Protein Design at Library Scale

Jacob Gershon

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington
2025

Reading Committee:
David Baker, Chair
Ge Liu
Frank DiMaio
Jesse Zalatan

Program Authorized to Offer Degree:
Molecular Engineering

University of Washington

Abstract

Protein Design at Library Scale

Jacob Gershon

Chair of the Supervisory Committee:
David Baker
Department of Biochemistry

Recent advances in de novo protein design have made it increasingly feasible to create proteins with novel functions, driven by rapid progress in both computational modeling and high-throughput experimentation. Modern tools can explore vast sequence-structure spaces and evaluate biomolecular interactions, while experimental assays can now screen billions of variants in parallel. Yet, a key limitation remains: our current predictive models still struggle to capture the complex physical and dynamical factors that underlie enzyme function. My thesis addresses this gap by developing an integrated experimental–computational framework for enzyme design that couples large-scale protein library construction with data-driven model development. I design and test extensive libraries of enzyme variants to both optimize catalytic activity and generate training data for next-generation predictors of protein function. Ultimately, this approach advances our ability to connect sequence, structure, and function.

Contents

Acknowledgments

The people who helped me through this :)

1 Introduction

Proteins are tools capable of operating at the molecular level. They are the basis of biological interactions mediating signaling events, immune modulation, catalysis, structural support networks and more. Every biological system relies on what is known as the *central dogma* first described by Francis Crick in the 1970s, to first transcribe deoxyribonucleic acid (DNA) to ribonucleic acid (RNA) and then translate RNA to protein [1]. While biochemical mechanisms of evolution take place at the DNA level to mutate or change the genetic code, these mutations are felt at the protein level when the gene is translated. Over the course of billions of years stochasticity inherent in the replication of genetic information has led to evolution of novel function in proteins [2]. Nature is smart about conserving evolutionary energy and often will rely on similar protein structures to achieve an array of chemistries or signaling events. Across all organisms protein sequences and structures are often conserved. For example immunoglobulin like folds in mammals are highly conserved protein structures which activate and interplay with the immune system to defend off pathogens [3]. Similarly, plants and algae across all species share common folds for proteins such as rubisco which fix carbon dioxide from the atmosphere into sugar building molecules [4]. There are many such examples one can find examining conserved functions in the biological kingdom. Furthermore, examples of convergent evolution offer insight into how different protein structures can be evolved for similar functions [5]. Looking through a specific class of enzyme functions, one can usually find a variety of protein folds which lend themselves to being able to do the same

chemistry [6]. This insight helped drive the field of protein design as people began to understand the value of being able to engineer proteins for a particular function. Early work in this space involved understanding and recognizing why certain mutations are beneficial for function. While it can often be difficult to rationalize how some mutations effect function, in many cases structural analysis such as NMR or x-ray crystallography can help understand why a mutation or group of mutations have a particular effect [7, 8]. It is believed there is an underlying physical explanation for the functional phenomenon of proteins, and many researchers around the world are working to explain and understand how atomic structure conveys function. To further our understanding for structure-function relationships, the field of protein design began to take form in the early 2000's as researchers first sought to better understand how and why proteins fold to a particular tertiary structure.

1.1 Molecular Mechanisms of Protein Folding

Proteins are often thought to have a tiered hierarchy which seeks to explain how they fold to a functional form. At the most basic level, a protein is a polymer composed of twenty canonical amino-acids. Each amino acid has a conserved amine group at one end and a carboxylic acid at the opposite end. Between these two functional groups lies another carbon atom, often referred to as the alpha carbon (α-carbon). The amine, carboxylic acid, and α-carbon are known as the protein *backbone*. Connected to the α-carbon lies a functional group, unique to every specific amino acid, and referred to as the *side-chain*. These functional groups have a variety of shapes and properties, some are hydrophobic, while others are polar and even charged. Secondary

structure is formed via hydrogen bonding of the backbone atoms. The nitrogen and oxygen atoms on the backbone are capable of hydrogen bonding to form two unique secondary structures: alpha-helices and beta-sheets. These secondary structure elements are conserved across all types of protein folds. At the tertiary structure level, amino acid sidechains are responsible for dictating the interactions and fold topology of the protein. Amino acid sidechains are capable of forming hydrophobic pockets, hydrogen bonds, charge-charge interactions, or covalent disulfide linkages. A the quaternary structure level, folded proteins may interact and complex with others to form larger complexes capable of assembling into machines, capsids, and more. While we have long understood the physical interactions which mediate protein folding, since the first protein structures were solved it has been a long outstanding challenge to predict the structure an amino acid sequence will fold to [9]. The protein data bank (PDB), founded in the 1970s is an ongoing effort to form a public repository of protein structures. This dataset is one of the greatest resources publicly available, containing over 100,000 structures.

1.2 Protein Modeling

In an effort to model a physical understanding of protein stability Rosetta was initially developed as a coarse grained force field model to predict protein stability [10]. As the community surrounding Rosetta grew, tools were developed for computing energies with respect to a wide variety of biomolecules allowing users to understand the interactions energetics [11, 12]. Designers used these tools to make new to nature topologies and found amino acid sequences which would stabilize the fold of interest

[13]. However, the Rosetta framework was unable generalize well to predicting the 3D structure of a 1D amino acid sequence alone. In 2020, Deepmind released Alphafold2, a deep neural network capable of predicting 3D structure from a 1D amino acid sequence [14]. Considered to be a breakthrough for protein structure prediction, this marked a major turning point which greatly increased the success and accessability of protein design. Modeling the conditional distribution of protein structure given sequence was possible using multiple sequence alignments (MSAs) which provide evolutionary context for a particular sequence. The families of sequences contained in an MSA allow co-evolutionary information to be derived which can allow the model to pick out where long range sequence contacts are predicted to be [15]. Often, short sequence motifs will fold to the same secondary structure and it is possible to learn what simple sequence patters are indicative of helix or beta strand formation. Predicting long range sequence contacts is often more challenging and Alphafold2 was able to do so by learning to pull the contact information from the MSA directly. The architecture can be broken down into a few components, a module which embeds the query sequence and MSA into a single ($\mathbb{R}^{L\times d}$) and pair ($\mathbb{R}^{L\times L\times d}$) feature representations which are carried into structure module to produce the coordinates ($\mathbb{R}^{L\times 3}$), where L is the length of the sequence to be folded [14]. This breakthrough not only offered researches the ability to predict the structure of unmown proteins, but it also was later discovered to be an excellent *in silico* oracle for designed proteins.

1.3 Extending Folding to Design

The success of the architecture for modeling sequence-structure relationships allowed researchers to make modifications allowing these models to be finetuned for design tasks. Reproductions of Alphafold2, such as RoseTTAFold, allowed researchers to experiment with architectural components of the network [16]. Deep network hallucination (also known as activation maximization) methods were first shown as a promising method for design [17]. These approaches involved computing gradient steps through a structure prediction network to optimize a sequence toward a low-entropy structure. Users were able to define distance constraints to bias the hallucination trajectory. Taking gradients through the bulky architecture is time-consuming and memory intensive, so researchers looked to the image modeling domain to see where improvements could be made. A common task in the world of image generative modeling is known as "inpainting" and involves completing a masked portion of the image [18]. This is similar to sentence completion in natural language processing, and requires the model to rely on context from the rest of the image to fill in what was left blank. In the world of protein generative modeling this translated well to a task where provided a motif the model was tasked with finding a scaffold to fix the motif in space. Known as "motif scaffolding", finetuning RoseTTAFold for inpainting yielded a model capable of scaffolding a wide variety of motifs in a single shot [17]. These inpainting models were great at conditional generation, but struggled to produce any diversity in unconditional generation tasks. Hallucination methods were good at giving diverse topologies, but at the time were still slow and memory intensive, as previously mentioned.

Diffusion denoising models were starting to become popular in the image domain [19]. These models learn to sample from the underlying data distribution by mapping from a well known distribution to the data. A natural case for the prior distribution is Gaussian. The properties are well studied and simple to sample from. Early work in adapting these models to the protein domain, parameterized the process in structure space. This was a natural choice as message passing neural network models (MPNN) have recently been widely popularized for their ability to design stable and soluble sequences when presented a backbone [20]. This led to the workflow of backbone generation followed by sequence design. However, this backbone first workflow makes it difficult to condition structural generation on sequence specific attributes. The interplay between sequence and structure is a critical element to protein folding and dynamics. There a subset of problems for which conditioning on sequence information is quite important.

1.4 Why De Novo Design?

Before moving forward, it is worthwhile taking a step back to understand why *de novo* design is a useful approach. The counter argument here is that nature has evolved all sorts of diverse proteins for a wide variety of applications and functions. It can seem futile to start from scratch when nature has done so much of the work up front. At the current state of the field it is often much easier to start with a native protein which has a function near your objective. However what happens when the function you're looking to encode is novel enough that no known protein has a function within the realm? For challenges where the function is new to nature we will need to design a

protein from scratch [21]. To build such a system completely *de novo* designers will require a robust set of filters to understand what metrics convey function. For this reason the exercise of designing *de novo* proteins will allow us to validate generative models for design, and discriminative models for their filtering power. A great example of this goes back to Alphafold2, where it was later discovered to be a fantastic oracle for *de novo* protein structure prediction. Many crystal structures have since been solved which match the predicted structure from Alphafold2 to near atomic accuracy. Interestingly, *de novo* proteins have no MSA as they have no evolutionarily related family, but structure prediction models are still able to fold these novel designs. This could be the reason for designed proteins often being thermostable, and we are a ways out from designing unstructured dynamic proteins. The lessons learned while designing proteins from scratch will enable us to dream up new to nature molecular interactions and chemistry.

1.5 The Promise of Protein Design

Designing proteins on demand will allow for everything from targeted therapeutic design to sustainable chemistry. Proteins are the molecular workhorses of biology and are enable complex and energetically unfavorable chemistry by lowering the activation energy of the reactions. From a therapeutic perspective, protein design promises the ability to treat disease [22]. This can involve targeting specific surface receptors to limit the off-target effects of a drug or replacing a missing enzyme with a *de novo* one which can be easily expressed. Designing antibodies to target a specific antigen is a highly sought after goal, with companies today investing hundreds of millions into

such design tasks. In addition to therapeutic applications, protein designing has applications in the broad space of sustainability. Modern manufacturing processes often require toxic reagents for compound synthesis, but design of enzymes would allow for some of these same chemical reactions to occur in aqueous environments allowing them to be environmentally friendly [23]. Furthermore, enzymatic degradation of waste products including plastics would allow for complex polymers to be recycled into monomers [24]. There is a need for us to build machines at the molecular scale in order to harness next generation of technologies.

1.6 Research Directions and Goals

In the following document I will lay out the central hypothesis to my graduate work along with a set of aims which will structure the approaches and proposals written here.

> **Hypothesis:** Guiding generative models with experimental data will increase the success of functional *de novo* protein design.

While the researchers are reliably able to design soluble and expressible proteins, challenges including enzyme design still suffer from low experimental success rate. The following work will outline the approaches I have taken and will take to guide generative models with experimental data. The following aims will outline and guide the work toward interrogating the central hypothesis.

Aims:

1. Build generative frameworks amenable for guidance with experimental

data.

2. Design libraries of proteins at scale to empower data generation.

These over arching aims will tie together the elements of my work, and will be referenced back to at times throughout the document.

2 Generative Modeling for Proteins

Generative modeling is a tool used to sample from an underlying distribution often denoted $p(\mathbf{x})$ [25]. When the manifold on which the data ($\mathcal{D}$) exists is difficult to sample from, there are a variety of procedures used to approximate and draw samples from $\mathcal{D}$. Often it is desirable to draw conditional samples $p(\mathbf{x}|y)$ where y is some conditioning information which restricts the sampling space to a subset of valid samples in the distribution which satisfy condition y. Generative modeling often requires some latent variable $\mathbf{z}$ which is often a stochastic parameter which conditions the sampling process. Both general adversarial networks (GANs) and variational autoencoders (VAEs) are of note to this space. These generative frameworks dominated the space of generative modeling for many years. More recently diffusion models have become very popular in the generative modeling space for a number of reasons to be discussed.

2.1 General Adversarial Networks

At a high level GANs are trained by two competing parameterized networks, a generator $G_\theta(\mathbf{z})$ and a discriminator $D_\phi(\mathbf{x})$ where $\mathbf{z}$ is a stochastic latent variable and $\mathbf{x}$ is a data sample [26]. The generator's goal is to propose samples $\hat{\mathbf{x}}$ which fool the discriminator. The generator is optimized by sampling $\hat{\mathbf{x}} \sim G_\theta$ and passing the sample through the discriminator to compute $D_\phi(\hat{\mathbf{x}})$ or the probability with which the generator has "fooled" the discriminator. In practice the loss function to be optimized is $\mathcal{L}_{\text{generator}} = -\log(D_\phi(\hat{\mathbf{x}}))$. The discriminator is tasked with accurately detecting if it has been provided a genuine sample $\mathbf{x} \sim \mathcal{D}$ or a fake $\hat{\mathbf{x}} \sim G_\theta(\mathbf{z})$. A common

optimization for the discriminator is as follows:

$\mathcal{L}_{\text{discriminator}} = -\log(D_\phi(\mathbf{x})) + \log(D_\phi(\hat{\mathbf{x}}))$. This is often referred to as a min-max game as the networks are competing for a similar objective. In practice these networks can be rather unstable to train and while some have been developed for protein generation, it has never been widely adopted.

2.2 Variational Autoencoders

VAEs are a longstanding alternative framework to GANs. In the VAE setup the network is attempting to learn a compression of $\mathbf{x} \sim \mathcal{D}$ to a latent $\mathbf{z}$ whose dimensionality is smaller than x [27]. The architecture consists of an encoder $E_\theta(\mathbf{z}|\mathbf{x})$ and a decoder $D_\phi(\hat{\mathbf{x}}|\mathbf{z})$. Here the encoder projects $\mathbf{x}$ to a with a mean $\mu_\mathbf{z}$ and variance $\sigma_\mathbf{z}$ that parameterize a multivariate Gaussian from which a latent $\mathbf{z}$ is sampled. The latent $\mathbf{z}$ is then passed through the decoder to recover the original input $\mathbf{x}$. Both $E_\theta(\mathbf{z}|\mathbf{x})$ and $D_\phi(\hat{\mathbf{x}}|\mathbf{z})$ are jointly optimized by the following global objective: $\mathcal{L} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{kl}}$. The reconstruction objective is most commonly formulated as the a mean squared error loss $\mathcal{L}_{\text{recon}} = ||\mathbf{x} - \hat{\mathbf{x}}||^2$, with the goal of learning a compressed latent $\mathbf{z}$ which contains the information needed to reconstruct $\mathbf{x}$. In addition a Kullback-Leibler (KL) divergence term is added to enforce that $\mathbf{z}$ is close to a normal Gaussian: $D_{KL}(E_\theta(\mathbf{z}|\mathbf{x}) \parallel \mathcal{N}(0, I))$. Typically a reparameterization trick is used to allow gradients to flow through the sampled latent. This involves predicting $\mu_\mathbf{z}$ and $\sigma_\mathbf{z}$ and then sampling the latent as follows $\mathbf{z} = \mu_\mathbf{z} + \sigma_\mathbf{z}\epsilon$ where $\epsilon \sim \mathcal{N}(0, 1)$. Ideally if the learned latents are close enough to a true multivariate Gaussian, in principle one can sample from the prior and pass through the decoder to sample from the generative

model. VAEs have found value in being applied to protein ensemble generation [28]. This allows a user to encode a latent vector corresponding to a particular state and then draw realizations of other probable ensemble states by sampling from the latent space.

2.3 Denoising Diffusion

More recently the field of image generation has shown significant progress in both unconditional and conditional generation using denoising diffusion probabilistic models (DDPMs), popularized in recent years [19]. These are latent variable models which define latents of the same dimension as the data. One can conceptualize such models as a map between a well known prior distribution which is easy to sample from and a posterior distribution defined over the dataset. A forward process is defined as a Markov chain which adds noise to the data to sample noisy estimates of the data $q(\mathbf{x}_{0:T}|\mathbf{x}_0)$. The forward process is usually defined by a noising schedule which determines the signal to noise ratio found at each step of the forward process. For different data domains there exist optimal choices of the schedule depending on the structure in the data. It is common for $t = 0$ to represent the timestep at which no noise has been added to the data and $t = T$ be the timestep at which the prior is sampled with no information from the data. The noising schedule is defined by parameters β_t, $\alpha_t := 1 - \beta_t$, and $\bar{\alpha}_t := \Pi_{s=1}^{t} \alpha_s$. Various options for β_t have been shown to work, but the authors of DDPM suggest $\beta_0 = 10^{-4}$ and $\beta_T = 0.02$. To sample from $q(\mathbf{x}_{0:T}|\mathbf{x}_0)$ is to compute $\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon$ where $\epsilon \sim \mathcal{N}(0, I)$. During training a timestep t is first sampled, the data is noised accordingly, and in the simplest

objectives the network is tasked with predicting the un-noised data $\mathbf{x}_0$ using a parametrization of $p_\theta(\mathbf{x}_0|\mathbf{x}_t, t)$. In addition to the providing the model with a noisy sample the model also receives the current timestep as conditioning information. The goal of diffusion modeling is to learn the score $\nabla_{\mathbf{x}} \log p(\mathbf{x})$ or the direction in which you should take a step to converge to the data. Most commonly the L2 objective used is of the following form: $\mathcal{L} = ||\mathbf{x}_0 - p_\theta(\mathbf{x}_0|\mathbf{x}_t, t)||^2$. In addition to conditioning on the time step information it is also possible to parametrize such networks to take in conditioning information from other sources, this could be an image label, or in the case of protein modeling a motif you wish to scaffold. Later work also proposed conditioning on the model's previous prediction of the data $\hat{\mathbf{x}}_0$ as this would allow the model a look ahead at what it last predicted. Since the seminal DDPM work, many modifications have been proposed including flow matching implementations which can be even simpler to implement. For the sake of this work I have focused here on diffusion models as they are most relevant. One particularly nice property of the diffusion framework is that it is amenable for gradient based guidance without needing to retrain the model. Classifier based guidance allows for conditional generation by computing gradients through a pretrained classifier [29]. To follow the conditional score, one can use Bayes' rule as has been previously shown to sample from the conditional score: $\nabla_{\mathbf{x}} \log p(\mathbf{x}|y) = \nabla_{\mathbf{x}} \log p(y|\mathbf{x}) + \nabla_{\mathbf{x}} \log p(\mathbf{x})$ where y is some conditioning label. This formulation is especially modular as it allows the pretrained diffusion model to be fixed and classifiers (or regressors) to be swapped in to bias the diffusion trajectories toward the conditional distribution.

2.4 Goals in Protein Generative Modeling

Before diving into my first body of work, I find it necessary to outline the challenges in protein design we are seeking to apply generative modeling to. Although many generative models in the protein design space are often evaluated on benchmarks for unconditional generations, there do not appear to be any use cases for unconditionally generated proteins. Depending on the application the designer has in mind a wealth of conditioning information may be accessible including but not limited to secondary structure, fold topology, structure and/or sequence motifs, amino acid composition, functional labels, experimental data and more. The scope of problems researches are looking to tackle is wide. Most commonly generative models are evaluated using *in silico* oracles such as Alphafold2 to look at "self-consistency" metrics which determine if a sampled protein (sequence and structure) is consistent with the oracles prediction. In the case of motif scaffolding where the structure of the motif is important one can also easily look at how well the oracle recapitulates the motif structure. Motif structure especially at the atom level is imperative to allowing for designed enzymatic function, however even when the geometric constraints are perfectly satisfied there are still dynamic and electronic effects which can impact the activity but are left unaccounted for. Certain topologies may be more optimal for a particular chemistry, and yet it is still necessary to find the most functional sequence for a particular fold. Finding the most functional sequence to fit a particular fold is an outstanding challenge in the space of protein generative modeling. Methods like ProteinMPNN will find sequences which are likely to fold to the correct structure, but generative paradigms have yet to condition on functional data.

2.5 Protein Structural Datasets

When training these generative models the choice of dataset is important. Luckily since the 1970's solved structures of proteins have been deposited into the protein data bank (PDB) and cataloged in an easy to access online database. These structures can be easily downloaded and processed as input for training. Each resolved structure comes with documentation as to how confident the resolution of each individual atom is. This allows for easy filtering to ensure you only train on high confidence structures. The PDB comprises roughly 300,000 individual protein crystal structures. However after clustering by sequence identity, roughly about 40,000 unique clusters are found. In the training of Alphafold2 and many other subsequent design models, a cluster is first sampled and then an item from that cluster. This prevents the overuse of any single protein structure seen during training. In addition to experimentally resolved structures in the PDB, the Alphafold database (AFDB) comprises millions of predicted protein structures with high confidence [30]. Due to the success of structure prediction, this greatly expands the space of potential training examples, and some models have found that training on such a dataset leads to better generalization. As the generative methods developed are described these datasets will be referenced as to what models were on which datasets.

2.6 Diffusion in Protein Sequence Space

One particular advantage of the diffusion as previously alluded too above is the ability for modular gradient based guidance in the framework. While an ongoing parallel effort sought to define the generative process in structure space, defining a diffusion

process in sequence space would allow for unique forms of conditioning and guidance [31], see figure 1a. Since sequence is a discrete modality, defining the diffusion process required some departure from the usual framework. At the time this effort began a few other works had already adapted the diffusion framework for discrete modalities. Some proposed diffusion on a simplex, which restricts the discrete domain to live on a true probability distribution manifold, however we found the proposed "analog bits" representation to be the best fit for our use case [32]. In the work they propose transforming the discretized one-hot space to live on the manifold of $[-1, 1]$ by the transformation $2\mathbf{x} - 1$ where $\mathbf{x}$ represents the one-hot initialization of the sequence. With this transformation it is now possible to sample Gaussian noise and add it directly to the input space as described in the denoising diffusion section above. This relaxed sequence representation is a continuous space amenable for easy gradient based guidance. In addition, this same work also proposed the concept of *self conditioning* where the model is able to condition on its previous prediction of $\hat{\mathbf{x}}_0$ [32]. We found that conditioning on the models previous prediction to be advantageous during training. As the framework for the architecture we used RoseTTAFold repurposing the 1D track as the input for which the noisy sequence would be provided. Importantly no MSAs were provided during training. The 1D track was expanded to allow for extra conditioning information such as a timestep feature, an indicator variable for if a particular position is diffused, and secondary structure conditioning information. In addition to providing sequence information as input, the RoseTTAFold architecture is also capable of conditioing on 3D structure. We used this to our advantage and occasionally provided structural information as conditioning.

We relied on previous works such as LM-diffusion for guidance on selecting a noise schedule and loss function appropriate for discrete data [33]. In the work, the authors suggest using a square-root schedule as it adds more corruption to the input for early time steps. This is especially important for discrete data as it can be easy for the model to cheat by effectively learning to argmax at low timesteps when not enough noise is added. Early on we conducted analysis to demonstrate that using the square-root schedule results in a linear decay of argmax accuracy over the timesteps from $t = 0$ to $t = T$. During training the timestep was sampled uniformly along the interval: $[0, T]$. The LM-diffusion and analog bits work also validate that categorical cross entropy (CCE) loss can be used in-place of an L2 loss as it is more natural for discrete domains. In addition LM-diffusion also suggests using a KL term on the reconstructed $\mathbf{x}_{t-1}$ input. They claim this helps to stabilize training, so we also implement this term in our work. In addition to these losses we also employ a variety of auxiliary losses used in training RoseTTAFold including the frame aligned point error (FAPE) loss, originally implemented by Alphafold, CCE loss for the predicted distogram, bond length and bond angle loss [14]. See algorithm 1 for more details. The training regiment consisted of a variety of tasks to help the model become robust to its understanding of all different kinds of conditioning. We found the self consistency of the sequence-structure pairing to be best when we split time evenly between structure prediction, fixed backbone sequence design, and the normal diffusion task. The structure prediction task is akin to sampling $t = 0$ without structural conditioning or MSAs provided. This can often be a difficult task for the model to learn as single sequence structure prediction without a language model

embedding like ESM is known to be difficult, but we found this to be valuable. Similarly, the fixed backbone sequence design task is as if you were to sample $t = T$, but the complete backbone structure is provided. This single shot sequence design task appeared to increase the quality of designed sequences. Finally, the diffusion task where the sequence is noised accordingly to the sampled timestep. If the diffusion task is sampled, there are two main masking schemes that get used. Half the time no conditioning information is provided, else the span or sparse masking schemes are used at random. For span masking 1-4 contiguous sections (usually between 5-10 residues) of the sequence and structure are provided as conditioning information. In the sparse masking scheme a random assortment of 3-5 residues far apart from each other in sequence space, but close in structure space. The sparse masking task should mimic scaffolding an enzyme active site where minimal information is provided. During training models were evaluated on a variety of tasks including unconditional generation and motif scaffolding.

To sample from the model in inference mode we ran a variety of tests and comparisons to observe that sampling with out direct interpolation to $\mathbf{x}_{t-1}$ worked best, a finding that is consistent with the LM-diffusion work. Beginning by sampling the prior (a normal Gaussian) each model prediction is treated to be an estimate of $\mathbf{x}_0$ and noise is added just as it is done during the forward $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\hat{\mathbf{x}}_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon$, see algorithm 2. Empirically we found that even with only 25 timesteps at inference we are able to achieve quality samples. At the end of the trajectories a final sequence would be gathered by argmaxing the output logits. To validate the unconditional generations from the model, we ordered and synthesized early unconditional samples

and found them to be soluble and monomeric. Each of the proteins ordered was a single gene block fragment that was then cloned into a bacterial expression system and purified. This experimental confirmation gave us confidence in the models ability to generate real proteins. We would later receive more confirmation with crystal structures that the designs behaved as expected in solution. For more details please see Lisanze et. al. [34].

2.7 Applications of Protein Sequence Diffusion Models

There are a variety of methods to guide diffusion models. Classifier guidance serves as a modular approach to conditioning, other approaches such as classifier free guidance rely on conditioning mechanisms which must be provided during training [35]. To demonstrate the promises of parameterizing the generative process in protein sequence space, we implemented potentials to enforce sequence composition. These potentials are akin to adding a force to the diffusion trajectory which pulls it into a conditional distribution that satisfies the amino acid composition. We began by applying simple potentials such as a charge bias to design proteins with a desired net charge, see figure 2d. This was done by simply adding some scaling factor to the output logits before noising. Using this approach we can demonstrate how the model is able to bias the output distribution for a variety of sequence based properties including charge and hydrophobicity index.

To generate proteins of a particular sequence composition a more complex potential was required. Naively we attempted to bias the logits toward a particular amino acid by up weighting along the dimension along a single amino acid in the logit space, but

this strategy would often lead to a form of mode collapse where the sequence devolved to be only the single amino acid for which the potential was applied. Rather what seemed to work better was a scheme where the user defines a fraction of the protein sequence they would like to upweight toward a particular amino acid, and at each step of the denoising trajectory a bias is added to the input fraction of residues which are predicted to most likely be the amino acid of interest. This avoids upweighting the entire sequence equally and instead only weights those positions already likely to end up as the amino acid of interest, see algorithm 3 for more detail. Using this potential scheme we demonstrated the ability for the model to guide toward rare amino acid composition to demonstrate the models ability to sample unique sequences and understand sequence to structure relationships, see figure 2e-h. Looking at the distribution of amino acids for which the model samples after training, it was clear that the distribution of amino acids the model samples is close to that of the native distribution with the exception of a few amino acids for which the model appeared to oversample including glutamate (this is similar to what ProteinMPNN observes as well). For the rare amino acids such as tryptophan, cysteine, methionine, histidine, and valine we employed the sampling procedure to see if the model could sample such sequences. We generated proteins of lengths 80-100 amino acids and biased 20% of the residues to be one of the rare amino acids listed above. When designing proteins with increased tryptophans we find the model is able to pack these large bulky side-chains into the core by ensuring the backbone leaves enough space to do so. Expressing these proteins found that a number of them were soluble with high 280nm absorbance consistent with what would be expected from a protein with many

tryptophans. In the case of unweighting cysteines in some of the designs the model will actually pair the cysteines to ensure they are disulfide bonded. Expression of these constructs revealed a number were soluble and mass spectroscopy revealed that the number of disulfides bonds formed matched the number found in the designs for some. For designs made with increased valine concentration we found that some displayed beta-sheet secondary structure, a finding that is consistent with increased valine content in nature. Circular dichroism analysis of these constructs revealed they had beta-strand character consistent with the designs. Additional circular dichroism analysis for all of these constructs revealed they highly thermostable and do not appear unfolded at temperatures as high as 95^0C. This hyperthermostable effect is consistent amongst other designed proteins.

Next we aimed to design proteins with conserved sequence repeats. Prominent throughout nature, proteins with conserved repeating units are common in biological materials. To design proteins with repeating sequences we symmetrized the noise and diffusion updates along the 1D sequence. For a user input of repeat unit length l and number of repeating units n we featurized an initial sequence of length $L = nl$. At each diffusion the logits in the first repeating unit were copied across all n units so the entire sequence received symmetric updates. We found a number of these proteins to express monomerically and were able get a crystal structure of one of them which demonstrated sub 4 Angstrom accuracy to the design model, see figure 3.

To attempt challenges with a functional application we investigated the model's ability to do structure agnostic sequence motif scaffolding. In the first class of problems, we provided with the model with a library of protein barcodes for it to

scaffold into monomers. Size exclusion chromatography (SEC) traces of the barcoded traces demonstrated the ability of the model to incorporate the barcodes into soluble and monomeric proteins. Protein barcodes, similar to DNA barcodes, allow designs to be tagged unique molecular identifier (UMI) which allows for experiments to be run in pooled based settings. Protein tags are advantageous for many assays where as it is a simpler system to express a protein with a UMI than other tagged systems such as cDNA display (cDNA display is a library preparation protocol where a protein is conjugated to its own DNA template).

In addition we tested the model's ability to design a protein which scaffolded a lytic peptide and a protease tag. When incubated with a protease ideally the lytic peptide would be release and lysis would occur, see figure 4. This system could serve as a precursor to a potential route for endosomal escape. When therapeutics are targeted to a cell they are often encapsulated into an endosomal compartment. One of the many challenges with administering these therapeutics is that drugs may become trapped and degraded in the endosome before they can take effect. To increase the relevance of such a design task we used a furin cleavage site upstream of a melittin peptide. Furin is one of the proteases commonly found in the endosome, and melittin is a common lytic peptide known for its powerful potential to lyse cells. As input to the model we provided the amino acid sequences for furin cleavage site and melittin peptide, as well as 100-120 masked resudeys at the N-terminus. We also provided secondary structure conditioning to scaffold the cleavage site in a loop, and the melittin peptide in a helix. Melittin does not appear to be a structured peptide, so we wanted to ensure it was stable in the scaffolded form. A few of the expressed designs

appeared to be monomeric by SEC and even demonstrated controllable release and lysis when incubated with furin. Lysis was determined by incubating the cleavage products with red blood cells. In a negative control where furin was not added no lysis was observed, hence we were able to achieve conditional lysis.

As a final application of we implemented a framework for multi-state design. In nature proteins which can adopt multiple different states are able to achieve unique functions. This can involve a rigid motion or something more dynamic where a helix can switch to a strand. Multi-state design fits nicely into the sequence diffusion framework as the end goal is to find a single sequence which is capable of adopting multiple structural states. In a structure diffusion framework it is possible to design multiple similar structures, but more difficult to enforce the structure are accessible from the same sequence. While it may not be the most principled implementation we found that we can initialize multiple diffusion trajectories with the same noise and provide different secondary structure conditioning for each. To demonstrate the ability of the model to find a single sequence which can adopt distinct secondary structure states, we formulated a challenge where a parent protein occupying an alpha/beta fold with a cleavage site at the center would cleave into two child proteins both with only alpha helical character. This would require that the residues which make up part of the beta-strand in the parent would need to switch to become alpha helices in the child. During the diffusion trajectories the secondary structure conditioning for the parent and children were full specified. After each denoising step the logits from the child and parent proteins are averaged over their respective indices. The representation across the parent and child proteins is synced before the next step,

see algorithm 4. The designed sequences were then filtered with Alphafold2 to ensure that the secondary structure switch was predicted to occur. A few of the constructs were ordered as separate parent and child genes to express. We were able to find that a subset of the designs appeared to express well and were monomeric by SEC. Next we attempted to quantify the secondary structure content of the proteins on CD, and it was promising to find that the resulting spectra matched with the secondary structures they were predicted to adopt. Working with a collaborator who solves structures with nuclear magnetic resonance (NMR) spectroscopy we sent some of our best candidates for structure determination. We were fortunate that the collaborator was able to solve high resolution NMR structures of two different parent proteins. Both revealed the correct alpha/beta fold that was intended in the design, see figure 5. Unfortunately, they were unable to solve structures of either of the child proteins, but these proteins still do look ordered and alpha-helical by CD. This proof of concept approach looks promising that we able to design fold switching proteins in this framework, but more excitingly this points to a potential transition in protein design. Proteins in solution are hardly static and currently our design frameworks mostly design for a particular state. In the case of enzyme design where a multistep reaction is required, it would be advantageous to be able to design a backbone which is capable of easily transitioning through each step of the reaction. It is important that the model has an explicit understanding of the sequence-structure relationship in order to design a sequence which has the ability to adopt multiple structural states. Current tools are good at being able to design multiple different structures, but there are no constraints to ensure these structures will be able be encoded by a single sequence.

2.8 Guidance with Experimental Data

To explore the ability of sequence space diffusion models to be guided using experimental data, we used a *combinatorially complete* dataset of for an IgG binding protein GB1. Four positions at the binding interface were mutated exhaustively such that every possible combination of four amino acids at the specified locations was sampled in the library. This amounts to $20^4 \approx 160,000$ unique variants. These variants were then assayed for binding activity and collected into a single dataset which maps the complete sequence-fitness landscape for the four specified positions at the interface. In our test experiment we attempt to compare using classifer guidance, to bayesian-optimization (BO) gradient based methods. To give both methods a fair start and equal footing, we initialized the pool of candidates for both methods with a sample of 96 designs proposed by the sequence diffusion model provided the context (structure and sequence) of the target with the positions of interest masked. The *combinatorially complete* dataset allows for any sequence which is proposed by the model to be evaluated. This is especially convenient and more realistic in terms of capturing the epistasis in the landscape. Next with the initial pool of 96 sequences we train a ranking based classifier to guide the diffusion model, and a gaussian process (GP) model for BO. Methods for BO are described later in this document as they pertain to the large scale library design work. Over 3 rounds of both methods proposing 96 sequences per round, the sequence diffusion model out performs basic BO methods in both the max activity of predicted variants and the mean of each pool as seen in figure 1d. The intuition for why the diffusion method should work better for this type of problem, comes from the fact that it has a good sequence to structure

relationship, whereas the BO method has only seen the limited set of sequences it has proposed for evaluation. While BO will select for sequences that appear to have high uncertainty associated with them because there is a chance these sequences can have high activity, the diffusion model is able to ignore a large part of the space for which it knows there is likely to be no function. It is promising to see that these diffusion models are able to enhance large sequence space searches, for reasons we discuss later there are concerns with these types of approaches as they can be cost limiting.

2.9 Atomistic Generative Modeling

Since the development of this initial sequence space diffusion model, the field has made great progress with being able to explicitly model atomistic systems. In Deepmind's Alphafold2, each protein residue is represented by a frame which can be thought of as the plane between the nitrogen, α-carbon, and carbon atoms of the backbone. Predicting frames in some aspects simplifies complexity as a translation and rotation can describe how the frame is oriented in euclidean space. This becomes more complicated however from a diffusion prospective as both the rotations and translations need to be treated as separate modalities with which the diffusion model must learn a joint denoising process. With the advent of Alphafold3 a new architecture overhaul was introduced that does away with the frame representation and instead favors a hierarchical representation of atoms and residues [36]. The computational complexity increases by over an order of magnitude when each atom is explicitly modeled in the system. To avoid this complexity, the team implemented a version of sparse attention where the atom attention operation is chunked into

sequence local patches. The encoding process involves iterations between the sparse atom attention and the full residue level attention. After each atom attention step, the atom representations are pooled into a representation which updates the residue level. Alphafold3 also relies on a diffusion model for the structure generation, conditioned on information from the MSA. In this case the diffusion process is formulated in atom space which removes a lot of the complexity associated with diffusing over the frame representation. Similarly to the feature encoder, each step of the diffusion process alternates between sequence-local attention at the atom level, and all-by-all attention at the residue level. Looking forward the field is interested in being able to adapt these structure prediction architectures for design.

2.10 Latent Diffusion Models

More recently work in the image domain has shifted from parameterizing diffusion models over the raw image modality to diffusing in the latent representation of a VAE [37]. In this case the VAE encoder compresses the data into a latent $\mathbf{z}$ which is a smaller dimensionality than the input data sample $\mathbf{x} \sim \mathcal{D}$. The intuition is that the smaller dimensionality of the latent space will be an easier manifold for the diffusion process to operate. In addition the, albeit small, KL term for VAE's will enforce Gaussian structure, and we know that transformation from a Gaussian prior to another Gaussian should be feasible. Compression in the image domain seeks to remove the perceptual information and retain the semantic information [37]. The perceptual information occupies much of the dimensionality, but does not appear to be necessary for high quality reconstruction because the decoder is able to learn how

to input the perceptual details. When a user is interested in specifying some conditioning information for which an image must be generated it is important those details can be satisfied and it appears as though the diffusion model is able to navigate the rich semantic representation contained in the latent. Latent diffusion models for image have reached state-of-the-art in generation tasks.

In the world of proteins the most challenging conditional design tasks involve finding proteins which are able to bring together atoms in space with specific geometric constraints of the active site. Protein sequences provide a natural way to compress over the number of atoms, but perhaps there is a smarter way to go about such compression so that high resolution can be retained near the active site while the atoms further away are clustered together in patches. The geometric constraints required to scaffold atoms at the active site are highly specific and atoms often need to be within an angstrom of where they are specified. As the number of atoms in the active site begins to scale the geometric constraints become harder and harder to satisfy. If $\mathbf{x} \sim \mathcal{D}$ is the atoms in a protein structure $\mathbf{x} \in \mathbb{R}^{N_{atom} \times 3}$ an ideal autoencoder would be able to compress the atomic representation to a much smaller dimension $\mathbf{z} \in \mathbb{R}^{N_{patch} \times d}$ where $N_{patch} << N_{atom}$ and d is a fixed size second dimension. Compressing to a smaller N_{patch} dimension would allow for all-by-all attention in the latent representation which would be ideal for training a diffusion model over the latent instead of the sequence local attention Alphafold3 implemented to do spare attention over the N_{atom} dimension.

As alluded to above, for active site conditioning it would be nice to preserve high resolution detail for the atoms in the active site, but further away from the active site

atoms can be aggregated together into the same token. In such a scheme one could imagine a Russian doll type approach where concentric shells around the active site dictate how many atoms are aggregated together. One could imagine first computing the min distance of each atom in the system to an atom in the active site, and depending on the distance to the active site will determine how many atoms it gets binned with. While it can be possible to patch non-contiguous atoms into the same patch, it seems more straightforward to patch atoms together with other atoms in along the N_{atom} dimension. The diffusion transformer architecture from Meta is a nice example of how such patching can be done. To find more information on my work here please see Yim et. al [38].

3 Library Scale Protein Design

Currently, the field of protein design is reliably able to design protein rocks, yet we are much more curious about designing functional proteins. Protein function is a highly complex interplay of actions and motions occurring at the molecular level. While the following works for any protein function, enzymes in particular are challenging to engineer [39]. These proteins are highly sensitive and require complex dynamics. For each turnover, the enzyme must bind the substrate effectively pulling it out of solution, transform the molecule in some way (often a multi-step process), and then release the products regenerating the starting state so that it may bind substrate again [40]. This requires the pocket where the small molecule binds to be accessible, and the molecular shape and pattern to accommodate the substrate. In order to bind the substrate we know the protein must be able to make favorable interactions which are energetically favorable. However they must not be too favorable or else the protein may bind the substrate too tightly and will be unable to turnover any product. The complexity of such a problem makes it highly difficult from a design perspective. With the current available tools we are able to confidently predict protein structure from sequence in a matter of seconds, however we do not have any filters which explicitly consider the dynamics of the system [41, 36, 42]. Newer systems are able to approximate molecular dynamics systems, but these methods are too expensive to screen a large batch of *in silico* designs. Structure prediction methods allow us to take a snapshot into the dynamic process in action and make some guess as to what is a state the system may occupy. Some people will argue that by running many

predictions to generate an ensemble we are sampling from the true statistical distribution underlying the potential states that the protein occupies. This can still be rather expensive when needing to run filtering for 10^6 different candidates. We are at the cusp of being able to design functional proteins, yet we will need to collect large scale functional datasets to understand how we can filter more appropriately for the particular function we are interested in, or train new models to filter for function. The Protein Data Bank (PDB) is a wealth of data made publicly available by the organizers who curated protein structures from labs and papers around the world [43]. This even went so far that Nature and Science journals signed on to promise that all structures published in a journal article would be deposited in the PDB. This ended up being a major endorsement of the PDB as these journals are highly sought after. Structural biologists who work directly for the PDB are responsible for the intake and quality assessment of new protein structures. This team of people ensure the data is formatted in a standardized and accessible way. For many years this repository has seen thousands of deposits and has enabled the age of AI structure prediction [14, 16]. Native proteins deposited to the PDB are functional for the most part in that biology automatically selects for functional proteins through the natural forces of evolution, although it is important to note that the function nature selects for is not necessarily well aligned with what designers are interested in. From the perspective of protein design we are capable of designing proteins which fold into a specific 3D structure, but lack a strong ability to know if such a protein will be functional. For this we will need to collect data at scale about function. The difficult part about such a task is being able to discriminate from all designs which fold those that are functional. Data will be

required to power such a feat, and we need an open source web portal where functional data can be deposited for all to use.

To align generative models so they may generate more functional proteins, large scale sequence to function datasets are necessary. Two of the most common approaches to generate large libraries involve error prone polymerase chain reaction (PCR) or gene fragment shuffling [44, 45]. To generate libraries with error prone PCR a promiscuous polymerase is used to amplify a gene of interest such that random mutations will be introduced into the amplified copies of the gene. Careful consideration is taken to control the number of mutations introduced. Too many random mutation often cause deleterious effects, so often only one or two mutations is desirable. On the other hand gene fragment shuffling is often used by both nature and engineers to explore a wider breadth of sequence space [46]. Variable chain recombination in antibodies is a tool the immune system uses to find optimal antibodies to a foreign antigen [47]. Higher order gene reshuffling can also be observed during sexual reproduction where whole genes can migrate to different chromosomes [48]. Inspired by these natural processes strategies for combinatorial library design have often been proposed for redesign and exploration of native proteins. This includes work like SCHEMA where the researchers propose splitting proteins into structural fragments and searching homology databases for other sequences which could be used to recapitulate the same structure in the fragment [49]. With the advent of tools for protein sequence design and folding, it would be ideal to incorporate such filters into the library design process.

It is worth defining the boundaries for which we wish to collect data before focusing on how to adapt library design strategies to new structural filtering tools. From a

protein design perspective, designers are often able to find an array of unique backbones for a particular campaign and it can be difficult to know which of these folds is most optimal for a particular function. Here we refer to this as the *fold optimization* problem. While highly desirable as a goal, collecting the dataset for such a prediction task is difficult and cost prohibitive with current approaches for DNA synthesis. In this problem statement we wish to build a dataset $\mathcal{D}$ for which a model $p_\theta(y|\mathbf{x})$ is able to predict the likelihood of function y for a particular protein fold $\mathbf{x}$. The difficulty in this formulation lies in collecting the data set. With modern day tools we are able to assign many sequences to a particular fold and we know from decades of experimental work that the sequence-functional landscapes are often very rugged [50]. It may be possible to compute the likelihood of function for a particular fold by assigning many sequences to a single backbone and determining what fraction of the assigned sequences are functional. However, to collect such a dataset for enough folds to be able to see good generalization will be cost prohibitive. In my work I focus on the subset problem which is finding the most optimal (functional) sequence for a single fold. This simpler problem is approachable from a library design perspective and may yield insights in the the grandeur problem of finding the most functional fold.

3.1 Sequence Function Landscapes

Before diving into the approaches in library design and optimization it is worth taking a step back to think more concretely about the space in which we are interested in collecting data from. While it is common to visualize the sequence-function landscape as a 3D map where the xy plane is a compressed sequence space representation and

the z axis defines some functional metric we wish to optimize, in reality the space is of high dimensionality and direct visualization is difficult. Figure 6 visualizes sequence-function space as described here. The vast majority of this space is non-functional, any random set of mutations chosen to a particular variant is highly likely to be non-functional given the enormous complexity (20^L where L is the length of the protein, and there are 20 possible amino acids at any given position). Natural evolution is able to explore this space greedily by taking small local steps and only accepting mutations which are not deleterious to function. Through these evolutionary mechanisms, nature has evolved high complex function, but it is unclear what are the boundaries of such a process and what the best way to navigate such a space is from a protein design perspective. Directed evolution (DE) is a foundational platform in protein engineering which aims to mimic the processes of evolution using an error prone polymerase to generated mutations to a target gene of interest [50]. In DE systems a strong evolutionary pressure is needed in order to extract new variants with enhanced properties for a function of interest. The phrase "you get what you screen for" is often used as a term of caution when setting up these screens. If there is an easy way to exploit the system without truly optimizing the protein of interest the system will likely find it. It is not well understood how to predict what this high dimensional landscape will look like, or even how rugged the landscape itself is. Using DE methods we can learn about the ruggedness of the local landscape, while a combinatorial shuffling approach will usually allow for sparser exploration where islands of function may be identified that are un-reachable by directed evolution methods. Recent work redesigning native enzymes has demonstrated that generative

backbone conditioned sequence design models are capable of proposing novel sets of mutations which increase function and solubility over the native equivalent [51]. While in an industrial setting we often are interested in maximizing the turnover rate of a particular enzyme, in a biological context these functional activities for enzymatic processes or signaling events may need to be carefully tuned in order to ensure the equilibrium states do not affect the homeostasis of the system. It should be noted that the most likely strategy to work is one that searches broadly through sequence space first and then performs greedy local search there after on promising candidates.

3.2 Prior Work

Exploring and predicting which mutations are most likely to be functional is a long outstanding challenge. Traditionally single site saturation mutagenesis (SSM) libraries are used to investigate how robust each residue is to all mutations at that site [52]. While this data can be valuable to understand which single amino acid mutations are beneficial, it is difficult to learn anything about the combinations of mutations from such data. When a model is trained on SSM data it can only learn to propose combinations of the most beneficial mutations. Often additivity does describe the way mutations interact with one another, but more complex epispastic effects are often common when combining mutations. Protein language models (PLMs) like Evolutionary Scale Model (ESM) are trained in an unsupervised manner with a masked language model objective to embed and recover sequence information [53]. This work has shown empirically that these models are able to learn about protein families and structures. Researchers have shown such models can be predictive in

some settings of functional mutations, but this has only been evaluated on native protein fitness landscapes for the most part. In a zero-shot setting where we are trying to asses what is the most beneficial mutation these models can ensure we do not propose something that falls out of distribution for the context, however the log-likelihoods associated with the sequences generated do not seem to correlate well with function. Furthermore, when training a supervised learning task on a set of activity data, recent work has shown the embeddings from these models often do not out perform simple one-hot encoded sequences. In the context of *de novo* proteins it is unclear if embeddings from a PLM is even useful.

Recently machine learning assisted directed evolution (MLDE) platforms have become quite popular. Often referred to as lab-in-a-loop pipelines, MLDE strategies involve iterative rounds of data collection, model training, and variant proposal [54, 55]. Two works in particular, Lipsh-Sokolik et. al. and Rapp et. al. employ combinatorial fragment strategies to optimize in quick iterations [56, 57]. In the Lipsh-Sokolik work, they split a native xylanase enzyme into 4 different fragments by structural domain and apply Rosetta based filtering metrics to identify which fragments lead to energetically favorable designs. New sequences from some of the fragments are proposed with Rosetta back-bone design. They screen a few thousand variants, train predictors, and demonstrate that in a follow up library they can use the predictors to enrich for active enzymes. Rapp et. al. take a similar approach in generating a library of fragments split at structural domains. Here they put an emphasis on building cloud lab setups to let agents explore the fitness landscape and attempt to optimize for a more thermo-stable variant. They are able to demonstrate the agents all converge to

similar candidates in the library. Both of these approaches use protein fragments as a strategy to create a large search space, but the space is limited in both cases, and neither are showing improvements in enzyme activity over baseline.

There are many interesting questions which remain to be addressed in this space. How do we build generalizable function predictors? How far do we need to search in sequence space to find higher functioning variants? As a platform to address some of these more foundational questions to sequence-function relationships. In the following sections I will work through and present a combinatorial library design strategy which is amenable to modern designer specific filters, an example of how such an approach can be applied to an enzyme, and a framework for discrete optimization which should allow for efficient search through the library. Other proposed library design strategies such as variational synthesis can generate libraries as large as 10^{15} unique variants, however the technology to synthesize such libraries is inaccessible and requires expensive DNA synthesizers to work [58].

3.3 The Need for Scalable Protein Libraries

Aside from the obvious use-cases already described in which an enzyme or other protein needs to be optimized there are two other reasons this type of work is becoming more and more valuable. As we continue to train better and better oracles we find that while the oracles are often strong predictors of true negatives, they lack the ability to discriminate true and false positives. In figure 7 we can see that the oracle's fitness landscape does have some overlap with the underlying fitness distribution, but there is room for improvement. Ideally the predictive oracles

perfectly overlap with the underlying experimentally determined functions. If this were to be the case it would be possible to train strong zero-shot predictors capable of knowing which of the proposed candidates will be most functional. In a similar effort to how human feedback has allowed LLMs to sound more human, we have the opportunity to further align biomolecular oracles to experimental data by sampling from the space where the oracle is particularly confident. It is trivial to explore non-functional space by generating random mutations, but it is more difficult to discriminate those sequences which are functional from those which are not. Additionally, the throughput of assays is continuing to approach the limits at which we are able to do deep protein sequencing. Some high throughput microfluidics assays or binding assays are able to generate confident data for billions of variants and we would like a platform to be able to generate libraries above the scale of these assays while remaining affordable.

3.4 Combinatorial Fragment Assembly

As protein design filters become increasingly more powerful and high-throughput assays more accessible, there will exist an increasing need to do library scale *de novo* protein design. Furthermore we have recognized that inverse folding models like proteinMPNN do not posses explicit knowledge of the functional space nor do structure prediction networks like alphafold. There is no guarantee any design which passes numerous filtering metrics will be functional. While it is trivial to design a non-functional protein discriminating which proteins are from structure prediction or more traditional Rosetta metrics is a challenge. This marks the need for better

quality datasets which map sequence to function. Such datasets will be vitally important for aligning models to produce functional proteins with fewer samples. High throughput assays using growth based selection, micro-fluidics, or other pull-down techniques can often screen libraries as large as 10^9 variants. In the interest of optimizing and searching through large expanses of protein sequence space controlling the size of the library is highly desirable. Furthermore, since we know most of the sequence space is non-functional, we would like to improve the probability of finding functional designs using novel filtering approaches such as Alphafold.

3.5 Simple Combinatorial Library Design

The following proposed workflow is a strategy which allows the user to control size and enrich the library *in silico* such that some fraction of the library will pass the desired filters. It begins with a protein fold of interest and many proposed sequences designed for the fold using a structure to sequence model. The number of starting sequences will depend on what is feasible space to sample in for the *in silico* filters and what the final desire library size is. The two main parameters which will define the final library size are both the number of fragments which the protein is split N, and the number of sequences ordered for each fragment $K = k_0, ..., k_N$. The total library complexity can be found by finding the product: $\Pi_{i=0}^{N} k_i$ (it can often be easier to get an estimate of the library size when the same number of sequences are ordered for each fragment in which case the complexity is k^N, see figure 8). Next the designed set of sequences are split by the fixed fragment boundaries into separate fragment pools. To sample a new sequence *in silico* a single sequence can be sampled for each

of these fragments and when concatenated together it will from a new possible sequence. Many of these newly assembled sequences are not likely to be *combinatorially compatible*, but a subset will be highly composable such that they are likely to pass the designer filters. To filter down the initial set of fragments to ones that are *combinatorially compatible* an *in silico* evolution strategy can be used. Simply said, over some number of rounds until the target library size and filter pass rate are reached, new sequences from the library are sampled and passed through the desired filters. Once the sequences are processed through the desired filters each fragment will get a score for *combinatorial compatability* computed by finding how many successful designs did a sequence for a particular fragment lead to. High scoring fragments are likely to recombine with others to produce sequences which pass the designer filters. Over many rounds it is possible to find a set of fragments which are highly composable with one another. In some examples the success rate for these libraries can get as high as 99% passing Alphafold2 catalytic residue root mean squared distance (RMSD) within 1Å, see figure 9. Further details are provided in the next section about an example application of the proposed design strategy.

This methodology is rather flexible and amenable to what parts of the proteins require redesign as well as how sparse the final library is. For questions about how in sequence space do we need to search to find functional variants, we can design a library which could interrogate such a question. If the library needs to be focused on the active site, then the fragment library can be focused to a specific area of the protein. One final consideration is to ensure fragments are roughly similar in length. This will make balancing the concentration of each fragment in the assembly step

easy. In addition when using an assembly method like golden-gate assembly (GGA) which relies of four base pair overhangs between fragments, it is important to leave a constant region at the start and end of each fragment to allow for GGA orthogonal overhang design. Online predictive tools can be used to find orthogonal overhangs which are not likely to recombine with one another.

3.6 Online Optimization for Combinatorial Library Design

The simple approach to combinatorial library design described above is appropriate for smaller libraries, but this approach does not scale well to large libraries where it is desirable to have hundreds of options for each fragment. There are two faults with the simple approach, the first being that the computational cost is quite high. In the simple approach we rely on value estimates for the fragments by oversampling the number of sequences to fold such that each fragment is represented in many sequences. This allows for a reasonable estimate on how valuable a single fragment is, but when multiple rounds of folding and selection are required the number of oracle queries becomes quite high. Additionally, if you wish to design on the order of hundreds of fragments, this will require starting with potentially tens-of-thousands of fragments so that the population can be narrowed down appropriated. These limitations necessitated the need for an alterative approach which saves on compute and is able to scale to any desired library size.

Online finetuning of large language models (LLMs) has helped drive the AI revolution in making these models sound more human and achieve better performance on a variety of benchmarks [59, 60]. This kind of online finetuning takes inspiration from

reinforcement learning (RL) principles [61]. Work in the RL field has enabled progress in robotics, but these techniques have also been quite valuable in the LLM space. In the traditional online framework we describe the actor as being parameterized by a policy $\pi_\theta(s)$, where s represents the state feature provided to the policy which it will condition on to propose an action. This policy proposes what the next action to take should be. Early work in this field led to algorithms like REINFORCE which rely on a policy gradient $\nabla_\theta \pi(s)$ to update the policy weights. This gradient is scaled by reward r which should be computed for any action taken in the environment. Traditionally this works in such a way that actions are sampled from the policy, rewards are computed for the proposed actions, and the policy is updated by scaling gradients with respect to the log-likelihood of the action. Intuitively we can see this as reweighting the actions according to their reward. If an action receives a strong reward we should up-weight the likelihood of resampling the action yet again. The policy gradient update $\nabla_\theta \pi(s) r$ notably does not require the reward r to be differentiable. This is especially useful in the context of robotics but is also useful in this case for protein optimization where it can be difficult to compute gradients for some of the metrics which we wish to optimize for. Hallucination style approaches work well for protein optimization when the gradients can be used directly to optimize the sequence, but this requires that the objective function is differentiable.

Recent work in this space has also led to the development of newer algorithms known as the family of proximal policy optimization (PPO) algorithms [62]. These algorithms usually converge much faster than something like REINFORCE because they perform multiple mini-batch updates on a set of data collected from policy

rollout. The name proximal policy comes from the fact that with each update made to the model, the data previously sampled becomes more and more off policy. The policy gradient theorem only holds for data which has been sampled from the policy. Too many updates will often cause the model to collapse. In these series of algorithms care is taken so as the model does not move too far off policy with sequential updates. Other improvements in this space include algorithms such as group relative policy optimization (GRPO) which seek increase the informativeness and lower the variance of the rewards by normalizing among the batch as so:

$$\mathrm{A}_i = \frac{R_{S_i} - \mu_{R_S}}{\sigma_{R_S}}$$

where A_i is the advantage estimate, R_{S_i} is the reward for a sequence in the batch, μ_{R_S} is the group average, and σ_{R_S} is the group standard deviation. Here the advantage is computed relative to the other rewards sampled in the batched rollout, and this has been shown to be a much more effective advantage and is simpler to compute [60]. Advantage estimation can be quite complicated with dedicated value models which need to be trained in addition to the policy. This framework for GRPO greatly simplifies the advantage function.

To adapt this framework to find optimal fragments we see proteinMPNN as the policy. Here the state feature we can think of as the protein backbone and the sequence proteinMPNN decoded as the action. For most enzymes there are many distance constraints we wish to satisfy between atoms in the active site and substrate for which the enzyme will operate on. Structure prediction models such as Alphafold

can be used as an oracle to co-predict protein structure with the ligand so that such measurements can be made. Each distance should be seen as an independent reward which is normalized using an upper and lower bound specified by the user such that the reward is in the range $[0, 1]$ where a value of 1 means the reward is perfectly satisfied. It is possible to take linear combination of many rewards where each reward m_i is assigned a weight w_i and their linear linear combination is as follows:

$$R_{S_i} = \frac{\sum_i w_i m_i}{\sum_i w_i} \tag{1}$$

Specifically we are interested in the learning the best possible fragments. To enforce this we first decode a batch of sequences from proteinMPNN, we then take the batch of sequences decoded and split them into fragments. Next we sample a larger set of sequences by independently sampling a fragment at each position (exactly how it is done in each round for the "simple" approach). Each sampled sequence is then folded and a reward is computed. To aggregate the rewards back to the fragment level we do as we did before and take an average over the sequence rewards R_s for which that fragment f_i was contained as so:

$$R_{f_i} = \frac{1}{|\{S : f_i \in S\}|} \sum_{S : f_i \in S} R_S \tag{2}$$

Then we use the same GRPO style advantage estimate and broadcast it appropriately to the corresponding fragments.

$$\hat{A}_{f_i} = \frac{R_{f_i} - \text{mean}(R_f)}{\text{std}(R_f)} \tag{3}$$

It should be noted that estimating the fragment likelihood in this way does leave room for some error as the likelihood estimates for each residue are always conditioned on the others in the neighborhood where it is decoded, and this means that when the fragments are swapped to other sequences the context is not quite aligned well. During training we optimize the following objective which GRPO builds on from PPO:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \frac{1}{G}\sum_{i=1}^{G}[\min(r_\theta \text{A}_i, \text{clip}(r_\theta, 1-\epsilon, 1+\epsilon)\text{A}_i)] - \beta\mathbb{D}_{kl}(\pi_\theta||\pi_{\text{ref}}) \tag{4}$$

where G is the group or batch of samples, $r_\theta = \frac{\pi_\theta}{\pi_{\theta_{\text{old}}}}$ the ratio of current to old policy, and the KL divergence is defined as $\mathbb{D}_{\text{KL}}(\pi_\theta||\pi_{\text{ref}}) = \frac{\pi_{\text{ref}}}{\pi_\theta} - \log\frac{\pi_{\text{ref}}}{\pi_\theta} - 1$. See algorithm 5 which describes the full online-learning loop in more detail. Deviating slightly from the traditional objective, $\pi_{\theta_{\text{old}}} = \pi_{\text{ref}}$ such that the updated parameters are discouraged from moving far way from the original policy. We take the base proteinMPNN model to be π_{ref} and use this to compute both r_θ and $\mathbb{D}_{kl}(\pi_\theta||\pi_{\text{ref}})$. Although there is some error in the estimates, they are still quite good and after around 100 iterations strong convergence is usually observed. Something of note is that if care is not taken to restrict the sequence space to search in the model will align itself to adversarial solutions which do align themselves strongly to the oracle of choice. There are two strategies we employ to combat this, the first is using a KL divergence term to the original model so that the parameters do not deviate too much from the original proteinMPNN model. Secondly, we also find that adding an extra reward to minimize the distance between the output sequences and the reference sequence is also quite powerful. In figure 14 it can be seen that while optimizing

specifically with these extra constraints to the reference sequence does not affect the other metrics. When restricting the search space to be within some edit distance from the reference (15-30 mutations) ensures the output sequences will fold well in multiple independent oracles. Future work on this front should include how we can do this optimization without the need for an explicit reference sequence.

3.7 Experimental Pipeline Evaluation

Before proceeding to examples and use cases for such libraries, we will go through how this platform enables rapid iteration. The first thing to notice is that since each of the proteins are split up into multiple fragments this means that we will use a method like GGA, as previously described, to combine several gene fragments together into a specific variant from the library. In practice, DNA can be purchased online from a synthesis company and prepared in such a way that on the shipped plate a unique DNA fragment will exist in each well. Most of the DNA synthesis providers will allow packaging in an ECHO (an acoustic liquid handling robot) compatible plate as an option. The ECHO is an acoustic liquid handling device which can transfer small volumes $< 5\mu$L between plates. This is very useful because instead of combining the desired fragments together manually, we can simply make a protocol and feed it to the ECHO. Since we know where each DNA fragment is located on the source plate, we know what sequence will be assembled in each well on the destination plate without having to do any sequencing. Avoiding sequencing in this step greatly saves on time and resources. Once the genes have been assembled with GGA we then amplify the plasmids with either PCR or rolling circle amplification (RCA). Both options are

plausible, however PCR requires the use of a thermocycler. This inherently bottlenecks the process, however RCA is isothermal. A bacterial polymerase locks onto the plasmid and continues to circle and amplify the plasmid repeatedly at a standing temperature. By using this RCA style reaction we are able to batch multiple plates together in a single run and increase the through put. Next, to express the protein of interest a small amount of amplified DNA can be combined with an off-the-shelf cell free protein synthesis kit. These kits have become much more common in recent years, and have proven themselves to be quite robust. Finally, once the protein has been expressed we can add the fluorogenic substrate directly to the plate and take a kinetic read on a plate reader (such as a NEO2) to observe the catalytic rate of the enzymes. The readout for the assay must be amenable to fluoresence or some other medium throughput screen compatible with cell free. Ultimately, for some of these libraries that require larger protein concentration, it is also possible to do this assay in bacteria by transforming the GGA product.

When done with the cell free prep, this whole process can be completed in a single day. This timeline is substantially faster than a DE which may take weeks to run plus a few extra weeks to account for sequencing and library preparation, granted many more samples are being tested in that case. Another important aspect to consider is biological replicates and controls. In most of the presented studies below we consider 3 replicates for each of the tested enzymes. Additionally, a fluorescent tag on the c-terminus of the protein allows for normalization with respect to the amount of protein in the reaction. This allows us the ability to estimate k_{cat} as the substrate is in excess and we know the protein concentration. Before considering the data usable,

we adhere to strict data quality standards. First each replicate must have some baseline level of protein expression ($> 0.5\mu$M), if so the background reaction rate (measured alongside each batch) will be subtracted from the measured rate and then normalized by the protein concentration. If the normalized rates have coefficient of variation ($\frac{\sigma}{\mu}$) less that 1 we consider the measurement to be valid. These considerations are highly important for the use of this data to train models. Quality data is most important for training quality predictors.

3.8 Hydrolase Library Design

As a test case for this library design approach, I worked with collaborators on sequence redesign campaign of a *de novo* designed serine hydrolase. The collaborators use a substrate which becomes fluorescent upon cleavage allowing for easy detection. While the k_{cat} of the enzyme is far below a typical native hydrolase (which can commonly reach values of 10^3, the activity is detectable and amenable for a medium throughput screening method [63]. The initial goal is to demonstrate the ability for this library design strategy to generate a search space, collect data from the library, and use it to predict a better set of fragment combinations to sample. Targeting a library size of 10^5 seemed reasonable as it would be possible to collect data for close to 1% of the library and use the data to train some models. The designers of the original enzyme suggested filtering the library on two metrics: low RMSD of the catalytic residues in the active site with Alphafold2, and pre-organization of the active site using a generative prediction tool for the small molecule and active site residues, conditioned on the backbone and sequence. The 160 residue enzyme was split into 4

fragments each approximately 60 amino acids in length. After several rounds of *in silico* library evolution a final library size of $\approx 300,000$ was reached with an estimated 98% of the library passing the active site RMSD filter, and 75% passing the pre-organization filter. To collect 1% of the ordered library $\approx 3,000$ random samples the following protocol was used. Step 1, use GGA to connect DNA fragments together. Step 2, amplify DNA with rolling circle amplification. Step 3, express protein with cell-free extract. Step 4, Add substrate and record fluorescence over time. This protocol was optimized to run in a 384 well plate format using an acoustic liquid handler to do the majority of the transfers. Especially for the fragment assembly step, having an acoustic liquid handler is helpful. For each variant, data across three biological replicates was collected (these replicates were formed at the expression step).

Once all the raw data had been collected, it was processed with two filters, one for expression and another for measurement consistency. The vector to which the design was cloned into has a c-terminal mscarlett tag which enables an estimate of protein concentration. This allows us to both normalize the measured rate by the concentration to get an estimate of k_{cat} and it also allows us to gate on expression. If all replicates for a single variant had at least 0.75uM of protein the variant would pass the first filter. Next the background rates were subtracted, and the concentration normalized by the estimated protein concentration. If the values for the replicates had an coefficient of variation $\frac{\sigma}{\mu} < 0.9$ the replicates are consistent and the data point is considered valid. After filtering the we were left with roughly 2,500 data points. This cleaned data was used as the training set. Two questions we were most interested in

answering with this dataset were: can we train a model that learns the sequence to function relationship? Is there a significant amount of epistasis in the landscape? To address these questions we trained two types of models, both multi-layer perceptrons (MLPs) and linear models to regress the normalized activity measurements. To split the data for training and testing, we clustered the data at 75% sequence identity and held out ≈ 250 examples for validation (drawn as full clusters). A linear model will not be able to reason about any context dependent affects in the input, as it is only able to assign a single weight to each mutation, so if the linear model is able to preform as well on the held out set as the MLP there is unlikely to be much epistasis in the landscape. Interestingly, both models performed just as well on the dataset achieving PearsonR scores of ¿0.8 consistently. To investigate how well the models would perform when evaluating the rest of the library, we tested the top 100 ranked predictions for both the linear and MLP models. It was clear that the using the models to filter the predictions results in a shifted distribution and even a variant which appears to have 6x higher activity than anything in the random sample, see figure 11. Interestingly, both the MLP and linear models appear to have strong predictive power when evaluated across the range of activities by the respective predictor as seen in figure 10. This is strong evidence that the design space is mostly linear and gives us confidence to continue designing larger spaces for search and optimization.

3.9 Click Chemistry Enzyme

In a separate effort to display the use of this method, we designed a library of fragments to optimize an enzyme capable of performing click chemistry. Click chemistry is a bioorthogonal mechanism between an alkyne and azide which results in the formation of a cyclic ring [64]. This kind of chemistry can be quite desirable for applications where we would like to selectively label something in a cell. This reaction is normally catalyzed with copper, but in a display of enzyme design a collaborator has developed an active enzyme for a click reaction. The enzyme is rather slow and successive rounds of partial diffusion and proteinMPNN redesign have failed to produce any active variants at all. After testing close to 500 unique redesigns with complete failure, we designed a fragment library by starting with many sequences from protein proteinMPNN and sub-selecting the population to pick out the top fragments which are *combinatorially compatible* just as was done for the hydrolase. To design the library we optimized for a few different objectives by folding the protein with the azide substrate, the alkyne substrate, and the product. When these three predictions are aligned, ideally the products and reactants overlap in the pocket of the enzyme. There is not particular catalytic residue responsible for the chemistry, but if both substrates are able to find there way into the pocket then the reaction will proceed if the distance between the azide and alkyne is compressed beyond the vanderwaals radius. This could be caused by natural fluctuations and dynamics of the protein. In this case we had already observed from the hydrolase library that the majority of the space could be explained by a simple linear predictor. With that knowledge instead of sampling random combinations, each fragment was tested independently

with the other parent designed fragments to understand the effect each fragment has on activity directly. In the next round we sampled combinations of the best fragments as previously seen. At this stage we could have trained a predictor using the data from the initial screening of each fragment independently, but the best the model can do is combine those fragments which worked best independently. Rather we sampled combinations of the best fragments to test in the following round. Then we train the predictor on both the first order and higher order assemblies to rank everything in the library for the first round of model predictions there after. At this round we can find variants which are 6-7 fold improved over the parent 12. These enzymes are close to being on par (or potentially exceed) the rate at which copper is able to catalyze this reaction at the same concentration as the protein.

Interestingly if we look at a comparisons of the fastest observed enzyme and the original models, we see that the original enzyme appears to have placed the substrates in such a position that they overlay better with the product, see figure 13. This is yet another example of how the filters we decide on can help point us in the direction of fucntion, but they are by no means perfect and we are still a ways out from understanding how to determine function. This further points to the need for better alignment of our oracles.

3.10 PETase Enzyme

One of the most widely recognized enzymes for plastic degradation, PETases were discovered in bacteria which could subsist off the polymer alone as a carbon source [65]. Since the initial discovery there has been a lot of work and effort to optimize

these enzymes for speed and thermostability [66]. A combination of rational engineering to introduce disulfide bonds and DE to optimize for thermostability and activity has led to improvements in these enzymes. Yet with all of these engineering strategies the enzymes still do not express very well. *De novo* designed proteins usually express well, and recently initial attempts to design enzymes which cleave PET have proven successful, yet these designs are much slower compared to their native counterparts. These *de novo* designs express well so if their activity was elevated to levels near the natives it would be greatly beneficial. Furthermore the enzymes are usually more robust to high temperatures and pressures where they will be used industrially.

After having evaluated smaller libraries for both the hydrolase and click-enzyme, we sought to dramatically scale up the size of the library to 10^{15} to increase our likelihood of finding an enzyme which is orders of magnitude more active than our starting construct. To optimize for such a construct we use the online learning framework to co-optimize for multiple objectives including the distance between the acyl oxygen on the PET polymer and two backbone oxyanion contacts (< 3.5Å), histidine nitrogen (NE2) to ester oxygen on PET polymer (< 4.0Å), histidine nitrogen (NE2) to serine nucleophilic oxygen (< 2.5Å), RMSD of predicted ligand to docked ligand after aligning on the backbone atoms (< 1.0Å), RMSD of active site residues after aligning on backbone atoms (< 1.0Å), iPTM confidence (> 0.8), PTM confidence (> 0.8), active site atom residue pLDDTs (> 0.8), and distance to the reference sequence (< 20 mutations). Each of these objectives requires the input of an upper and lower bound for which the metric is normalized. Once normalized we can

then subtract 1 from the metric if we intend to decrease it (e.g. RMSD). The distance to the reference sequence seems to be especially important as this ensures the output sequences are not adversarial. This proves to be a challenge without the distance to reference sequence constraint. Without this constraint the optimization objective may find sequences which appear to be adversarial to the oracle as they do not fold well in other structure prediction oracles not used in the online optimization loop. In the case for this design run Alphafold3 was used as the oracle to optimize against, and designs were folded in both RF3 and Boltz to ensure the sequences are not adversarial after optimization. This is certainly an area of future work, as it would be helpful to have a more robust framework that will avoid finding adversarial sequences. In figure 14 we can see that it is possible to find sequences which satisfy the necessary objectives while constraining the search space to be closer to the original reference sequence. When the final sets of sequences are sampled, we can see that some sequences are able to predict well in Alphafold3, boltz, and RF3 15.

INSERT PETASE RESULTS HERE

3.11 Approaches for Iterative Optimization

Many are focused on building better frameworks for protein optimization from a search perspective. The goal of this work is to spend time engineering the space so that the search becomes easy. However as the size of the space grows and the complexity increases it is worth while understanding the various frameworks available for search. Bayesian Optimization (BO) is one of the most notorious frameworks for black-box style optimization relying on uncertainty estimates to propose the next set

of candidates to test [67]. While other frameworks such as active learning seek to purely reduce uncertainty in the model, BO deals with the exploration-exploitation trade off to find the global optimum in the search space [68]. This trade off of explorations vs. exploitation is quite central to the principles of BO. In a pure exploration setting we would only care about reducing uncertainty, while in an exploitation only setting we risk becoming overconfident in our models predictions. It is important to be able to balance both of these in order to do efficient search well. More formally stated the goal of BO is to find $\max_{\mathbf{x}\in\mathcal{X}} f^*(\mathbf{x})$ where $f^*(\mathbf{x})$ is a non differentiable function we do not have access to. By initially collecting a sample of data points from the space of possible inputs $\mathcal{X}_{\text{pool}}$ and their respective evaluations $f^*(\mathbf{x})$ we can now train a surrogate function $f_\theta(\mathbf{x})$. It is important that for each prediction $f_\theta(\mathbf{x})$ both a mean $\mu_\mathbf{x}$ and variance $\sigma_\mathbf{x}$ are output. That is to say that the surrogate function must have some notion of uncertainty. For this reason Gaussian Process (GP) models are a popular choice. These are usually simple models which parametrize each point with a Gaussian and use a kernel function to estimate how similar the data point is to all other data points in $\mathcal{X}_{\text{pool}}$ [69]. Data points which are similar should also have lower uncertainty associated with them as we would expect the kernel function to operate such that similar inputs have similar measured values from $f^*(\mathbf{x})$. While a GP is a common framework used to get uncertainty estimates associated with predictions, these models do not scale well to large datasets and high-dimensional input spaces (usually input dimensions < 20 are desirable). In the case of protein sequence-function data the datasets can be large and the inputs are often high dimensional. More recently deep ensembles have been widely popularized

because they are simple to train and do not require any specification of a similarity kernel [70]. Each model in the ensemble predicts mean $\mu_{\mathbf{x}}$ and variance $\sigma_{\mathbf{x}}$ which are learned by the negative log likelihood objective: $-\log(y|\mathbf{x}) = \log(\sigma_{\mathbf{x}}) + \frac{(y-\mu_{\mathbf{x}})^2}{\sigma_{\mathbf{x}}}$. The negative sign is just semantics as most optimizers are setup to minimize the function of interest.

Using the learned surrogate function, an acquisition function can then be used to find the next best points to acquire. At each round of iterative optimization some new candidates will be proposed to test, and past work has found that certain acquisition functions tend to minimize long term regret. That is to say that by finding the samples which maximize the acquisition function you should be able to converge to the global optimum quickly. These acquisition functions take the general form $\alpha(\mathbf{x}, f_\theta) \in \mathbb{R}$, where the returned value is a scalar. Two of the most common acquisition functions are upper confidence bound (UCB) which is simply $\mu_{\mathbf{x}} + \sigma_{\mathbf{x}}$ and lower confidence bound (LCB) which is $\mu_{\mathbf{x}} - \sigma_{\mathbf{x}}$. UCB can be seen as an optimistic estimator where as LCB is a pessimistic estimator. Other acquisition functions include probability of improvement, expected improvement, knowledge gradient, and more. There are many extensions which deal with noisy data as well. In addition there exist multi-objective acquisition functions such as expected hyper volume improvement which seeks to expand the Pareto frontier. While traditionally these acquisition functions return a single sample to evaluate next, in many settings such as a biological assay it is often easier to test a batch of points in each round. Many of these acquisition functions have been adapted to the batched setting where we care about finding a diverse set of sample which maximize the acquisition function [71].

Considering the diversity among the set of samples to be tested is important as it would be undesirable to evaluate many similar samples in parallel as this goes against the notion of input locality (similar inputs are more likely to have similar experimental outcomes).

3.12 Optimizing in a Discrete Space

In the case of a protein sequence space where we wish to optimize, each sequence in the space is discrete. This also applies to the fragment libraries proposed in the preceding sections. Fortunately, previous theoretical work outlines a framework for optimizing an acquisition function over a discrete input space. In their work they make an interesting connection to the field of reinforcement learning (RL), specifically in how a policy π_ϕ can be learned to find a set of optimal actions to take [72]. Similarly, we can think of the proposed sequences we wish to test next as an "action" and so we wish to find the optimal policy π^*_ϕ which corresponds to the best set of sequences to test next. To find such an optimal policy we can use a policy gradient where the reward function is the acquisition function:
$\nabla_\phi \mathbb{E}_{\mathbf{x}\sim\pi_\phi}[\alpha(\mathbf{x}, f_\theta)] \approx \frac{1}{N}\sum_{i=1}^{N} \alpha(\mathbf{x}_i, f_\theta)\nabla_\phi \log \pi_\phi(\mathbf{x}|\phi)$. Notably, optimizing in this discrete framework does not require you to compute gradients through the acquisition function. In traditional BO gradients are computed through the acquisition function following: $\nabla_{\mathbf{x}}\alpha(\mathbf{x}, f_\theta)$, however in the policy gradient case this is not necessary. In my work I have demonstrated, just as the original authors do on their paper, that optimizing the acquisition function using this policy gradient strategy consistently yields high valued batches when compared to optimizing the traditional BO

acquisition function objective in a relaxed discrete space. Not needing to compute the gradient directly through the acquisition function means the surrogate model can use features which are not differentiable with respect to the input such as Rosetta energies, Alphafold metrics, distances between various atoms in a predicted structure, or any other metrics one wishes to compute.

3.13 Limitations

It should be noted that it is unclear at this moment if optimizing for sequences which are *combinatorially compatible* restricts us to a part of the sequence space which will never sample a set of mutations capable of increasing the enzyme activity by multiple orders of magnitude. More generally it is unclear if the ability to match the speed of native enzymes is solely a problem in sequence space or of protein backbone space. The most likely answer is that both factors are important, but we do not yet understand the complex interplay allowing us to boost the activity of *de novo* enzyme activity by orders of magnitude. Other limitations of this approach include the need to start with an enzyme with some activity, as without this reference sequence we are poised to find adversarial ones.

3.14 Future Directions

In continuing this work, while not described in detail here, some of my work has focused on ways to generate pools of designs to test each round as opposed to individual variants. This workflow is especially powerful for high-throughput assays capable of evaluating millions to billions of sequences. Another area of interest for

continued work is adapting the online optimization framework presented to be more robust to adversarial attack. It is clear that we can easily find adversarial solutions when we optimize without constraints to a reference sequence. While this is fine in the case where we wish to explore around a single enzyme, it would be valuable to use such optimization tools more generally to find sequences which satisfy the constraints designers need. Optimizing on multiple backbones simultaneously, stronger KL, or using multiple oracles are all potentially strategies which could help mitigate the adversarial effects observed.

References

[1] F Crick. Central dogma of molecular biology. *Nature*, 227(5258):561–563, August 1970.

[2] J L Patton. Evolutionary mechanisms. *Science*, 216(4543):287–288, April 1982.

[3] P. Bork, L. Holm, and C. Sander. The immunoglobulin fold. *Journal of Molecular Biology*, 242(4):309–320, September 1994.

[4] Inger Andersson and Anders Backlund. Structure and function of rubisco. *Plant Physiology and Biochemistry*, 46(3):275–291, 2008.

[5] David L Stern. The genetic causes of convergent evolution. *Nature Reviews Genetics*, 14(11):751–764, 2013.

[6] Bernd Gerhartz, Andre J Niestroj, and Hans-Ulrich Demuth. Enzyme classes and mechanisms. *Proteinase and peptidase inhibition: recent potential targets for drug development. London and New York: Taylor and Francis*, pages 1–20, 2002.

[7] MS Smyth and JHJ Martin. x ray crystallography. *Molecular Pathology*, 53(1):8, 2000.

[8] Lewis E Kay. Nmr studies of protein structure and dynamics. *Journal of magnetic resonance*, 213(2):477–491, 2011.

[9] Christopher M Dobson. Protein folding and misfolding. *Nature*, 426(6968):884–890, 2003.

[10] Carol A Rohl, Charlie EM Strauss, Kira MS Misura, and David Baker. Protein structure prediction using rosetta. In *Methods in enzymology*, volume 383, pages 66–93. Elsevier, 2004.

[11] Ora Schueler-Furman, Chu Wang, Phil Bradley, Kira Misura, and David Baker. Progress in modeling of protein structures and interactions. *science*, 310(5748):638–642, 2005.

[12] Po-Ssu Huang, Yih-En Andrew Ban, Florian Richter, Ingemar Andre, Robert Vernon, William R. Schief, and David Baker. Rosettaremodel: A generalized framework for flexible backbone protein design. *PLOS ONE*, 6(8):1–8, 08 2011.

[13] Brian Kuhlman, Gautam Dantas, Gregory C Ireton, Gabriele Varani, Barry L Stoddard, and David Baker. Design of a novel globular protein fold with atomic-level accuracy. *science*, 302(5649):1364–1368, 2003.

[14] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.

[15] David J Thomas, Georg Casari, and Chris Sander. The prediction of protein contacts from multiple sequence alignments. *Protein Engineering, Design and Selection*, 9(11):941–948, 1996.

[16] Minkyung Baek, Frank DiMaio, Ivan Anishchenko, Justas Dauparas, Sergey Ovchinnikov, Gyu Rie Lee, Jue Wang, Qian Cong, Lisa N Kinch, R Dustin

Schaeffer, et al. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373(6557):871–876, 2021.

[17] Jue Wang, Sidney Lisanza, David Juergens, Doug Tischer, Joseph L Watson, Karla M Castro, Robert Ragotte, Amijai Saragovi, Lukas F Milles, Minkyung Baek, et al. Scaffolding protein functional sites using deep learning. *Science*, 377(6604):387–394, 2022.

[18] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5505–5514, 2018.

[19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[20] Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J Ragotte, Lukas F Milles, Basile IM Wicky, Alexis Courbet, Rob J de Haas, Neville Bethel, et al. Robust deep learning–based protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022.

[21] Po-Ssu Huang, Scott E Boyken, and David Baker. The coming of age of de novo protein design. *Nature*, 537(7620):320–327, 2016.

[22] Aaron Chevalier, Daniel-Adriano Silva, Gabriel J Rocklin, Derrick R Hicks, Renan Vergara, Patience Murapa, Steffen M Bernard, Lu Zhang, Kwok-Ho Lam,

Guorui Yao, et al. Massively parallel de novo protein design for targeted therapeutics. *Nature*, 550(7674):74–79, 2017.

[23] Roger A Sheldon and Dean Brady. Streamlining design, engineering, and applications of enzymes for sustainable biocatalysis. *ACS Sustainable Chemistry & Engineering*, 9(24):8032–8052, 2021.

[24] Baotong Zhu, Dong Wang, and Na Wei. Enzyme discovery and engineering for sustainable plastic recycling. *Trends in biotechnology*, 40(1):22–37, 2022.

[25] Ruslan Salakhutdinov. Learning deep generative models. *Annual Review of Statistics and Its Application*, 2(1):361–385, 2015.

[26] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[27] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[28] Sanaa Mansoor, Minkyung Baek, Hahnbeom Park, Gyu Rie Lee, and David Baker. Protein ensemble generation through variational autoencoder latent space sampling. *Journal of Chemical Theory and Computation*, 20(7):2689–2695, 2024.

[29] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[30] Mihaly Varadi, Stephen Anyango, Mandar Deshpande, Sreenath Nair, Cindy Natassia, Galabina Yordanova, David Yuan, Oana Stroe, Gemma Wood, Agata Laydon, et al. Alphafold protein structure database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic acids research*, 50(D1):D439–D444, 2022.

[31] Joseph L Watson, David Juergens, Nathaniel R Bennett, Brian L Trippe, Jason Yim, Helen E Eisenach, Woody Ahern, Andrew J Borst, Robert J Ragotte, Lukas F Milles, et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976):1089–1100, 2023.

[32] Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*, 2022.

[33] Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35:4328–4343, 2022.

[34] Sidney Lyayuga Lisanza, Jacob Merle Gershon, Samuel W K Tipps, Jeremiah Nelson Sims, Lucas Arnoldt, Samuel J Hendel, Miriam K Simma, Ge Liu, Muna Yase, Hongwei Wu, Claire D Tharp, Xinting Li, Alex Kang, Evans Brackenbrough, Asim K Bera, Stacey Gerben, Bruce J Wittmann, Andrew C McShan, and David Baker. Multistate and functional protein design using RoseTTAFold sequence space diffusion. *Nat. Biotechnol.*, 43(8):1288–1298, August 2025.

[35] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[36] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J Ballard, Joshua Bambrick, et al. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, pages 1–3, 2024.

[37] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[38] Jason Yim, Marouane Jaakik, Ge Liu, Jacob Gershon, Karsten Kreis, David Baker, Regina Barzilay, and Tommi Jaakkola. Hierarchical protein backbone generation with latent and structure diffusion, 2025.

[39] Jason Yang, Francesca-Zhoufan Li, and Frances H. Arnold. Opportunities and challenges for machine learning-assisted enzyme engineering. *ACS Central Science*, 10(2):226–241, 2024.

[40] Siyuan Du, Rachael C. Kretsch, Jacob Parres-Gold, Elisa Pieri, Vinícius Wilian D. Cruzeiro, Mingning Zhu, Margaux M. Pinney, Filip Yabukarski, Jason P. Schwans, Todd J. Martínez, and Daniel Herschlag. Conformational ensembles reveal the origins of serine protease catalysis. *Science*, 387(6735):eado5068, 2025.

[41] Saro Passaro, Gabriele Corso, Jeremy Wohlwend, Mateo Reveiz, Stephan Thaler, Vignesh Ram Somnath, Noah Getz, Tally Portnoi, Julien Roy, Hannes Stark, David Kwabi-Addo, Dominique Beaini, Tommi Jaakkola, and Regina Barzilay. Boltz-2: Towards accurate and efficient binding affinity prediction. *bioRxiv*, 2025.

[42] Nathaniel Corley, Simon Mathis, Rohith Krishna, Magnus S. Bauer, Tuscan R. Thompson, Woody Ahern, Maxwell W. Kazman, Rafael I. Brent, Kieran Didi, Andrew Kubaney, Lilian McHugh, Arnav Nagle, Andrew Favor, Meghana Kshirsagar, Pascal Sturmfels, Yanjing Li, Jasper Butcher, Bo Qiang, Lars L. Schaaf, Raktim Mitra, Katelyn Campbell, Odin Zhang, Roni Weissman, Ian R. Humphreys, Qian Cong, Jonathan Funk, Shreyash Sonthalia, Pietro Liò, David Baker, and Frank DiMaio. Accelerating biomolecular modeling with atomworks and rf3. *bioRxiv*, 2025.

[43] Helen M. Berman, John Westbrook, Zukang Feng, Gary Gilliland, T. N. Bhat, Helge Weissig, Ilya N. Shindyalov, and Philip E. Bourne. The protein data bank. *Nucleic Acids Research*, 28(1):235–242, 01 2000.

[44] Nikolaos E Labrou. Random mutagenesis methods for in vitro directed enzyme evolution. *Current Protein and Peptide Science*, 11(1):91–100, 2010.

[45] Andreas Crameri, Sun-Ai Raillard, Ericka Bermudez, and Willem PC Stemmer. Dna shuffling of a family of genes from diverse species accelerates directed evolution. *Nature*, 391(6664):288–291, 1998.

[46] W P Stemmer. Dna shuffling by random fragmentation and reassembly: in vitro recombination for molecular evolution. *Proceedings of the National Academy of Sciences*, 91(22):10747–10751, 1994.

[47] David G Schatz and Yanhong Ji. Recombination centres and the orchestration of v (d) j recombination. *Nature reviews immunology*, 11(4):251–263, 2011.

[48] James F Crow. Advantages of sexual reproduction. *Developmental genetics*, 15(3):205–213, 1994.

[49] Michelle M Meyer, Lisa Hochrein, and Frances H Arnold. Structure-guided schema recombination of distantly related β-lactamases. *Protein Engineering, Design and Selection*, 19(12):563–570, 2006.

[50] Philip A Romero and Frances H Arnold. Exploring protein fitness landscapes by directed evolution. *Nature reviews Molecular cell biology*, 10(12):866–876, 2009.

[51] Kiera H Sumida, Reyes Núñez-Franco, Indrek Kalvet, Samuel J Pellock, Basile IM Wicky, Lukas F Milles, Justas Dauparas, Jue Wang, Yakov Kipnis, Noel Jameson, et al. Improving protein expression, stability, and function with proteinmpnn. *Journal of the American Chemical Society*, 146(3):2054–2061, 2024.

[52] Lei Zheng, Ulrich Baumann, and Jean-Louis Reymond. An efficient one-step site-directed and site-saturation mutagenesis protocol. *Nucleic acids research*, 32(14):e115–e115, 2004.

[53] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, et al.

Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.

[54] Nathan C. Frey, Isidro Hötzel, Samuel D. Stanton, Ryan Kelly, Robert G. Alberstein, Emily Makowski, Karolis Martinkus, Daniel Berenberg, Jack Bevers, Tyler Bryson, Pamela Chan, Alicja Czubaty, Tamica D'Souza, Henri Dwyer, Anna Dziewulska, James W. Fairman, Allen Goodman, Jennifer Hofmann, Henry Isaacson, Aya Ismail, Samantha James, Taylor Joren, Simon Kelow, James R. Kiefer, Matthieu Kirchmeyer, Joseph Kleinhenz, James T. Koerber, Julien Lafrance-Vanasse, Andrew Leaver-Fay, Jae Hyeon Lee, Edith Lee, Donald Lee, Wei-Ching Liang, Joshua Yao-Yu Lin, Sidney Lisanza, Andreas Loukas, Jan Ludwiczak, Sai Pooja Mahajan, Omar Mahmood, Homa Mohammadi-Peyhani, Santrupti Nerli, Ji Won Park, Jaewoo Park, Stephen Ra, Sarah Robinson, Saeed Saremi, Franziska Seeger, Imee Sinha, Anna M. Sokol, Natasa Tagasovska, Hao To, Edward Wagstaff, Amy Wang, Andrew M. Watkins, Blair Wilson, Shuang Wu, Karina Zadorozhny, John Marioni, Aviv Regev, Yan Wu, Kyunghyun Cho, Richard Bonneau, and Vladimir Gligorijević. Lab-in-the-loop therapeutic antibody design with deep learning. *bioRxiv*, 2025.

[55] Kadina E Johnston, Patrick J Almhjell, Ella J Watkins-Dulaney, Grace Liu, Nicholas J Porter, Jason Yang, and Frances H Arnold. A combinatorially complete epistatic fitness landscape in an enzyme active site. *Proceedings of the National Academy of Sciences*, 121(32):e2400439121, 2024.

[56] R. Lipsh-Sokolik, O. Khersonsky, S. P. Schröder, C. de Boer, S.-Y. Hoch, G. J.

Davies, H. S. Overkleeft, and S. J. Fleishman. Combinatorial assembly and design of enzymes. *Science*, 379(6628):195–201, 2023.

[57] Jacob T Rapp, Bennett J Bremer, and Philip A Romero. Self-driving laboratories to autonomously navigate the protein fitness landscape. *Nature chemical engineering*, 1(1):97–107, 2024.

[58] Eli N. Weinstein, Mattia G. Gollub, Andrei Slabodkin, Cameron L. Gardner, Kerry Dobbs, Xiao-Bing Cui, Alan N. Amin, George M. Church, and Elizabeth B. Wood. Manufacturing-aware generative model architectures enable biological sequence design and synthesis at petascale. *bioRxiv*, 2024.

[59] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. *arxiv*, 2022.

[60] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z F Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong

Li, J L Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R J Chen, R L Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S S Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W L Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X Q Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y K Li, Y Q Wang, Y X Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y X Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z Z Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu,

Zhongyu Zhang, and Zhen Zhang. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning, September 2025.

[61] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[62] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[63] Anna Radzicka and Richard Wolfenden. A proficient enzyme. *Science*, 267(5194):90–93, 1995.

[64] Ellen M. Sletten and Carolyn R. Bertozzi. Bioorthogonal chemistry: Fishing for selectivity in a sea of functionality. *Angewandte Chemie International Edition*, 48(38):6974–6998, 2009.

[65] Shosuke Yoshida, Kazumi Hiraga, Toshihiko Takehana, Ikuo Taniguchi, Hironao Yamaji, Yasuhito Maeda, Kiyotsuna Toyohara, Kenji Miyamoto, Yoshiharu Kimura, and Kohei Oda. A bacterium that degrades and assimilates poly(ethylene terephthalate). *Science*, 351(6278):1196–1199, 2016.

[66] Elizabeth L Bell, Ross Smithson, Siobhan Kilbride, Jake Foster, Florence J Hardy, Saranarayanan Ramachandran, Aleksander A Tedstone, Sarah J Haigh, Arthur A Garforth, Philip J R Day, Colin Levy, Michael P Shaver, and Anthony P Green. Directed evolution of an efficient and thermostable PET depolymerase. *Nat. Catal.*, 5(8):673–681, August 2022.

[67] Peter I Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

[68] Richard M Felder and Rebecca Brent. Active learning: An introduction. *ASQ higher education brief*, 2(4):1–5, 2009.

[69] Eric Schulz, Maarten Speekenbrink, and Andreas Krause. A tutorial on gaussian process regression: Modelling, exploring, and exploiting functions. *Journal of mathematical psychology*, 85:1–16, 2018.

[70] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[71] Maximilian Balandat, Brian Karrer, Daniel Jiang, Samuel Daulton, Ben Letham, Andrew G Wilson, and Eytan Bakshy. Botorch: A framework for efficient monte-carlo bayesian optimization. *Advances in neural information processing systems*, 33:21524–21538, 2020.

[72] Samuel Daulton, Xingchen Wan, David Eriksson, Maximilian Balandat, Michael A Osborne, and Eytan Bakshy. Bayesian optimization over discrete and mixed spaces via probabilistic reparameterization. *Advances in Neural Information Processing Systems*, 35:12760–12774, 2022.

Figures

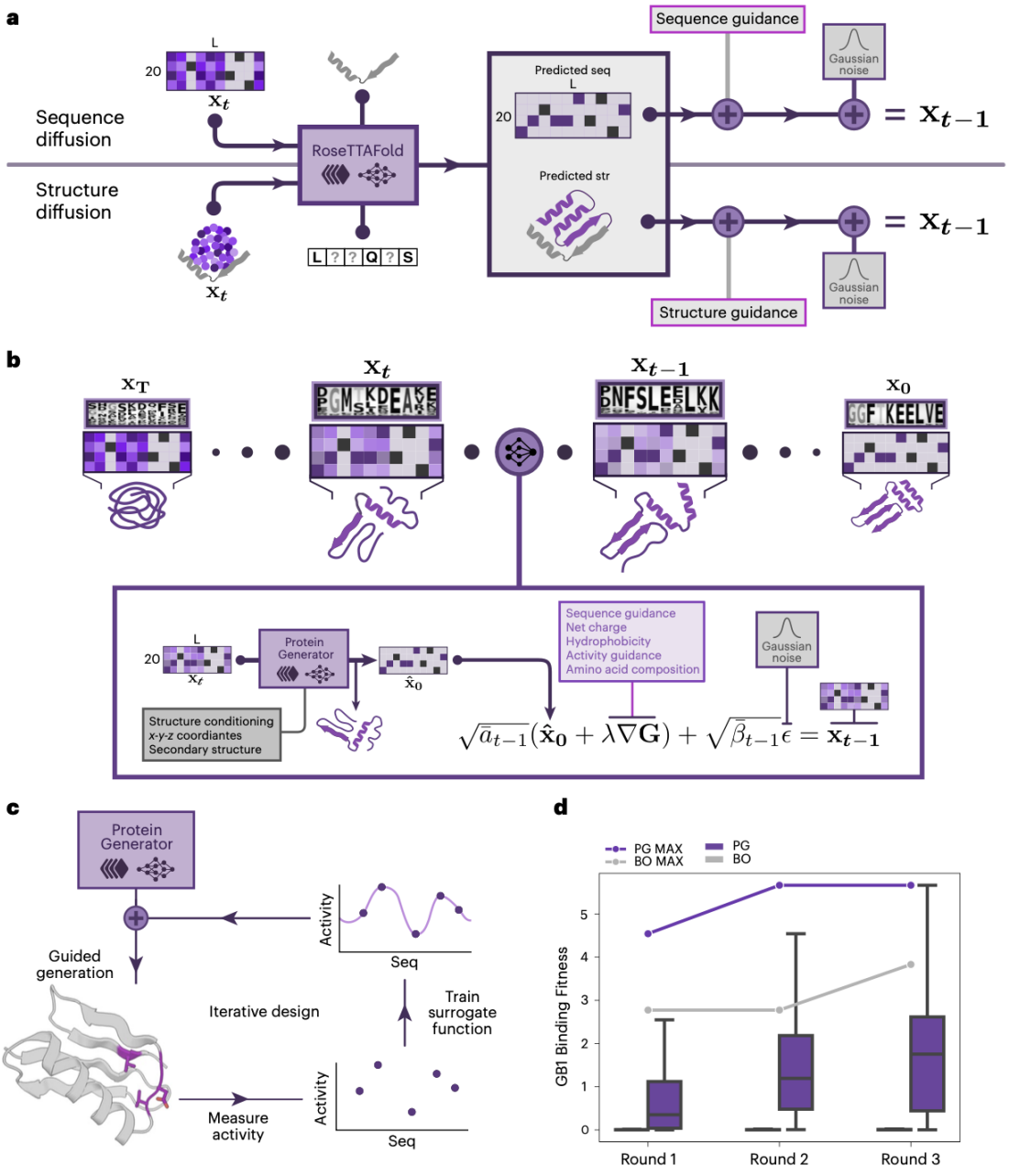

Figure 1

Figure 1: a, Comparison of diffusion in sequence and structure space. PG and RFdiffusion take as input noised sequence (PG) or structure (RFdiffusion) data and problem specific sequence and structure constraints. At each denoising step, the RoseTTAFold architecture generates complete protein sequences and structures, and this is used to generate the next step in the trajectory in sequence (PG) or structure (RFdiffusion) space. Although specific structural or sequence features can be fixed in the input to RoseTTAFold in both approaches, biases toward particular sequence features during the diffusion update at each step are more readily incorporated in PG (as are biases toward structural features, such as symmetry, in RFDiffusion). b, Schematic of PG inference trajectory. At each step in the diffusion process the sequence $\mathbf{x}_0$ is predicted from sequence $\mathbf{x}_t$ by RF conditioned on any desired structural information, combined with any desired sequence bias, and noised to generate the $\mathbf{x}_{t-1}$. This process is repeated for T steps as the sequence–structure pair converges on a high-confidence solution shaped by the structural and sequence guidance information. c, Iterative design schematic demonstrating how PG can be used in an experimental feedback loop. Designs generated by the model are evaluated for activity; a surrogate function approximating sequence to function relationships is fit; and gradients from the surrogate can then be used to guide PG toward active design space. d, In silico demonstration of iterative design using GB1 fitness landscape for binding and comparison with Bayesian optimization (BO). In round 0, not shown in the plot, 96 designs are generated with PG without guidance, and a surrogate function is trained to discriminate high and low activity designs. In rounds 1-3, gradient-based guidance is used to generate 96 designs for each method; a surrogate function is fit; and the process is repeated. Line plots show maximum activity sampled, and box plots show distribution sampled over the batch of 96. Mean activities for each round are statistically significant between the two populations ($p<0.05$, two-sided Mann–Whitney U-test, n=96 designs per round). Box plots boundaries indicate upper and lower quartiles, and whiskers indicate the nearest quartile + 1.5× interquartile range. seq, sequence; str, structure.

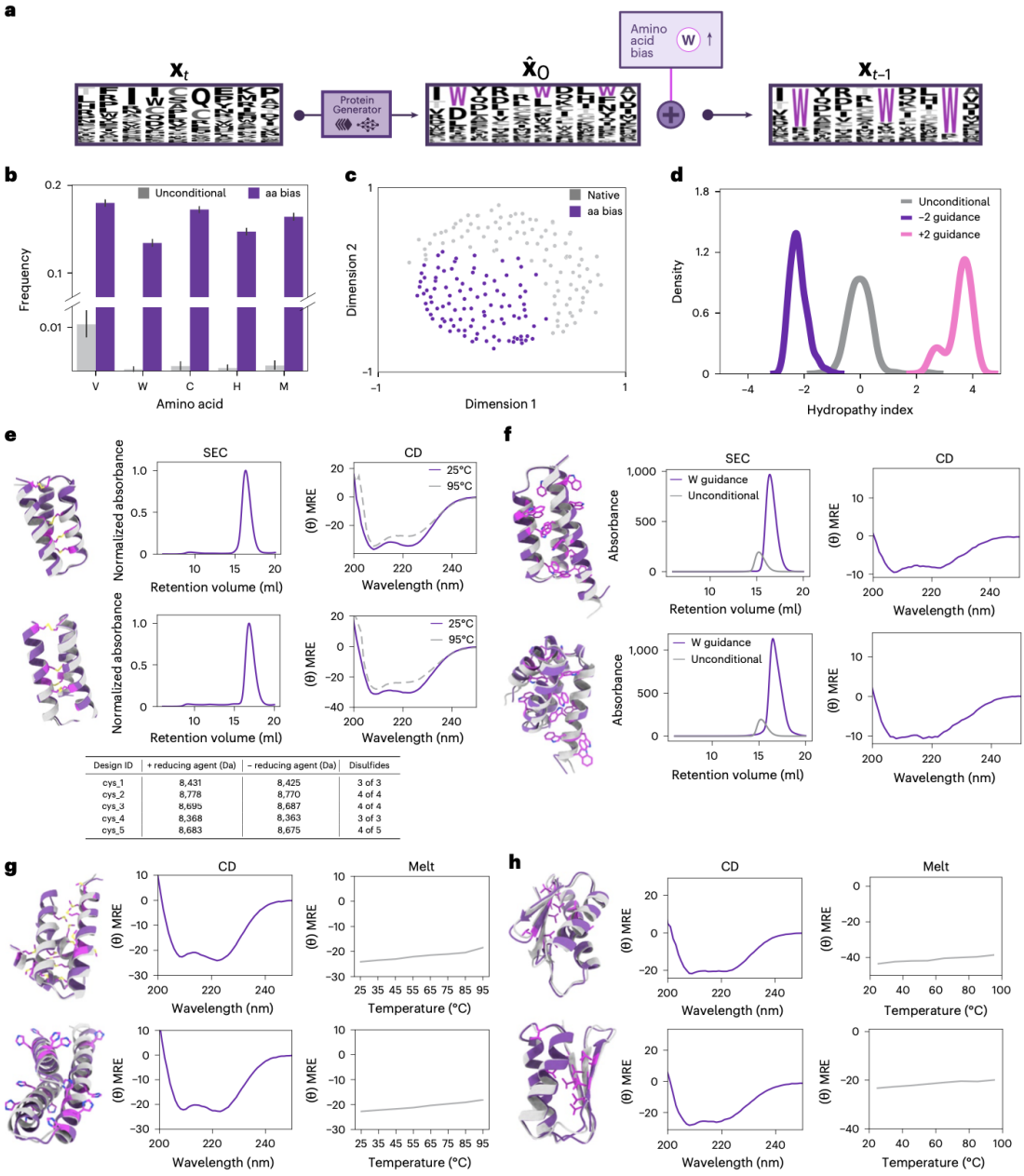
a
x_t
Protein Generator
$\hat{x}_0$
Amino acid bias
W
x_{t-1}
b
Unconditional
aa bias
Frequency
Amino acid
V
W
C
H
M
c
Native
aa bias
Dimension 2
Dimension 1
d
Unconditional
−2 guidance
+2 guidance
Density
Hydropathy index
e
SEC
CD
Normalized absorbance
Retention volume (ml)
(θ) MRE
Wavelength (nm)
25°C
95°C
Design ID | + reducing agent (Da) | − reducing agent (Da) | Disulfides
cys_1 | 8,431 | 8,425 | 3 of 3
cys_2 | 8,778 | 8,770 | 4 of 4
cys_3 | 8,695 | 8,687 | 4 of 4
cys_4 | 8,368 | 8,363 | 3 of 3
cys_5 | 8,683 | 8,675 | 4 of 5
f
SEC
CD
W guidance
Unconditional
Absorbance
Retention volume (ml)
(θ) MRE
Wavelength (nm)
g
CD
Melt
(θ) MRE
Wavelength (nm)
Temperature (°C)
h
CD
Melt
(θ) MRE
Wavelength (nm)
Temperature (°C)

Figure 2

Figure 2: a, Amino acid compositional bias schematic. b, Comparison of amino acid frequency in unconditional (gray) and amino acid biased (purple) generation; separate PG trajectories were carried out for each enriched amino acid. Error bars are standard deviation. Biased distributions are significantly different from unconditional amino acid frequencies (P¡0.05, two-sided Mann–Whitney U-test, n=200 designs per amino acid). Box plot boundaries indicate upper and lower quartiles; whiskers indicate the nearest quartile + 1.5× interquartile range; and the center line is the median. c, Multidimensional scaling of native and amino acid biased sequences shows that they occupy distinct regions of sequence space. d, Hydropathy guidance. Biasing the sequence toward or away from hydrophobic amino acids results in a shifted distribution of hydropathy scores compared to unconditional generation (P¡0.05 two-sided Mann–Whitney U-test, n=122 designs per condition). e, Experimental validation of cysteine biased designs (design in gray, AF2 in purple). Proteins are monomeric by SEC and alpha helical by CD at 25°C and 95°C. Mass spectrometry indicates the presence of the designed number of disulfide bonds. f, Experimental validation of tryptophan biased designs (design in gray, AF2 in purple). Designs are monomeric by SEC, have considerably higher absorbance at 280nm than unconditional designs and are alpha helical by CD. g, Experimental validation of histidine and methionine biased designs (design in gray, AF2 in purple). h, Experimental validation of valine biased designs (design in gray, AF2 in purple). Valines highlighted in pink on the designs are present in the beta-fold secondary structure. CD traces and melt curves at 222nm are to the right of the designs. CD traces and melt curves at 222nm are to the right of the designs. aa, amino acid.

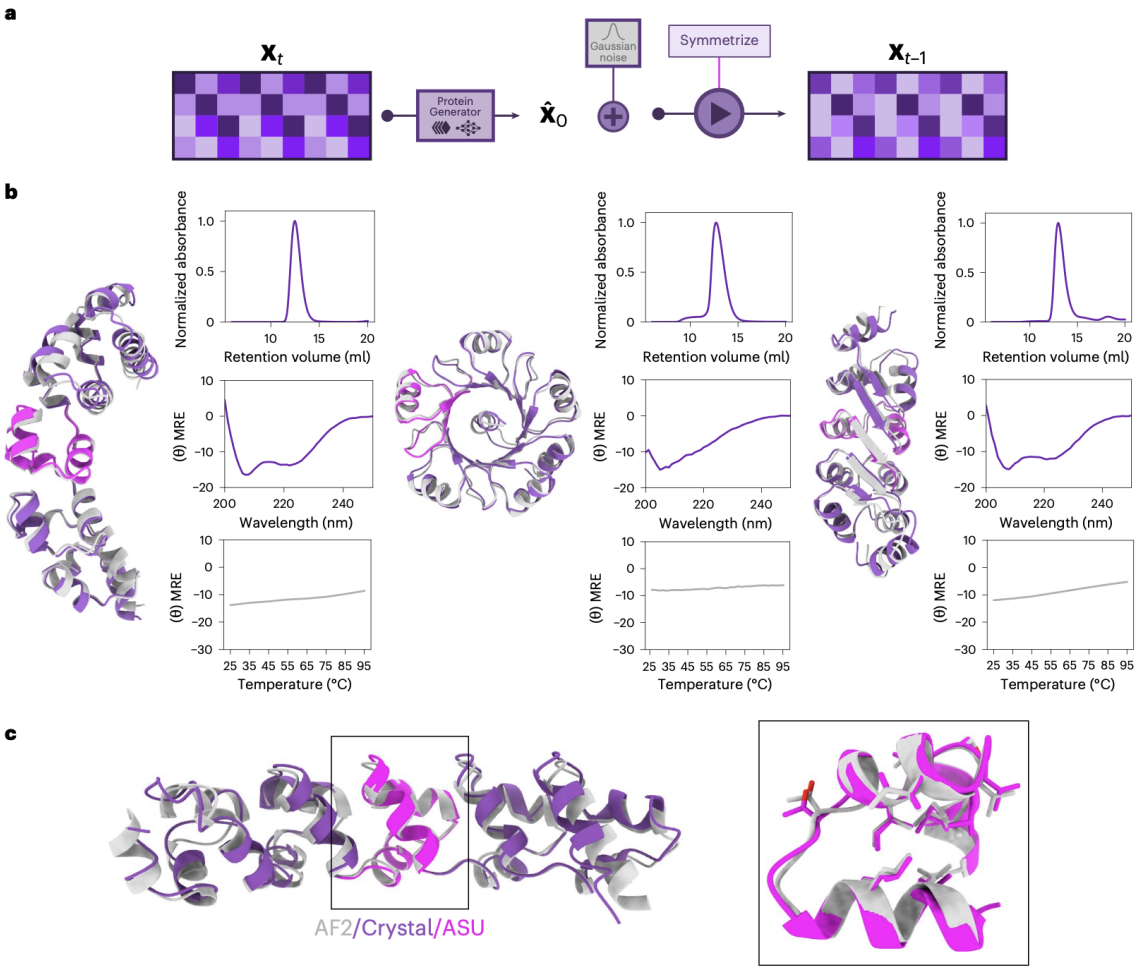

Figure 3: a, Symmetric sequence diffusion to design proteins with sequence symmetry. b, Experimental validation of sequence repeat proteins. Designs in gray are overlaid with AF2 predictions in purple, and asymmetric units are highlighted in pink. SEC and CD traces and melting curves demonstrate stability of these designs. c, 3.70-Å crystal structure of designed repeat protein: AF2 model in gray, crystal structure in purple and asymmetric unit in pink. Box on the right highlights the accuracy of designed side chains in the asymmetric unit.

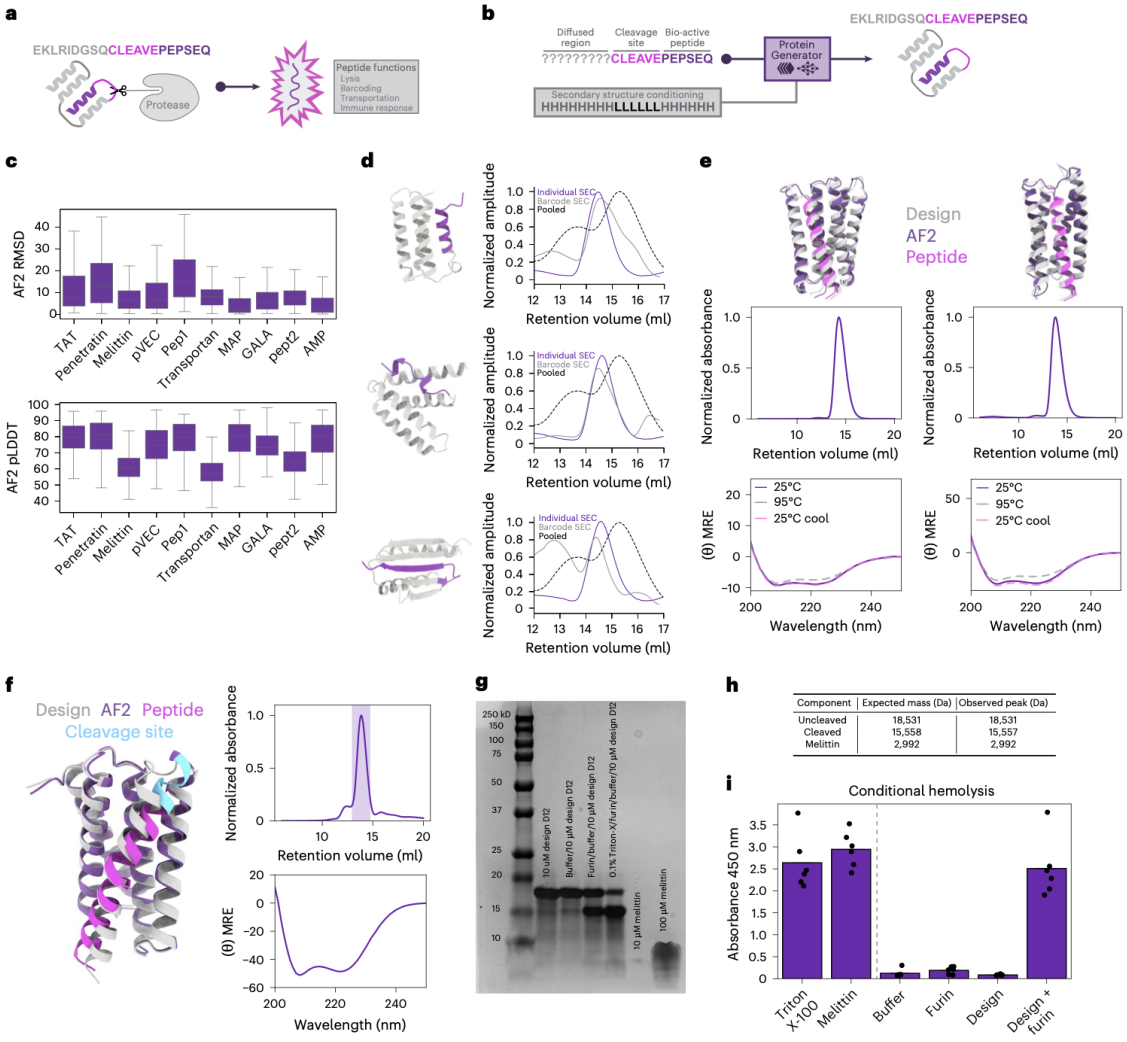
a
EKLRIDGSQCLEAVEPEPSEQ
Protease
Peptide functions
Lysis
Barcoding
Transportation
Immune response
b
Diffused region
Cleavage site
Bio-active peptide
??????????CLEAVEPEPSEQ
Protein Generator
EKLRIDGSQCLEAVEPEPSEQ
Secondary structure conditioning
HHHHHHHHLLLLLLHHHHHH
c
AF2 RMSD
AF2 pLDDT
TAT
Penetratin
Melittin
pVEC
Pep1
Transportan
MAP
GALA
pept2
AMP
d
Normalized amplitude
Individual SEC
Barcode SEC
Pooled
Retention volume (ml)
e
Design
AF2
Peptide
Normalized absorbance
Retention volume (ml)
(θ) MRE
25°C
95°C
25°C cool
Wavelength (nm)
f
Design AF2 Peptide
Cleavage site
Normalized absorbance
Retention volume (ml)
(θ) MRE
Wavelength (nm)
g
250 kD
150
100
75
50
37
25
20
15
10
10 µM design D12
Buffer/10 µM design D12
Furin/buffer/10 µM design D12
0.1% Triton-X/furin/buffer/10 µM design D12
10 µM melittin
100 µM melittin
h
Component | Expected mass (Da) | Observed peak (Da)
Uncleaved | 18,531 | 18,531
Cleaved | 15,558 | 15,557
Melittin | 2,992 | 2,992
i
Conditional hemolysis
Absorbance 450 nm
Triton X-100
Melittin
Buffer
Furin
Design
Design + furin

Figure 4

Figure 4: a, Schematic overview of functional peptide scaffolding for downstream tasks such as protease cleavage for lysis and peptide barcoding. b, Sequence-only motif scaffolding and secondary structure conditioning to generate proteins with embedded functional sequences. Cleavage sites can be specified at the N or C terminus of the peptide to allow for protease cleavage. c, In silico design metrics for sequence-only bioactive peptide scaffolding. RMSD of AF2 predictions to designs on the top and AF2 pLDDT of designs on the bottom. Box plot boundaries indicate upper and lower quartiles; whiskers indicate the nearest quartile + 1.5× interquartile range; and the center line is the median. n=2,000 designs per condition. d, Mass spec peptide barcoding assay. Scaffolding barcodes with PG results in soluble and monomeric designs by SEC. SEC traces for individual designs are in gray. When the same designs are expressed in a pooled library (black), and fractions are digested with trypsin, analytical mass spectroscopy of each fraction is able to recapitulate the SEC trace shown in purple. e, Melittin scaffolded designs with furin cleavage site. Designs are shown in gray, and AF2-predicted structures are shown in purple, with melittin peptide highlighted in pink. Designs are soluble and monomeric by SEC and folded with helical secondary structure by CD. f, Melittin scaffolded design D12. D12 design model is in gray; AF2-predicted structure is overlayed in purple for scaffold; cyan is for the cleavage site; and pink is for melittin. SEC fraction of monomeric D12 used for downstream assays is highlighted with the purple bar. CD trace of D12 is consistent with the designed helical secondary structure. g, Representative SDS-PAGE of uncleaved D12 (18kD), cleaved D12 (15kD) and melittin peptide (3kD) (n=3 biological replicates). h, Mass spec of the cleavage reaction products confirms the presence of uncleaved D12, cleaved D12 and melittin. Melittin mass was calculated with an additional c-terminal 'GS' due to the expression vector used. i, Absorbance at 450nm for six technical replicates of washed RBCs after incubation with design with and without furin protease. Positive controls Triton X-100 and melittin are shown to the left of the vertical bar. Design with furin lyses RBCs significantly more than samples without design (P=0.002, two-sided Mann–Whitney U-test) or furin (P=0.005, two-sided Mann–Whitney U-test) and is on par with positive controls Triton X-100 (P=0.127, two-sided Mann–Whitney U-test) and melittin (P=0.132, two-sided Mann–Whitney U-test).

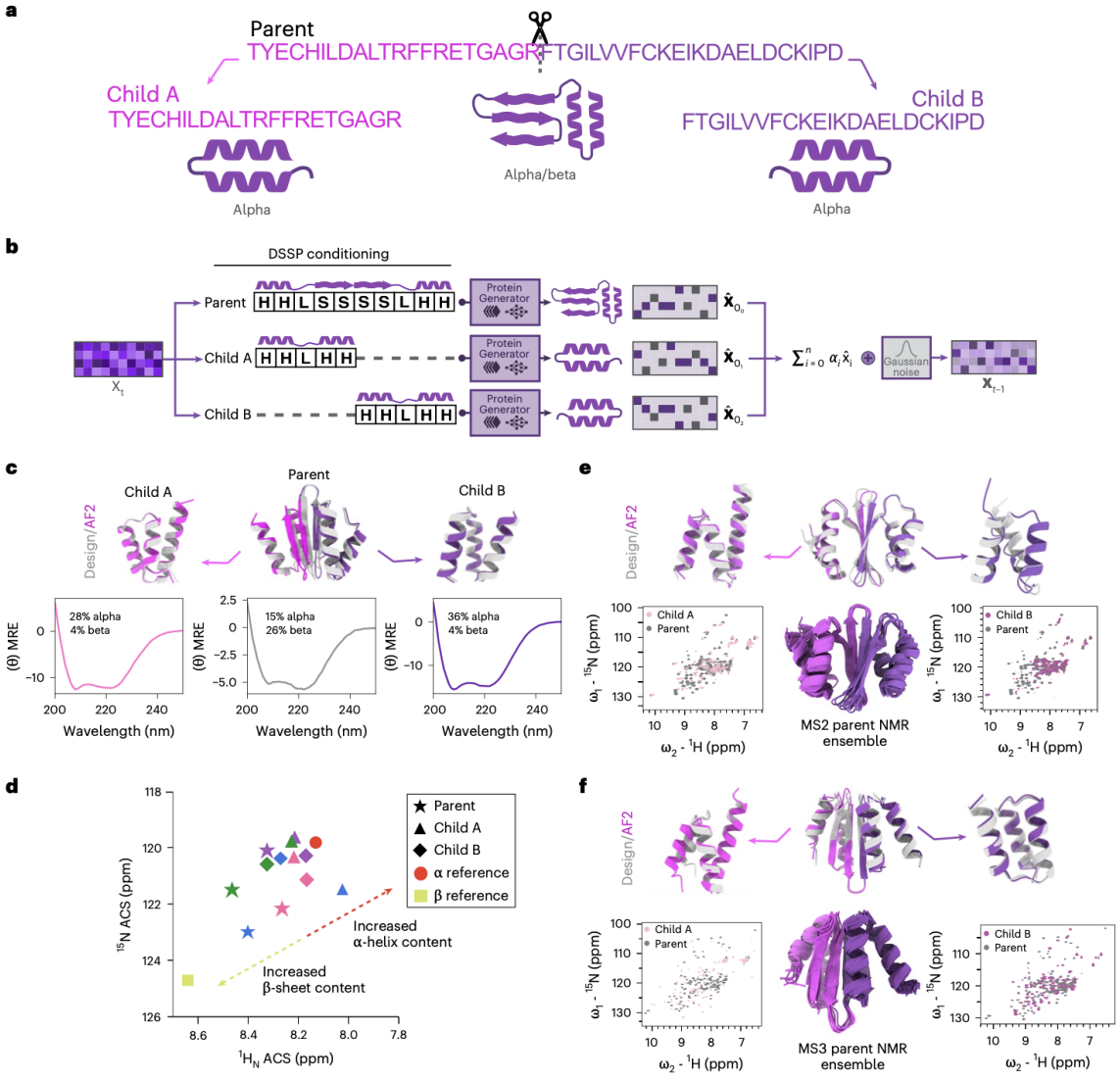
a
Parent
TYECHILDALTRFFRETGAGRFTGILVVFCKEIKDAELDCKIPD
Child A
TYECHILDALTRFFRETGAGR
Child B
FTGILVVFCKEIKDAELDCKIPD
Alpha
Alpha/beta
Alpha
b
DSSP conditioning
Parent
Child A
Child B
Protein Generator
Gaussian noise
c
Child A
Parent
Child B
Design/AF2
28% alpha
4% beta
15% alpha
26% beta
36% alpha
4% beta
(θ) MRE
Wavelength (nm)
d
Parent
Child A
Child B
α reference
β reference
Increased α-helix content
Increased β-sheet content
15N ACS (ppm)
1HN ACS (ppm)
e
Design/AF2
Child A
Parent
Child B
MS2 parent NMR ensemble
ω1 - 15N (ppm)
ω2 - 1H (ppm)
f
Design/AF2
Child A
Parent
Child B
MS3 parent NMR ensemble

Figure 5

Figure 5: a, Multistate DSSP conditioning is used to generate a sequence with an alpha/beta fold in the parent state and all alpha in the child A and child B states. b, Implementation of multistate DSSP sequence conditioning. Different DSSP conditioning strings are applied to a full-length parent sequence and two subsequences (child A and child B). RoseTTAFold predictions and model logits are output for parent, child A and child B. A linear combination of output logits is used as a potential to guide the model toward finding one sequence that satisfies all DSSP conditioning strings for parent, child A and child B. c, MS1 family adopts distinct folds by CD. Top, high pLDDT design and AF2 models of family MS1. Bottom, CD spectra and deconvolution of family MS1 indicating 26% beta content in the parent compared to 4% beta content in child A and child B, respectively. d, ACS of 1HN and 15N chemical shifts values obtained from MS1–MS4 HSQC spectra. Reference average ACS values of primarily α-helical proteins (red circle) and primarily β-sheet proteins (yellow square) are shown calculated from 1HN–15N correlations using chemical shift information obtained from the Biological Magnetic Resonance Bank. ACS values are compared for multistate sequences among parent (α/β mix fold), child A (α-helical fold) and child B (α-helical fold). MS1 in pink, MS2 in purple, MS3 in blue, MS4 in green. MS2 (e) and MS3 (f) families are designed by PG to adopt distinct folds in the parent and child states with high AF2 confidence (top row). HSQC overlays of MS2 and MS3 child A and B compared to parent (bottom row; ω indicates chemical shift). NMR structures of MS2 and MS3 parent fold into the intended secondary structures with atomic-level accuracy (bottom middle).

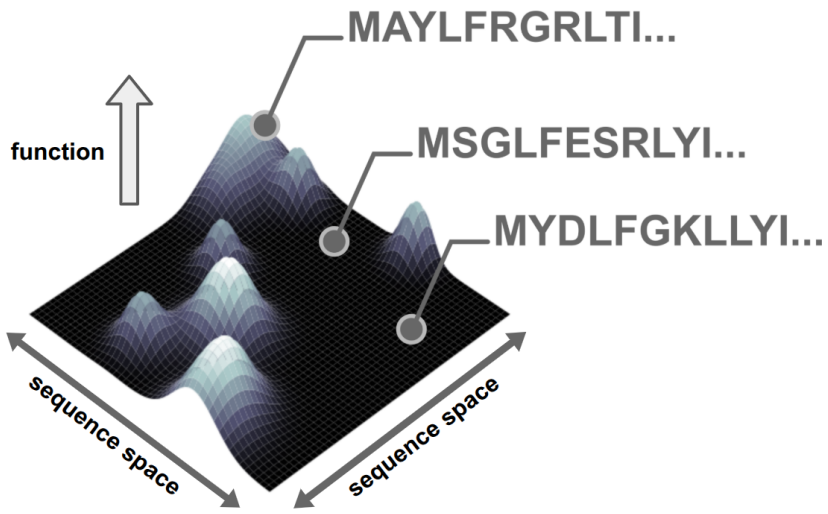

Figure 6: Sequence-function landscape representation.

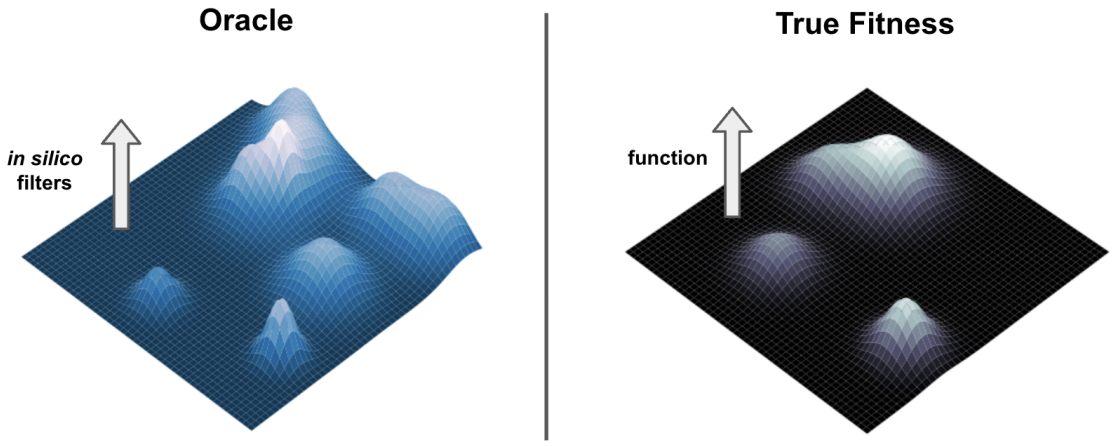

Figure 7: On the left we can see what the oracle's fitness landscape and on the right we can see how that might compare to the underlying functional landscape. We can see that there is good agreement in areas where the oracle does not show any filters passing, but there are many spaces where the oracle is confident that do not actually have any functional activity when experimentally measured.

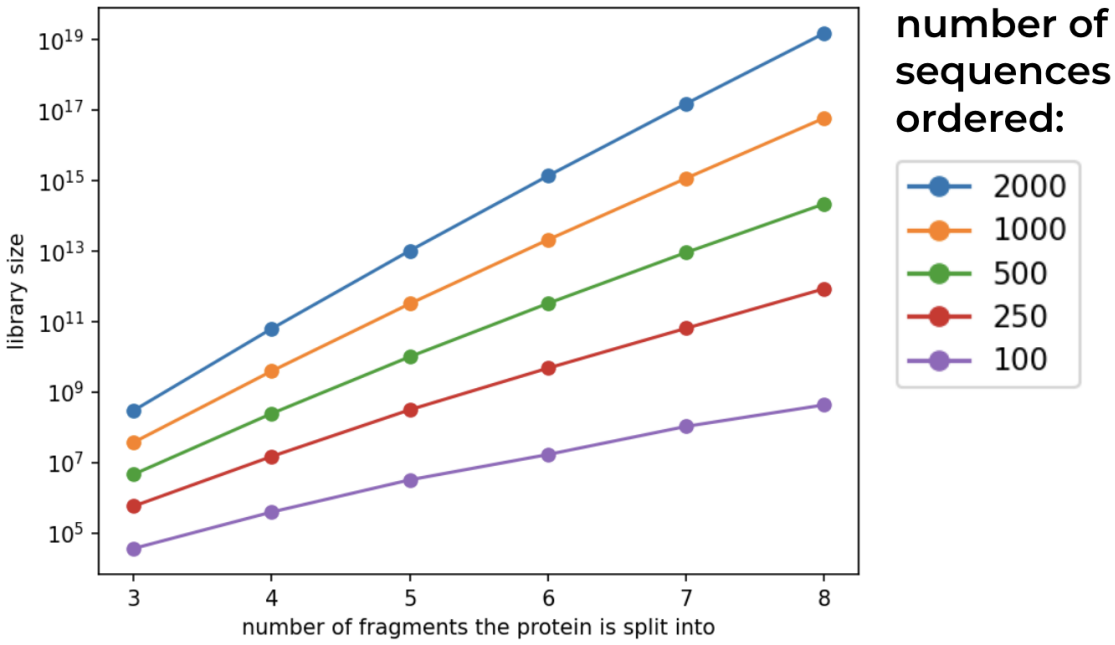

Figure 8: Scaling laws define how many fragments and total sequences needed to achieve the library size of interest.

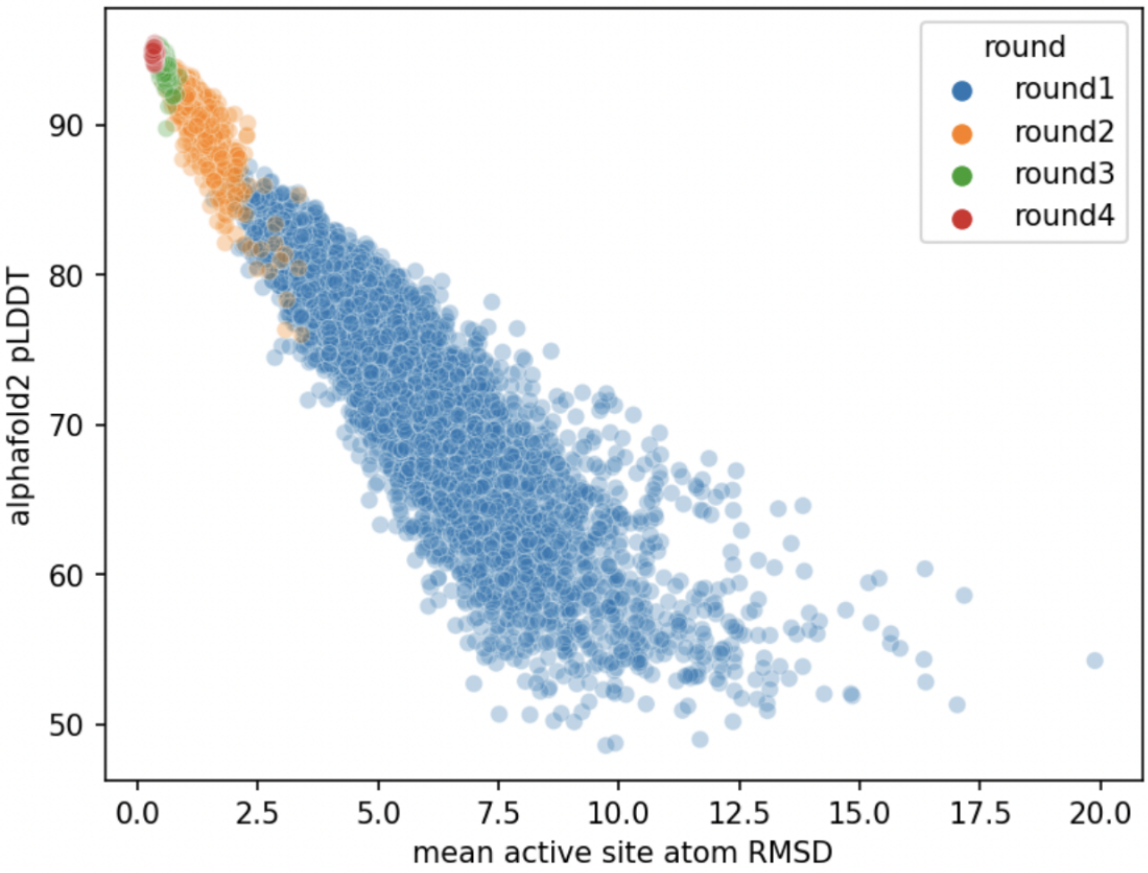

Figure 9: Visualization of *in silico* evolution process to optimize the combinatorial space for high filter pass rate.

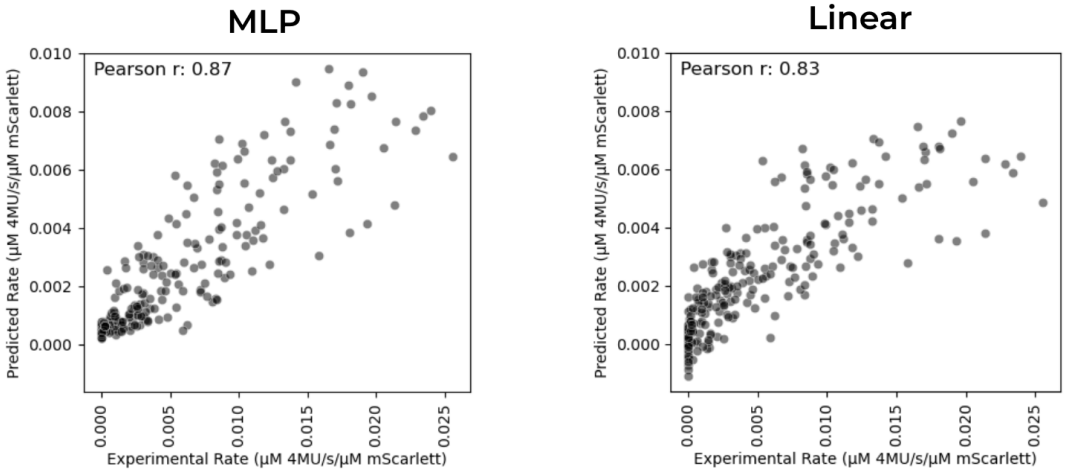

Figure 10: Comparing random sampling, with trained linear and MLP models for selecting high activity variants in the library.

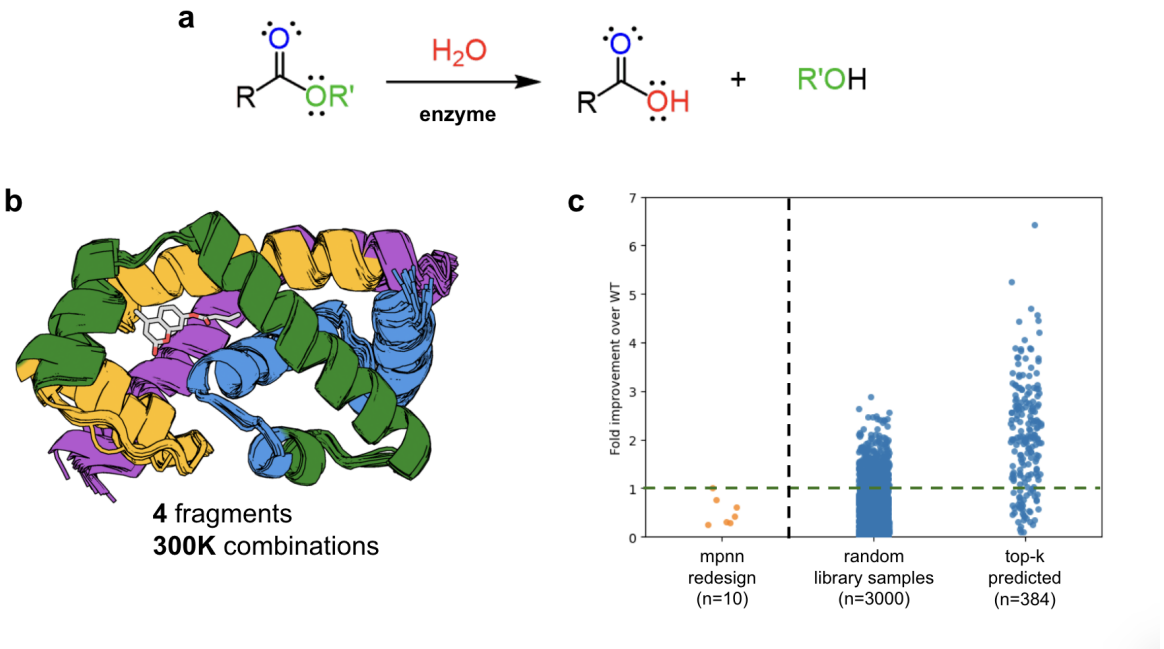

Figure 11: a, Ester hyrdolysis example reaction scheme. b, Overlay of designs in the library colored by where the protein was split into different fragments (4 fragments, 300K library size). c, Experimentally evaluated data. On the left side is a comparison to redesign with proteinMPNN, on the right side of the figure is multiple rounds and strategies used to improve the activity of the enzyme (k_{cat})

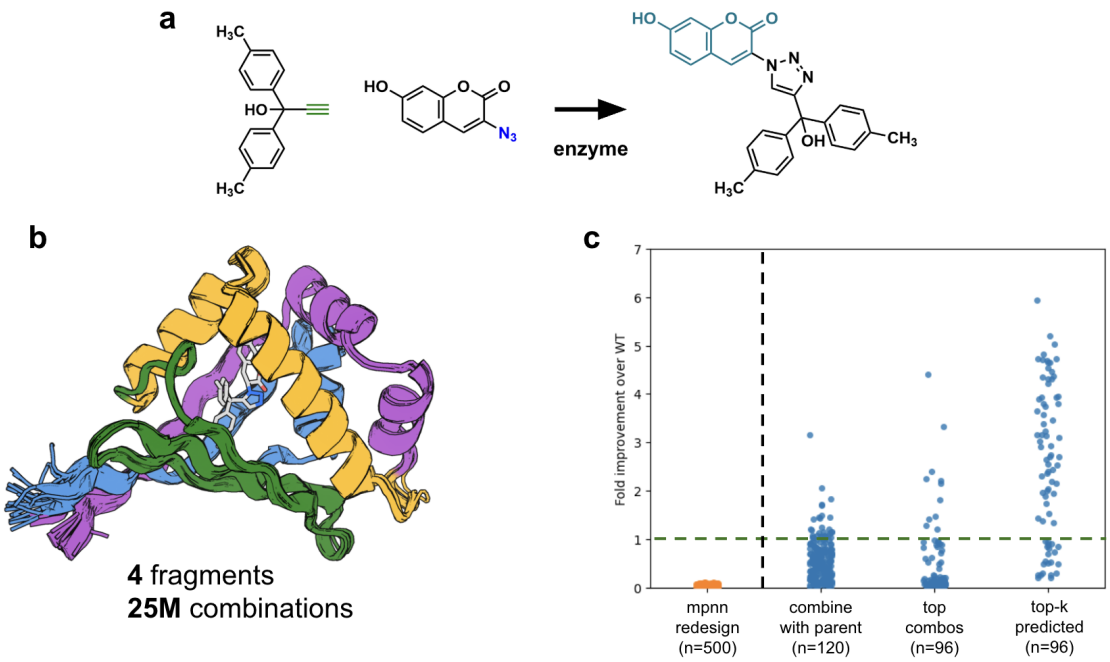

Figure 12: a, Click reaction scheme. b, Overlay of designs in the library colored by where the protein was split into different fragments (4 fragments, 25M library size). c, Experimentally evaluated data. On the left side is a comparison to redesign with proteinMPNN, on the right side of the figure is multiple rounds and strategies used to improve the activity of the enzyme (k_{cat})

Figure 13: Comparison of overlay of AF3 predictions for the original design on the left and the fastest found design from the library on the right.

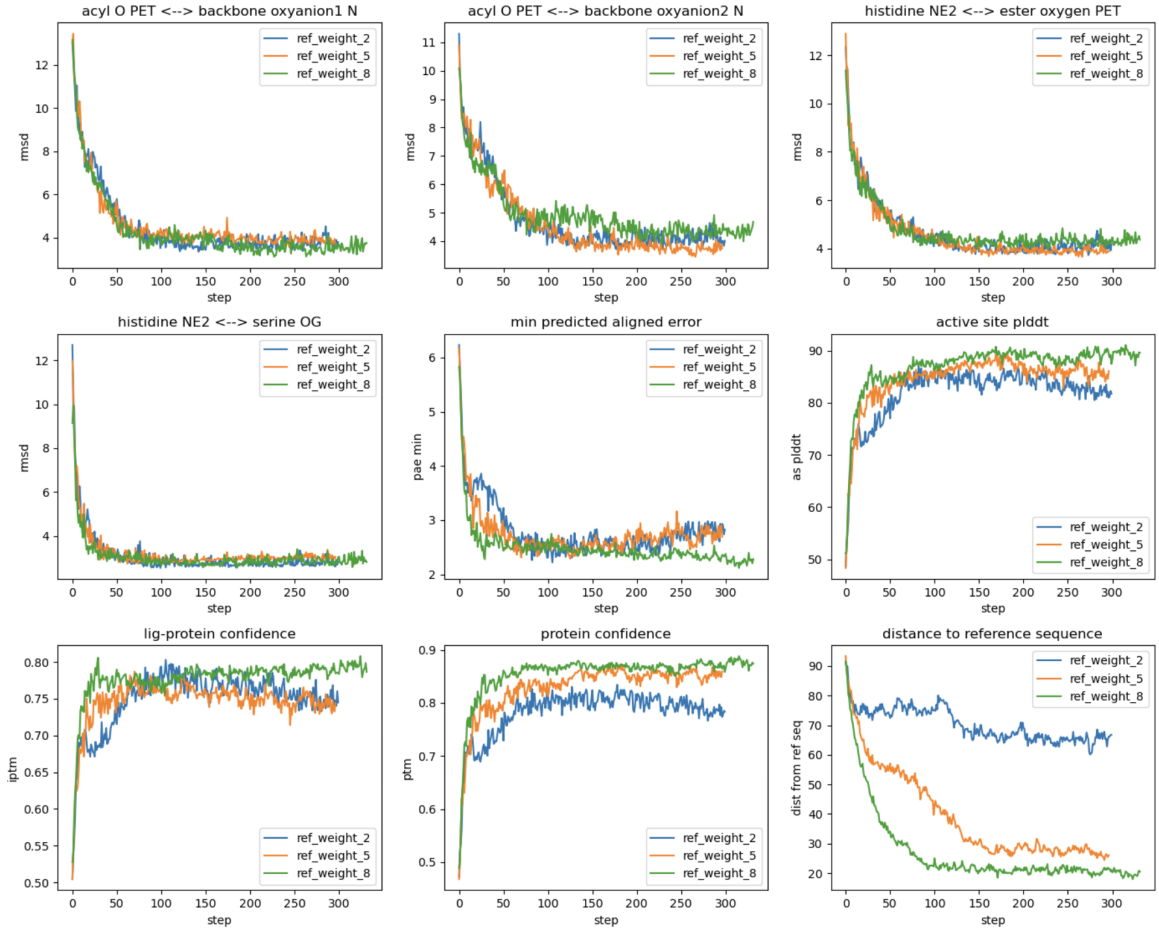

Figure 14: PETase online optimization of multiple objectives. Each trajectory is an independent training run with a different weight on the reference sequence reward (blue, 2; orange, 5; green, 8).

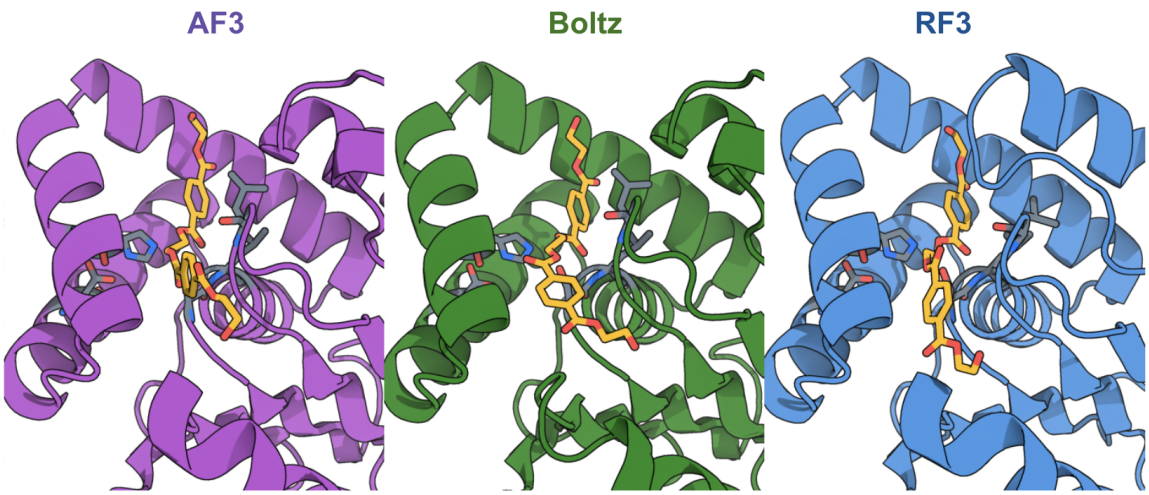

Figure 15: Folding library sequences with multiple oracles.

Algorithms

Variables

1. seq : Discrete protein amino acid sequence.
2. xyz : True 3D Cartesian coordinates of the protein structure.
3. xyz_{pred} : The model's predicted 3D coordinates.
4. $model$: The neural network model used for prediction.
5. $\mathbf{x}_0$: The "clean" continuous representation of the sequence, typically one-hot encoded and scaled to $[-1, 1]$.
6. $\mathbf{x}_t$: The continuous sequence representation noised to diffusion step t.
7. $\mathbf{x}_{0_{\text{pred}}}$: The model's prediction of the clean data $\mathbf{x}_0$ from a noisy input $\mathbf{x}_t$.
8. $\mathbf{x}_{0_{\text{selfcond}}}$: The predicted $\mathbf{x}_0$ from a previous step, used as an additional input for self-conditioning.
9. T : The total number of diffusion steps.
10. t : The current diffusion timestep, $t \in \{1, \ldots, T\}$.
11. β_t : The variance schedule, defining the noise level at step t.
12. α_t : The signal rate, defined as $1 - \beta_t$.
13. $\bar{\alpha}_t$: The cumulative signal rate, $\prod_{s=1}^{t} \alpha_s$.
14. ϵ : Random noise sampled from a standard normal distribution, $\mathcal{N}(0, I)$.

15. c : Conditioning information for a single state (e.g., structure, sequence constraints).

16. C : A set of different conditioning information for multistate design.

17. L : The length of the sequence (number of amino acids).

18. G : A guidance function $G : \mathbb{R}^{L\times 20} \to \mathbb{R}$ that scores a sequence based on desired properties.

19. $\mathbf{g}$: The gradient of the guidance function, $\nabla_{\mathbf{x}} G(\mathbf{x}_0, t)$.

20. λ : The guidance scale, controlling the strength of the guidance gradient $\mathbf{g}$.

21. γ_j : A weighting factor for combining predictions from state j during multistate inference.

22. p : The fraction of the sequence to which amino acid bias is applied.

23. idx_{aa} : The index for a specific amino acid to be biased.

24. idx_{bias} : The indices of residue positions where the bias is applied.

25. π_{ref} : The initial reference policy for sequence generation.

26. π_θ : The optimizable policy, parameterized by θ.

27. b : The batch size for sampling sequences from the policy π_θ.

28. β : A parameter in the GRPO algorithm (distinct from the diffusion β_t).

29. I, M : The number of outer and inner loop iterations for policy optimization.

30. S, F : A batch of sequences (S) and their corresponding fragments (F).

31. S_{sampled} : A new set of sequences sampled from the fragment pool F.

32. A : The advantage estimation for each fragment, guiding the policy update.

Algorithm 1 Training

function TRAINSTEP(seq, xyz, c)
  $\mathbf{x}_0 \leftarrow \text{Onehot}(seq) * 2 - 1$ ▷ convert sequence to [-1,1]
  $t \sim \text{Uniform}(\{1, \ldots, T\})$
  $\epsilon \sim \mathcal{N}(0, I)$
  $\mathbf{x}_t \leftarrow \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon$
  $\mathbf{x}_{t+1} \leftarrow \sqrt{\bar{\alpha}_{t+1}}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_{t+1}}\epsilon$
  $\mathbf{x}_{0_{\text{selfcond}}}, xyz_{pred} \leftarrow \text{stopgrad}(\text{model}(\mathbf{x}_{t+1}, t+1, c, \mathbf{x}_{0_{\text{selfcond}}}))$
  $\mathbf{x}_{0_{\text{pred}}}, xyz_{pred} \leftarrow \text{model}(\mathbf{x}_t, t, c, \mathbf{x}_{0_{\text{selfcond}}})$
  $\text{cce\_loss} \leftarrow \text{CCE\_loss}(\mathbf{x}_0, \mathbf{x}_{0_{\text{pred}}})$
  $\text{kl\_loss} \leftarrow \text{KL\_reconstruction\_loss}(\mathbf{x}_0, \mathbf{x}_{0_{\text{pred}}}, t)$
  $\text{str\_loss} \leftarrow \text{Structure\_loss}(xyz, xyz_{\text{pred}})$
  Take gradient descent step on $\text{cce\_loss} + \text{kl\_loss} + \text{str\_loss}$
end function
function KL_RECONSTRUCTION_LOSS($\mathbf{x}_0, \mathbf{x}_{0_{\text{pred}}}, t, \beta_t, \beta_{t-1}, \alpha_t, \bar{\alpha}_t, \bar{\alpha_{t-1}}$)
  $\mathbf{x}_{t-1_{\text{pred}}} \leftarrow \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_{0_{\text{pred}}} + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t$
  $\mathbf{x}_{t-1} \leftarrow \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t$
  $\sigma_t^2 \leftarrow \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$
  return $\text{KL}(\mathbf{x}_{t-1}, \mathbf{x}_{t-1_{\text{pred}}}, \sigma_t^2)$
end function

Algorithm 2 Inference

function SAMPLE(T, L, λ,c,G)
  $\mathbf{x}_t \sim \mathcal{N}(0, I)_{L,20}$
  $\mathbf{x}_{0_{\text{selfcond}}} \leftarrow$ None
  for i in range(T) **do**
    $t \leftarrow T - i$
    $\mathbf{x}_0, xyz \leftarrow \text{model}(\mathbf{x}_t, t, c, \mathbf{x}_{0_{\text{selfcond}}})$
    $\mathbf{g} \leftarrow \nabla_{\mathbf{x}} G(\mathbf{x}_0, t)$
    $\mathbf{x}_t \leftarrow \sqrt{\bar{\alpha}_{t-1}}(\mathbf{x}_0 + \lambda \mathbf{g}) + \sqrt{1 - \bar{\alpha}_{t-1}}\epsilon$
    $\mathbf{x}_{0_{\text{selfcond}}} \leftarrow \mathbf{x}_0$
  end for
  $seq \leftarrow \text{argmax}(\mathbf{x}_0)$
  return seq, xyz
end function

Algorithm 3 Sequence Bias

function AMINOACIDBIAS($\mathbf{x}_{0_{\text{pred}}}, L, p, idx_{aa}$)
  $\mathbf{g} \leftarrow 0^{L \times 20}$
  $n \leftarrow \text{int}(pL)$
  $idx_{bias} \leftarrow \text{topK}(\mathbf{x}_{0_{\text{pred}}}[:, idx_{aa}], n)$
  $\mathbf{g}[idx_{bias} : idx_{aa}] \leftarrow 1$
  return g
end function

Algorithm 4 Multistate Inference

function SAMPLEMULTISTATE(T, L, λ, C, γ)
  $\mathbf{x}_t \sim \mathcal{N}(0, I)_{L,20}$
  $\mathbf{x}_{0_{\text{selfcond}}} \leftarrow$ None
  for i in range(T) **do**
    $t \leftarrow T - i$
    for j, c in enumerate(C) **do**
      $\mathbf{x}_{0_j}, xyz_j \leftarrow \text{model}(\mathbf{x}_{t_c}, t, c, \mathbf{x}_{0_{\text{selfcond}}})$
    end for
    $\mathbf{x}_0 \leftarrow \sum_{j=0}^{|C|} \gamma_j \mathbf{x}_{0_j}$
    $\mathbf{x}_t \leftarrow \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_{t-1}}\epsilon$
    $\mathbf{x}_{0_{\text{selfcond}}} \leftarrow \mathbf{x}_0$
  end for
  $seq \leftarrow \text{argmax}(\mathbf{x}_0)$
  return seq, xyz
end function

Algorithm 5 Online Combinatorial Library Optimization

function FRAGMENTGRPO($\pi_{\text{ref}}, b, \beta$)
  $\pi_\theta \leftarrow \pi_{\text{ref}}$
  for i in range(I) **do**
    Draw batch of sequences $S \sim \pi_\theta(b)$
    Split sampled sequences S into fragments F
    Sample new set of sequences S_{sampled} from F
    Compute rewards for sequences S_{sampled}
    Aggregate rewards from sequence to fragment level (2)
    Compute advantage A for each fragment (3)
    for j in range(M) **do**
      Sub-sample batch of sequences and advantages $(S_b, A_b) \sim (S_{\text{sampled}}, A)$
      Update policy π_θ using S_b, A_b to maximize the GRPO objective (4)
    end for
  end for
  return π_θ
end function