Progress in Computing Applications

Print ISSN: 2278 – 6465
Online ISSN: 2278 – 6473

DLINE JOURNALS

PCA 2025: 14 (1)
https://doi.org/10.6025/pca/2025/14/1/38-55

Integrating a Fold-specific Regularization Component in Protein Engineering

Trevor S. Frisby
Computational Biology Department
School of Computer Science
Carnegie Mellon University, Pittsburgh, PA, USA
tfrisby@andrew.cmu.edu

Christopher J. Langmead[1]
Computational Biology Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, USA
cjl@cs.cmu.edu

ABSTRACT

Directed Evolution (DE) is a method used in protein engineering that entails multiple cycles of mutagenesis and screening to identify sequences that enhance a specific characteristic (e.g., binding strength to a designated target). However, the fundamental optimization challenge is not fully determined, meaning that alterations to boost the chosen property may negatively impact unmeasured yet significant attributes (e.g., subcellular localization). We aim to tackle this challenge by integrating a fold-specific regularization component into optimisation. This regularization component steers the search towards designs similar to sequences within the fold family of the protein. We implemented our approach on an extensive collection of protein GB1 mutants, measuring their binding affinities to IgG-Fc. Our findings reveal that the regularized optimization process yields more native-like GB1 sequences while only slightly compromising binding affinity. Specifically, the log-odds of our designs, assessed under a generative model of the GB1 fold family, are approximately 41-45% greater than those achieved without regularization, with merely a 7% reduction in binding affinity. Therefore, our technique successfully balances competing characteristics. Additionally, we show that our active-learning-based method lowers the experimental workload needed to pinpoint optimal GB1 designs by 67%, compared to recent findings from the Arnold lab using the same dataset.

Keywords: Regularization Components, Protein Engineering, Directed Evolution, Protein Sequences

Received: 15 September 2024, Revised 9 December 2024, Accepted 18 December 2024

1. Introduction

The field of protein engineering seeks to design molecules with novel or improved properties [15]. The primary techniques used in protein engineering fall into two categories: *rational design* [22] and *directed evolution* (DE) [1]. Rational design uses model-driven *in silico* combinatorial searches to identify promising candidate designs, which are then synthesized and tested experimentally. Directed evolution, in contrast, involves iterative rounds of saturation mutagenesis at select residue positions, followed by *in vitro* or *in vivo* screening for desirable traits. The most promising sequences are then isolated and used to seed the next round of mutagenesis. Traditionally, directed evolution is a model-free approach. That is, computational models are not used to guide or simulate mutagenesis.

Recently, a technique for incorporating Machine Learning (ML) into the DE workflow was introduced [32]. Briefly, this ML-assisted form of DE uses the screening data from each round to update a model that predicts the effects of mutations on property being optimized, $f(s_k) \rightarrow y$. Here, y is the measured trait, s_k is the choice of residues at $k \ll n$ positions, and n is the length of the protein's primary sequence. The mutagenesis step in the current round of DE is then biased towards generating sequences with high predicted fitness under the model, as opposed to generating a uniformly random sample. ML-assisted DE has been shown to reduce the number of rounds needed to find optimal sequences, relative to traditional (i.e., model-free) DE [32].

Significantly, the models learned in ML-assisted DE are *myopic* in the sense that they only consider the relationship between s_k and the screened trait, y (ex binding affinity). Therefore, the underlying optimization problem is under-determined, and so the technique may improve the measured trait at the expense of those that are unmeasured, but nevertheless important (ex. thermostability, solubility, subcellular localization, etc). The primary goal of this paper is to introduce an enhanced version of ML-assisted DE that is biased towards native-like designs, while optimizing the desired trait. By "native" we mean that the optimized design still has high probability under a generative model of the fold family to which the protein belongs. The intuition behind this approach is that any high-probability sequence is likely to respect factors that are not directly accounted for by the fitness model, f, such as epistatic interactions between the mutated residues and the rest of the protein [27], among others.

Our method performs Bayesian optimization [16] and incorporates a regularization factor derived from a generative model of the fold family. Here, we evaluate two choices of generative models: Markov Random Fields (MRF) generated by the gremlin algorithm [2], and profile HMMs [11]. We demonstrate our method by re-designing the B1 domain of streptococcal protein G (GB1) to maximize binding affinity to the IgG Fc receptor. Our results demonstrate that the regularization term leads to more native-like GB1 sequences, with minimal impact on binding affinity. Like previous studies, our results also show that ML-assisted DE outperforms the traditional, model-free approach to DE. Additionally, we demonstrate that our approach reduces the wet-lab burden to identify optimal GB1 designs by 67%, relative to recent results from previous results on the same data [32].

2. Background

The method introduced in this paper combines several technologies: directed protein evolution, *Bayesian* optimization, and generative modeling of protein fold families. The following subsections provide brief summaries of these techniques.

2.1 Directed Protein Evolution

Directed evolution (DE) is an iterative technique for designing molecules. It has been used to create proteins with increased stability (ex. [6]), improved binding affinity (ex. [9]), to design new protein folds (ex. [7]), to change an enzyme's substrate specificity (ex. [26]) or ability to selectively synthesize enantiomeric products (ex. [32]), and to study fitness landscapes ([24]), among others. Given an initial sequence, the primary steps in directed evolution are: (i) *mutagenesis*, to create a library of variants; (ii) *screening*, to identify variants with the desired traits; and (iii) *amplification* of the best variants, to seed the next round. Each step can be performed in a variety of ways, giving rise to multiple options for performing DE. For example, the mutagenesis step can be performed one residue at a time, called a single mutation walk (Fig. 1-top), or simultaneously at multiple positions, followed by genetic recombination (Fig. 1-bottom).

Rational design and directed evolution have complementary strengths and weaknesses, and in practice it is not unusual for protein engineers to use both. The computational models used in rational design are typically physics- or knowledge based [4], and will therefore filter designs that are predicted to be energetically or statistically unfavorable. This filtering, in turn, may reduce the total number of designs that need to be synthesized/cloned and tested in the lab. Directed evolution, in contrast, performs what is in effect a parallel in vitro or in vivo search over designs. Most of the designs will lack the desired trait, which is inefficient in the sense that consumable resources are wasted. On the other hand, DE's approach results in the screening of a larger number of designs than rational design, because it generates large libraries of variants, as opposed to individual designs. That is, DE may ultimately be the faster route to finding optimal designs due to its parallel and high throughput nature, which may also increase the odds of serendipitous discoveries. Moreover, the traditional approach to DE is model-free, and therefore not subject to the limitations of computational models which are, at best, only approximations to the underlying physics and are not intended to reflect phenomena at higher scales (i.e., chemistry and biology).

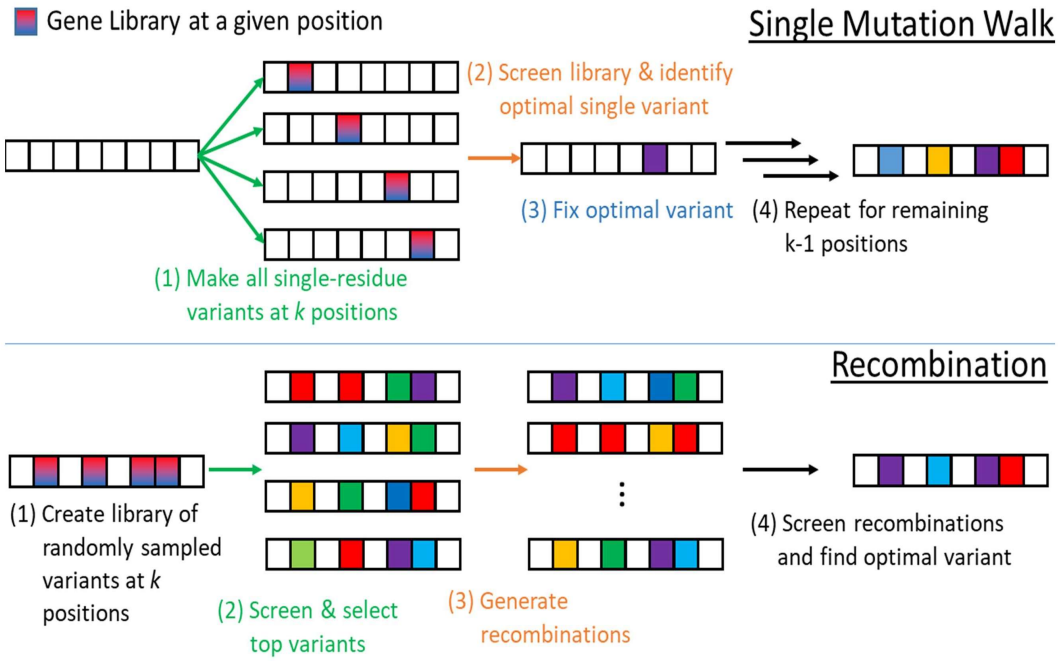

Figure 1. Traditional, model-free approaches to directed evolution

Machine Learning-assisted directed protein evolution

While effective, the mutagenesis, screening, and amplification steps in DE are expensive and time-consuming, relative to rational design's in silico screens. In an effort to reduce these experimental demands, a Machine Learning-assisted approach to DE was introduced recently [32]. This ML-assisted form of DE is summarized in Fig. 2. The key difference between traditional and ML-Assisted DE is that the data generated during screening, $S = \{s_k, y\}_{i=1:n}$ are used to train a model, $f(s_k) \rightarrow y$, that maps the set of mutations to the trait. The model, f, can be a classifier or regression model, and is used to perform an *in silico* screen over designs. Promising designs are then synthesized/cloned and screened in the lab. The key assumption made by ML-assisted DE is that the cost of performing an *in silico* screen is much lower than running wet-lab experiments.

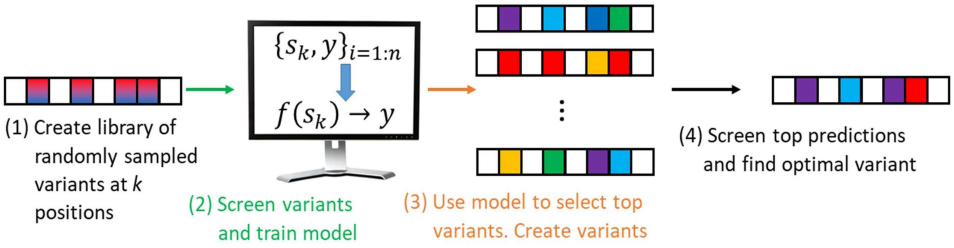

Figure 2. Machine Learning-assisted directed evolution

Like rational design, ML-assisted DE uses computational models, but the nature of those models is rather different. For one, the models used in ML-assisted DE make predictions corresponding to the quantity measured in the screening step, whereas the models used in rational design tend to be based on physical or statistical energy functions, and are therefore making predictions about the energetic favorability of the design. Second, the models used in ML-assisted DE are updated after each DE round to incorporate the new screening data, and thus adapt to protein-specific experimental observations. The models used in rational design, in contrast, are typically fixed. Finally, the models used in ML-assisted DE are myopic, in the sense that they only consider the relationship between a small subset of sequence positions and the measured quantity. The models used in rational design, in contrast, generally consider the entire sequence, and are thus better suited to filtering energetically unfavorable designs. The technique introduced in this paper seeks to combine the strengths of both methods; our method uses a fitness model that adapts to the experimental data, but also considers the favorability of the mutations across the entire sequence.

2.2 Bayesian Optimization

Bayesian optimization [16] is a technique for optimizing black-box objective functions, f. It has been used in a variety of contexts, including robotics (ex. [14]), particle physics (ex. [10]), and hyper-parameter optimization in deep learning (ex [3]). The first published form of ML-assisted DE [32] did not employ Bayesian optimization, but the same lab subsequently introduced a version that does [33].

In the Bayesian optimization framework, our goal is to find x^*, the point that maximizes (or minimizes) $f(x)$. This is challenging because f is both unknown and expensive to evaluate. Therefore, we want to minimize the number of times $f(x)$ is evaluated. Because it is unknown, f is treated as a random *function* with a suitably defined prior. Given experimental observations, a posterior distribution over f is computed, which becomes the prior for the next round.Typical choices for priors/posteriors include Gaussian Processes [21] and Tree-

structured Parzen estimators [3]. In the context of this paper, *f* is the function that maps designs to fitness values. The evaluation of *f* is expensive, because it requires the previously described DE mutagenesis and screening steps.

The posterior over *f* becomes the input to an acquisition function, which is used to select the next point(s) to evaluate [30]. A variety of acquisition functions have been proposed, including: expected improvement (EI), upper (or lower) confidence bounds, Thompson sampling [28], and probability of improvement. In general, an acquisition function defines some trade-off between exploring the design space, and simply selecting the point that has the best expected value under the posterior (aka exploitation). In this paper, we seek to maximize f, and use expected improvement criterion as the acquisition function.

2.3 Generative Modeling of Protein Fold Families

The proposed approach uses a modified acquisition function that considers not only the expected improvement in fitness for a given set of mutations, *sk*, but also whether the overall design, *sn*, resembles proteins within the same fold family. The latter criterion is implemented using a regularization term, as described in Sec. 3. Our assumption is that the statistical properties of the proteins within a given fold family have been optimized through natural evolution to ensure that they have a full range of physical, chemical, and biological properties to function properly in a complex cellular environment. Therefore, during the process of protein engineering, we should seek designs that are as native-like as possible [8, 12, 17].

To accomplish this task, we propose to use fold family-specific generative models of sequences. We evaluate two options for such models – profile HMMs and Markov Random Fields (MRF), as generated by the gremlin algorithm [2]. HMM and MRF models can be learned from known sequences from a given fold family. The primary difference between these models is that whereas HMMs only encode dependencies between adjacent residues, the gremlin algorithm can detect and encode both sequential and long-range dependencies. Either way, the models encode a joint distribution over residue types at each position in the primary sequence, $P(S_1, ..., S_n)$, which can be used to compute the probability (or related quantities, like log-odds) of given design. We assume that any design with a high probability or log-odds under the generative model is native-like.

3. Methods

3.1 Generative Models of Protein G IgG Fc binding domain (GB1)

Our method incorporates generative models of protein sequences for the fold family to which the target protein belongs. We evaluated two options for such models, (i) a Markov Random Field (MRF) model learned using the gremlin algorithm [2], and (ii) a profile Hidden Markov Model (HMM). We downloaded the profile HMM [11] for the fold family to which GB1 belongs (Pfam id: PF01378) from the Pfam database [5]. We also downloaded the multiple sequence alignment that was used to train the HMM from Pfam, and then used the alignment to train the gremlin model. Thus, the gremlin and HMM models were trained from the same sequence data. We used these models to compute the log-odds of each design. These log-odds are used as a regularization factor in the Bayesian optimization (see Sec 3.2). The two models make different assumptions about the conditional independencies among the residues in the distribution over GB1 sequences, and thus will output different log-odds scores for the same design, in general.

3.2 Fold family-regularized Bayesian Optimization for Directed Protein Evolution

The Bayesian optimization is performed using Gaussian Process (GP) regression as the prior over the unknown fitness function, *f*. A GP requires a kernel, *K*, which describes the covariance between sequences s_i and s_j. In our models, we use the squared exponential kernel given by:

$$K_{s_i,s_j} = \exp\left(-\frac{d(s_i,s_j)^2}{2\ell^2}\right)$$

where $d(\cdot,\cdot)$ is the Euclidean distance and ℓ the length scale. A one-hot encoding of the variants at the selected positions was used to compute distances. Hyperparameter θ is optimized while fitting the GP to data by maximizing the log marginal likelihood:

$$\log p(y|\theta) = -\frac{N}{2}\log 2\pi - \frac{1}{2}\log\det|K+\sigma^2 I| - \frac{1}{2}y^T(K+\sigma^2 I)^{-1}y$$

The term y is a vector of the given property (e.g. fitness) of N sequences, σ^2 the variance of observations, and I is the $N \times N$ identity matrix. Once fitted, the GP encodes a distribution, P, which is used to obtain a posterior mean function $\mu_{\mathcal{P}}(s_k)$ and variance over the unknown fitness function f:

$$\mu_{\mathcal{P}}(s_k) = \mathbb{E}[f(s_k)] = K_{s_k,s}(K+\sigma^2 I)^{-1}y$$
$$\mathrm{Var}[f(s_k)] = K_{s_k,s_k} - K_{s_k,s}(K+\sigma^2 I)^{-1}K_{s,s_k}$$

$K_{sk,s}$ refers to the row vector of kernel function values between sequence s_k and all other sequences, denoted by subscript s. Additionally, $K_{s,sk} = K^T_{sk,s}$.

The GP becomes the argument to an acquisition function, which is used to select sequences for wet-lab screening. The data produced via the screening step are used to update the GP for the next round. In our experiments, we used the expected improvement (EI) criterion as our acquisition function. Two versions of EI were considered: (i) the standard version, which is often used in Bayesian optimization, and (ii) a regularized form of EI. The standard form of EI is given by:

$$\mathrm{EI}(s_k;\mathcal{P}) = \mathbb{E}_{\mathcal{P}}\left[\max(0, f(s_k) - \mu_{\mathcal{P}}(x^+)\right] \quad (1)$$

where x^+ is the location of the (estimated) optimal posterior mean.

Regularized Expected Improvement

We also evaluated a *regularized* form of EI by scaling the standard EI by a design-specific scaling factor, $F(s_k; P)$. In our experiments, F refers to the log-odds score obtained by either an MRF or profile HMM, as described in Section 3.1. Our regularized *EI* is defined as:

$$\mathrm{EI}_{\mathcal{F}}(s_k;\mathcal{P}) = \mathrm{EI}(s_k;\mathcal{P})\mathcal{F}(s_k) \quad (2)$$

We will demonstrate in Section 4 that this small modification to the acquisition function results in a substantial shift in the designs discovered via ML-assisted DE. In particular, our method finds designs that are substantially more native-like, with only a small decrease in expected fitness.

3.3 Directed Evolution with Machine Learning and *in Silico* Traditional Approaches

Our experiments contrast the performance of "standard" ML-assisted DE (i.e., non-regularized) to the regularized version. We also compare the results to simulated forms of "traditional" DE (i.e., without ML), as was also done in [32]. Specifically, we simulated both the single mutation walk and recombination versions of DE (see Fig. 1). We note that the single mutation walk approach is deterministic, given the starting sequence. With the single step, we start each trial with a randomly chosen sequence from the GB1 variant library. At each of positions 39, 40, 41, and 54, we observe the experimentally determined fitness values for all possible single-residue mutations. Having observed these mutations, we then fix in place the single-residue mutant which has the highest fitness. With this residue fixed, we then repeat this procedure for the remaining unfixed residue positions. Continuing in this manner, the trial ends when all residues have been fixed. All observed fitness values within a trial thus represent a DE determined fitness function approximation.

For the recombination method, we mimic saturation mutatgenesis experiments by starting with n randomly chosen sequences from the GB1 variant library. From these, we identify the top three sequences that have highest fitness (as was done in [32]), and use these sequences to perform recombination. A recombinant library is simulated *in silico* by computing the Cartesian product $S_{39} \times S_{40} \times S_{41} \times S_{54}$, where the set S_m refers to the variant residues found at position m among the three highest fitness sequences in the initial random library. The resulting list of 4-tuples defines the recombinant library. Here, the DE fitness function approximation is given by observing fitness values for the n starting sequences as well as the recombined sequences.

4. Results

In this section, we report the results of five approaches to performing DE: (i) single mutation walk (see Fig. 1-top); (ii) recombination (see Fig. 1-bottom); (iii) Bayesian optimization using standard expected improvement (EI), denoted by "GP+EI"; (iv) Bayesian optimization using regularized EI with MRF-derived log-odds, denoted by "GP+EI+gremlin"; and (v) Bayesian optimization using regularized EI with HMM-derived log-odds, denoted by "GP+EI+HMM". Gaussian Process regression models are used for (iii)-(v). The standard form of expected improvement (Eq. 1) is used for (iii), and the regularized version of EI (Eq. 2) is used for (iv) and (v).

Each method was allowed to screen (i.e., obtain fitness values for) a total of 191 variants. This number was chosen to be similar to the number of sequences screened by the deterministic single mutant walk so that each method had similar experimental burden. Each model was initially trained on 20 randomly selected sequences. The small number of initial sequences simulates the scenario where the available fitness data is limited, prior to DE. The Bayesian optimization methods selected the top 19 sequences during each acquisition round. Each model is then updated with the experimentally measured fitness values for the chosen batch of 19 sequences, and this process is repeated for 9 batches (ie. 20 initial sequences plus 9 batches of 19 designs per batch, giving $20 + 19 \times 9 = 191$ variants selected). We refer to a complete set of variant selection batches as a trial. We performed 100 total trials with each selection strategy with different random initial starting sequences. 20% of the data were held out for testing purposes.

4.1 Data

Protein G is an antibody-binding protein expressed in Streptococcus. The B1 domain of protein G (GB1) interacts with the Fc domain of immunoglobulins. We performed our experiments on an existing dataset generated by

Metric	Mean	Median	Variance	(1)	(2)	(3)
(1)Fitness	0.08	0.003	0.16	1		
(2)gremlin	0.54	0	1.06	0.17	1	
(3)HMM	2.71	2.56	1.12	0.1	0.67	1

Table 1. Descriptive statistics for fitness, gremlin log odds score, and HMM log odds score. The final three columns show correlations between each respective score

Wu et al. [31], who performed saturation mutagenesis at four carefully chosen sites in GB1 in order to investigate the protein's evolutionary landscape. The four chosen residues (V39, D40, G41, and V54) are collectively present in 12 of the protein's top 20 pairwise epistatic interactions, meaning these sites are expected to contain evolutionarily favorable variants [20].

The fitness criterion for their study was binding affinity to IgG-Fc. Experimental measurements were obtained for 149,361 out of 160,000 (i.e. 20^4) possible variants at these four loci using mRNA display [23], followed by high-throughput Illumina sequencing. Briefly, this approach to measuring binding affinity works by first creating an input mRNA-protein fusion library from GB1 variants. This input library is then exposed to the GB1 binding target IgG-Fc. Any variant that binds to the target is subsequently sequenced for identification. By measuring the counts of each variant contained in the input library, c^{in}, and output "selected" library, c^{out}, the relative fitness w of the ith variant is calculated as follows:

$$w_i = \gamma \frac{c_i^{\text{in}}}{c_i^{\text{out}}} \qquad (3)$$

Here, γ is a normalizing factor that ensures the wildtype sequence has fitness 1, and sequences with improved fitness are greater than 1. The range of fitness scores is from 0 to 8.76, with mean 0.08. Only 3,643 sequences in the dataset ($\approx 2.4\%$) have fitness greater than 1.

The MRF and HMM models described in Sec. 3 were used to compute the log-odds of each of the 149,361 variants in the GB1 library. The log-odds were scaled to match the range of the fitness scores (0 - 8.76). As described in Sec. 3.2, the (scaled) log-odds were used as regularization terms. Table 1 displays the descriptive statistics of the fitness and the scaled log-odds values used in our experiment. While most fitness and gremlin scores lie close to 0, the HMM scores have mean and median values of 2.71 and 2.56, respectively. The difference between the gremlin and HMM log-odds is not unexpected, because the HMM makes strong assumptions about the conditional independencies between residues, and therefore does not penalize as many interaction pairs. We note that because gremlin and HMM scores describe statistics related to position specific and pairwise patterns, wildtype sequence {V39, D40, G41, V54} takes the maximum value under these metrics. Further, gremlin and HMM scores have weak positive Pearson's correlation with fitness (0.17 and 0.12, respectively). Conversely, gremlin and HMM have relatively high correlation (0.67) to each other.

4.2 Protein fold family-regularization biases variant selection

Traditionally, DE techniques aim to identify sequences that score highly in one property. In Figure 3 we demonstrate that ML-assisted DE outperform simulated traditional approaches (i.e., single-mutant walk and

recombination). Over each cumulative fractions of trials, each of the three ML-assisted methods identify higher fitness variants than simulated traditional approaches. Of the two traditional approaches, the single mutant walk method fares better than recombination, though both consistently identify variants that have higher fitness than wildtype. On average, the single mutant walk procedure finds a variant with maximum fitness 5.13, whereas recombination yields a variant with maximum fitness 4.85.

GP+EI on average identifies variants with 7.28 fitness, followed by GP+EI+gremlin and GP+EI+HMM at 6.76 and 6.74, respectively. This suggests that protein fold familyregularization via gremlin or HMM induces a small reduction ($\approx 7\%$) in overall fitness. Since GP+EI+gremlin and GP+EI+HMM are regularized with the intent to select variants with higher gremlin or HMM scores, we expect that this fitness cost is actually a trade off with these metrics. Figure 4 supports this notion. In Figure 4 top left, GP+EI achieves highest per-batch average maximum fitness, and is able to do so in fewer batches. This is consistent with the result in Figure 3. However, GP+EI+gremlin and GP+EI+HMM are able to find variants of equal fitness levels to GP+EI if given more batches. We note that 20 initial sequences plus 9 batches of 19 corresponds to 191 mutagensis and screening experiments, a relatively modest burden.

The per-batch average gremlin and HMM scores of selected variants reveal a different pattern. In the top right and bottom of Figure 4, it is clear that variants selected by GP+EI+GREM and GP+EI+HMM have highest per-batch average maximum log-odds, and this holds true for every batch. Thus, the 7% decrease in binding affinity is more than compensated for by a 41 to 45% increase in the the log-odds of our designs under a generative model of the GB1 fold family.

That GP+EI+gremlin and GP+EI+HMM have high overlap is not unexpected, given the high correlation between these two metrics. GP+EI at no point selects variants with log-odds scores on par with GP+EI+gremlin and GP+EI+HMM. Since GP+EI gives no consideration to these scores when making variant selection decisions, it should not be expected that this performance difference will ameliorate itself given more batches. Thus, protein fold family-regularization can bias variant selections towards those with favorable fitness and fold family conservation statistics.

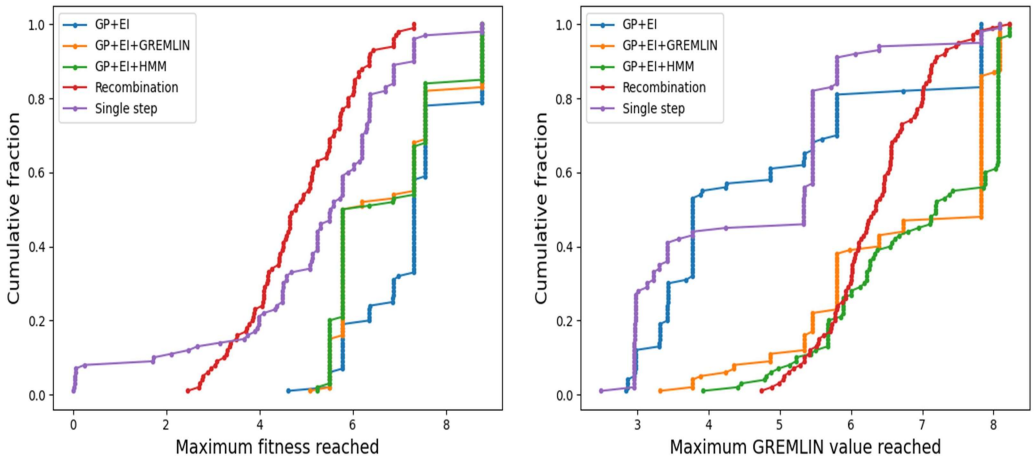

Figure 3. ML-assisted Directed Evolution techniques identify high fitness variants in fewer experiments

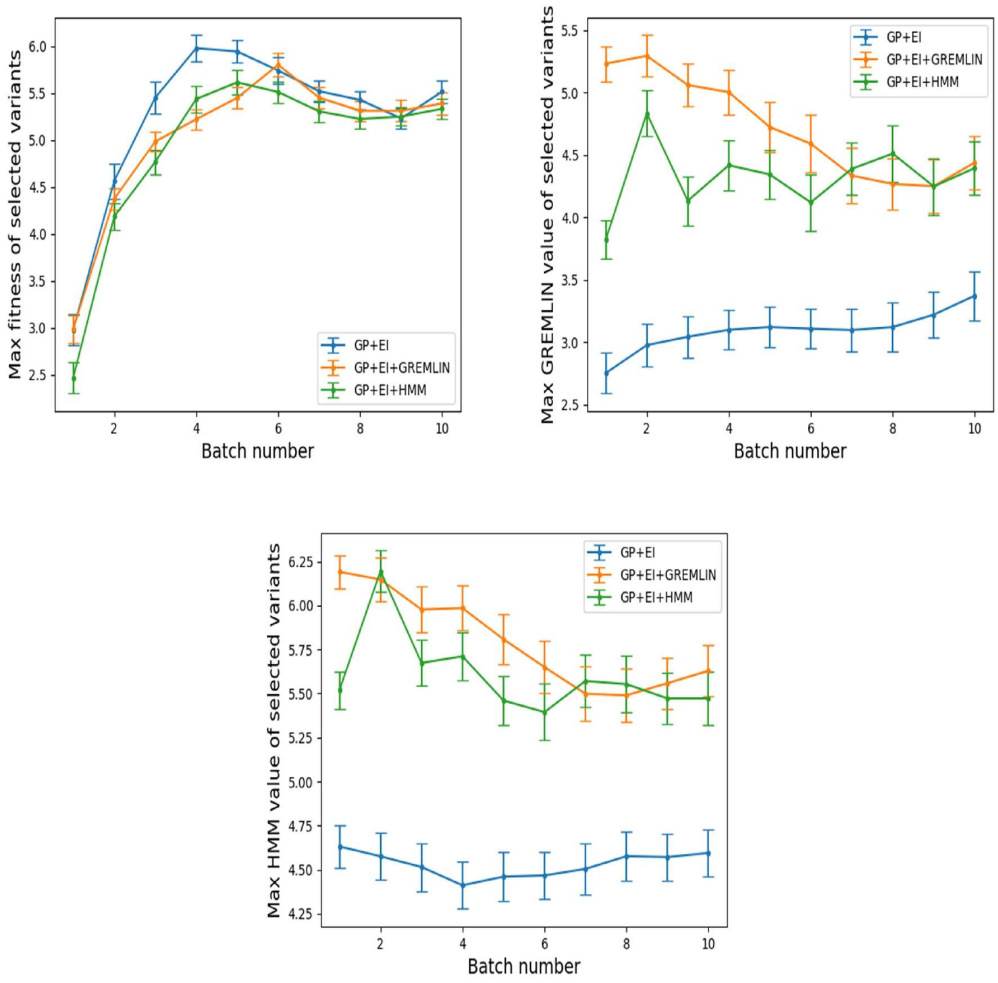

Figure 4. Protein fold family-regularization biases variant selections towards those with native-like position-specific and pairwise statistics with little cost to fitness

To further demonstrate the utility of protein fold family-regularization, Figure 5 compares the maximum fitness and maximum gremlin scores of variants selected in each of 100 trials. For brevity, we omit similar results with HMM scores from this discussion. To show how this process evolves as selections are made, the left figure shows selections from the first batches of each trial, and the right figure shows the final selections made in each trial. First batch selections use models that have only observed 20 random sequences, while the last batches have also observed all choices made in previous batches. There is clear separation between max gremlin scores of GP+EI and GP+EI+gremlin in the first batch. GP+EI+gremlin identifies variants with maximum gremlin score of at least 3 in most trials, whereas most GP+EI selected variants have gremlin scores of 4 or less. On the other hand, the max fitness of selected variants have very similar means (GP+EI = 2.98, GP+EI+gremlin =2.99) and variance (GP+EI = 2.57, GP+EI+gremlin = 2.21). Even as the models are able to select new variants and update themselves, we observe that this general trend persists (Figure 5, right hand side). Thus, selections made by GP+EI+gremlin are most strongly biased toward variants with high gremlin scores, while still being able to identify variants of fitness on par with GP+EI.

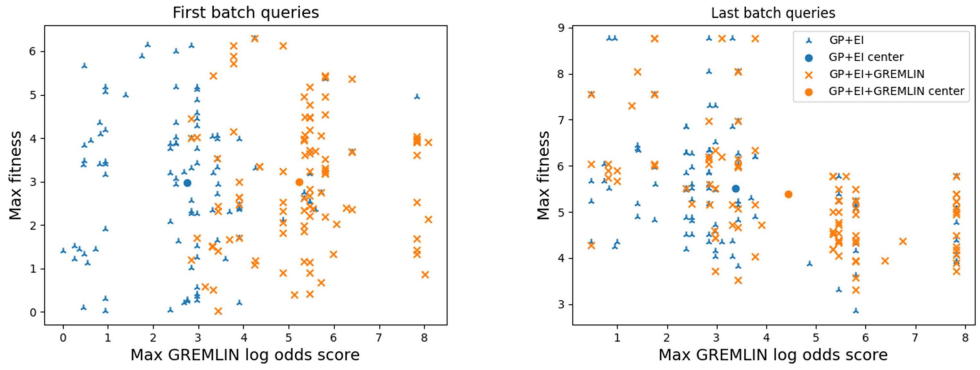

Figure 5. Biased variant selection persists over many batches when using protein fold family-regularization

4.3 Sequence characteristics of selected variants

Thus far, we have characterized each ML-assisted DE technique in terms of the average maximum fitness and the log-odds of selected variants. We now consider the residue-specific behavior of choices made under each model. Figure 6 shows residue-specific entropy of variant selections. A high entropy grid (dark blue) indicates that the model selects many different residue types at that position within a given batch. Low entropy (light blue) indicates that the model selects only few residue types. The relative entropy of selections thus provides a sense of the confidence the model has that a particular variant residue is informative. Each model has relatively low entropy at residues 41 and 54, where GP+EI has lowest entropy at residue 54. The models attain low entropy statuses at these residues during the early batches, and maintain low entropy for the duration of the trial. Conversely, residues 39 and 40 have high entropy throughout, even increasing in later batches. This suggests that all the models attain certainty at residues 41 and 54, but are uncertain at residue positions 39 and 40.

In Figure 7, we show sequence logos obtained from each model after the final batch selections. Again, positions 41 and 54 have highest information content among all positions. Interestingly, all models select similar residue types at each position. In positions 40, 41, and 54, the top two selections are the same residues, just in different orders. Most notably, the top choice at each position yields a variant sequence that is most optimal in terms of the applied regularization (or lack thereof). GP+EI has consensus sequence, {W39,W40,C41,A54}which has a very high fitness (7.28), but comparably low gremlin (0.47) and HMM (1.71) scores. Meanwhile, GP+EI+gremlin with consensus sequence {I39,W40,G41,A54} (fitness=2.75,GREMLIN=3.77,HMM=4.34) and GP+EI+HMM with consensus sequence {I39,W40,G41,A54} (fitness=2.34,GREMLIN=3.42,HMM=4.67) strike a balance between finding variants with increased fitness yet high gremlin and HMM score.

4.4 Predictions on unseen data

In the previous sections, we characterized the batched variant sequence selections made by ML-assisted DE techniques with and without protein fold family-regularization. These selections allowed us to iteratively update models for variant fitness using sequences that each model expected to be informative. Since each model selected different sequences according to their own unique selection objective, each model should also have unique predictive capabilities on unseen sequences. To demonstrate this, we used models obtained from each trial to predict the fitness of a held out test set of approximately 30,000 variants (Figure 8). We emphasize that these sequences were never seen by the models during the iterative variant selection stage of each trial, and that they

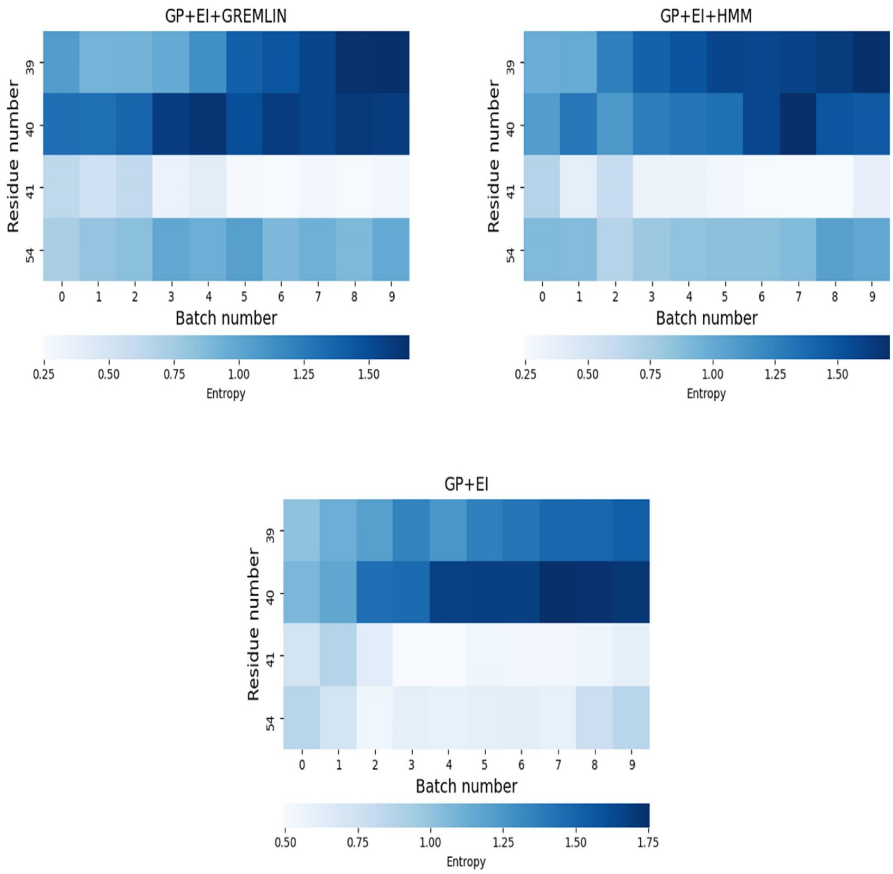

Figure 6. Bayesian selection techniques quickly identify informative sequence patterns

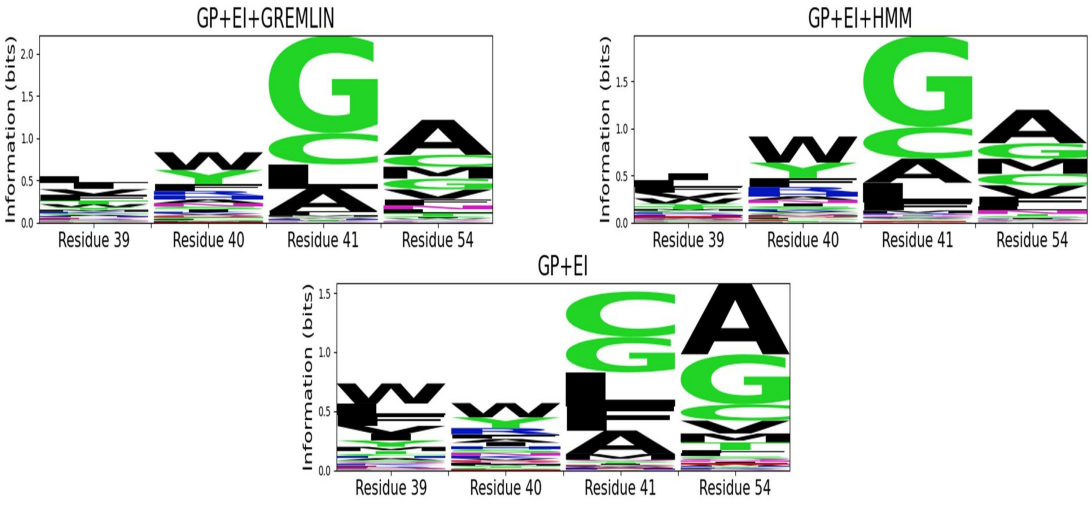

Figure 7. Protein fold family-regularization biases sequence logos towards sequences with desired properties

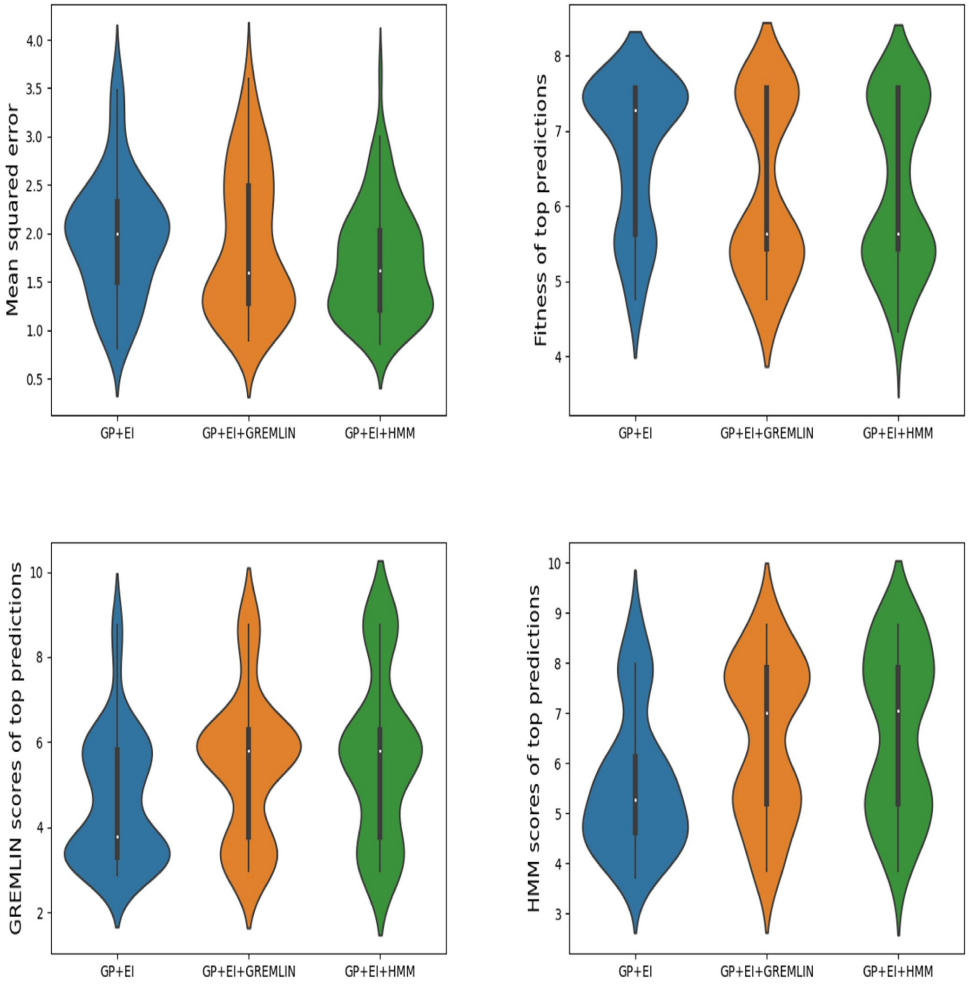

Figure 8. The sequential active learning strategies of ML-assisted DE approaches yields low-accuracy models, but are able to maintain any biases induced by protein fold family-regularization

constitute a random subsample of the GB1 data set. While the models generally yield low-accuracy in terms of predicting actual fitness values, we find that each are still able to discern between high fitness and low fitness variants, and that regularization introduced during selection and training influences model predictions.

Each plot in Figure 8 shows distributions over 100 trials of each model type. The thick black bar shows the interquartile range, and the white dot the median value. The width of each plot corresponds to the mass of the distribution at a given value. The top left figure shows the mean squared error of predictions. All models tend to have error greater than 1, meaning each model has relatively low accuracy in terms of predicting fitness of unseen sequences. The remaining plots show the true fitness, GREMLIN, and HMM scores of the top 100 sequences ranked by predicted fitness for each model. With respect to fitness (top right Figure 8), top predictions made by GP+EI have higher mass than those of GP+EI+gremlin or GP+EI+HMM. Conversely, with respect to gremlin and HMM scores (bottom Figure 8), the opposite trend exists, where the mass of the distributions are higher in GP+EI+gremlin and GP+EI+HMM than in GP+EI.

5. Discussion

Our work extends recent successes of ML-assisted DE. In particular, we show that GP regression techniques with and without protein fold family-regularization are able to identify high fitness variants with greater efficiency than traditional laboratory-based methods. These results echo those found by [32], though our ML-based strategies differ from theirs in key ways. First, they used an ensemble of different regressor types followed by extensive model selection, whereas we use a single Bayesian framework. Additionally, while they use a more traditional supervised learning setup, we adopt an active learning strategy where models iteratively select variants and are able to update themselves based on these selections. This more naturally mimics the workflow of real experimental labs. The batch sizes of an active learning setup can be easily tuned to match the throughput of a given lab, which is useful if incorporating ML-assisted DE into a laboratory-based experimental design. Finally, our models are able to identify high fitness variants while observing true fitness values of fewer sequences compared to the approach in [32]. Specifically, our sequential method only required 191 fitness value acquisitions (20 initial observations plus 9 batches of 19) to identify designs with high fitness values. In contrast, the experiment in [32], which used the same GB1 data, required 470 initial observations plus a single batch of 100 to find similarly fit designs. This is a 67% reduction in the number of sequences tested, which demonstrates the merits of active learning and Bayesian optimization, as the models are able to correct for mistakes made early in the learning process.

A common weakness with traditional DE approaches is that they only seek to optimize sequences with respect to a single property, such as fitness. However, there are any number of other characteristics that describe a protein, such as solubility, thermostability, or subcellular location, and in general, the relationships between these variables may be nonlinear. It is often favorable to identify sequences that score favorably according to multiple such metrics. In principle, one might consider using a set of regression models, each specialized to make predictions about a particular trait. Unfortunately, obtaining sufficient data to train such models is challenging, and we may not know all the relevant properties to consider. More importantly, such models would likely be trained on data from a range of fold families, and may therefore not include factors that are unique to a specific fold family. This motivated our use of fold family-regularization. Our results demonstrate that such regularization produces more native-like designs, with only a slight reduction in fitness.

We note that there are other ways that one might introduce fold family-regularization into this problem. A separate approach that we tried includes creating a new label for each sequence that is the product of fold family score F and variant fitness. Since the highest products should correspond to jointly high fitness and log odds scores, we expected that identifying variants that score highly according to this product should identify mutually favorable variants with respect to each individual score. However, we found that this approach does not obtain a desirable balanced tradeoff between fitness and log odds scores (data not shown, average maximum fitness of selected variants, gremlin×fitness = 4.62, HMM×fitness = 4.52, average maximum log odds score: gremlin×fitness = 7.37, HMM×fitness = 7.04). One possible explanation is that the generative fold family model is conditioned on the entire protein sequence, whereas our prediction task was constrained over only a few specific residues. Another is that the small number of training samples (between 20 and 190, in our experiments) may not be sufficient to learn such a complex function.

Finally, we noted a trend where selections by each model had low entropy at two residue positions. This coincided with each model being able to identify high fitness variants early within each active learning trial. This may suggest that the models quickly identified

informative sequence patterns. When we looked at the types of residues selected by each model, we saw that each model had similar but different selection patterns. With respect to residue positions 41 and 54, the top two residues identified by each model were 41G vs 41C and 54A vs 54G. Interestingly, these selections correspond to some known biology. Olson *et al.* [20] used this same dataset to study epistasis in GB1. They observed that the mutation V54A is by itself neutral, but often beneficial (ie. increase fitness) in the presence of other mutants, and that this effect was most pronounced at residue 41. While 41G represents the wildtype sequence, each model seems to also favor the mutation G41C. Olson *et al.* find that, with a V54A background, the G41C mutation provides the highest fitness. Even at residue positions where the models had high entropy, it seems they identify informative sequence features. Each model consistently chooses variant D40W, and both in the background of mutant V54A and by itself as single mutant, this mutation increases fitness. Thus, our Bayesian ML-assisted DE methods are capable of identifying informative sequence patterns.

6. Conclusions and Future Work

We have introduced protein fold family-regularization for Bayesian ML-assisted directed evolution. Using log-odds scores from gremlin and Pfam HMM's, we calculate an adjusted expected improvement score that is biased towards selecting variants with native-like sequences. Using an active learning approach, we show that we can efficiently trade off small losses in variant fitness for larger gains in gremlin and HMM scores, while also outperforming traditional laboratory methods.

In our experiments, we investigated variants at four residue positions, totaling nearly 150,000 sequences. While we only needed small training sets to identify favorable variants, it could be the case that the amount of necessary training data increases as the number of residue positions under investigation increases. A downside with using GP as the underlying model is that they do not scale well with large data, as inference with a GP has complexity $O(n3)$. A natural extension to our procedure then is to incorporate recent advances in learning sparse GPs, using either pseudoinputs or variational techniques [29, 13]. We could also imagine improving our model by using kernels that are specifically designed with sequential data in mind [18, 19].

Finally, we have so far observed that using fold family-regularization induces a fitness tradeoff. However, our model does not attempt to harness this tradeoff in any way. We can imagine a utility where a user specifies a tradeoff parameter which dictates by how much the regularization should favor the generative model for a protein fold family compared to fitness. It could be informative to use such a utility to identify a Pareto optimal frontier over variant sequences (ex. [25]), where sequences on the boundary correspond to those that jointly optimize fitness and probability under a given fold family model.

References

[1] Arnold, Frances H. (2018). Directed evolution: Bringing new chemistry to life. *Angewandte Chemie International Edition, 57*(16), 4143–4148. https://doi.org/10.1002/anie.201708408

[2] Balakrishnan, S., Kamisetty, H., Carbonell, J. C., Lee, S. I., Langmead, C. J. (2011). Learning generative models for protein fold families. *Proteins: Structure, Function, and Bioinformatics, 79*(6), 1061–1078.

[3] Bergstra, James S., Bardenet, Rémi., Bengio, Yoshua., Kégl, Balázs. (2011). Algorithms for hyperparameter

optimization. In Shawe-Taylor, J., Zemel, R. S., Bartlett, P. L., Pereira, F., Weinberger, K. Q. (Eds.), *Advances in Neural Information Processing Systems 24* (pp. 2546–2554). Curran Associates, Inc.

[4] Boas, F. Edward., Harbury, Pehr B. (2007). Potential energy functions for protein design. *Current Opinion in Structural Biology, 17*(2), 199–204. https://doi.org/10.1016/j.sbi.2007.03.006

[5] El-Gebali, Sara., Mistry, Jaina., Bateman, Alex., Eddy, Sean R., Luciani, Aurélien., Potter, Simon C., Qureshi, Matloob., Richardson, Lorna J., Salazar, Gustavo A., Smart, Alfredo., Sonnhammer, Erik L. L., Hirsh, Layla., Paladin, Lisanna., Piovesan, Damiano., Tosatto, Silvio C. E., Finn, Robert D. (2018). The Pfam protein families database in 2019. *Nucleic Acids Research, 47*(D1), D427–D432.

[6] Gatti-Lafranconi, Pietro., Natalello, Antonino., Rehm, Sascha., Doglia, Silvia Maria., Pleiss, Jürgen., Lotti, Marina. (2010). Evolution of stability in a cold-active enzyme elicits specificity relaxation and highlights substrate-related effects on temperature adaptation. *Journal of Molecular Biology, 395*(1), 155–166. https://doi.org/10.1016/j.jmb.2009.10.026

[7] Giger, Lars., Caner, Sami., Obexer, Richard., Kast, Peter., Baker, David., Ban, Nenad., Hilvert, Donald. (2013). Evolution of a designed retro-aldolase leads to complete active site remodeling. *Nature Chemical Biology, 9*(8), 494–498. https://doi.org/10.1038/nchembio.1276

[8] Goldenzweig, Adi., Fleishman, Sarel J. (2018). Principles of protein stability and their application in computational design. *Annual Review of Biochemistry, 87*(1), 105–129. https://doi.org/10.1146/annurev-biochem-062917-012102

[9] Hawkins, Robert E., Russell, Stephen J., Winter, Greg. (1992). Selection of phage antibodies by binding affinity. *Journal of Molecular Biology, 226*(3), 889–896. https://doi.org/10.1016/0022-2836(92)90639-2

[10] Ilten, P., Williams, M., Yang, Y. (2017). Event generator tuning using Bayesian optimization. *Journal of Instrumentation, 12*(04), P04028. https://doi.org/10.1088/1748-0221/12/04/P04028

[11] Krogh, Anders., Brown, Michael., Mian, I. Saira., Sjölander, Kimmen., Haussler, David. (1994). Hidden Markov models in computational biology. *Journal of Molecular Biology, 235*(5), 1501–1531. https://doi.org/10.1006/jmbi.1994.1104

[12] Kuhlman, B., Baker, D. (2000). Native protein sequences are close to optimal for their structures. *Proceedings of the National Academy of Sciences of the United States of America, 97*(19), 10383–10388. https://doi.org/10.1073/pnas.97.19.10383

[13] Liu, Haitao., Ong, Yew-Soon., Shen, Xiaobo., Cai, Jianfei. (2020). When Gaussian process meets big data: A review of scalable GPs. *IEEE Transactions on Neural Networks and Learning Systems*, 1–19.

[14] Lizotte, Daniel J., Wang, Tao., Bowling, Michael H., Schuurmans, Dale. (2007). Automatic gait optimization with Gaussian process regression. In Veloso, Manuela M. (Ed.), *IJCAI* (pp. 944–949). Retrieved from http://ijcai.org/Proceedings/07/Papers/152.pdf

[15] Lutz, Stefan., Bornscheuer, Uwe Theo. (2012). *Protein engineering handbook*. Wiley-VCH.

[16] Mockus, Jonas. (1989). *Bayesian approach to global optimization: Theory and applications* (Vol. 37, Mathematics and Its Applications). Springer Netherlands. https://doi.org/10.1007/978-94-009-0909-0

[17] Movahedi, Marziyeh., Zare-Mirakabad, Fatemeh., Arab, Seyed Shahriar. (2016). Evaluating the accuracy of protein design using native secondary sub-structures. *BMC Bioinformatics, 17*(1), 353. https://doi.org/10.1186/s12859-016-1199-y

[18] Nojoomi, Saghi., Koehl, Patrice. (2017a). String kernels for protein sequence comparisons: Improved fold recognition. *BMC Bioinformatics, 18*(1), 137. https://doi.org/10.1186/s12859-017-1560-9

[19] Nojoomi, Saghi., Koehl, Patrice. (2017b). A weighted string kernel for protein fold recognition. *BMC Bioinformatics, 18*(1), 378. https://doi.org/10.1186/s12859-017-1795-5

[20] Olson, C. Anders., Wu, Nicholas C., Sun, Ren. (2014). A comprehensive biophysical description of pairwise epistasis throughout an entire protein domain. *Current Biology, 24*(22), 2643–2651. https://doi.org/10.1016/j.cub.2014.09.072

[21] Rasmussen, Carl Edward., Williams, Christopher K. I. (2006). *Gaussian processes for machine learning*. MIT Press.

[22] Richardson, Janes S., Richardson, David C. (1989). The de novo design of protein structures. *Trends in Biochemical Sciences, 14*(7), 304–309. https://doi.org/10.1016/0968-0004(89)90070-4

[23] Roberts, Richard W., Szostak, Jack W. (1997). RNA-peptide fusions for the in vitro selection of peptides and proteins. *Proceedings of the National Academy of Sciences, 94*(23), 12297–12302. https://doi.org/10.1073/pnas.94.23.12297

[24] Romero, Philip A., Arnold, Frances H. (2009). Exploring protein fitness landscapes by directed evolution. *Nature Reviews Molecular Cell Biology, 10*(12), 866–876. https://doi.org/10.1038/nrm2805

[25] Salvat, Regina S., Parker, Andrew S., Choi, Yoonjoo., Bailey-Kellogg, Chris., Griswold, Karl E. (2015). Mapping the Pareto Optimal Design Space for a Functionally Deimmunized Biotherapeutic Candidate. *PLoS Computational Biology, 11*(1), e1003988. https://doi.org/10.1371/journal.pcbi.1003988

[26] Shaikh, Fathima Aidha., Withers, Stephen G. (2008). Teaching old enzymes new tricks: Engineering and evolution of glycosidases and glycosyl transferases for improved glycoside synthesis. *Biochemistry and Cell Biology, 86*(2), 169–177. https://doi.org/10.1139/O07-149

[27] Starr, Tyler N., Thornton, Joseph W. (2016). Epistasis in protein evolution. *Protein Science, 25*(7), 1204–1218. https://doi.org/10.1002/pro.2897

[28] Thompson, W. R. (1933). On the likelihood that one unknown probability exceeds another in view of the

evidence of two samples. *Biometrika, 25*(3-4), 285–294. https://doi.org/10.1093/biomet/25.3-4.285

[29] Titsias, Michalis. (2009). Variational learning of inducing variables in sparse Gaussian processes. In D. van Dyk & M. Welling (Eds.), *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics* (Vol. 5, pp. 567–574).

[30] Wilson, James., Hutter, Frank., Deisenroth, Marc. (2018). Maximizing acquisition functions for Bayesian optimization. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 31* (pp. 9884–9895). Curran Associates, Inc.

[31] Wu, Nicholas C., Dai, Lei., Olson, C. Anders., Lloyd-Smith, James O., Sun, Ren. (2016). Adaptation in protein fitness landscapes is facilitated by indirect paths. *eLife, 5*, e16965. https://doi.org/10.7554/eLife.16965

[32] Wu, Zachary., Kan, S. B. Jennifer., Lewis, Russell D., Wittmann, Bruce J., Arnold, Frances H. (2019). Machine learning-assisted directed protein evolution with combinatorial libraries. *Proceedings of the National Academy of Sciences, 116*(18), 8852–8858.

[33] Yang, Kevin K., Wu, Zachary., Arnold, Frances H. (2019). Machine-learning-guided directed evolution for protein engineering. *Nature Methods, 16*(8), 687–694. https://doi.org/10.1038/s41592-019-0496-6