximum Likelihood Estimation model where a probability could be assigned that a given dock would pass a given filter. By binning the rapid design trajectory by a given metric, and looking at the fraction of designs in each bin that pass the final filter, one can estimate the probability that a design with a given metric value will pass the filter. These graphs invariably take on a sigmoid shape and as such, a sigmoid is fitted to the distribution in order to fully eliminate the ability to memorize the input data. These sigmoids range from 0 to 1 and are multiplied together to arrive at the final probability that a dock will pass all of the filters. The final functional form of the predictor is exactly the same as a logistic regression function, however, the difference lies in how the fitting parameters are obtained.

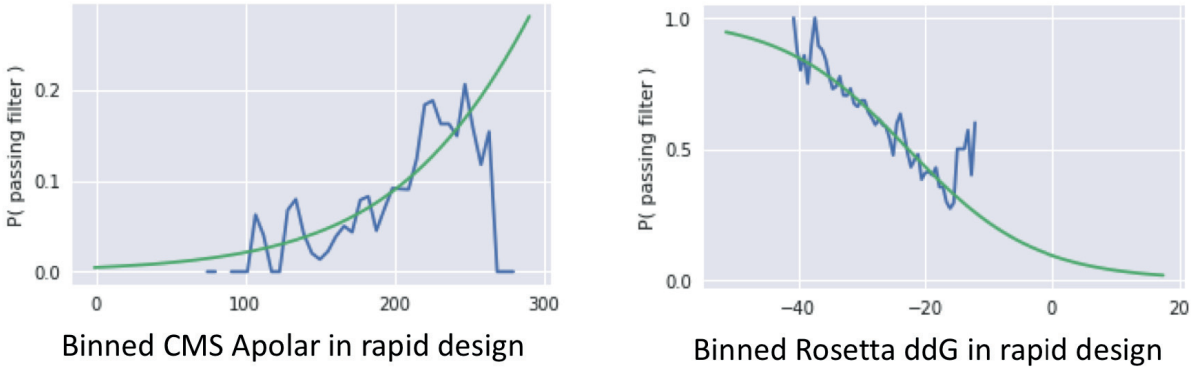

Fig 6.5.1.A: Example MLE fit graphs: In blue are the success rates of passing a CMS filter or a ddG filter after a full Rosetta Design calculation while the x-axis shows the calculated value during the rapid design procedure.

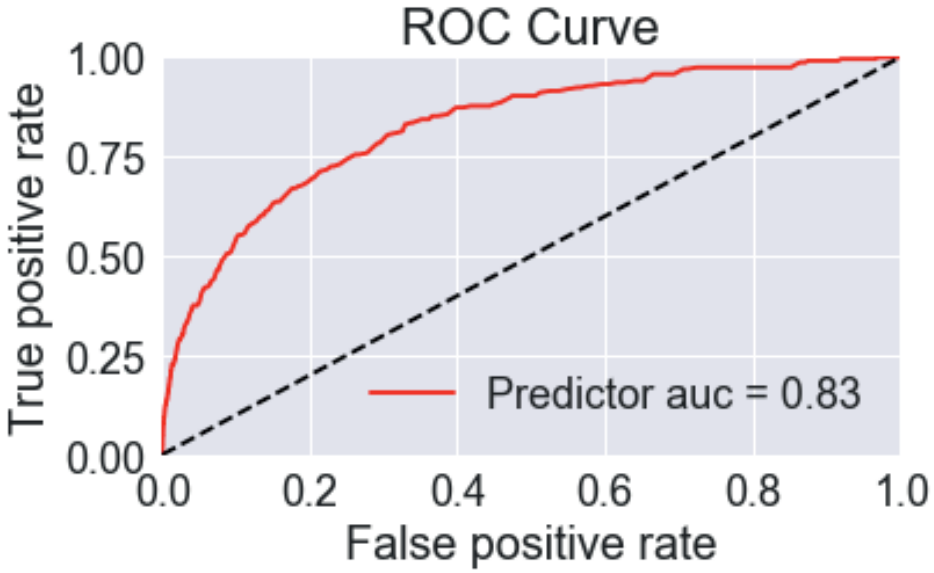

Fig 6.5.1.B: Example ROC plot for the predictor: Here we see a typical ROC plot for the predictor. Using the MLE estimates for each term and multiplying them together. A predictor with decent predictive power is produced.

These predicted values can then be used to rank the designs. While the estimated probabilities do not necessarily correspond to the actual final probabilities (as the metrics are not independent of one another), the relative ranking of the designs is roughly correct. Then by taking the top designs, the poor scoring docks can be eliminated leaving only the docks that appear to be promising.

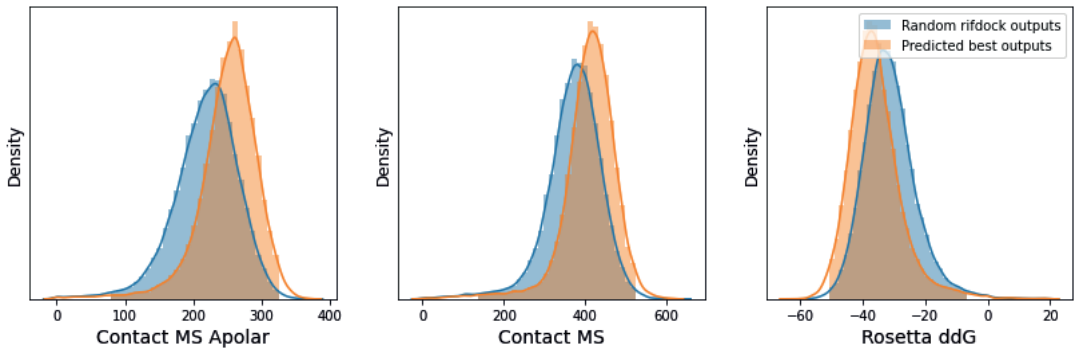

Fig 6.5.1.C: Example predictor metric shifts: Shown are the results of designing 30,000 out of 300,000 RifDock outputs against IL7 Receptor alpha (PDB: 3DI3)[25]. The predictor does a great job at selecting the best designs to move forward with.

An added benefit of this method is that even in the total failure case where there is almost no correlation between the predicted value and the actual value, the method selects the top scoring designs by each method. This is what one would naively do anyways, simply take the top fraction from each metric, and it is convenient that this is the default behavior.

6.5.2. Rosetta Design optimizations:

Optimizing the final Rosetta Design operation is critical because this step is the only way in which the final amino acids are chosen. A poor Rosetta Design operation will never achieve good docks because it isn't necessarily possible to "get lucky". If the energetics and sampling aren't set up right, achieving the right set of amino acids may never happen.

Two important optimizations have already been discussed in their own sections. The Sap Score (*6.1. SAP score*) and the approximate_buried_unsat_penalty (*6.3. Penalizing buried unsaturated hydrogen bonds*). These optimizations give the designer more control over what the outputs from Rosetta Design look like, but would be insufficient without the next optimization.

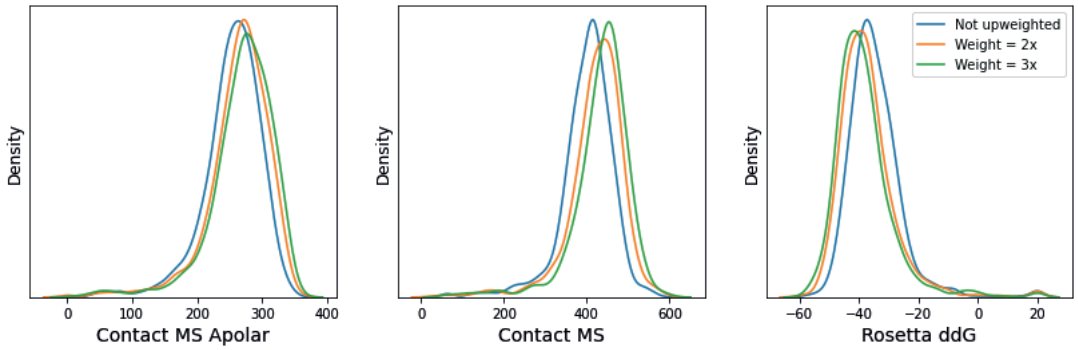

Fig 6.5.2.A: Effect of upweighting interface edges: A quick test against IL7 Receptor alpha (PDB: 3DI3)[25] shows that by upweighting the interface edges using the ProteinProteinInterfaceUpweighter that all interface metrics improve.

Perhaps the most important change to Rosetta Design was to tell the score function that the interface energy is more important that the energy of the monomer. By default, the score function treats a monomer self-interaction as just as important as a cross interface interactions. While the validity of this may be debated, it certainly does not result in interfaces with good metrics. A sort of null-solution here is to leave the interface void and to instead pack highly favorable interactions into the monomer. Instead, if we upweight the interface, usually by a factor of 2 or 3, we end up with interfaces that score much better. While it may be the case that we have only shifted the poor docks from having poor scoring interfaces to poor scoring monomer characteristics, at least with the interface upweighted, the critical interface interactions were made and it's now clear that the monomer could not support them.

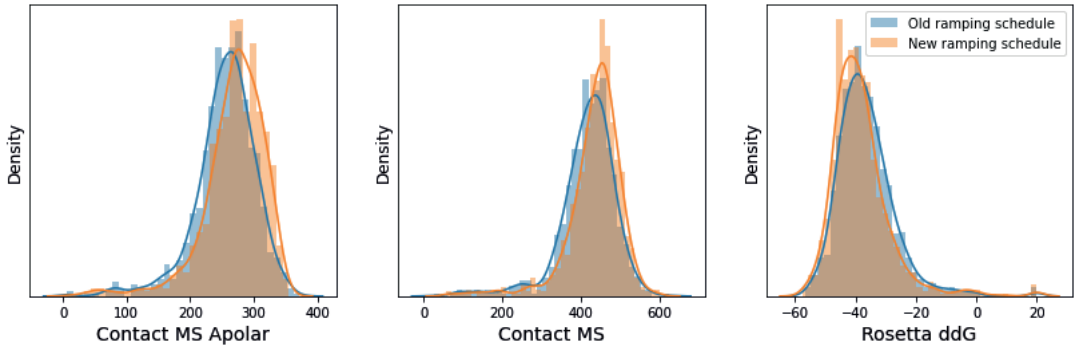

Fig 6.5.2.B: Effect of better Rosetta Design ramping schedule: A quick test against IL7 Receptor alpha (PDB: 3DI3)[25] shows that by using the new ramping schedule that all of the interface metric improved. (Here the old FastRelax[30] schedule was compared with rosettacon2018[31] using standard beta_nov16 ref weights).

The next most important factor that was optimized for Rosetta Design was the choice of the repulsive ramping schedule. The original ramping schedule often led to structures that got smaller with time. Although the goal was to allow large amino acids to fit into small places, in reality, the backbone instead compressed, and the amino acids got smaller. Jack Maguire found that by using a slightly different scheme that wasn't as soft, that structures maintained their initial size[31]. The same held true for interface design.

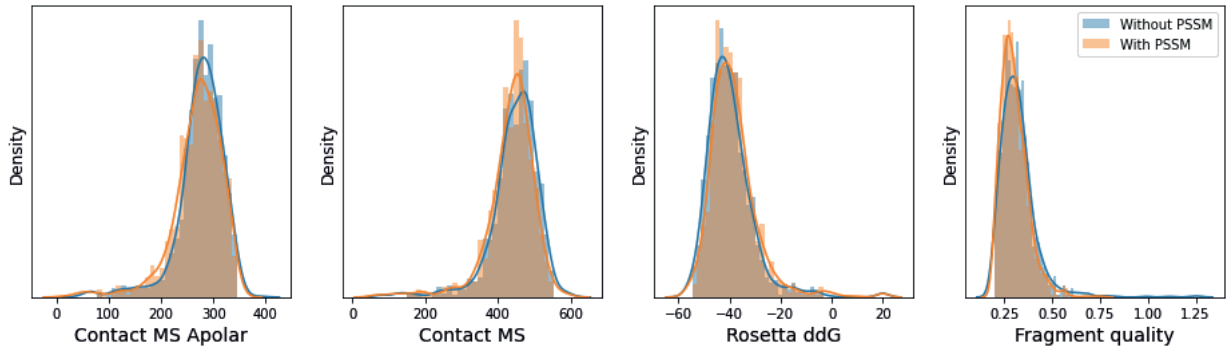

Fig 6.5.2.C: Effect of using a fragment-based PSSM: A quick test against IL7 Receptor alpha (PDB: 3DI3)[25] shows that the fragment-based PSSM had almost no effect on anything. The interface metrics didn't get worse which is good,

but the fragment quality didn't improve. Larger effects are seen for monomer design. (StructProfileMover used with consider_topN_frags="100", RMSthreshold="0.6", burialWt="0", and res_type_constraintset to 1.5.)

Another important consideration for design is that the monomer needs to actually fold correctly. For proteins of this size, the most important part of folding are the loop amino acids. We found that by incorporating a fragment-based lookup method to derive PSSMs[32] that we were able to improve the folding characteristics of our designs. This method looks at the CA positions of every 9-amino acid stretch and looks them up by RMSD into the database of known proteins structures. Then it derives weights for each amino acid type at each position.

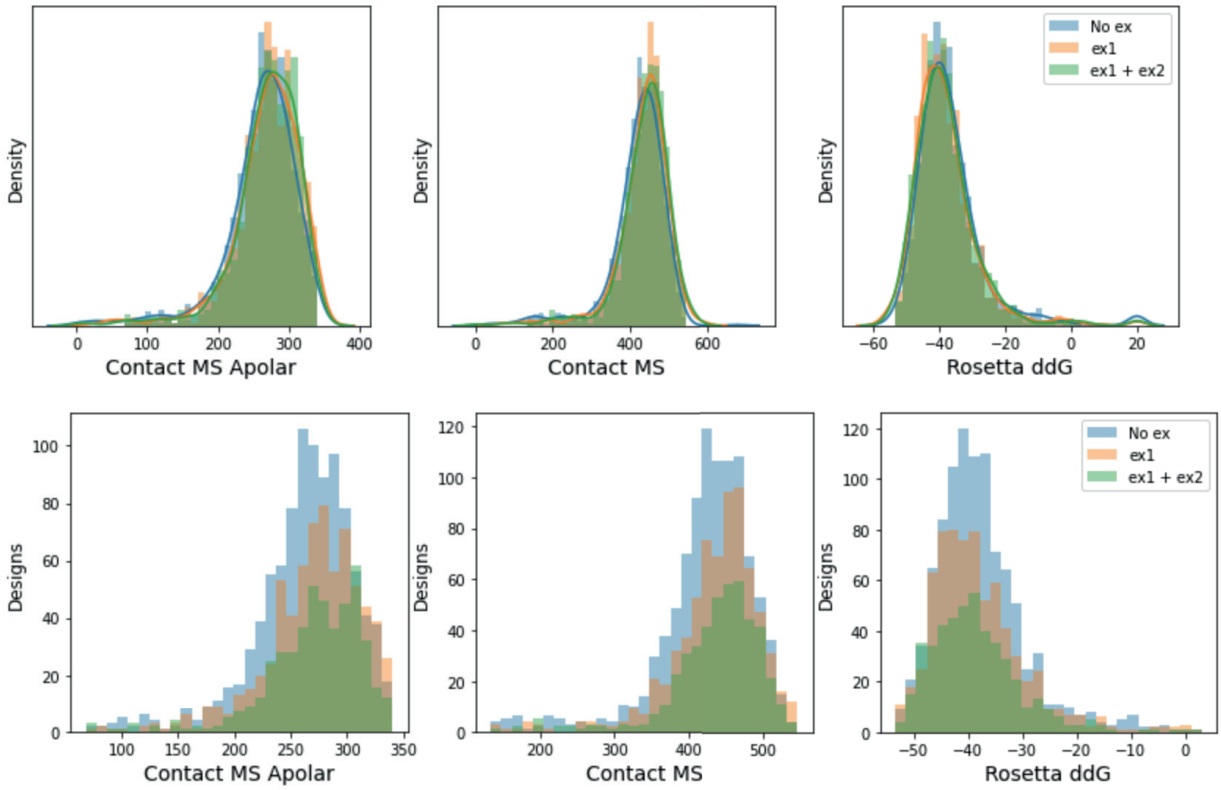

Fig 6.5.2.D: Effect of extra chi flags: Shown are the results of using the -ex1 and -ex2 flags on Rosetta Design for designs against IL7 Receptor alpha (PDB: 3DI3)[25]. The top graphs show that compared output-to-output, perhaps -ex1 and -ex2 combined perform better than the others. The bottom graphs show what happens when we correct for CPU time however. With the three protocols taking on average 190, 260, and 400 seconds per output, the number of outputs were scaled down proportionately. Now it is clear that the best method is either -ex1 or no -ex flags for the most good outputs per CPU time.

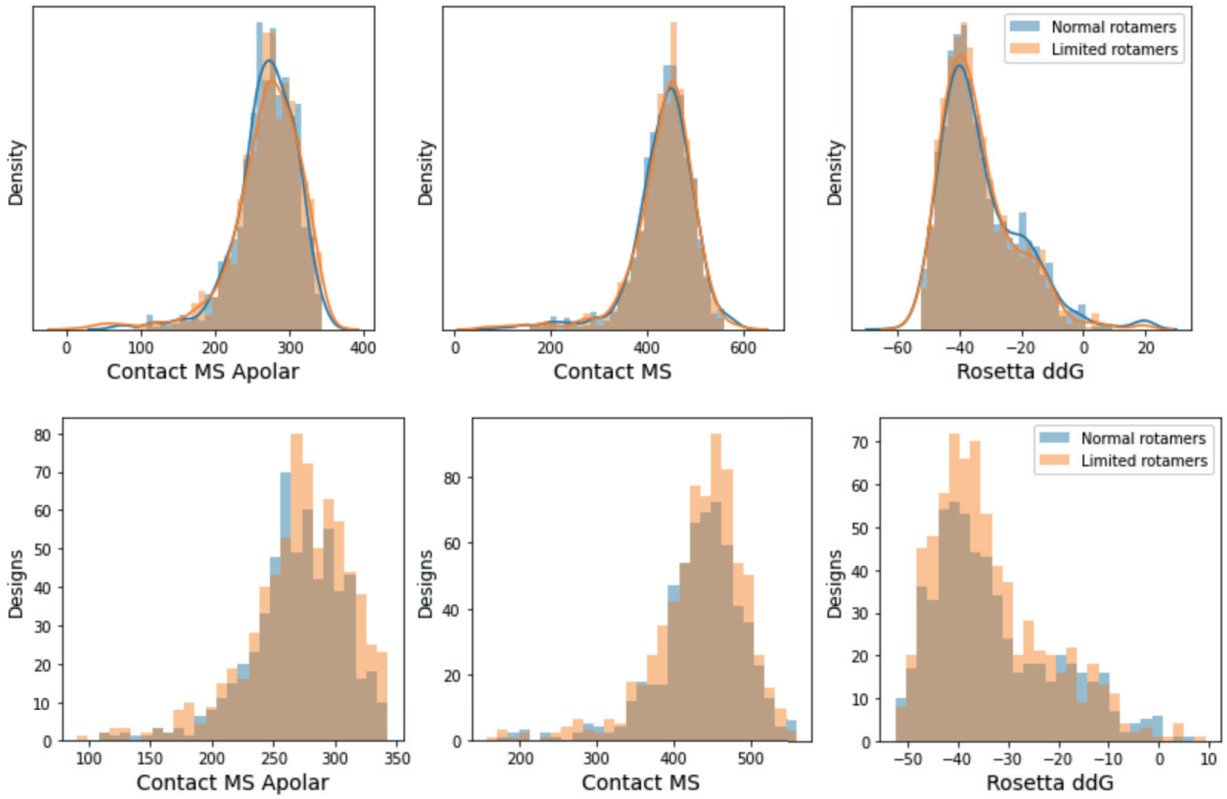

Fig 6.5.2.E: Effect of reduced rotamers: Here we see the results when either the standard rotamers are used, or the rare rotamers are pruned by setting all four -dunbrack_prob* flags to 0.8 against IL7 Receptor Alpha (PDB: 3DI3)[25]. The top graphs show that output-for-output, not much changed with the extra flags. However, since the limited rotamers run faster than the normal rotamers with runtimes of 260 and 320 seconds, by down sampling the outputs to constant CPU time, we see that the limited rotamers produce slightly more good outputs per CPU time.

A variety of other small factors were tried and accepted or rejected. Most of these deal with the tradeoff between runtime and sampling space. In terms of rotamers, Rosetta gives the option to allow extra rotamers at chi1 and chi2 that differ by slight perturbations. It was found that the extra chi1 rotamers were worth the extra runtime cost while the chi2 rotamers were not. There is another way to limit rotamers which is to restrict the number of base rotamers that get built (rotamers that differ by more than 30 degrees). When building rotamers, Rosetta accumulates the probability of seeing the rotamers it chooses, where choosing all rotamers leads to a probability of 1. By default, Rosetta uses between 0.87 and 0.98 as the cutoffs for various types of positions. If instead one sets all of the values to 0.80, the Rosetta Design calculations take half as long. It has been found that the resulting designs aren't as good, but the faster runtime makes up for the difference in a break-even fashion.

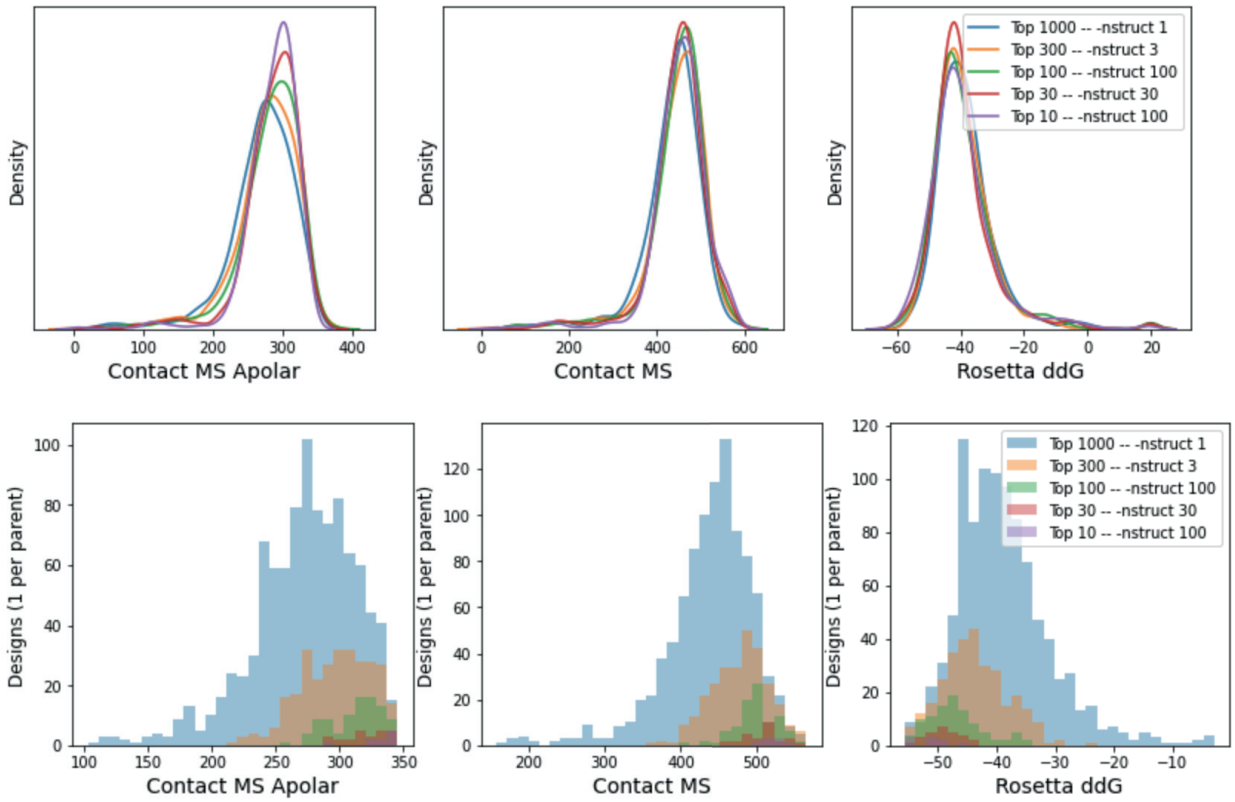

Fig 6.5.2.F: Effect of Rosetta Design repetitions vs docks: Here we see the results of performing many replicates of a few designs versus one replicate of many designs. RifDock outputs against IL7 Receptor Alpha (PDB: 3DI3)[25] were subjected to Rosetta Design. The top graphs show that output-for-output, that focusing efforts on only the very best outputs leads to the best results. However, if we instead limit the output to only the best member of a given -nstruct (bottom graphs), we see that spending all of the time on a few docks does not give the diversity we need. Based on these graphs, either -nstruct 1 or -nstruct 3 appears to be the winner.

The final question with Rosetta Design is how many replicates to perform. Experience shows that taking the best of 10 Rosetta Design trajectories is almost always better than performing 1 trajectory. However, things get more complicated when the alternative is to do 1 trajectory on 10 different docks. The decision to perform 1 trajectory on multiple docks was chosen in the hope that it's better to search through more docks than it is to hammer down on a few.

6.6. Scaffold design optimizations:

In order to design good binders, you need good scaffolds. At the outset of this project, we had a set of scaffolds created by Gabe Rocklin and Eva Strauch. Soon after, Longxing Cao made a set. Although these scaffolds were workable, they left much to be desired.

These scaffolds were all validated by the Protease Assay[12], which tests whether the protein is resistant to protease degradation. A critical oversight in this process is that just because the protein sequence is stable, does not mean that the computational model is correct. It is likely

that these early scaffolds set us back several months. As shown in *8.5. Scaffold topology correlations*, we quickly determined that certain topologies were never going to work. Manual inspection of the scaffolds also indicated that there was no way their structures could be correct. Longxing painstakingly cleaned out the scaffold set by hand. Meanwhile, more scaffolds were generated in 3-helical and 4-helical topologies.

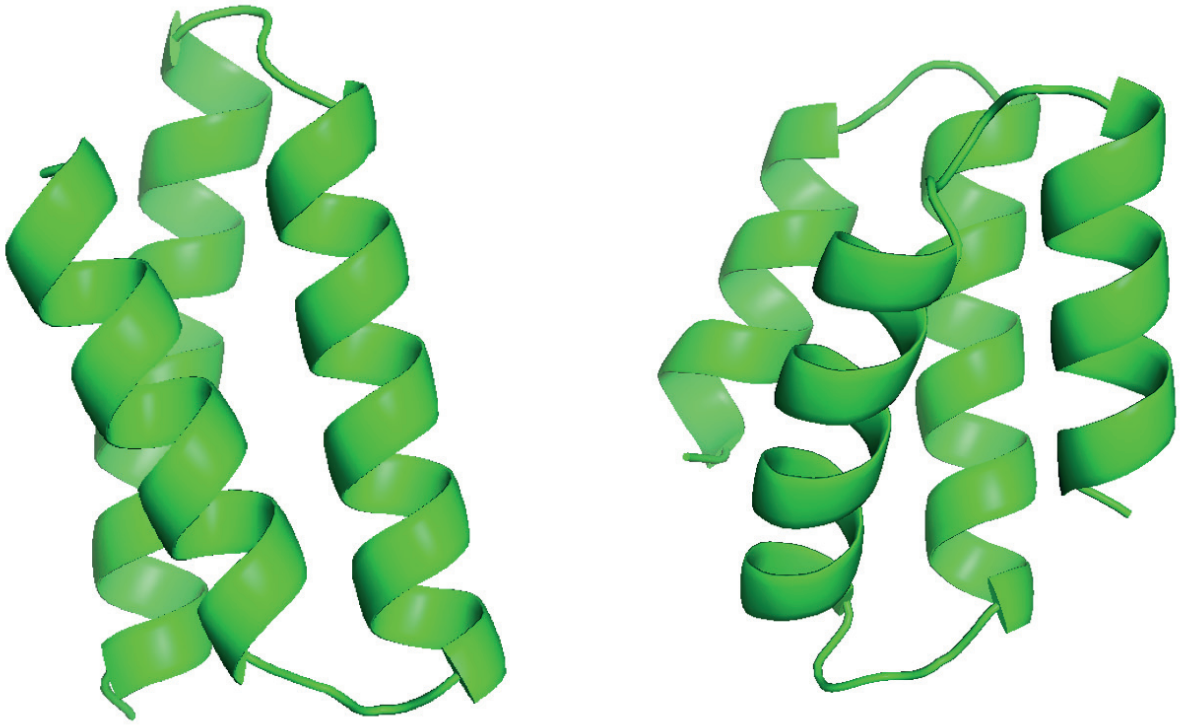

Fig 6.6.A: Example 3 and 4 helical bundles: These are what the 3 and 4 helical bundles being described look like. All arrangements of helices in both cases are enumerated.

The limiting factor in the scaffold sets at this time was the extremely slow algorithms used to make scaffolds. A fragment-insertion based method[33] was used to insert sets of 9-amino acid backbone angles into an extended protein chain. This would slowly fold the protein up into a well-folded protein. This approach has numerous problems. First and foremost, after a 1-minute trajectory, only 10% of the outputs would even look like a folded protein. Of those 10%, only a further 10% would have backbone complementarity worth designing. These inefficiencies led to a production rate of 1 usable scaffold backbone every 100 minutes. Standard procedure at this time was to then subject this scaffold to 5 rounds of Rosetta Design taking about 20 minutes to arrive at a final design. (Actually, Rosetta Design was run on the 10% of outputs that looked like folded proteins making this even less efficient). Then, if that wasn't slow enough, only 10% to 1% of the designed proteins actually passed the metrics. This meant that on average, 10-20 CPU hours were required per scaffold; 100-200 CPU hours at the more stringent cutoffs.

It was desired to fully enumerate the 3-helical and 4-helical topology spaces at 48-65 amino acids. “Full enumeration” depends on one’s definition of similarity, and based on the grafting experiment (*Fig 8.4.1.A: Success of grafted designs versus RMSD of graft*) would likely be the point at which every possible bundle exists at a 0.2Å RMSD threshold. While the coverage of these sets was never quantified, it was clear that millions of backbones would be needed rather than the tens of thousands that could be generated by the methods above. The solution was to speed up the backbone generation part and use tiers of filtering on the Rosetta Design stage.

6.6.1. Faster scaffold generation:

Inspired by the Worms protocol[34], a protocol was developed that stitched fully formed secondary structural elements together rather than trying to build them from phi and psi[33]. An important metric for scaffolds is known as worst9mer and represents the 9-amino acid segment in the scaffold that has the farthest CA RMSD from the Protein Data Bank[32]. The metric mostly identifies strange turns and tells you that this turn has never been seen before in nature. Scaffolds that come from the phi-psi insertion protocol often fail this metric because although the process starts with 9mers, it ends with 3mers, and can generate loops that are unlike anything seen in nature.

The worst9mer problem was easy to solve when using a pre-enumerated collection of turns. The faster scaffold generation method started by extracting all turns and helices from the previously validated scaffold sets. By extending the turns with ideal helices, it was possible to run the worst9mer metric, and poorly scoring turns (with worst RMSD > 0.4Å) were eliminated before they ever entered a scaffold. In this way, it wasn’t possible to build a scaffold with a poor worst9mer score because all components had been pre-validated. The turns and helices were then clustered to remove redundancy. The 4-helical bundles were the first set created by this method and used 9 helices and 274 turns. The 3-helical bundles came next and used 513 helices and 26,333 turns. In hindsight, the 4-helical bundles did not use enough turns or helices and need to be resampled.

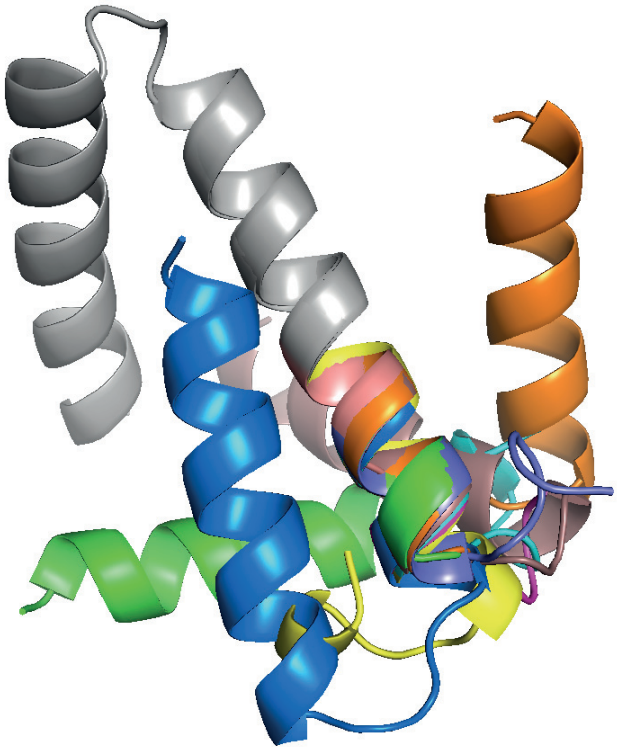

Fig 6.6.1.A: Helical worms representative picture: This picture was compiled from the start of a helical worms trajectory. The grey helices were built first, and then all of the colored helices were attempted. None of the colored helices were accepted.

Generating helical bundles from the helices and turns was accomplished by simply using a "growing worm" type procedure. The turns were all cut such that 4 helical residues remained on each terminus. These overlaps were screened for perfect RMSD alignment at every position along each of the helices. In this way, a database of all compatible grafts between the turns and every truncation of the helices was generated.

To generate a bundle, first a helix was placed at the origin. Then, at a random position along the helix, a turn was added with the remaining helix truncated. At the other end of the turn, a helix was added at a random position along its length. From here another turn was added and so on until a helical bundle of the desired topology was created. After each addition, the parts of the growing bundle were clash checked against one another and an addition was rejected if severe clashes were detected. If an unpromising worm was detected, or a successful bundle was output, a random rollback occurred removing several helices and turns in order to avoid redundant output.

Once a non-clashing bundle of the right topology was generated, a variety of metrics were used to determine if this bundle was likely to be worth designing. Simple metrics like the radius of gyration and maximum diameter could quickly weed out extremely bad bundles. Perhaps the

best metric for identifying sane-looking bundles is the fraction of sequence positions on the bundle that are classified as "core positions" by "side chain neighbors"[35]. Side chain neighbors uses a weighted-cone projected from the CA-CB vector of a residue to determine how many CA neighbors the residue has. Trial and error has dictated that a number of greater than 5.2 neighbors indicates a core position. By ensuring that at least 22% of the positions on the protein were designated as core, one can be fairly certain that a bundle forms an ordered structure rather than a disordered one with helices extending into space.

A slower, but more accurate measure was then used to ensure that sequence design would likely produce a well-packed structure. A motif hash[36] method was used to identify if good hydrophobic interactions could be realized between every pair of positions. By taking the 6D backbone orientations of two positions and looking into a hash table of all known hydrophobic interactions (extracted from the Protein Data Bank), one can determine if the backbone positions are oriented such that favorable interactions can exist. It was found that collapsing this metric into a Boolean value worked best where either an interaction exists or does not exist. The more naïve method used for the 4-helical bundles was to simply count the pairs of positions that had interactions and only use bundles with greater than X number of pairs. For the 3-helical bundles, a more sophisticated method was used, where the fraction of 7mer segments that had at least 1 match was counted. The more sophisticated method had the advantage that it would identify structure that contained "bad" segments because these segments would not have any motif hits. The summation method for the 4-helical bundles could allow a single bad helix to exist, as long as the other 3 helices packed well together.

This scaffold generation method was incredibly fast. Although written in python, the use of NumPy for vectorized operations[37], Numba for JIT compilation of python code to C++[38], and a custom protein representation called Npose[39] allowed the protocol to process as many as 20,000 splices per second. As many as 1,000 completed bundles were generated per second, and using the very strict filters above, this resulted in 1 very promising bundle per second. Comparing this to the 1 bundle per 100 minutes of the previous method, the new backbone generation method was roughly 6,000 times faster. Using a mere 7,000 CPU hours each, 25 million 3-helical and 4-helical bundle backbones were generated. (The 25 million 4-helical bundles were then screened with motif hash to bring the number down to 5 million. Only the 3-helical bundles had motif hash built into the generation script).

6.6.2. Faster/smarter scaffold design:

The 3-helical and 4-helical bundle campaigns had different target metrics. The 4-helical bundles came first and targeted Rosetta Score. The 3-helical bundles came second and targeted Secondary Structure Shape Complementarity as well as a highly favorable LDDT prediction from a neural network (unpublished, Deep Accuracy Net).

At this point, previous attempts at designing scaffolds would run 5 rounds of Rosetta Design on every backbone. While this method is simple and does not accidentally discard good structures, it is not very efficient because unpromising backbones are optimized unnecessarily. At the scale of 25 million, running 5 rounds of Rosetta Design on everything requires 8.3 million CPU hours. At this scale, optimization is critical.

Since Rosetta Score could be calculated from within the Rosetta Design procedure, a single protocol for the 4-helical bundles was developed. By using a pilot run, Rosetta Score cuts were developed that would remove bad designs while keeping good designs after every stage of design. After the very first invocation of the packer in the first round of Rosetta Design, 50% of the overall population was removed while only losing 5% of the good population. Using this same procedure at all stages in the protocol, by the final rounds, only 10% of the initial population was designed.

For the 3-helical bundles, the LDDT estimation from the neural network could not be performed from within Rosetta. Instead, it was noted that after 1 round of Rosetta Design, LDDT could be accurately estimated for the result after all 5 rounds. So, after a pilot study, all 25 million 3-helical bundles were designed once with Rosetta Design, the best 5 million selected, and those designed for an additional 4 rounds.

After design and filtering, 30,000 4-helical bundles remained, and 17,000 3-helical bundles remained. The 3-helical bundles faced much more stringent requirements, and if the 4-helical bundles are filtered with the same metrics, only 4,000 remain. These number are dramatically lower than the 25 million backbones generated and lend credence to the idea that well-packed monomer cores are hard to come by. Perhaps the filters were too stringent here (SecondaryStructureShapeComplementarity > 0.80 and LDDT > 0.9), but with so many unknowns in the protocol, it seemed like a good idea to ensure the scaffolds were of the highest quality.

See *8.6. Helical bundle protease assay results* for the exciting conclusion for the 4-helical bundles.

7. Data collection and processing

This section describes how the experimental data was collected and the best strategies developed for analyzing it.

7.1. Experimental data collection

For the binding experiments, the primary source of data is Next Generation sequencing data. This data represents all of the DNA sequences observed in a pool of yeast cells. By looking for known sequences and performing a summation, one can arrive at the “counts” of a given protein binder in a given pool.

These pools come from a Fluorescence Activated Cell Sorter (FACS) which decides whether to keep or reject each individual cell (and therefore each individual design). The binders contain a C-terminal c-myc tag that allows for labeling with a green fluorophore. Meanwhile, the target protein itself is labeled with a red fluorophore. Inside the FACS machine, each cell's red and green fluorescence is measured. Green cells have successfully expressed the protein, while green and red cells have also bound to the target protein. In this way, cells can be selected that bind to the target protein.

A typical experiment begins by transforming the 10,000 – 100,000 DNA variants obtained from a chip-synthesized oligonucleotide synthesis into yeast cells. The cells are then labeled and sorted by green fluorescence to identify cell lines that successfully express binders. From here, several consecutive sorts (with grow-ups in between) are performed with high target concentration to enrich the pool for binders that actually bind. The final step is to perform a series of sorts where the target protein concentration is systematically decreased. In this way, the strength of each binder can be observed by examining its fraction in the pool as the concentration decreases.

7.2. Enrichment versus Apparent KD:

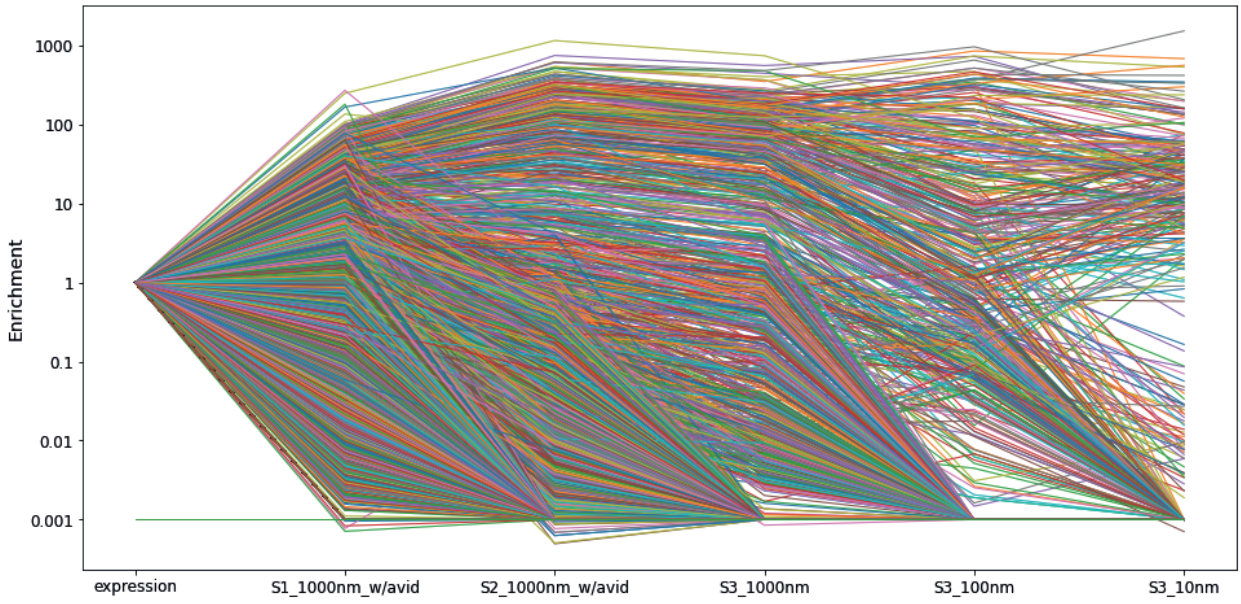

Fig 7.2.A: Enrichment of all designs for Insulin Receptor: Each line represents an individual design as it passed through the experimental conditions on the x-axis. The counts in each pool were divided by the number of counts in the expression pool to arrive at an enrichment. Most designs die out while only a few survive.

When examining the Next Generation Sequencing data, there are several ways to quantify the binding potential of each design. The standard procedure when the project started was to take the number of counts in one of the later titration rounds, and divide it by the number of counts in the expression round. This results in a distribution of "enrichments" that typically range from

0 to 1000. Most designs have a 0 enrichment because most designs disappeared; however, a few designs bind and enrich to very high levels. Standard procedure then was to call anything with an enrichment over 10 a “good binder” and anything over 2 a “binder”.

If one’s goal is simply to extract the best binders from a library, this procedure works just fine. The binders with the largest enrichment clearly beat out the rest of the pool and are the best binders. However, if one wishes to perform data analysis across multiple experiments, these enrichments have several serious problems.

The primary issue with enrichments is that they are by-definition relative. Consider two binding experiments, one in which most binders bind very tightly, and another in which none of the binders work. In the strong-binder experiment, there won’t be any binders that dramatically enrich because with so many other strong binders, all pools will end up with lots of binders. In the weak-binder experiment, there also won’t be any binders that dramatically enrich because almost nothing is being collected; binders will randomly get through the gate due to noise without any strong selection. When it comes time to analyze this data by enrichment, we’ll see similar patterns in both experiments with modest level of enrichments. What level of enrichment shall we pick to call “good binders”? The answer is that we cannot pick a single level. For the strong-binder experiment, there may be binders that have enrichments less than 1 that still bind strongly. However, for the weak-binder experiment, even selecting the binder with the highest enrichment is not enough.

There is a solution to the enrichment problem, and that is to look at the fraction of cells the FACS machine kept. In the strong binder case, we’ll see that the FACS machine was keeping nearly all of the cells. However, in the weak binder case, almost none of the cells were collected. Using this information and looking at the number of counts before and after each sort, it’s possible to determine the fraction of each binder that was accepted by the FACS machine. If these two experiments performed a sort at the same concentration, we could calculate these “survival fractions” and directly compare the two experiments. What we would see is that in the strong binder experiment, the cells were being kept at a much higher rate than the weak binder experiment, and we could directly compare their relative strengths. It’s also directly possible to classify binders using this approach. We could say for instance that any design with a survival fraction above 25% is a binder.

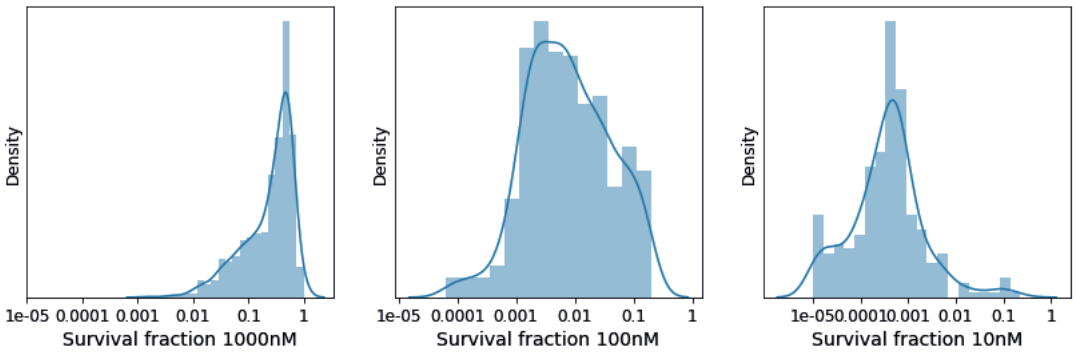

Fig 7.2.B: Example binding fractions for Insulin Receptor: These graphs show the survival fraction for each design in the 3 titration rounds. Only binders are shown. While no single design can be tracked here, since the majority of binders in this experiment were around 1000nM KD, we can see the population shift towards lower and lower fractions as the concentration decreases.

We can do better than binder-nonbinder classification. Using the definition of KD, we can directly convert a fraction bound to a KD value. If 25% of cells survive a 1µM sort, then we can say that the "Apparent KD of the yeast cell" is 3µM. (KD = concentration * (1 – fraction) / fraction). Further, by using all of the information obtained from the titration sorts, we can estimate the Apparent KD using several different concentrations and arrive at an average value with error bars for the binding strength. This is a good metric to quantify binding.

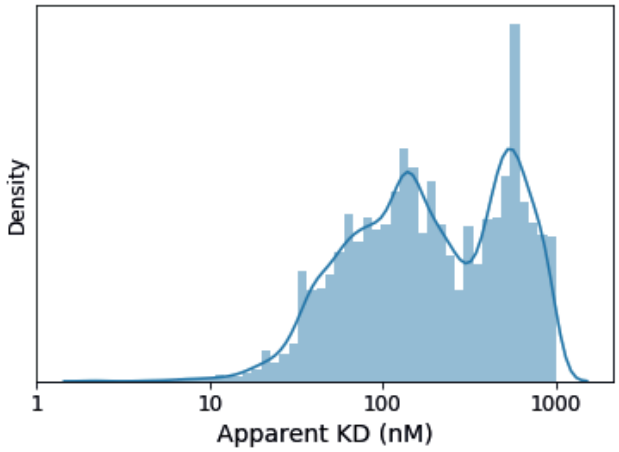

Fig 7.2.C: Apparent KD for Insulin Receptor: This graph shows the calculated Apparent KD values for all binders from Insulin Receptor. KD values above 1000 nM were truncated.

While various factors can affect the Apparent KD of a design, it is likely the most robust way to analyze the binding data. It falls victim to differences in expression rates and whole-pool competition effects, but without internal controls, it seems to be the most natural way to fit a number to the data.

7.3. Approaches to metric correlation:

Finding the best way to look for correlations in the data was a process that took over a year. It is tempting to simply produce a plot of "Apparent KD" versus a metric and look for a correlation. While this works to some degree, there are two problems. The first is the high degree of noise within the Apparent KD measurements. Although these numbers are likely the best way to quantify the data, the level of inaccuracy is severe. The second problem is that analyzing the data like this discards 99% of the data. In a typical binding experiment, only 0.1% - 1% of the binders actually bind. This means that for a given experiment, there may only be 10 – 1,000 sequences that have Apparent KDs. (These numbers were very near 10 for the first several experiments). With only 10 data points from a noisy experiment, drawing any conclusion is suspect.

A method was needed that looked at all of the data at once. The final solution was to instead classify the binders as binder or non-binders and to look at the fraction of binders versus a metric of interest. A typical approach would be to split the data into 10 groups at each of the 10th percentiles of a metric and to look at the success rate in each group. In this way, the massive number of nonbinders could be the denominator, while the binders show up in the numerator. This method has the advantage that correlations can easily be seen by eye if the binders cluster to one part of the graph.

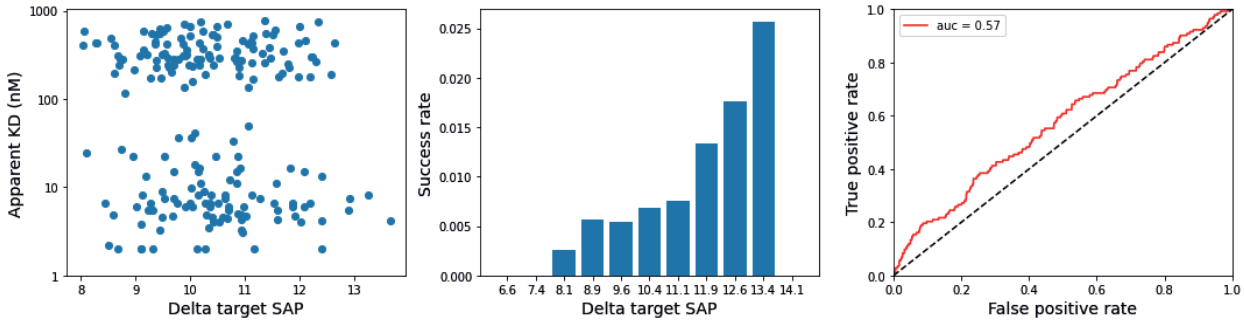

Fig 7.3.A: The three ways to plot correlations: Here we see the correlation between Delta Target SAP and binding for PD-L1 binder. While it is tempting to make a scatter plot, only ~ 100 / 25,000 proteins showed binding signal. Looking only at the scatter plot, one would assume no correlation; however, looking at the other two, we see predictive power.

A quantitative value may be assigned to such a correlation by using the Area Under the Curve (AUC) of a Receiver Operating Characteristic (ROC). In this method, the metric is used as a classifier to decide whether a design will bind or not. If the metric identifies a large number of binders before non-binders, the AUC will be greater than 0.5 (with a max a 1.0). If the metric has no separation power, it will identify binders and non-binders at an equal rate and produce an AUC of 0.5.

Although the AUC value provides a quantitative number, it cannot be used alone to analyze the data. The percentile graphs are needed to identify metrics where there is a "happy medium".

Some metrics cannot be too high, or too low, and for these cases, the AUC would be equivalent to a random metric, even though there is strong signal to the data.

8. In-depth results:

Here we will discuss all of the findings that originated from experiments in the wet-lab. These experiments were performed by a variety of people including: Longxing Cao, Inna Goreshnik, Samer Halabiya, Aza Allen, Cami Cordray, and Buwei Huang.

8.1. Success by target:

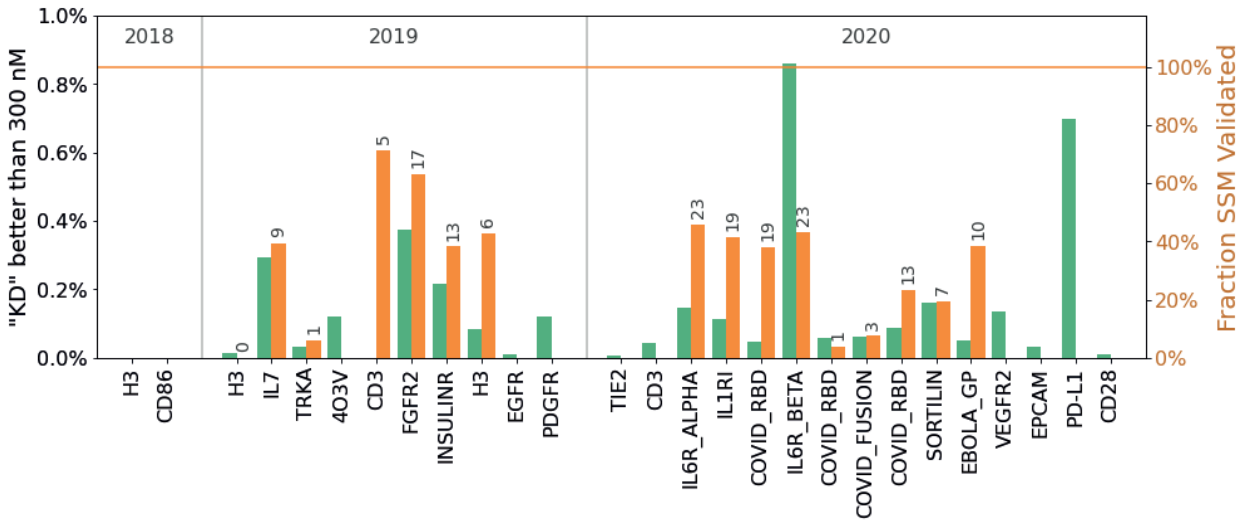

Fig 8.1.A: Full experimental summary graph: This graph shows the success rate in terms of Number of Binders (green) and Fraction SSM Validated (orange). It's difficult to use this graph to say whether the success rate increased or decreased with time because every target is different (and multiple campaigns against the same target often targeted different locations).

Binders were attempted for a variety of targets, and for nearly every target, successful binders were identified. While success rates of 0.1% are depressing, from the biological standpoint, only 1 binder is needed for therapeutics. A good portion of the success rate of this graph can be explained by Delta Target (*Fig 8.2.1.B: Target success rate vs max Target Delta SAP*).

8.2. Initial screen metric correlations:

This section is devoted to the findings obtained from the initial library screens where 10,000 – 100,000 individual designs were tested at once against a target protein. These correlations help to identify which binders to order and which proteins to target. Unfortunately, there is very little validation in this dataset. The vast majority of binders identified in these screens remain a single data-point from a single experiment; they were not followed up, replicated, or verified. As such,

a vast majority of the following data is likely to be composed of false positives; that is, protein sequences that appeared to bind, but whose design model does not reflect the experimental reality. “Seeing through the noise” is required here, and a bit of intuition is required to decide whether a correlation is real or not.

8.2.1. Hydrophobicity metrics:

Although in hindsight it should come as no surprise, metrics involving the level of exposed hydrophobicity showed the strongest correlations. The most important of the hydrophobicity measures is the level of exposed hydrophobic on the target that were buried upon complexation. (See *6.1.1. Delta SAP score* for details).

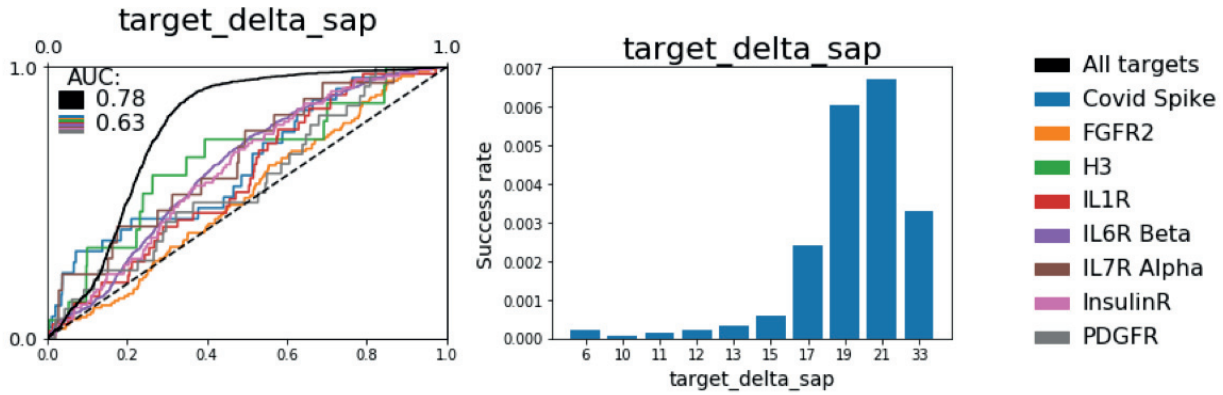

Fig 8.2.1.A: Correlation of Target Delta SAP with experimental data: Shown are the correlations between Target Delta SAP and binder success rate for: All pooled data (Black) or each individual target (colors). The AUCs represent the AUC of the black line, or the mean of the color lines. The black line indicates the power of the metric to differentiate between targets, whereas the colored lines indicate one’s expected outcome when making binders against a single target. The right graph is an alternate representation of the black line showing the metric binned at the 0-10th percentile, 10th-20th percentile, and so on. Only data from 3-helical bundles were plotted to avoid spurious correlations that prefer 3-helical bundles to 4-helical bundles (because 3-helical bundles have a 3x higher success rate).

The overall correlation here suggests that the differing levels of hydrophobicity lead to different success rates on different targets based on the level of hydrophobicity. The correlation on a per-target basis is remarkable, however, because it indicates that the specific amount of hydrophobicity covered on a specific target is very important (and also suggests that the binders bind specifically and not randomly to the target). If the per-target correlation did not exist, we could argue that the binders stick randomly to the target, however, this shows that the specific binding mode is important. The overall correlation is so strong however that the following plot can be made.

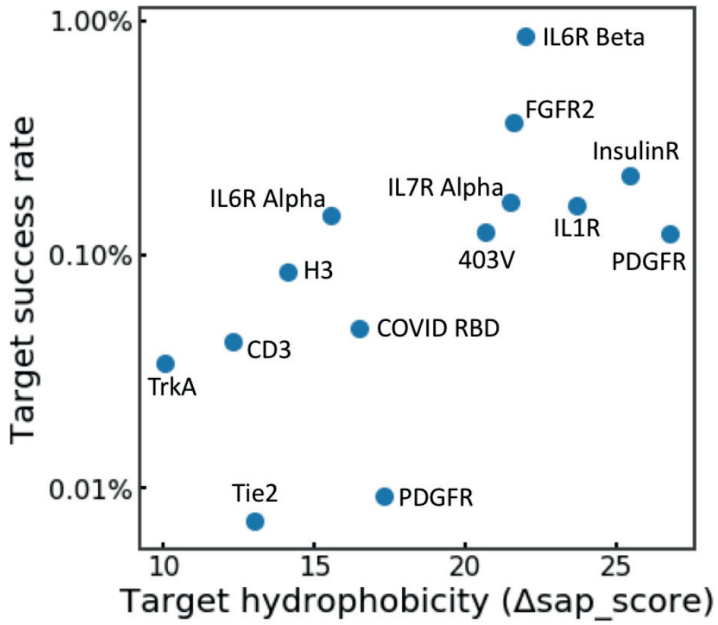

Fig 8.2.1.B: Target success rate vs max Target Delta SAP: This graph shows that the SAP score alone can predict the success rate of a target. Importantly, this value can be precomputed before design begins. Two caveats with this graph are that the quality of the ordered binders improved as time went on that only a small fraction of the binders achieved the given Delta Target SAP. If one can put all binders at the max Delta Target SAP for a target using the information in this dissertation, success rates may be 10-fold higher than what is shown.

This plot can serve as a guide to target feasibility. We can see that the most successful experiments buried the most SAP on the target. There are a few caveats to this plot, like the fact that these experiments were produced at different points in time (as the methods got better) and that only a very small fraction of the binders actually achieved the listed delta SAP score. However, even so, this graph can serve as a starting point for deciding whether or not to start a binding experiment.

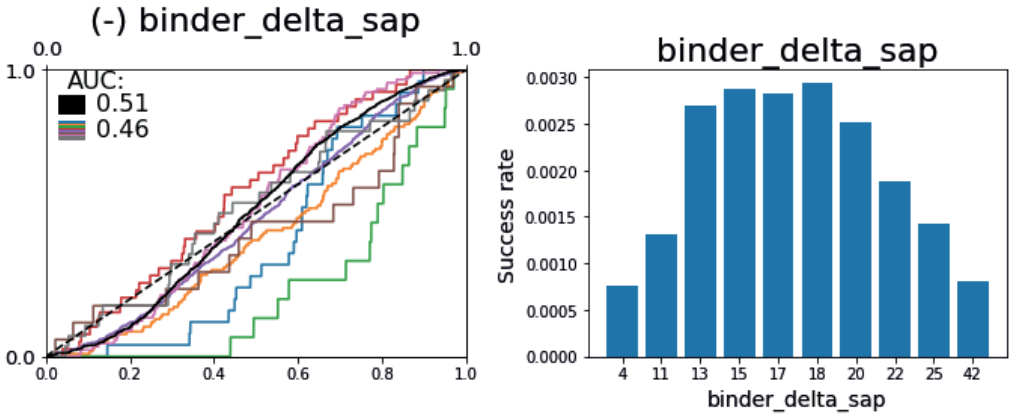

Fig 8.2.1.C: Correlation of Binder Delta SAP with experimental data: See *Fig 8.2.1.A*

On the binder side, we see a much different trend. No longer is ever-increasing hydrophobicity a good thing; instead, we see a sweet spot for the binders. Various explanations exist for this curve, but the most likely one is as follows: At high Binder Delta SAP, the non-specific behavior of the binders dominate their behavior. They likely stick to themselves forming homodimers and to other proteins. This limits their availability to serve as protein binders. At low Binder Delta SAP, there is simply not enough driving force for the binders to stick to something. Their well solvated surface would rather be exposed to solvent than to another protein.

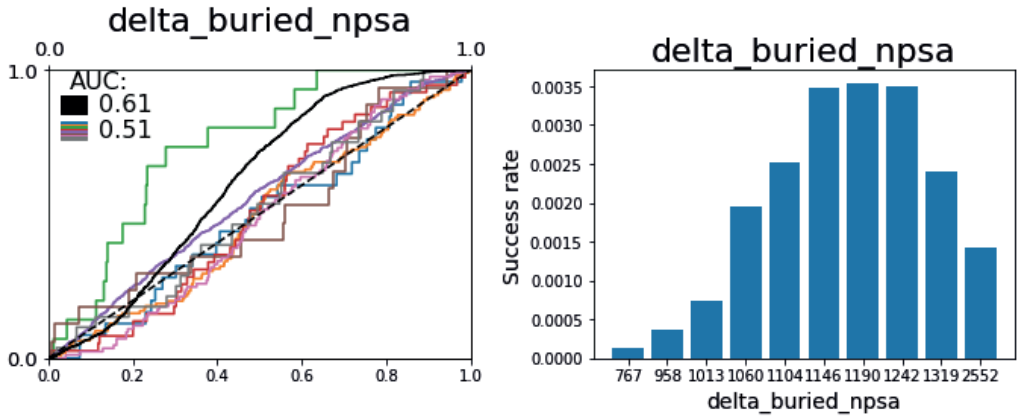

Fig 8.2.1.D: Correlation of Delta Buried Non-Polar Surface Area with experimental data: See *Fig 8.2.1.A*

Looking back at the older method of calculating the change in nonpolar surface area, we see a superposition of the Binder and Target Delta SAP graphs. Using this metric alone would lead to complications because it doesn't separate binder and target side. While it is possible to use complicated tricks to allow SASA to measure each side individually, the new metrics of SAP and CMS do a far better job.

8.2.2. Packing measures:

The second most important feature discovered for interface formation is the packing of the interface. Well packed interfaces featuring large areas of close contacts appear to do best. It is tempting to try to use Delta SASA and SC for this task, but *6.2.1. Problems with Shape Complementarity and Delta SASA* explains why this isn't a good idea.

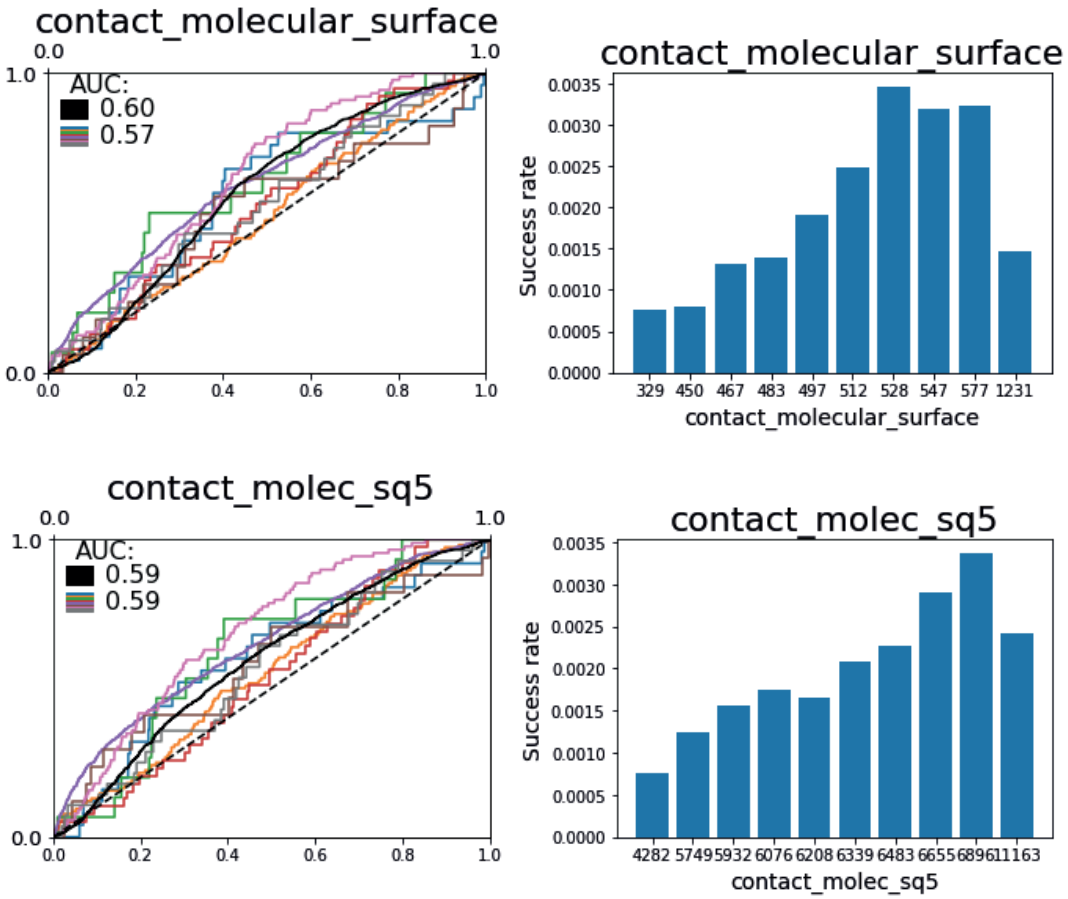

Fig 8.2.2.A: Correlation of CMS and density variant with experimental data: See *Fig 8.2.1.A*

CMS and the local density variant show some ability to differentiate weak from binders indicating that more contacts are better. On some targets, the local density variant performs better, but it is hard to say with certainty that it is indeed better. An important caveat is that on later targets, these metrics were filter heavily (which can remove some correlation).

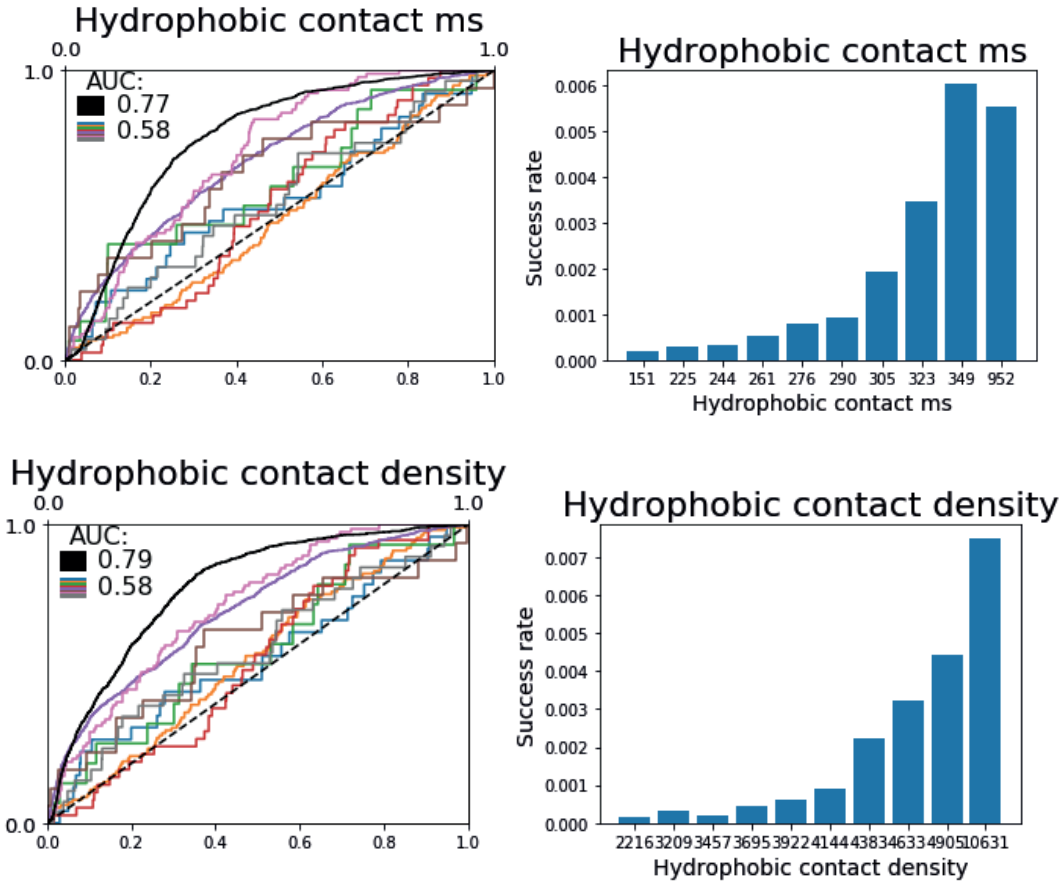

Fig 8.2.2.B: Correlation of Apolar CMS and density variant with experimental data: See *Fig 8.2.1.A*

Only allowing CMS and the density variant to look at hydrophobic atoms greatly increases their predictive power across targets. It is difficult to separate the effects of overall hydrophobicity with specific hydrophobic targeting, however, the unchanged individual contribution as well as in increase in overall AUC to that of the Delta SAP score indicates that perhaps, this is simply another way to capture hydrophobicity. An important caveat is that on later targets, these metrics were filter heavily (which can remove some correlation).

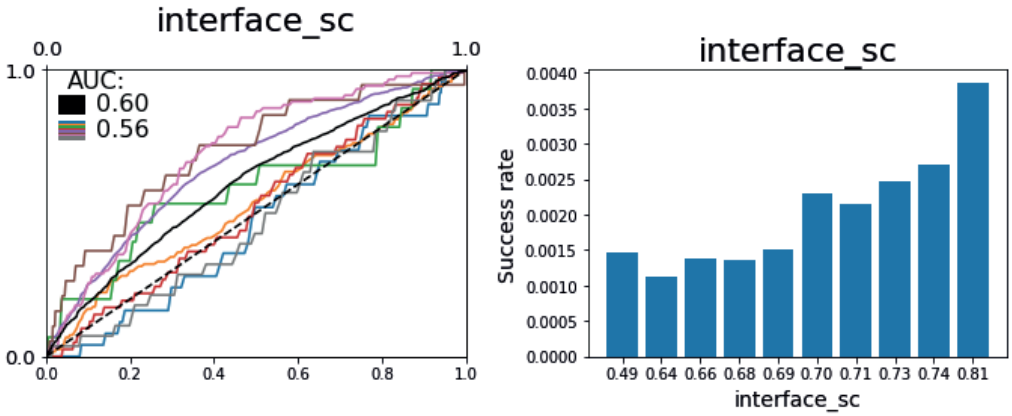

Fig 8.2.2.C: Correlation of Shape Complementarity with experimental data: See *Fig 8.2.1.A*

It is of little surprise that SC[SC] shows correlation with the data. Better docks tend to have better SC. Additionally, SC was not used as a filter on these designs, so a larger range of values is available than for the other packing terms. However, as noted in *6.2.1. Problems with Shape Complementarity and Delta SASA*, using SC as a filter has its problems.

8.2.3. Rosetta Score metrics:

Rosetta has several metrics that seem important for interface design. Most notably is the calculated ddG which seeks to estimate the real ddG of a binder.

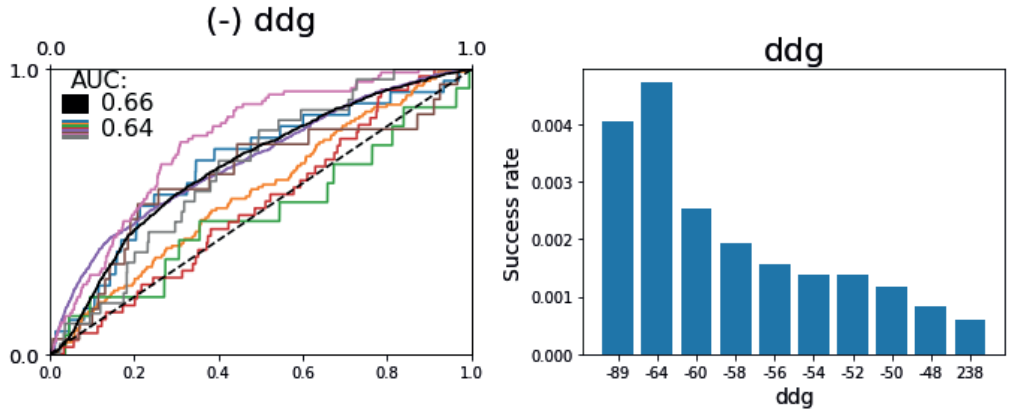

Fig 8.2.3.A: Correlation of Rosetta ddG with experimental data: See *Fig 8.2.1.A*. ddG calculated with repacking unbound. Not repacking gives similar correlations.

Indeed, Rosetta ddG does a fairly good job at predicting the data, achieving the highest per-target correlation of any metric. There is a slight caveat here in that Rosetta ddG was not used as a filter in some of the later rounds, but nonetheless, it appears to be a good filter.

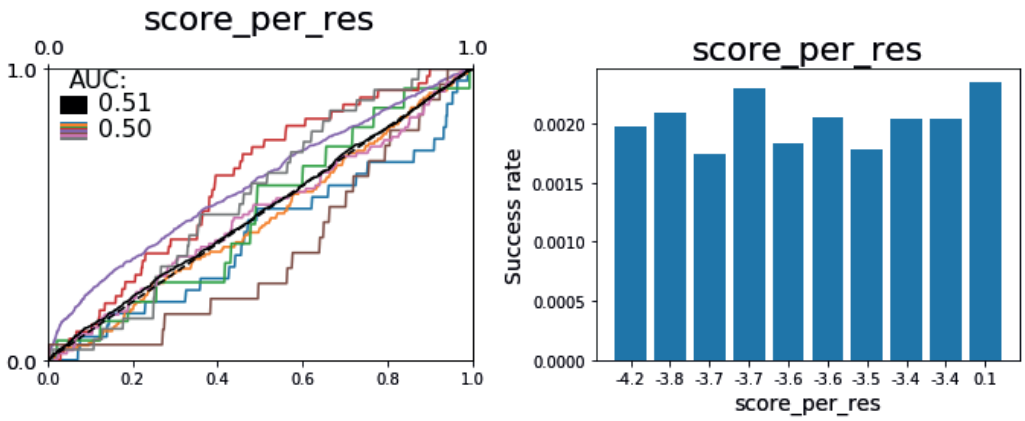

Fig 8.2.3.B: Correlation of score_per_res experimental data: See *Fig 8.2.1.A*. score_per_res calculated in the monomer state.

Rosetta score divided by the number of residues provides an estimate of Rosetta's assessment of the quality of a monomer. Although this metric is useful in predicting the effects of point-mutants during an SSM, it is not a good discriminator between different scaffolds.

8.2.4. Buried unsats:

Buried unsatisfied polar atoms should be a bad thing in interface design. Arguments can be made for why they cause energetic penalties; however, no way of quantifying them that we have tried has shown any correlation.

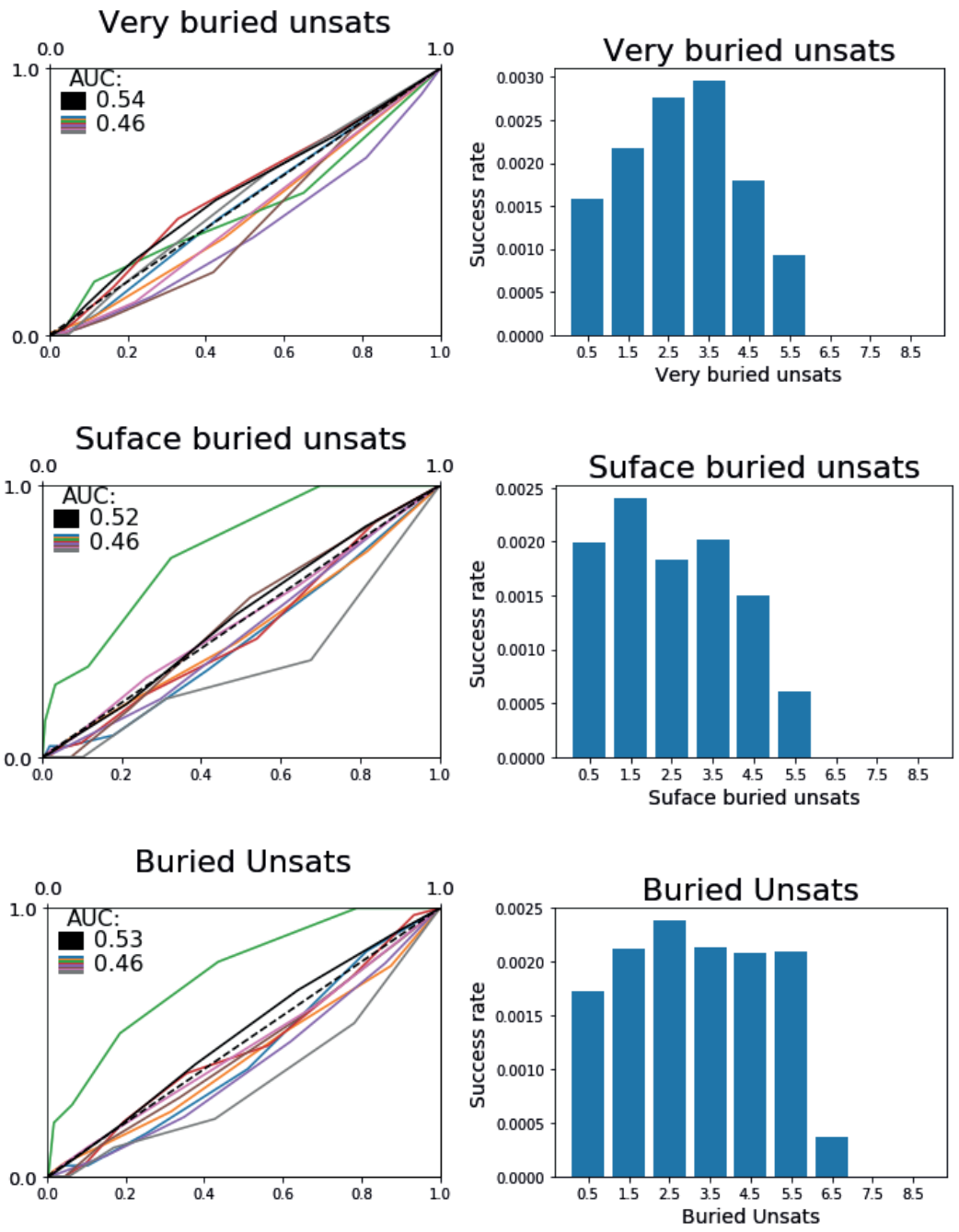

Fig 8.2.4.A: Correlation of VBUNS, SBUNS, and BUNS with experimental data: See *Fig 8.2.1.A*. Very buried unsats use the atomic depth method to mark out a region of space deeper than 5.5Å from the molecular surface. Anything inside this range is considered buried even if near a water-sized hole. Surface buried unsats are anything less deep that 5.5Å and use the SASA method.

Why is there no correlation with buried unsats? Perhaps the best explanation is that we do not consider explicit water. Buried water molecules may contact buried polar atoms by rearranging the local packing to make contacts. Without modeling this, and with the low-resolution nature of our binders, it may not be possible to tease this out. Additionally, the

approximate_buried_unsat_penalty is used during design. Perhaps this is biasing the designs by eliminating buried unsats such that we never see truly egregious ones.

8.2.5. Net charge:

Net charge was a metric that was not seriously considered until it became the dominant predictor of binding signal.

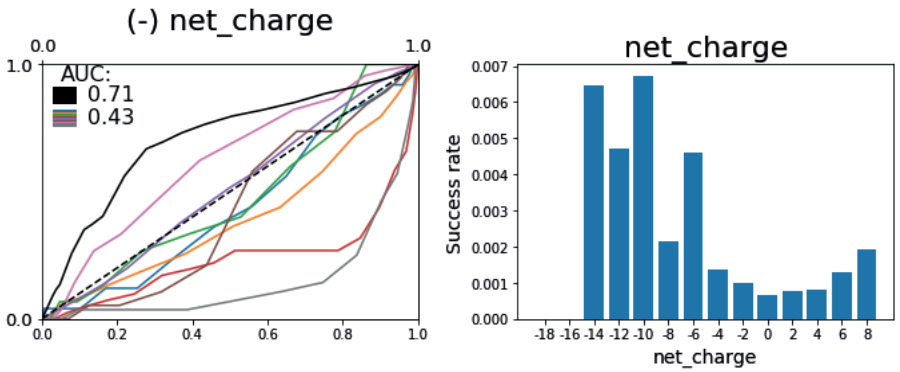

Fig 8.2.5.A: Correlation of net charge with experimental data: See Fig 8.2.1.A

There is clearly very strong signal here, but what does the signal mean? More information can be found in *8.3.4. SSM net charge sweeps*, but for the initial screen case, it would appear that having some level of net charge, either positive of negative, helps with binding. The best explanation could be that having net charge makes the binders more likely to be monomers and less likely to stick to other proteins. On further investigation, it was found that positively charged binders don't seem to express very well and might be misbehaved. It has been decided that going forward, all binders are designed using a roughly -7 net charge.

8.2.6. Other metrics:

A few more correlations are needed to tie up loose ends.

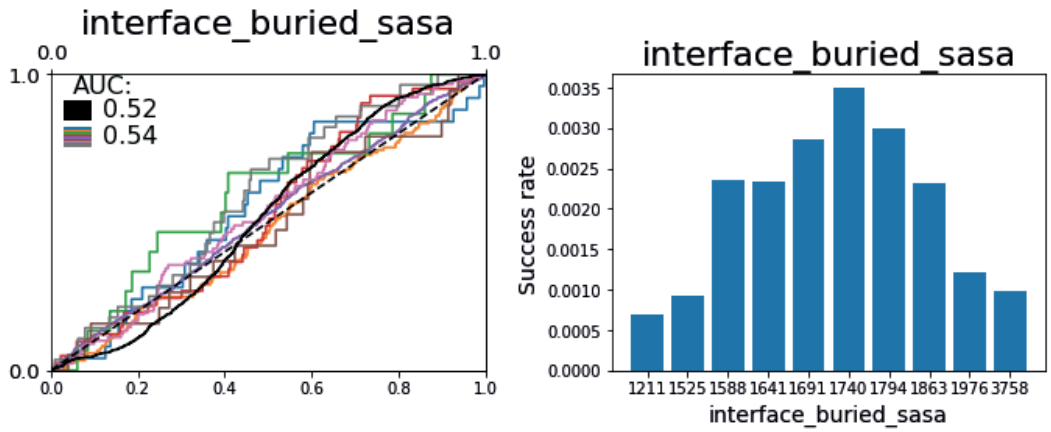

Fig 8.2.6.A: Correlation of Delta SASA with experimental data: See *Fig 8.2.1.A*

This graph serves purely to show that maxing out Delta SASA does not lead to better binder. This was actually attempted, and the results are that the issues with the specific Delta SASA calculation appear and large terrible interfaces are designed. CMS can also be maxed out, but on the whole, bigger interfaces aren't always better.

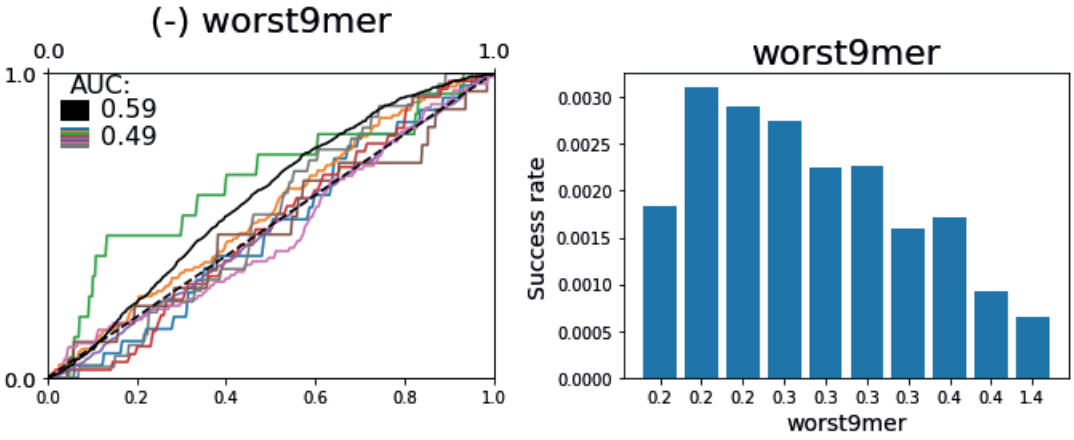

Fig 8.2.6.B: Correlation of worst9mer with experimental data: See *Fig 8.2.1.A*

Worst9mer is a sequence-independent measure of backbone quality that asks what 9-amino acid stretch has the worst RMSD to the Protein Data Bank[32]. Higher numbers indicate that a fragment is novel (which isn't a good thing). Although there is little correlation in this graph, this metric proved to be invaluable. Heavy filtering resulted in nearly all of the binders being below 0.4Å on this metric. This removed any correlation that would be here, however, it is strongly advised to use this for all future scaffolds. (Beta-sheet containing scaffolds seem to like a cut of around 0.7Å).

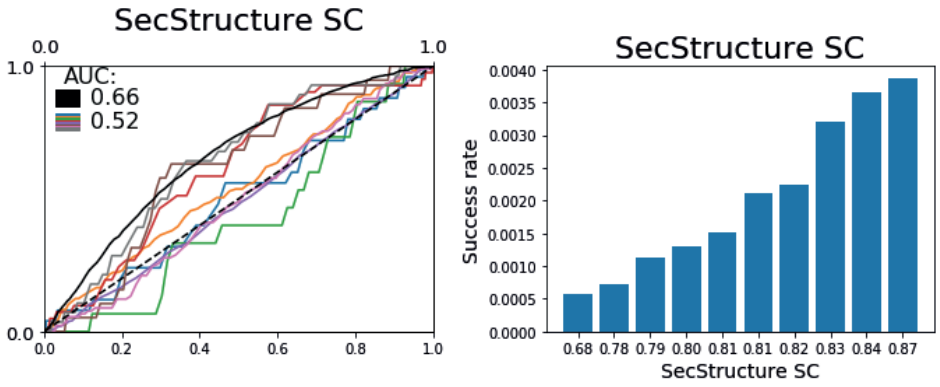

Fig 8.2.6.C: Correlation of SecondaryStructureShapeComplementarity with experimental data: See *Fig 8.2.1.A*

SecondaryStructureShapeComplementarity (ss_sc) is another metric that was only observed after it repeatedly showed correlation with experimental data. Although the correlation is not especially strong here, this is one of the few monomer packing quality metrics that ever shows correlation. A caveat is that this was eventually used as a filter in the later experiments which may hurt the correlation. Additionally, the numbers shown above were computed after cartesian FastRelax which artificially inflates the numbers. Cutting at 0.80 seems to be the right tradeoff for non-cartesian designs.

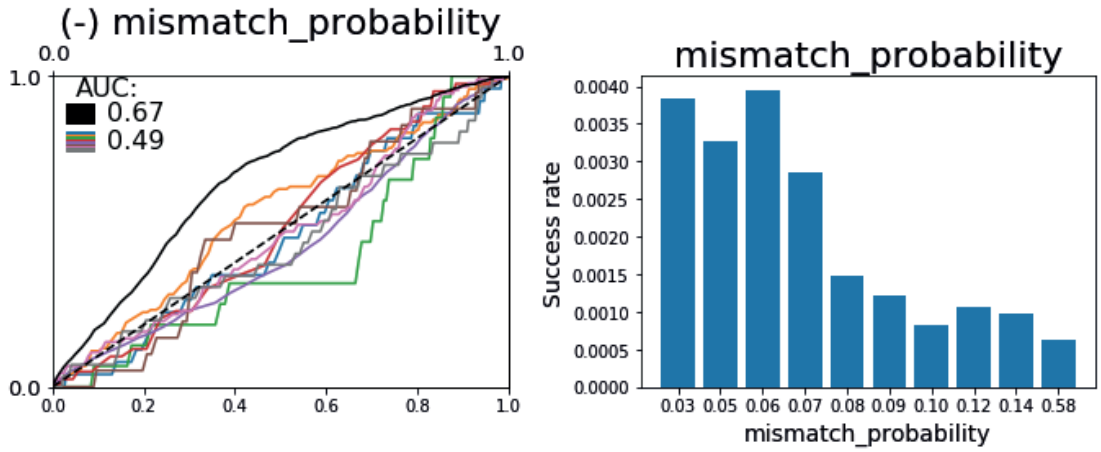

Fig 8.2.6.D: Correlation of PsiPred[40] Mismatch Probability with experimental data: See *Fig 8.2.1.A*

PsiPred[40] can be used to calculate the probability that a given sequence will adopt its given secondary structure. By multiplying the probabilities that this sequence will fold into something else, one arrives at the Mismatch Probability. The correlation above again suffers from the fact that this metric is heavily filtered. A cutoff of 0.15 seems to work well for helices while beta sheets may require higher cutoffs.

8.3. SSM results:

Validating a result from the initial screen experiments is challenging. The perfect validation is a crystal structure of the binder in complex with the target, but this is extremely low throughput and takes months. Expression in E. coli and testing by Biolayer Interferometry is a good test that the sequence does indeed bind, but does not provide any structural validation of the binder or binding mode.

Instead, a method was devised called Site Saturation Mutagenesis (SSM) where every mutation is made at every position along the binder. A yeast library may then be prepared containing every mutant for several different designs. By performing the standard sorting experiments and quantifying the effect of each mutation, one may arrive at a better picture of whether or not the design model is correct.

8.3.1. SSM validation process:

With the experimental results of an SSM experiment in hand, one can try to determine whether or not a binder is “real” by careful examination of the mutations. At the time of writing, this process is still very manual and subjective. Efforts to objectify this process are difficult, because in some ways, they require one to fully understand interface energetics.

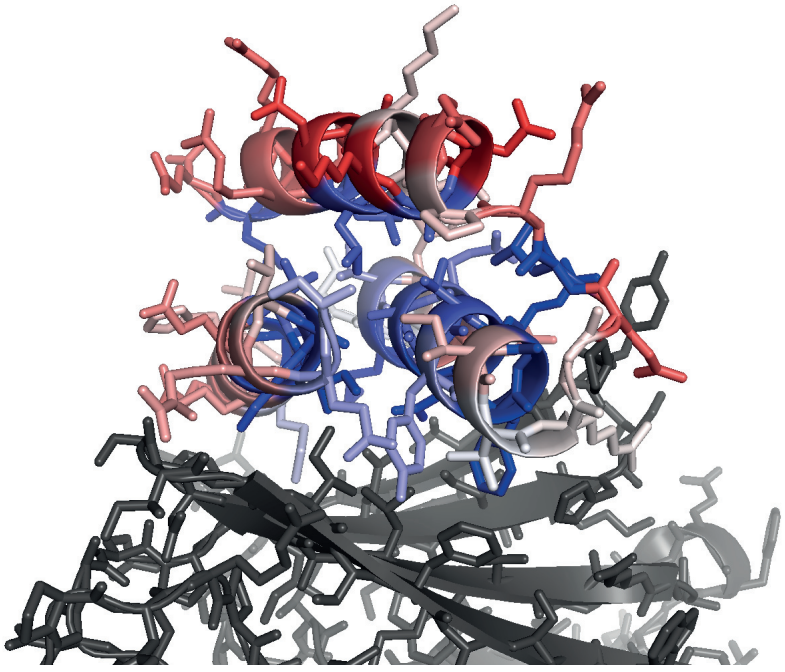

Fig 8.3.1.A: Example SSM result colored by Shannon Entropy: A FGFR2 (PDB: 1EV2)[41] binder is shown colored by the Shannon Entropy of the experimental sequencing counts of each position. Blue means conserved while red means variable. The conservation of the core and variability of the surface lead to confidence that this binder may be correct.

The most straightforward metric is to color the design model the Shannon Entropy of the sequencing data at each position. Here, positions with more allowable mutations appear red while fewer mutations appear blue. One can then make the argument that positions with few mutations must be “critical” to the function because they cannot be mutated whereas positions that allow many mutations do not matter. While these assumptions are not perfectly correct, this analysis can give an overall “feeling” to a design. If the core and interface are conserved while the surface is variable, there’s a good chance the binder is well folded and using its designed interface to bind. If the whole binder is conserved or variable, it’s likely a misfolded protein or a disordered protein.

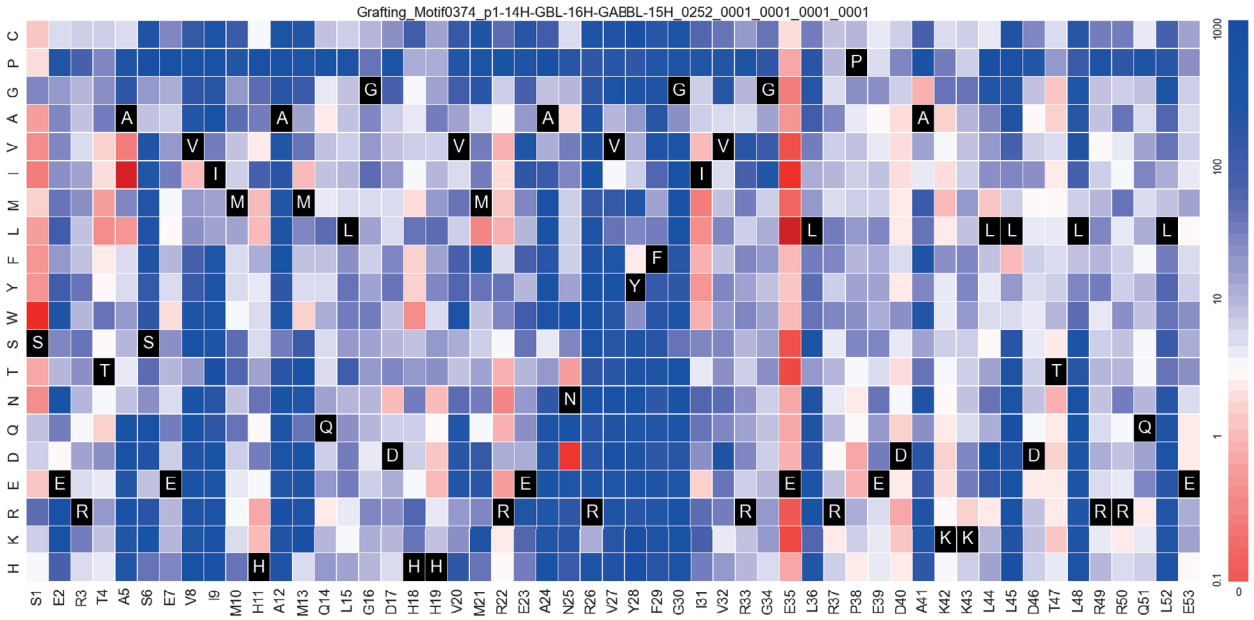

Fig 8.3.1.B: SSM effect graph: This graph shows the full results of an SSM experiment for the same binder pictured above. Blue mutations hurt binding while red mutations enhanced binding. The Apparent KD of each mutation is shown on the colorbar on the right. After looking at many of these graphs, one can decide whether a model is likely to be good or bad simply from the patterns shown. For instance, this graph clearly depicts a 3-helical bundle with the right helix as the upper helix (since only the core positions matter). Then, of the left two helices, clearly the middle one is the most important.

The second bit of information one can use is the specific mutations that are allowed or disallowed. This information is the only way to actually validate that the binder is sticking to the correct place on the target; just looking at the overall entropy pattern simply tells you that the binder is folded and using its designed interface. By examining the pattern of small to large mutations, or polar to apolar mutations, one can again get a “feeling” for whether or not the design model looks consistent with the data. Efforts have been made to quantify this process, however, without a full understanding of interface energetics, this process is very difficult.

After the sequence entropy and specific mutations have been observed, it is up to the designer to decide whether or not the binder is real. This process is obviously highly subjective and

different opinions can be cast depending on one's desired outcome. For the analysis here, a skeptical attitude was attempted where it is better to throw away a good binder than to accept a bad one; however, mistakes were certainly made in both directions.

8.3.2. SSM validation rates:

When one looks at SSM plots, one can usually categorize the results into a few categories very quickly: A) This design might be real. B) This binds for the wrong reasons C) This protein doesn't actually bind at all. It is only the binders from A) that are even important; however, more binders get categorized in categories B) and C) than one may expect.

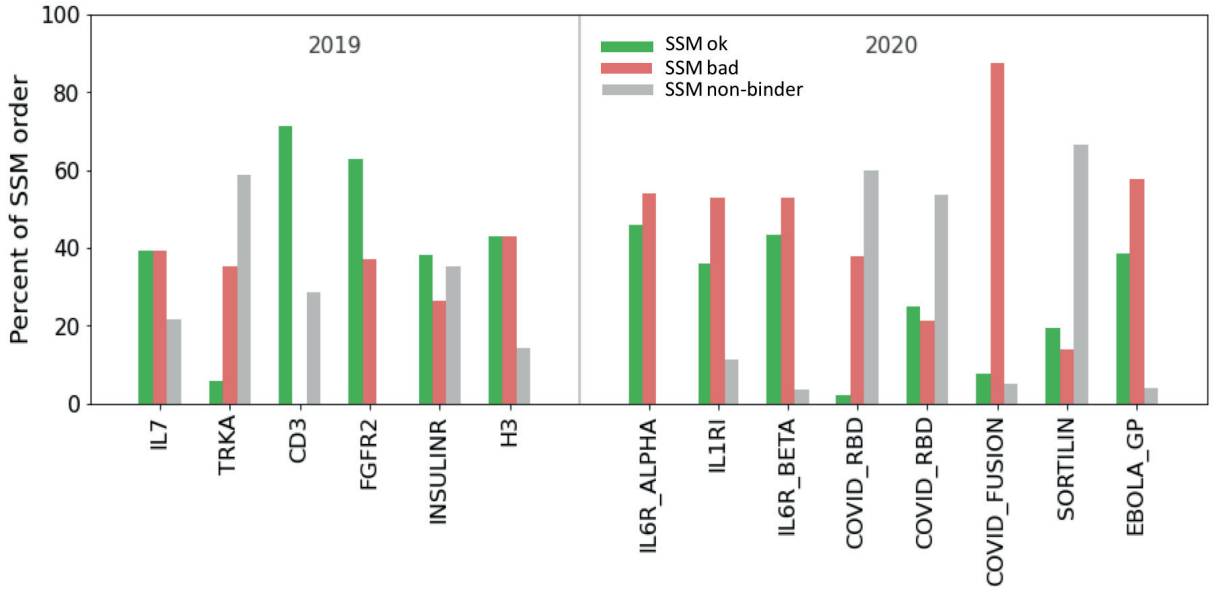

Fig 8.3.2.A: Validation rates of SSM experiments: The classification of the SSM results into the 3 categories above are shown for all SSM experiments. The designs in green (A)), look good enough that the might be correct. Designs in red (B)), appear to bind but do not appear to be correct by SSM analysis. Finally, designs in grey (C)), don't bind and shouldn't have been ordered.

Serious attempts to figure out what the proteins in category B) have not been attempted. Some of these are disordered proteins or are clearly misfolded. Structure prediction and docking prediction would simply be the start of such an analysis.

8.3.3. SSM non-binders:

The nonbinders (category C) above) are perhaps the strangest of all the categories. These designs should not have been classified as binders in the initial screen library. In an effort to stop ordering these, the following plot was made:

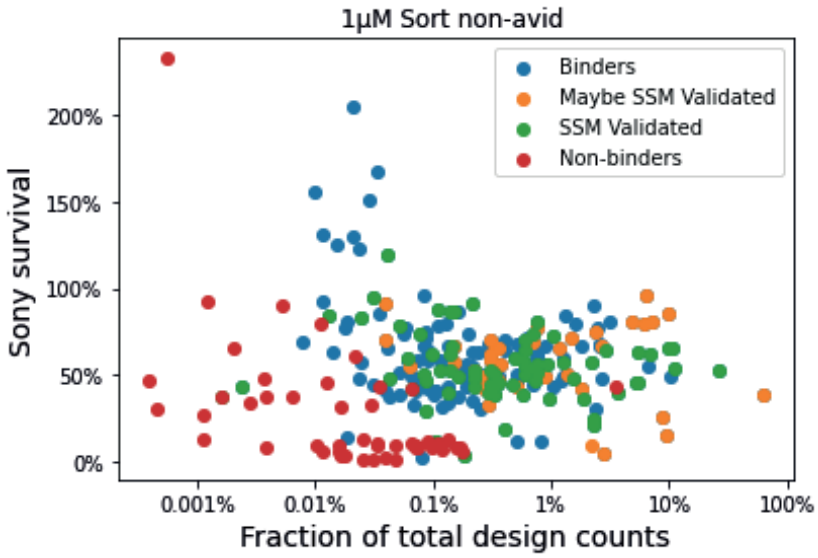

Fig 8.3.3.A: Initial screen results of designs ordered for SSM: This graph shows the initial screen characteristics of all of the designs that were ordered for SSM. The Non-binders in this graph are the concerning points because they should not have been ordered. It is clear that with a few simply heuristics, the majority can be eliminated, however, the few mixed in with the other populations are harder to differentiate.

This plot can clearly divide the non-binders into two categories: binders that were sorted on the edge of noise and binders that looked like strong binders. These edge-of-noise binders are rather straightforward to identify when plotted like this and lead to a healthy skepticism of designs with low counts. What about the designs that look like real binders though?

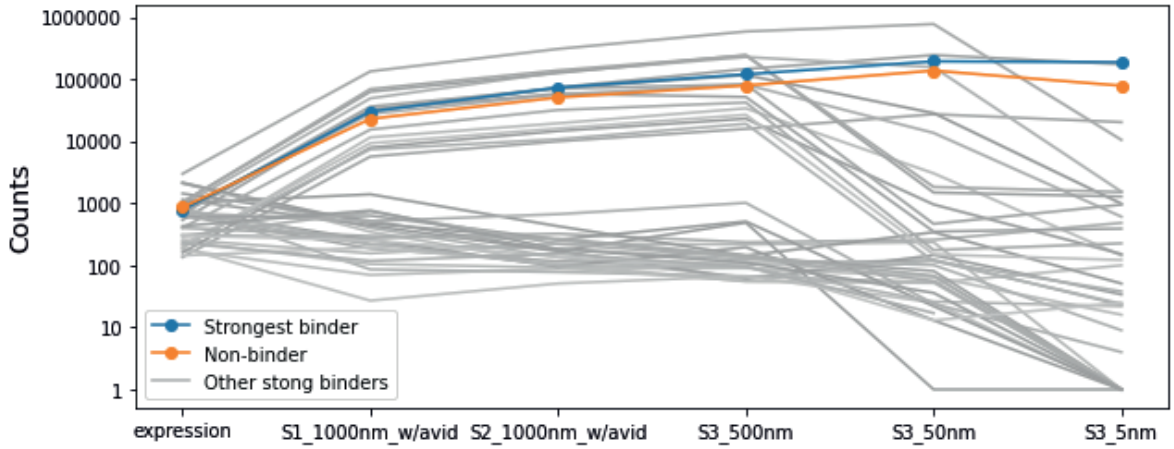

Fig 8.3.3.B: Potential hitch-hiker design against IL7 Receptor Alpha: This graph shows the enrichment traces of the top 40 designs against IL7 Receptor Alpha. The very best performing design is in blue, and a design from category C) in orange. This orange design showed no binding in later experiments even though it looks very good here. A potential explanation is that the blue binder contained two plasmids and carried the orange binder along through the sorts.

While this may not be the only explanation of these data points, it would appear that a “hitch-hiker” was caught red-handed in this experiment. The hypothesis is that a yeast cell contained two plasmids: one of a very strong binder and another of a non-binder. The strong binder carried the non-binder’s DNA through all of the experiments making it appear to be a strong binder. The evidence for this is the strong correlation between the counts of the binder and the “hitch-hiker”. Issues like this can easily be resolved with replicate experiments using separate transformations, but it remains a source of error for all of the data collected in this project.

8.3.4. SSM net charge sweeps:

This was a sort of bonus experiment added into SSM chips in order to gain deeper insight into the effects of net charge investigated previously. The basic idea was to use surface mutations to vary the net charge of the binders from -20 to +20. For each net charge, two different sequences were attempted; however, it was not always possible to generate a unique sequence for every net charge, and as such, there are blanks in the data.

There were several hypotheses going into this experiment. 1) Binders cannot have 0 net charge if they wish to bind. 2) Binders should have the opposite net charge of their target protein.

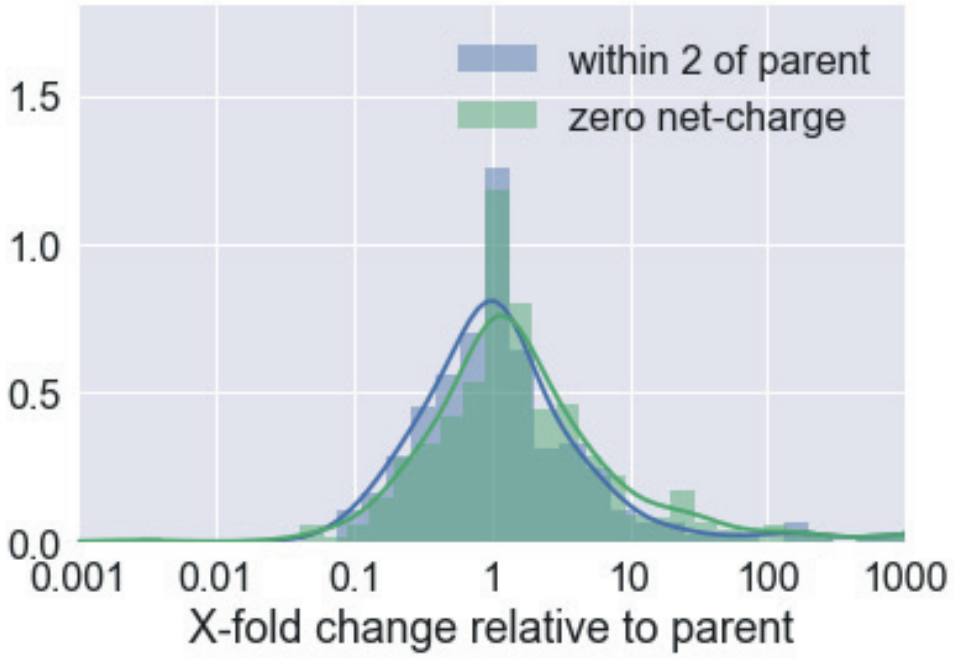

Fig 8.3.4.A: Effect of 0 net charge: Plotted are the changes relative to the parent design for the net charge sweeps mentioned in the text. The blue curve is the control with only net charges within 2 of the parent while the green curve spans from -2 to +2. Although there might be a small shift here, it is nothing like one would expect after seeing the net charge correlations in *8.2.5. Net charge*.

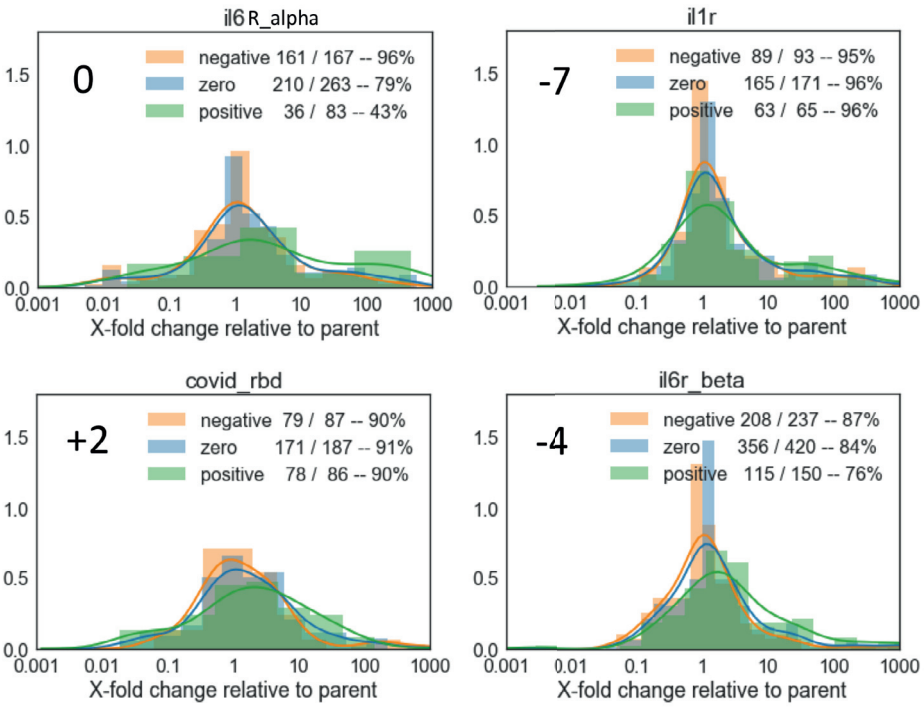

Fig 8.3.4.B: Effect of net charge on different targets: Plotted is the success rate of various net charges across targets with different net charges (the target net charge is the large number in the upper left). The numbers in the legend represent the fraction of the designs that still showed binding. Notably, there does not appear to be much of a trend in any of the graphs. This indicates that charge complementarity between the binder and the target may not have a very large effect.

8.4. Grafting experiment results:

At a time in this specific experiment, where success on future targets seemed unlikely, a new kind of experiment was proposed that at the very least would generate a large number of binders. Whereas the initial screen libraries generated low quality data for a large number of binders and the SSM libraries generated high quality data for a low number of binders, there was no experiment that provided high quality data for a large number of binders.

The plan for the experiment was to take the 11 SSM-validated interfaces for IL7 Receptor Alpha[25] and to try to graft both of the interface helices into a new set of scaffolds. (All of these interfaces consisted of 3-helical bundles using 2 helices to make contact). By grafting the same scaffold into multiple interfaces using multiple pairs of helices, there was hope that it would be possible to validate the 3D models of the scaffold proteins by showing that different faces were correct in this experiment.

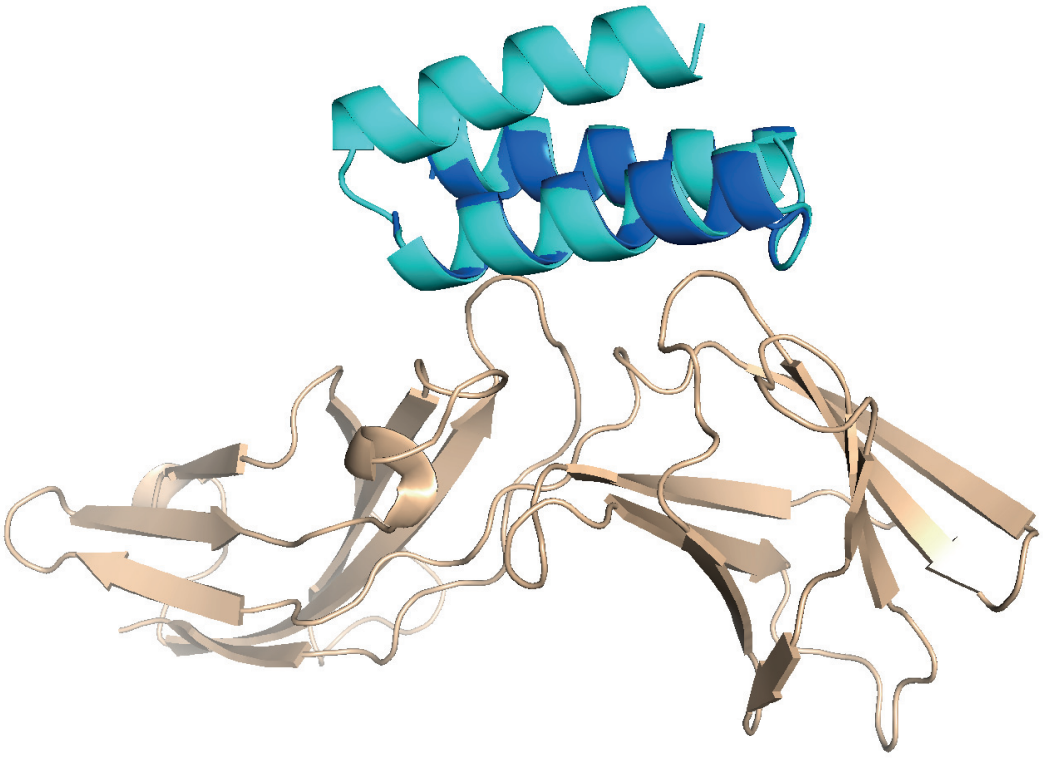

Fig 8.4.A: Schematic for the grafting experiment: An example of a graft with a very low RMSD. Here the cyan design has been grafted onto the dark blue interface. The amino acids will then be copied onto the cyan scaffold. The target structure is IL7 Receptor Alpha (PDB: 3DI3)[25].

When the grafting began, it became clear there would be a critical parameter for this experiment. That parameter was the CA RMSD of the scaffold helices to the SSM validated interface. A perfect match would achieve 0Å RMSD; however, it was unknown what a reasonable upper cutoff would be. A value of 1.5Å was guessed to be the value and most grafts were under this value. However, the optimistic experimenter allowed grafts as poor as 3.0Å to make sure they didn't "miss" the cutoff value.

The other important parameter was the topological connection of the helices at the interface. Some scaffolds contained the exact same loop as the designed interface, while others had longer loops, shorter loops, or different connection patterns all-together. It was desired to test a wide variety of these connection types.

Finally, in order to prevent false positives, a series of 6 identical mutations was made to all members of a given SSM graft. With the goal of producing 2 affinity-increasing mutations, 2 neutral mutations, and 2 deleterious mutations, these mutations would hopefully "fingerprint" the interface such that misfolded proteins could easily be identified.

8.4.1. Grafting experiment RMSD results:

After the experimental data was collected, a shocking discovery was made: the vast majority of the binders did not bind. While at first this seemed like an error, the following graph clearly explains the trend:

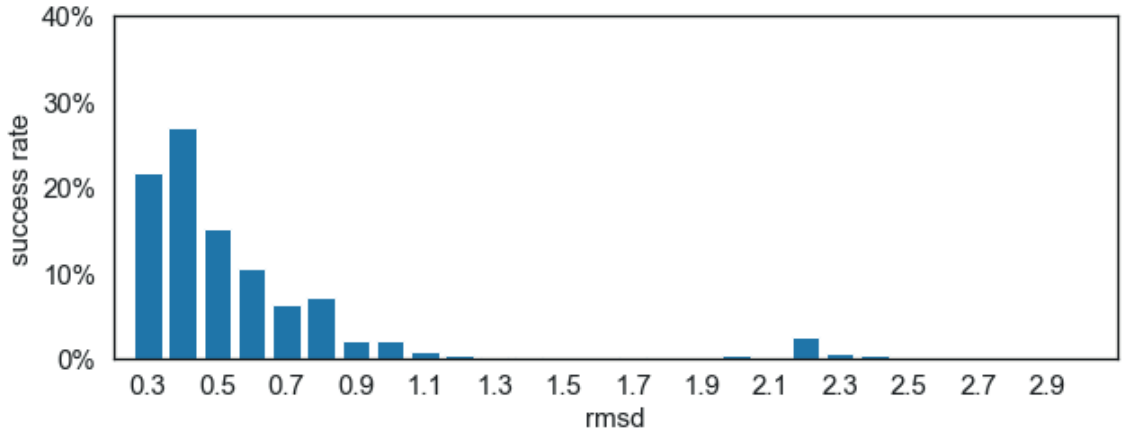

Fig 8.4.1.A: Success of grafted designs versus RMSD of graft: This graph plots the success rate of binders in various CA RMSD bins. The RMSD was measured versus the SSM validated structure. While not shown, there was an even distribution among all RMSD values, so the trend here is very striking. The slight success at 2.2Å likely represents mis-categorization of interfaces.

Evidently in this experiment, RMSD values greater than 0.7Å were different enough to knock out binding. This fact immediately eliminated 90% of the designs that were ordered. It is notable; however, how high the success rate is at the low RMSD values. As many as 25% of the designs with these low RMSD values worked.

These trends immediately suggest that the quality of the binder set is actually very good. If, for instance, the binders were only accurate to 1.0Å RMSD, this graph would not start to fall until around 1.0Å RMSD. The fact that it does fall off so quickly, and that the success rate at the low end is very good, suggests that most of the binders are rather accurate.

8.4.2. Grafting Experiment topology trends:

Limiting the data to only those grafts with RMSD < 0.7Å severely affects the number of data points. However, a striking trend is that the most favored connection type of the grafts was the exact match category.

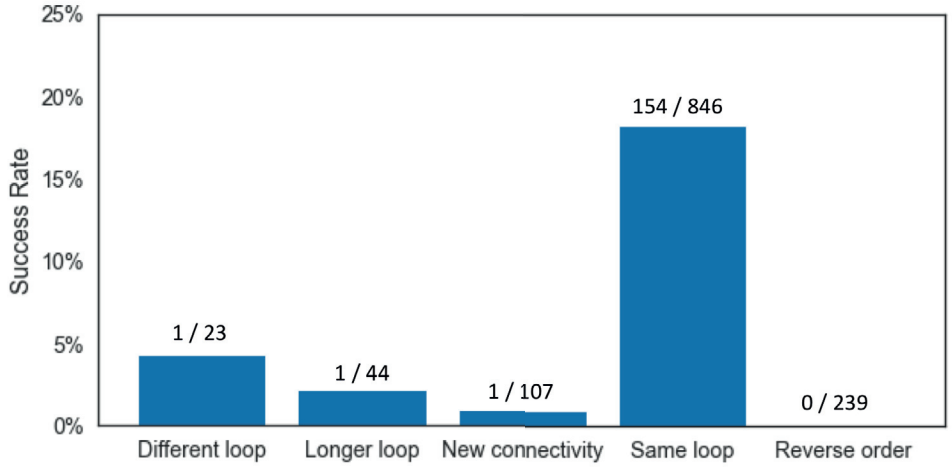

Fig 8.4.2.A: Grafting success by loop type: Given the RMSD results, only grafts with an RMSD < 0.7Å were used for the graph. This severely limits the number of designs to look at and the only piece of meaningful data that can be extracted is that using the same loop has a far greater chance of success than any of the other connectivities.

There are a few ways to look at this. The first is that the loop sometimes stabilizes a few critical interface residues and as such this could be a spurious correlation. The next way to look at this is to say that perhaps there is something more going on than these static protein snapshots show. Perhaps the specific loop helps with interface dynamics. Finally, there is the null hypothesis where we simply say that the same loop allows for nearly the same overall amino acid sequence and therefore it doesn't matter how accurate the 3D model is because they all use the same amino acid sequence. It's hard to say which is the correct answer, but the last point is worth keeping in mind.

8.4.3. Grafting experiment point mutants:

While the goal of the point mutants was to provide a fingerprint to identify misbehaving binders, something even more interesting occurred in the data. When analyzing the data, there was an idea to cluster the binders by which mutations worked and didn't with the goal of teasing out minute details of their structure. Instead, it appeared that each mutation had a specific, but consistent effect on the overall binding strength.

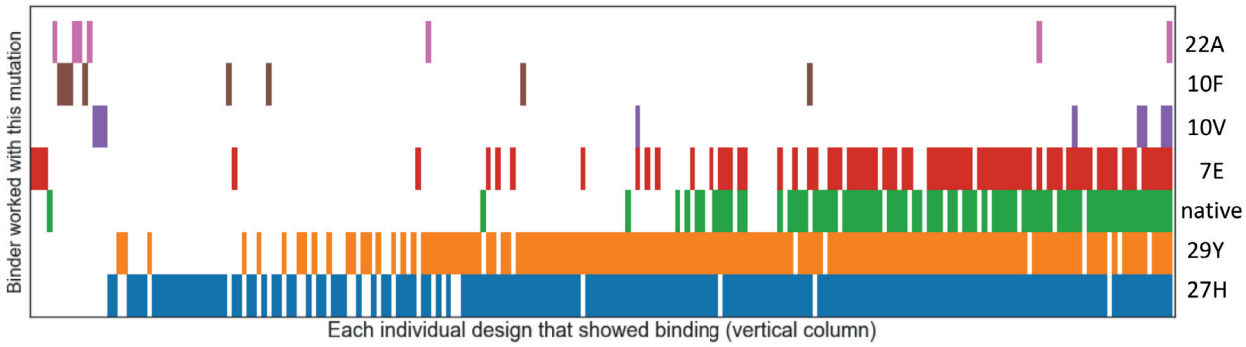

Fig 8.4.3.A: Which mutations allowed binding for each graft: This graph shows the success or failure of each point mutant for each design. Each vertical column represents the 6 points mutants (and native) where the presence of color indicates binding. Binders were sorted from left to right by binding strength.

This graph shows that the strongest binders were able to withstand even the knockout mutations while the weakest binders were only able to bind with the best possible mutations. A plot like this also makes it easy to spot erroneous binders who do not follow the pattern.

While many attempts were made to find a metric that correlated with a binder's position on this graph (i.e. how strong of a binder it was), unfortunately, nothing, not even RMSD, showed a correlation.

8.5. Scaffold topology correlations:

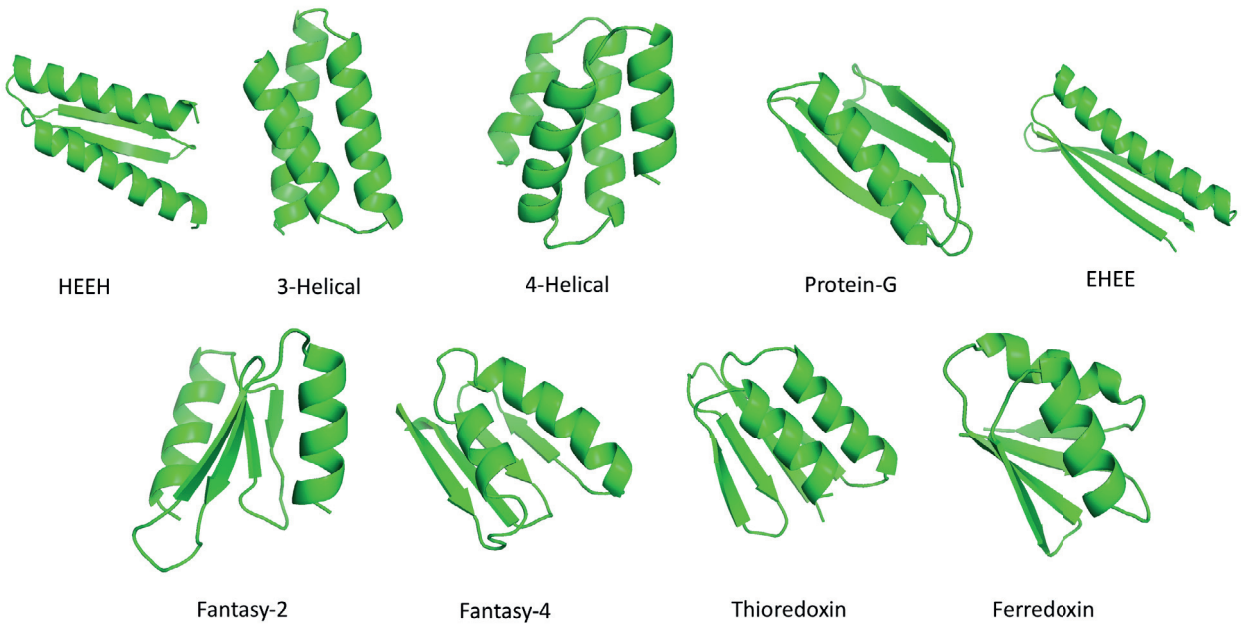

Fig 8.5.A: The 9 scaffold topologies tested in this work: From top left to bottom right, HEEH, 3-helical bundle, 4-helical bundle, Protein-G, EHEE, Fantasy-2, Fantasy-4, Thioredoxin, Ferredoxin. In all cases the scaffolds had to be 65 amino acids or fewer.

While unknown at first, the choice of topology turned out to be one of the most critical factors in designing binders. The data is incomplete, because early on, it was discovered that the protein-G like topologies did not seem to be working. However, examining the data from back then, we can see why this decision was made:

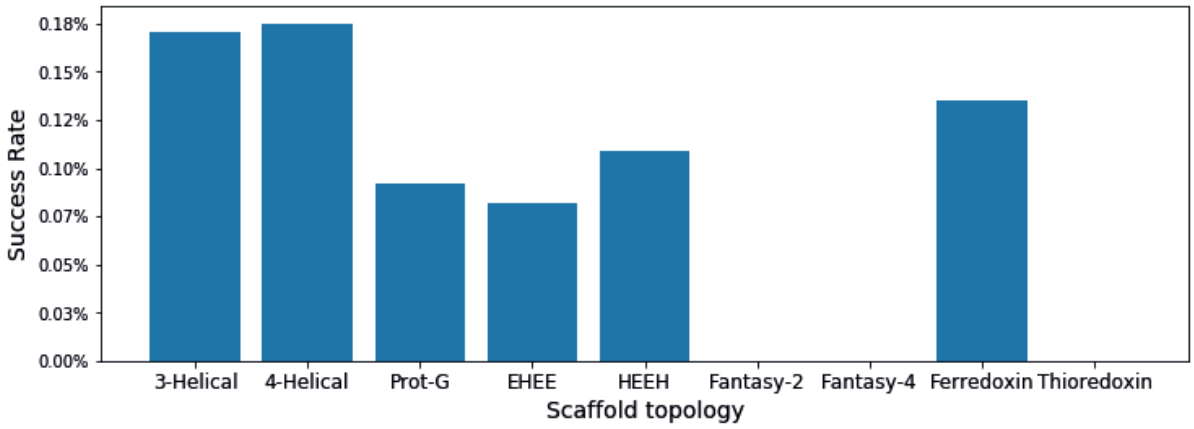

Fig 8.5.B: Early success rate by scaffold topology: This graph shows the success rates for the pooled data of IL7 Receptor Alpha, TrkA, and 4O3V. The Fantasy folds and thioredoxin did not work at all as binders. While it would appear that Protein-G, EHEE, and HEEH allowed for binders, closer inspection of the binding models made this seem unlikely (extremely non-polar beta sheets), and none of these could be expressed in E. coli.

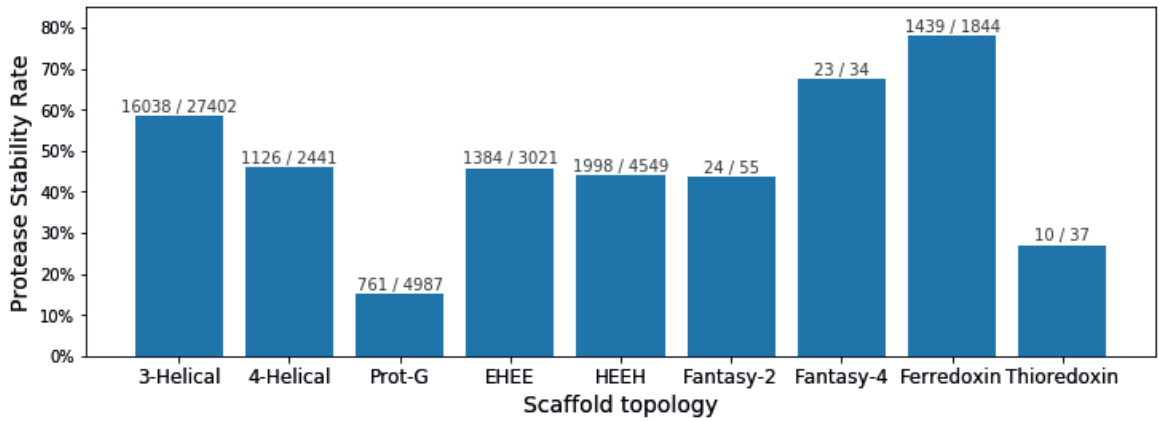

Fig 8.5.C: Protease resistance data from early experiments: This graph shows the protease resistance of the first H3 library, the CD86 library, and the IL7 library. We see that Ferredoxins had the highest stability while Protein-G had the lowest.

Although these protein-G topologies appeared to have modest success rates, none of these binders could be expressed. It was decided that these binders were not binding for the right reasons but were rather molten globules. The protease data above shows the same trend with as many as 80% of the protein-G topologies failing to be protease resistant. Although passing the protease assay does not guarantee accuracy (see *8.6. Helical bundle protease assay results*) failing it is not a good sign.

These early experiments led to focusing exclusively on 3 and 4 helical bundles. The data on these is plentiful and convincing.

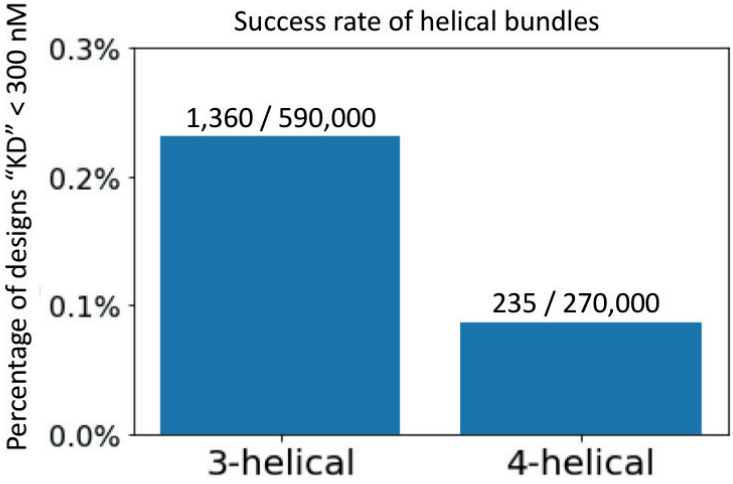

Fig 8.5.D: Success rates of 3-helical bundles and 4-helical bundles: Data pooled from all experiments shown. 3-helical bundles seem to work better than 4-helical bundles.

As to why the 3-helical bundles work better than the 4-helical bundles, there are three main hypotheses: 1) With fewer helices and turns, there are fewer ways the structure can be wrong. 2) The overall helical packing of the 3-helical bundles often “looks more structurally sound” than the arrangements of the 4-helical bundles. 3) The helices of the 4-helical bundles were too short and led to instabilities. These hypotheses were never explicitly tested, but the fact remains that for high success rates, 3-helical bundles are the topology to use.

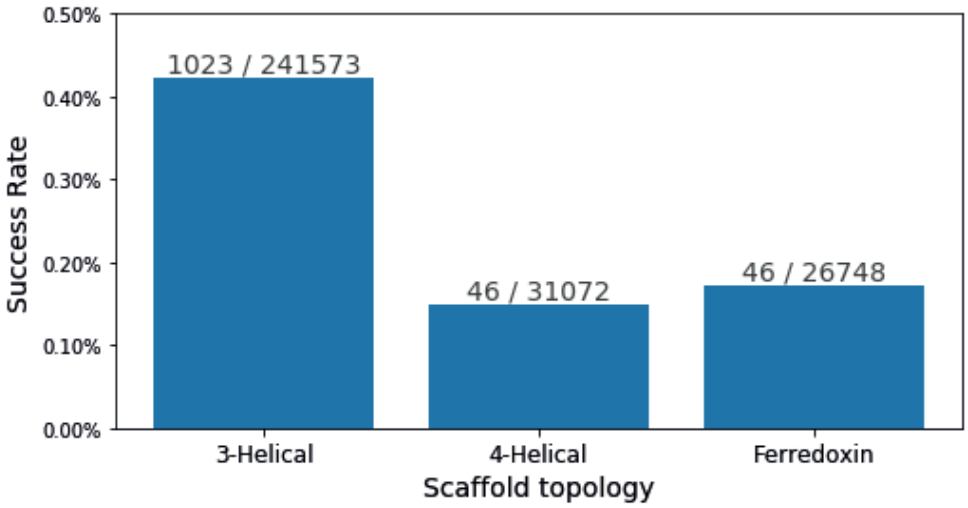

Fig 8.5.E: Success rate by topology of later experiments: The data for IL6 Receptor Alpha, IL6 Receptor Beta, and IL1 Receptor were pooled to show the success rates of different topologies. Although the 3-helical bundles remain the winners, the Ferredoxins appear to have success rates similar to 4-helical bundles.

Of all the beta-sheet topologies at the 65 aa length, the ferredoxins appear to be the best binders. The best hypothesis for this fact is that with a higher helical content, they are better behaved than the single helix variants. As to their success over the other two-helical containing topologies, perhaps they have the superior connectivity pattern.

8.6. Helical bundle protease assay results:

As mentioned in *8.5. Scaffold topology correlations*, the Protease Assay[12] experiment was used to infer that the protein-G like topologies were not working as desired. While the protease assay is great for differentiating well-folded proteins from totally mis-folded proteins, its ability to detect atomic accuracy is of doubt. Additionally, there are concerns that rather than measuring protein folding, it is simply measuring hydrophobicity.

After the 4-helical bundles were generated (*6.6. Scaffold design optimizations*), it was decided that they should be tested on the protease assay to determine their atomic accuracy. A skeptic of the assay, the author decided to add several controls to the pool to test the concerns mentioned previously.

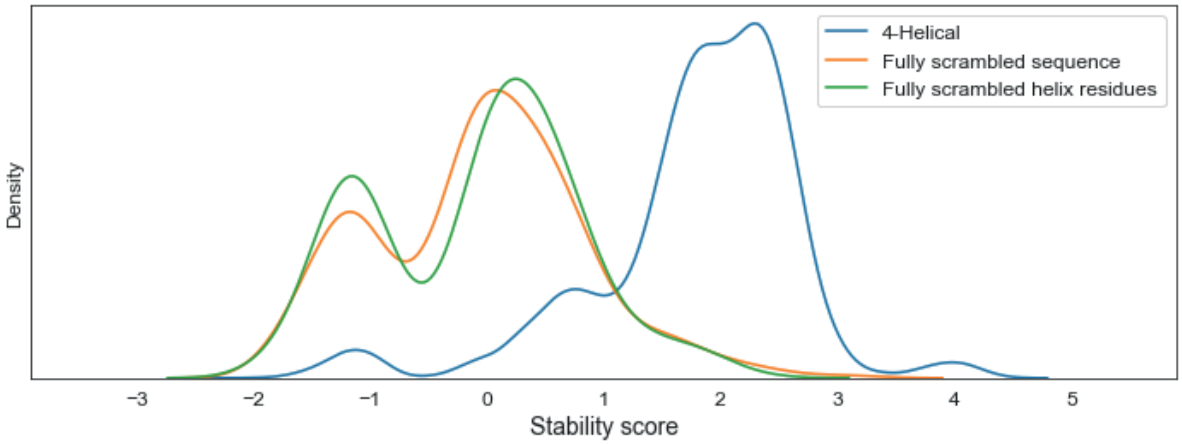

Fig 8.6.A: Protease resistance of 4-helical bundles: This graph shows the extreme protease resistance of the generated 4-helical bundles. Notably, the scrambles are quite protease resistant too. A value of stability score > 1 is typically used as the cutoff between stable and unstable.

The first thing to note here is the exceptionally high success rate of both the designs and the scrambles. Typically, one chooses a cutoff to call "validated" where only 5% of the scrambles remain, here we see that that may not work. Another approach is to pick the value of 1 on this scale to call success (which was done on the original experiments) in which case we see that nearly all designs are protease stable.

The question of course is then, are they atomically accurate? An attempt to answer this question was made by burying an Aspartic Acid at the deepest point in the protein structure, or burying two Aspartic Acids as close together in the protein structure.

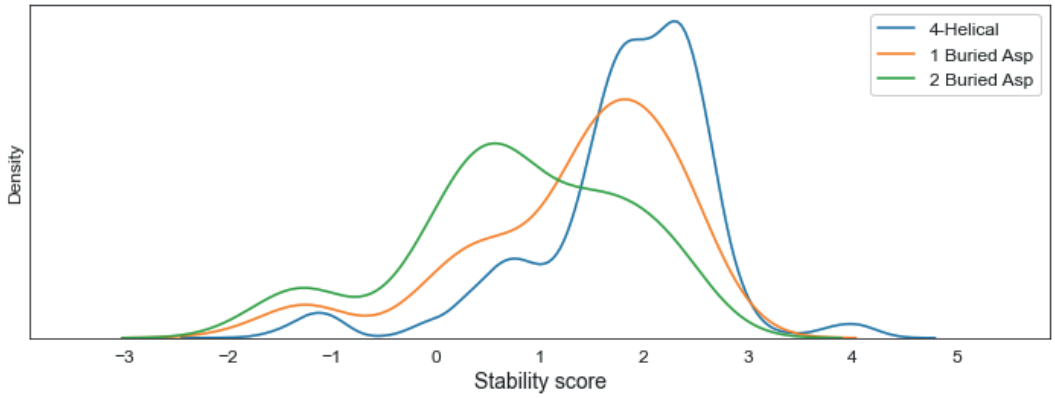

Fig 8.6.B: Protease stability of buried aspartate controls: This graph shows the stability of the buried aspartate controls in relation to the normal helical bundles. Although many designs were greatly destabilized, not all of them were.

Although there is a shift in these distributions, not all of the designs were knocked out by these mutations, even with 2 buried Aspartic Acids. It is potentially conceivable that these proteins are

so stable that they can bury two aspartic acids, but it is also conceivable that these proteins folded into a different shape that was also protease stable.

The final experiment was to test the hypothesis that the only thing the protease assay was measuring was hydrophobicity. For a series of 200 designs, all hydrophobic amino acids were mutated to either A, V, L, or F. The idea was to create a series of hydrophobicities with the expected outcome that A would be less stable than V, which is less stable than L, which is less stable than F. Most of the cores of the proteins were L anyways, so we would then expect the F designs to be more stable than the original designs.

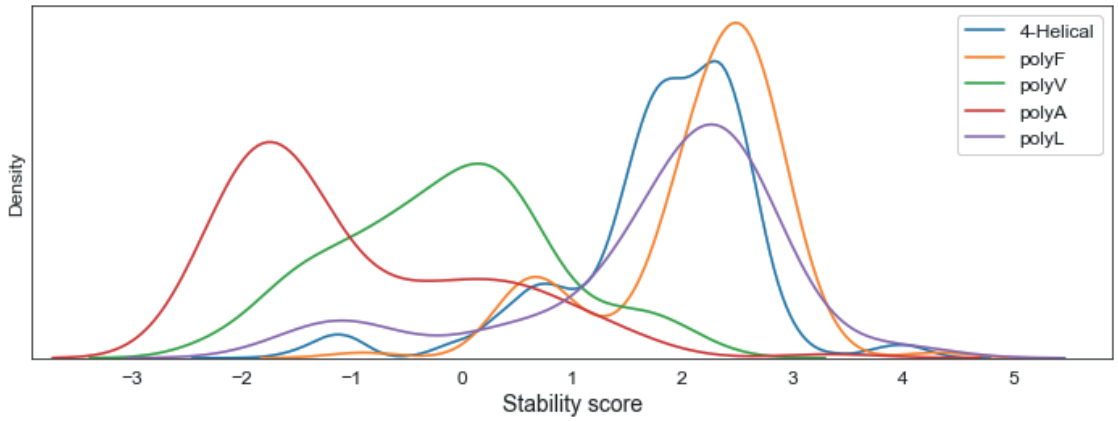

Fig 8.6.C: Protease stability of poly X hydrophobic proteins: This plot shows the stability of the poly X controls. These controls were created by mutating every hydrophobic amino acid to the designated letter. No attempt was made at ensuring these structures were physically realistic; the mutations happened at the sequence level.

And this hydrophobicity-driven result is exactly what was observed. Examining the structures of the poly-F designs, it's clear there's no way these proteins could fold as designed. Instead, this experiment shows that at least for helical-patterned proteins, that maxing out hydrophobicity is a null solution.

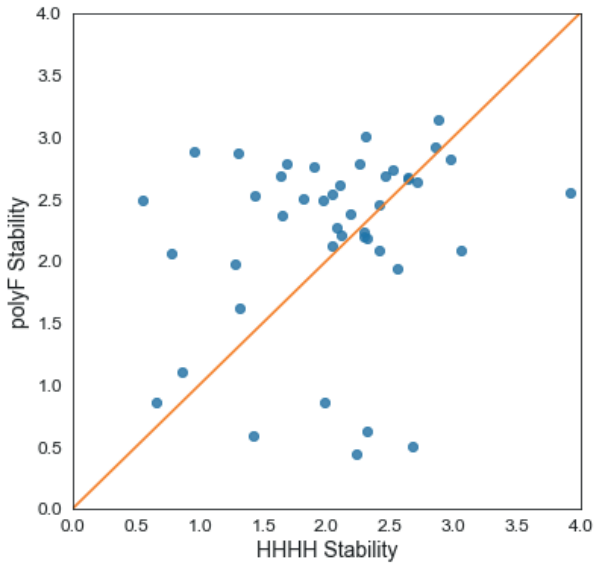

Fig 8.6.D: Protease stability of individual polyF variants: This plot shows the protease stability of designs before and after mutating all hydrophobic residues to phenylalanine. In some cases, there was no change (which is peculiar) and in other cases, there is a knockout effect (which is to be "expected"). However, in other cases, we see the polyF mutant greatly stabilizing the structure which is not consistent with a structurally based model for the protease assay.

One final point to back up this observation is that the poly-F cores actually "saved" some designs. The parent design did not work, but the poly-F variant did.

9. What remains unknown:

Although this work provides a framework to reliably make mini-protein binders to hydrophobic patches on natural targets, it also highlights the fact that there is still a lot we don't know about how protein's fold and interact.

9.1. Why is the success rate so low?

The highest success rate of any of the experiments was 1% while typically the success rate was around 0.2%. It is totally unclear why these success rates are so low. This is not an experimental artifact, because the SSM libraries can have success rates as high as 30%. Do we blame protein folding? Poor interface quality? Off-target interactions? The grafting experiment (*Fig 8.4.1.A: Success of grafted designs versus RMSD of graft*) suggests that some of the scaffolds are very high quality. The correlations with binder hydrophobicity (*Fig 8.2.1.C: Correlation of Binder Delta SAP with experimental data*) show that off-target trapping is certainly possible, but also suggest a way to overcome it. Perhaps it is poor interface quality, which is supported by the next point.

9.2. Why can't we gain binding energy from polar interactions?

The trends in *8.2.1. Hydrophobicity metrics* and *8.2.2. Packing measures* suggest that hydrophobicity and packing are the only factors involved in binding here. Looking at the polar contacts of interfaces often leads to negative correlations, but it is difficult to separate the effect of non-polarity here since the polar-contacts often take the place of non-polar contacts (because the binders are all roughly the same size). Some native interfaces are mediated entirely through polar contacts. It is a puzzle as to why we cannot seem to do the same thing that nature does. Does water play a role here? Does polarization play a role? Interface dynamics? Variable dielectric? Whatever the real answer, it does not appear to be part of our current tools.

9.3. What affect does water have on interfaces?

Our models in this study use only an implicit solvent model that does little more than to penalize polar atoms for being buried in an interface. In reality, water is a complex beast making specific, but transient h-bonds to both sides of the interface and itself. Additionally, water carries with it entropy that is modified when the two surfaces come together. In this work, we pay almost no attention to this fact and just use proxies like SAP Score and buried unsats. Correct consideration of water would likely require Molecular Dynamics simulations which would likely be too expensive for a dataset of this size.

9.4. What affect does dynamics play on interfaces?

Rosetta can be viewed as a crystal structure tool; it only deals with the lowest energy state of the system and does not consider dynamics. There are many sources of dynamics that can be envisioned in this system; however, without any real way to investigate them using Rosetta. Examples of dynamics include: transient surface interactions like salt bridges, loss of side-chain conformational entropy upon binding, loss of backbone conformational entropy upon binding, harmonized side-chain or backbone movements by both partners of the interfaces, or partial unbinding events.

The partial unbinding events also bring up the topic of on-rate or off-rate dynamics. Could the specific sequence of events that lead to the final binding complex be important? Could the ability of the interface to recover from partial unbinding events be important? These are all questions that remain unanswered by this study. Long Molecular Dynamics trajectories would likely be needed to start investigating these ideas.

10. Acknowledgements:

This work was a team effort spanning multiple years. Most notably, **Longxing Cao**, should be recognized as he and I worked on this problem side-by-side from the start until the end. Additionally, he did the vast majority of the experiments until mid 2020 when the UW BioFab took over the experiments.

Wet-lab experiments were performed primarily by:

- Longxing Cao
- Inna Goreshnik
- Samer Halabiya
- Aza Allen
- Cami Cordray
- Buwei Huang
- Lisa Kozodoy
- Lauren Miller

Methods development and scaffold development that made it into the final pipeline were created by:

- Longxing Cao
- Gabe Rocklin
- Eva Strauch
- Will Sheffler
- TJ Brunette
- Daniel Silva
- Franziska Seeger
- Hugh Haddox
- Ta-Yi Yu
- Stephanie Berger
- Danny Sahtoe
- Brian Weitzner
- Vikram Mulligan
- Scott Boyken

Countless discussions about this project came from:

- Adam Moyer
- Derrick Hicks
- Florian Praetorius

- Dmitri Zorine
- Ian Hayden
- Nate Bennett
- Tim Huddy
- Harley Pyles
- Danny Sahtoe
- Basile Wicky
- Tim Craven
- Wei Yang
- Chris Norn
- Yang Hsai
- Ajasja Ljubetic
- Brian Koepnick

This project stressed computational resources to the max and these people kept them going:

- Luki Goldshmidt
- David Kim
- Patrick Vecciato

This project would not have been possible without the help and support of the Baker Lab and the Institute for Protein Design. **David Baker** has been a tremendous help through the process and Lance Stewart has made all of the organizational steps happen. Additionally, all of the members of the Baker Lab deserve recognition for creating a great working environment as well as helpful conversations along the way.

Funding for this work came from the following sources:

- The Audacious Project
- The Open Philanthropy Project
- The DARPA Synergistic Discovery and Design (SD2) Project DARPA SD2 HR001117S0003
- TACC Frontera Pathways (compute)
- 1U19AG065156-01
- R01AG063845-01
- Funding from Eric and Wendy Schmidt by recommendation of the Schmidt Futures program

Additionally, the following supercomputers were used for design calculations:

- The digs (IPD)
- Rosetta@Home[42] (IPD)

- Hyak ikt (UW)
- Hyan mox (UW)
- Maverick (TACC, SD2)
- Wrangler (TACC, SD2)
- Lonestar5 (TACC, SD2)
- Stampede2 (TACC, SD2)
- Frontera (TACC, Pathways)

Supercomputer acknowledgements:

- This work was facilitated through the use of advanced computational, storage, and networking infrastructure provided by the Hyak supercomputer system at the University of Washington
- The authors acknowledge the Texas Advanced Computing Center (TACC) at The University of Texas at Austin for providing {HPC, visualization, database, or grid} resources that have contributed to the research results reported within this paper. URL: http://www.tacc.utexas.edu

11. References

1. Baker D. A surprising simplicity to protein folding. Nature. 2000;405: 39–42. doi:10.1038/35011000
2. Verdecchia P, Cavallini C, Spanevello A, Angeli F. The pivotal link between ACE2 deficiency and SARS-CoV-2 infection. Eur J Intern Med. 2020;76: 14–20. doi:10.1016/j.ejim.2020.04.037
3. Lorenzen N, Lapatra SE. Immunity to rhabdoviruses in rainbow trout: the antibody response. Fish Shellfish Immunol. 1999;9: 345–360. doi:10.1006/fsim.1999.0194
4. Shen C, Assche GV, Colpaert S, Maerten P, Geboes K, Rutgeerts P, et al. Adalimumab induces apoptosis of human monocytes: a comparative study with infliximab and etanercept. Aliment Pharmacol Ther. 2005;21: 251–258. doi:https://doi.org/10.1111/j.1365-2036.2005.02309.x
5. Carbonetti S, Oliver BG, Vigdorovich V, Dambrauskas N, Sack B, Bergl E, et al. A method for the isolation and characterization of functional murine monoclonal antibodies by single B cell cloning. J Immunol Methods. 2017;448: 66–73. doi:10.1016/j.jim.2017.05.010
6. Silva D-A, Yu S, Ulge UY, Spangler JB, Jude KM, Labão-Almeida C, et al. De novo design of potent and selective mimics of IL-2 and IL-15. Nature. 2019;565: 186–191. doi:10.1038/s41586-018-0830-7

7. Strauch E-M, Bernard SM, La D, Bohn AJ, Lee PS, Anderson CE, et al. Computational design of trimeric influenza-neutralizing proteins targeting the hemagglutinin receptor binding site. Nat Biotechnol. 2017;35: 667–671. doi:10.1038/nbt.3907
8. Chevalier A, Silva D-A, Rocklin GJ, Hicks DR, Vergara R, Murapa P, et al. Massively parallel de novo protein design for targeted therapeutics. Nature. 2017;550: 74–79. doi:10.1038/nature23912
9. Fleishman SJ, Whitehead TA, Ekiert DC, Dreyfus C, Corn JE, Strauch E-M, et al. Computational design of proteins targeting the conserved stem region of influenza hemagglutinin. Science. 2011;332: 816–821. doi:10.1126/science.1202617
10. Kryshtafovych A, Schwede T, Topf M, Fidelis K, Moult J. Critical assessment of methods of protein structure prediction (CASP)—Round XIII. Proteins Struct Funct Bioinforma. 2019;87: 1011–1020. doi:https://doi.org/10.1002/prot.25823
11. Hughes RA, Ellington AD. Synthetic DNA Synthesis and Assembly: Putting the Synthetic in Synthetic Biology. Cold Spring Harb Perspect Biol. 2017;9: a023812. doi:10.1101/cshperspect.a023812
12. Rocklin GJ, Chidyausiku TM, Goreshnik I, Ford A, Houliston S, Lemak A, et al. Global analysis of protein folding using massively parallel design, synthesis, and testing. Science. 2017;357: 168–175. doi:10.1126/science.aan0693
13. Alford RF, Leaver-Fay A, Jeliazkov JR, O'Meara MJ, DiMaio FP, Park H, et al. The Rosetta All-Atom Energy Function for Macromolecular Modeling and Design. J Chem Theory Comput. 2017;13: 3031–3048. doi:10.1021/acs.jctc.7b00125
14. Leaver-Fay A, Tyka M, Lewis SM, Lange OF, Thompson J, Jacak R, et al. ROSETTA3: an object-oriented software suite for the simulation and design of macromolecules. Methods Enzymol. 2011;487: 545–574. doi:10.1016/B978-0-12-381270-4.00019-6
15. Lauer TM, Agrawal NJ, Chennamsetty N, Egodage K, Helk B, Trout BL. Developability Index: A Rapid In Silico Tool for the Screening of Antibody Aggregation Propensity. J Pharm Sci. 2012;101: 102–115. doi:10.1002/jps.22758
16. Lawrence MC, Colman PM. Shape Complementarity at Protein/Protein Interfaces. J Mol Biol. 1993;234: 946–950. doi:10.1006/jmbi.1993.1648
17. Black SD, Mould DR. Development of hydrophobicity parameters to analyze proteins which bear post- or cotranslational modifications. Anal Biochem. 1991;193: 72–82. doi:10.1016/0003-2697(91)90045-U
18. Croll TI, Smith BJ, Margetts MB, Whittaker J, Weiss MA, Ward CW, et al. Higher-Resolution Structure of the Human Insulin Receptor Ectodomain: Multi-Modal Inclusion of the Insert Domain. Structure. 2016;24: 469–476. doi:10.1016/j.str.2015.12.014
19. O'Meara MJ, Leaver-Fay A, Tyka MD, Stein A, Houlihan K, DiMaio F, et al. Combined Covalent-Electrostatic Model of Hydrogen Bonding Improves Structure Prediction with Rosetta. J Chem Theory Comput. 2015;11: 609–622. doi:10.1021/ct500864r
20. Coventry B, Baker D. Protein sequence optimization with a pairwise decomposable penalty for buried unsatisfied hydrogen bonds. bioRxiv. 2020; 2020.06.17.156646. doi:10.1101/2020.06.17.156646
21. Vijay-Kumar S, Bugg CE, Cook WJ. Structure of ubiquitin refined at 1.8Åresolution. J Mol Biol. 1987;194: 531–544. doi:10.1016/0022-2836(87)90679-6

22. Xu D, Zhang Y. Generating Triangulated Macromolecular Surfaces by Euclidean Distance Transform. PLOS ONE. 2009;4: e8140. doi:10.1371/journal.pone.0008140
23. Xu D, Li H, Zhang Y. Protein Depth Calculation and the Use for Improving Accuracy of Protein Fold Recognition. J Comput Biol. 2013;20: 805–816. doi:10.1089/cmb.2013.0071
24. Buckle AM, Schreiber G, Fersht AR. Protein-protein recognition: Crystal structural analysis of a barnase-barstar complex at 2.0-.ANG. resolution. Biochemistry. 1994;33: 8878–8889. doi:10.1021/bi00196a004
25. McElroy CA, Dohm JA, Walsh STR. Structural and Biophysical Studies of the Human IL-7/IL-7Rα Complex. Structure. 2009;17: 54–65. doi:10.1016/j.str.2008.10.019
26. Dou J, Vorobieva AA, Sheffler W, Doyle LA, Park H, Bick MJ, et al. De novo design of a fluorescence-activating β-barrel. Nature. 2018;561: 485–491. doi:10.1038/s41586-018-0509-0
27. Schneidman-Duhovny D, Inbar Y, Nussinov R, Wolfson HJ. PatchDock and SymmDock: servers for rigid and symmetric docking. Nucleic Acids Res. 2005;33: W363–W367. doi:10.1093/nar/gki481
28. Frishman D, Argos P. Knowledge-based protein secondary structure assignment. Proteins Struct Funct Bioinforma. 1995;23: 566–579. doi:https://doi.org/10.1002/prot.340230412
29. Zhang Y, Skolnick J. TM-align: a protein structure alignment algorithm based on the TM-score. Nucleic Acids Res. 2005;33: 2302–2309. doi:10.1093/nar/gki524
30. Tyka MD, Jung K, Baker D. Efficient sampling of protein conformational space using fast loop building and batch minimization on highly parallel computers. J Comput Chem. 2012;33: 2483–2491. doi:https://doi.org/10.1002/jcc.23069
31. Maguire J, Haddox H, Strickland D, Halabiya S, Coventry B, Cummins M, et al. Perturbing the energy landscape for improved packing during computational protein design. Preprints; 2020 May. doi:10.22541/au.158986804.41133682
32. Brunette TJ, Bick MJ, Hansen JM, Chow CM, Kollman JM, Baker D. Modular repeat protein sculpting using rigid helical junctions. Proc Natl Acad Sci. 2020;117: 8870–8875. doi:10.1073/pnas.1908768117
33. Koga N, Tatsumi-Koga R, Liu G, Xiao R, Acton TB, Montelione GT, et al. Principles for designing ideal protein structures. Nature. 2012;491: 222–227. doi:10.1038/nature11600
34. Divine R, Dang HV, Ueda G, Fallas JA, Vulovic I, Sheffler W, et al. Designed proteins assemble antibodies into modular nanocages. bioRxiv. 2020; 2020.12.01.406611. doi:10.1101/2020.12.01.406611
35. Dang B, Wu H, Mulligan VK, Mravic M, Wu Y, Lemmin T, et al. De novo design of covalently constrained mesosize protein scaffolds with unique tertiary structures. Proc Natl Acad Sci. 2017;114: 10852–10857. doi:10.1073/pnas.1710695114
36. Fallas JA, Ueda G, Sheffler W, Nguyen V, McNamara DE, Sankaran B, et al. Computational design of self-assembling cyclic protein homo-oligomers. Nat Chem. 2017;9: 353–360. doi:10.1038/nchem.2673
37. Harris CR, Millman KJ, van der Walt SJ, Gommers R, Virtanen P, Cournapeau D, et al. Array programming with NumPy. Nature. 2020;585: 357–362. doi:10.1038/s41586-020-2649-2
38. Lam SK, Pitrou A, Seibert S. Numba: a LLVM-based Python JIT compiler. Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC. New York, NY, USA: Association for Computing Machinery; 2015. pp. 1–6. doi:10.1145/2833157.2833162

39. Coventry B. bcov77/npose. 2020. Available: https://github.com/bcov77/npose
40. McGuffin LJ, Bryson K, Jones DT. The PSIPRED protein structure prediction server. Bioinformatics. 2000;16: 404–405. doi:10.1093/bioinformatics/16.4.404
41. Plotnikov AN, Hubbard SR, Schlessinger J, Mohammadi M. Crystal Structures of Two FGF-FGFR Complexes Reveal the Determinants of Ligand-Receptor Specificity. Cell. 2000;101: 413–424. doi:10.1016/S0092-8674(00)80851-X
42. Das R, Qian B, Raman S, Vernon R, Thompson J, Bradley P, et al. Structure prediction for CASP7 targets using extensive all-atom refinement with Rosetta@home. Proteins Struct Funct Bioinforma. 2007;69: 118–128. doi:https://doi.org/10.1002/prot.21636