Proximal Exploration for Model-guided Protein Sequence Design

Zhizhou Ren [*1,2] **Jiahan Li** [*1,3] **Fan Ding** [1] **Yuan Zhou** [4] **Jianzhu Ma** [3] **Jian Peng** [1,2,5]

Abstract

Designing protein sequences with a particular biological function is a long-lasting challenge for protein engineering. Recent advances in machine-learning-guided approaches focus on building a surrogate sequence-function model to reduce the burden of expensive in-lab experiments. In this paper, we study the exploration mechanism of model-guided sequence design. We leverage a natural property of protein fitness landscape that a concise set of mutations upon the wild-type sequence are usually sufficient to enhance the desired function. By utilizing this property, we propose Proximal Exploration (PEX) algorithm that prioritizes the evolutionary search for high-fitness mutants with low mutation counts. In addition, we develop a specialized model architecture, called Mutation Factorization Network (MuFacNet), to predict low-order mutational effects, which further improves the sample efficiency of model-guided evolution. In experiments, we extensively evaluate our method on a suite of in-silico protein sequence design tasks and demonstrate substantial improvement over baseline algorithms.

1. Introduction

Protein engineering aims to discover novel proteins with useful biological functions, such as fluorescence intensity (Biswas et al., 2021), enzyme activity (Fox et al., 2007), and therapeutic efficiency (Lagassé et al., 2017), where the functions of a particular protein is determined by its amino-acid sequence (Crick, 1958; Nelson et al., 2008). The *protein fitness landscape* (Wright, 1932) characterizes the mapping between protein sequences and their functional levels (*aka.* fitness scores). It formulates the engineering of proteins as a sequence design problem. The evolution in nature can be regarded as a searching procedure on the protein fitness landscape (Smith, 1970), which is driven by natural selection pressure. This natural process inspires the innovation of *directed evolution* (Arnold, 1998), the most widely-applied approach for engineering protein sequences. It mimics the evolution cycle in a laboratory environment and conducts an iterative protocol. At each iteration, enormous variants are generated and scored by functional assays, in which sequences with high fitness are selected to form the next generation. Recent attempts focus on using machine learning approaches to build a surrogate model of protein landscape (Yang et al., 2019; Wittmann et al., 2021) and designing model-guided searching schemes (Biswas et al., 2018). This paradigm can effectively reduce the burden of laboratory experiments by performing in-silico population selection through model-based fitness prediction.

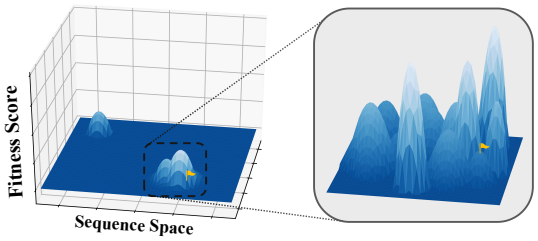


Figure 1. (Left) Functional proteins are rare in the sequence space. The yellow flag corresponds to the wild-type sequence. (Right) The local landscape near the wild type is rugged. A few amino-acid mutations may dramatically alter the protein fitness.

The exploration mechanism of directed evolution is a simple greedy strategy. It starts from the wild-type sequence and continuously accumulates beneficial mutations in a hill-climbing manner on the fitness landscape (Romero & Arnold, 2009). As the evolution is conducted in both the real landscape and the surrogate model, this unconstrained greedy search may lead to sequences far from the wild type. These sequences with high mutation counts are laborious to synthesize for mass production (Storici & Resnick, 2006; Fowler et al., 2014). In addition, it is hard for the training procedure of the surrogate model to reuse previous samples that are far from the current search region.

However, as shown in Figure 1, the protein landscape is

[*]Equal contribution [1]HeliXon Limited [2]Department of Computer Science, University of Illinois at Urbana-Champaign [3]Institute for Artificial Intelligence, Peking University [4]Yau Mathematical Sciences Center, Tsinghua University [5]Institute for Industry AI Research, Tsinghua University. Correspondence to: Jian Peng <jianpeng@illinois.edu>.

Proceedings of the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

known to be sparse in the vast sequence space, where the local landscape near the wild-type point is rugged and multi-peaked. It implies that a few amino-acid mutations on key positions are sufficient to dramatically alter the fitness score of the protein (Weinreich et al., 2005). Searching through regions with low mutations around wild-type sequence can improve searching effectiveness and reduce wet-lab labor.

Regarding the natural property of the protein landscape, we propose a novel exploration mechanism that goes beyond the classical paradigm of directed evolution. Our exploration method, named *Proximal Exploration* (PEX), aims to search for effective candidates of low-order mutants (*i.e.*, mutants near the wild-type sequence) rather than greedily accumulate mutations along the evolutionary pathway. By formulating the local search around the wild-type sequence as a proximal optimization problem (Parikh & Boyd, 2014), we derive a regularized objective for searching the steepest improvement upon the wild type with the limited counts of amino-acid mutations. Furthermore, we design a specialized network architecture, called *Mutation Factorization Network* (MuFacNet), to model low-order mutational effects based on local neighbor information, which additionally improves the sample efficiency of our model-guided sequence design. Our method shows great performance over baselines on multiple protein sequence design benchmarks.

2. Related Work

ML for Protein Landscape Modeling. Advances in experimental technologies, such as deep mutational scanning (Fowler & Fields, 2014), yield large-scale generation of mutant data. It enables data-driven approaches to model the protein fitness landscape, which is one of the crucial problems for protein engineering. Recent works start to focus on leveraging co-evolution information from multiple sequence alignments to predict fitness scores (Hopf et al., 2017; Riesselman et al., 2018; Luo et al., 2021) and utilizing pre-trained language models to conduct transfer learning or zero-shot inference (Rao et al., 2019; Alley et al., 2019; Meier et al., 2021; Hsu et al., 2022). The learned ML model can be used to screen enormous designed sequences in silico (Ogden et al., 2019; Bryant et al., 2021; Gruver et al., 2021; Shan et al., 2022), which serves as a surrogate of expensive wet-lab experiments.

Exploration Algorithms for Sequence Design. Directed evolution is a classical paradigm for protein sequence design, which has achieved several successes (Bloom & Arnold, 2009; Dalby, 2011; Arnold, 2018). Under this framework, machine learning algorithms play an important role in improving the sample efficiency of evolutionary search. Angermueller et al. (2020a) proposes an ensemble approach that performs online adaptation among a heterogeneous population of optimization algorithms. Angermueller et al. (2020b) formulates sequence design as a sequential decision-making problem and conducts model-based reinforcement learning for efficient exploration. Brookes & Listgarten (2018) and Brookes et al. (2019) use adaptive sampling to generate high-quality mutants for batch Bayesian optimization. Zhang et al. (2021) proposes a probabilistic framework that unifies black-box sequence design and likelihood-free inference and presents a methodology to develop new algorithms for sequence design.

Proximal Optimization. Proximal methods are a family of optimization algorithms that decompose non-differentiable large-scale problems to a series of localized smooth optimization (Parikh & Boyd, 2014). The most representative approach is proximal gradient (Combettes & Wajs, 2005), which conducts a convex step-wise regularization to smooth the optimization landscape without losing the optimality of convergence. It is related to natural gradient (Amari, 1998) and trust-region methods (Sorensen, 1982). These methods aim to find the steepest improvement direction in a restricted small region. In this paper, we adopt the formulation of proximal methods to deploy localized optimization around the wild-type sequence. We propose an exploration algorithm to find proximal maximization points (Rockafellar, 1976) for black-box sequence design.

3. Problem Formulation and Background

Protein Sequence Design upon Wild Type. The protein sequence design problem is to search for a sequence s with desired property in the sequence space $\mathcal{V}^L$, where $\mathcal{V}$ denotes the vocabulary of amino acids and L denotes the desired sequence length. We aim to maximize a protein fitness function $f : \mathcal{V}^L \to \mathbb{R}$ by editing sequences. The protein fitness mapping $f(s)$ is a black-box function that can only be measured through wet-lab experiments. As a reference to the design problem, a wild-type sequence $s^{\mathtt{wt}}$ is given as the starting point of sequence search, which is the sequence occurring in nature. In general, the wild type $s^{\mathtt{wt}}$ has already expressed decent fitness under the natural selection evolutionary process towards the target function (Fisher, 1958; Chari & Dworkin, 2013). In this paper, we focus on the modification upon the wild-type sequence $s^{\mathtt{wt}}$ to enhance the desired protein function. Specifically, we aim to design low-order mutants that have low mutation counts compared to the wild-type sequence.

Batch Black-Box Optimization. Protein sequence design with respect to a fitness function $f(s)$ is a batch black-box optimization problem. Advances in experiment technology have enabled the parallel fitness measurement over a large batch of sequences (Kosuri et al., 2013; Peterman & Levine, 2016). However, each round of parallel experiments in the wet lab is expensive and time-consuming. It leads to an online learning problem with high throughput but limited

interactions. Formally, we conduct T rounds of batch optimization with batch size $M \gg T$ for sequence-fitness measurements within each round.

Model-directed Evolution. Building a surrogate model for in-silico evolutionary selection is an effective approach to reduce the burden of expensive wet-lab experiments. It trains a fitness model $\hat{f}_\theta$, parameterized by θ, to predict the fitness of mutant sequences. More specifically, the surrogate model optimizes the following regression loss for fitness prediction:

$$\mathcal{L}(\theta) = \mathop{\mathbb{E}}_{s \sim \mathcal{D}} \left[(\hat{f}_\theta(s) - f(s))^2 \right], \tag{1}$$

where $\mathcal{D}$ is a dataset of experimentally measured sequences. The learned surrogate model $\hat{f}_\theta$ can be used to predict the fitness of unseen sequences, which can thus guide in-silico sequence searching and improve the sample-efficiency of directed evolution.

4. Proximal Exploration for Sequence Design

In this section, we introduce our approach, Proximal Exploration (PEX), which prioritizes the search for high-fitness sequences with low mutation counts upon the wild type. We formulate this localized search as a proximal optimization problem and implement the exploration mechanism through a model-guided approach. In addition, we propose a specialized network architecture to model the local landscape around the wild type, which further improves the efficiency of proximal exploration.

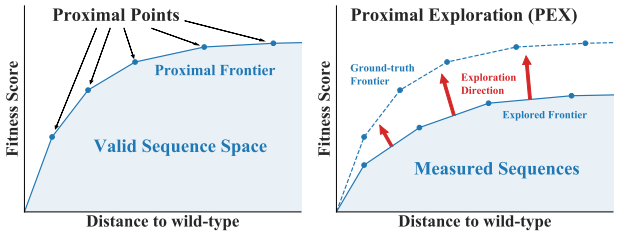
4.1. Overview of Proximal Exploration

In the natural evolutionary process, it is usually sufficient for a protein to significantly enhance its fitness score by mutating only a few amino acids within a long sequence. Motivated by this natural principle, we consider restricting the search space near the wild type and seek for high-fitness mutants with low mutation counts. Based on the paradigm of directed evolution, the basic idea is to adopt a regularized objective that prioritizes low-order mutants in the sequence selection step. *i.e.*, our algorithm prefers to select sequences with low mutation counts for artificial evolution. Formally, we employ the proximal optimization framework to formulate such a localized searching scheme.

Evolution via Proximal Optimization. Following the formulation of proximal optimization (Parikh & Boyd, 2014), we define a regularized objective function that penalizes the deviation from the wild-type sequence $s^{\mathtt{wt}}$:

$$f_\lambda^{\text{prox}}(s) = f(s) - \lambda \cdot d(s, s^{\mathtt{wt}}), \tag{2}$$

where $f(\cdot)$ is the original objective function, *i.e.*, the protein fitness score. $\lambda \geq 0$ denotes the regularization coefficient.



(a) Frontier of Proximal Points (b) Proximal Exploration

Figure 2. (a) A geometric illustration of the definition of proximal points and proximal frontier. (b) The goal of proximal exploration is finding the ground-truth proximal frontier.

$d(\cdot, \cdot)$ denotes a distance metric in the sequence space. In this paper, without further specification, the metric $d(\cdot, \cdot)$ refers to evolutionary distance (*i.e.*, Levenshtein edit distance). We aim to find a *proximal maximizer* (or *proximal point* (Rockafellar, 1976)) with respect to the wild-type sequence $s^{\mathtt{wt}}$:

$$\text{prox}_\lambda(s^{\mathtt{wt}}, f) = \mathop{\arg\max}_{s \in \mathcal{V}^L} f_\lambda^{\text{prox}}(s). \tag{3}$$

The value of λ controls the weight of proximal regularization. With larger values of λ, the searching procedure driven by Eq. (2) would become more concentrated around the wild-type sequence, and go far otherwise. The degraded case with $\lambda = 0$ is equivalent to the maximization of the original objective function: $\text{prox}_{\lambda=0}(s^{\mathtt{wt}}, f) = \arg\max_{s \in \mathcal{V}^L} f(s)$. By varying the value of regularization coefficient λ, we define the *proximal frontier* as follows:

$$\text{prox}(s^{\mathtt{wt}}, f) = \left\{ \text{prox}_\lambda(s^{\mathtt{wt}}, f) : \lambda \geq 0 \right\}. \tag{4}$$

Geometric Interpretation. Figure 2a shows a geometric illustration of proximal frontier $\text{prox}(s^{\mathtt{wt}}, f)$. It interprets why this set of proximal points is named as a *frontier*. We plot a coordinate system where each protein sequence is represented by its fitness score $f(s)$ and its distance $d(s, s^{\mathtt{wt}})$ to the wild-type sequence $s^{\mathtt{wt}}$. The sequences in $\text{prox}(s^{\mathtt{wt}}, f)$ are located at the upper frontier on the convex closure of valid sequence space. These proximal points are Pareto-efficient solutions to trade off between maximizing protein fitness score $f(s)$ and minimizing the evolutionary distance $d(s, s^{\mathtt{wt}})$ to the wild-type sequence, *i.e.*, the trade-off between the major objective and the regularization term. The sequences lying on the left-upper region of this coordinate system correspond to the mutants with high fitness and low mutation counts, which is the goal of our method.

Proximal Exploration (PEX). In classical evolutionary algorithms, only top-fitness sequences are selected and mutated within the evolution cycle. Such a greedy evolutionary

Algorithm 1 Proximal Exploration (PEX)

1: **Input:** wild-type sequence $(s^{\mathtt{wt}}, f(s^{\mathtt{wt}}))$
2: Initialize model parameter θ
3: Initialize buffer $\mathcal{D}^{(0)} \leftarrow \{(s^{\mathtt{wt}}, f(s^{\mathtt{wt}}))\}$
4: **for** $t = 1$ **to** T **do**
5: Generate query sequence batch ▷ section 4.2
$$\{s_i^{(t)}\}_{i=1}^M \leftarrow \text{PEX.QueryBatch}(\hat{f}_\theta, \mathcal{D}^{(t-1)})$$
6: Measure ground-truth fitness by wet-lab experiments
$$\mathcal{D}^{(t)} \leftarrow \mathcal{D}^{(t-1)} \cup \{(s_i^{(t)}, f(s_i^{(t)}))\}_{i=1}^M$$
7: Update fitness model $\hat{f}_\theta$ using $\mathcal{D}^{(t)}$ ▷ section 4.3
8: **end for**

process hinders the efficiency of exploration and does not utilize the property of the natural protein fitness landscape. We consider a different mechanism that performs exploration along the proximal frontier and seeks high-fitness mutant sequences with low mutation counts. As shown in Figure 2b, our exploration algorithm, Proximal Exploration (PEX), aims to extend the proximal frontier instead of only optimizing the fitness score. We consider this proximal exploration mechanism as a regularization of the search space of protein sequences.

The overall procedure of our algorithm is summarized in Algorithm 1. Following the problem formulation introduced in section 3, our algorithm conducts T rounds of interaction with the laboratory. At each round, we propose a query batch containing M sequence candidates (line 5) and measure their fitness scores through the wet-lab experiments (line 6). The measured protein fitness data are used to refine the fitness model $\hat{f}_\theta$ where θ denotes the model parameter (line 7). The major technical improvement of our method contains two parts:

1. We formalize the mechanism of proximal exploration through a model-guided approach (section 4.2).
2. We design a specialized model architecture to predict low-order mutational effects of proximal points, which further boosts the efficiency of model-guided exploration (section 4.3).

4.2. Model-guided Exploration on Proximal Frontier

In each round of batch black-box optimization, the exploration algorithm is required to generate a query batch given the experimental measurements collected in previous rounds. More specifically, a model-guided exploration algorithm constructs the query batch based on the fitness model $\hat{f}_\theta$ and a dataset of measured sequences $\mathcal{D}$. The query generation procedure of PEX is summarized in Algorithm 2. The same as other model-guided methods, we first mutate

Algorithm 2 PEX.QueryBatch$(\hat{f}_\theta, \mathcal{D})$

1: **Input:** fitness model $\hat{f}_\theta$, measured buffer $\mathcal{D}$
2: Generate a candidate pool $\mathcal{D}^{\text{pool}}$ by generating random mutations upon the measured sequences $\mathcal{D}$
3: Initialize query batch $\mathcal{D}^{\text{query}} \leftarrow \emptyset$
4: **while** $|\mathcal{D}^{\text{query}}| < M$ **do**
5: Find proximal frontier based on fitness model $\hat{f}_\theta$
$$\mathcal{D}^{\text{prox}} \leftarrow \text{prox}(s^{\mathtt{wt}}, \hat{f}_\theta, \mathcal{D}^{\text{pool}})$$
6: Update query batch $\mathcal{D}^{\text{query}}$ and candidate pool $\mathcal{D}^{\text{pool}}$
$$\mathcal{D}^{\text{query}} \leftarrow \mathcal{D}^{\text{query}} \cup \mathcal{D}^{\text{prox}}, \quad \mathcal{D}^{\text{pool}} \leftarrow \mathcal{D}^{\text{pool}} \setminus \mathcal{D}^{\text{prox}}$$
7: **end while**
8: **return:** M sequences from $\mathcal{D}^{\text{query}}$

sequences in measured dataset $\mathcal{D}$ to obtain a large set of random sequences $\mathcal{D}^{\text{pool}}$ (*i.e.*, a candidate pool of offspring in evolution) and then perform a model-based selection over these mutants.

Classical evolutionary algorithms perform a greedy selection to determine the query batch. The selection criteria is given by the model fitness prediction $\hat{f}_\theta(s)$. *i.e.*, only sequences with high predicted fitness scores will be sent to laboratory measurements. In comparison, PEX considers a different objective to construct the query batch. To expand the frontier shown in Figure 2b, we compute the proximal frontier within the candidate pool $\mathcal{D}^{\text{pool}}$:

$$\text{prox}(s^{\mathtt{wt}}, \hat{f}_\theta, \mathcal{D}^{\text{pool}}) = \left\{ \arg\max_{s \in \mathcal{D}^{\text{pool}}} \hat{f}^{\text{prox}}_{\theta,\lambda}(s) : \lambda \geq 0 \right\}, \quad (5)$$

which is expected to explore Pareto-efficient sequences with either higher fitness or lower mutation count.

At each round of interaction with the wet lab, the query budget can support the experimental measurement of M sequences. In addition to the sequences exactly lying on the proximal frontier defined in Eq. (5), we diversify the query batch by including the nearby region of the frontier. We conduct an iterative query generation mechanism as shown in Algorithm 2. We iteratively compute the proximal frontier over the current candidate pool $\mathcal{D}^{\text{pool}}$ (line 5) and remove them from the pool (line 6). It enables us to effectively deploy proximal exploration in the batch-query setting.

Remark. The computation of $\text{prox}(s^{\mathtt{wt}}, \hat{f}_\theta, \mathcal{D}^{\text{pool}})$ defined in Eq. (5) is equivalent to find the upper convex closure of $\mathcal{D}^{\text{pool}}$ in the coordinate system illustrated by Figure 2a, where the fitness scores are predicted by the surrogate model $\hat{f}_\theta(s)$. In our implementation, we use Andrew's monotone chain convex hull algorithm (Andrew, 1979), which can be completed in $O(n \log n)$ time with $n = |\mathcal{D}^{\text{pool}}|$. Implementation details are included in Appendix A.2.

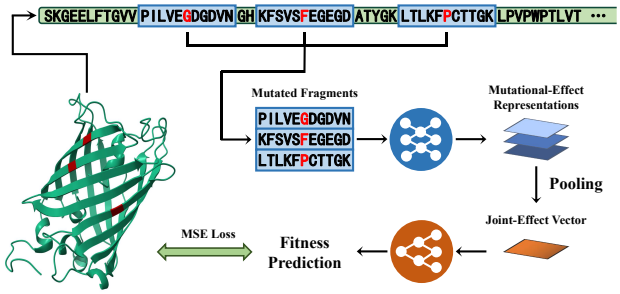


Figure 3. The model architecture and the forward flow graph of Mutation Factorization Network. The red marked sites are mutated amino acids upon the wild type. The illustrative example corresponds to the green fluorescent protein (PDB code: 2O29).

4.3. Modeling the Landscape of Low-order Mutants

One challenge of landscape modeling is the high sensitivity of protein fitness concerning amino-acid mutations. As illustrated in Figure 1, a few amino-acid substitutions may dramatically alter the fitness scores. Classical model architectures, such as CNN and RNN, represent the protein landscape as a sequence-fitness mapping. Since the fitness function is highly non-smooth in the sequence space, these models usually require a large amount of mutant data to capture the complicated mutational effects, which do not meet the demand of sequence design in data efficiency.

To address the challenge of non-smoothness for landscape modeling, we consider to leverage the algorithmic property of proximal exploration. Note that our exploration mechanism prioritizes the search for low-order mutants close to the wild-type sequence. This special property motivates us to design a specialized network architecture to model the local fitness landscape around the wild type. We propose a Mutation Factorization Network (MuFacNet) that factorizes the composition of mutational effects to the interactions among single amino-acid mutations. The major characteristic of MuFacNet is its input representation. It represents a mutant sequence by the corresponding set of amino-acid mutations upon the wild type, which emphasizes the effects of each single mutation site.

Forward View: Aggregation. Figure 3 presents the architecture of MuFacNet. Given an input mutant sequence, we first locate the sites of mutations in comparison to the wild type. We take the neighbor fragment of each mutated amino acid as its context information, *i.e.*, a context window centered at the mutated amino acid. Each context window is then fed into an encoder network to generate the feature vector of the corresponding mutation site. We would obtain K separate feature vectors for K mutations, which are computed by a shared encoder network. These feature vectors encode the functionality of the given mutations. We perform a sum-pooling operator to aggregate them into a unified vector that represents the joint effect of mutations. Finally, the protein fitness prediction is given by a decoder network taking the joint-effect vector as its input.

Backward View: Factorization. The supervision of MuFacNet is the same as the standard model training procedure. It learns from sequence-fitness data through an end-to-end paradigm. The data obtained in wet-lab measurements only indicate the joint effects of mutation compositions, and the design purpose of MuFacNet is to factorize the joint fitness value to local representations of every single mutation site. The pooling operator is the core component to realize this joint-to-local factorization. It characterizes the interaction among mutation sites through the arithmetical composition of their feature vectors.

Remark. The architecture of MuFacNet is a variant of set model that represents functions defined on sets. The sum-pooling operator ensures the model output is invariant to the order of input mutation context windows. Zaheer et al. (2017) proves that such a sum-pooling aggregation operator can preserve the universal approximability of neural networks. *i.e.*, with sufficiently expressive encoder and decoder, a set model with sum-pooling aggregation can represent any functions mapping sets to values. This property ensures that MuFacNet can learn non-additive epistasis effects.

5. Experiments

In this section, we present experiments to evaluate the performance of our method. We show that PEX together with MuFacNet achieves state-of-the-art performance and significantly outperforms baselines in section 5.2 and 5.3. In addition, we conduct several ablation studies to investigate the functionality of each designed component, including the design principle of MuFacNet and the local-search mechanism of PEX.

5.1. Protein Engineering Benchmarks

We evaluate our exploration algorithm and baseline methods on a suite of in-silico protein engineering benchmarks. Following prior works, we simulate the ground-truth protein fitness landscape by an oracle model in replace of the wet-lab measurements. As the black-box optimization setting defined in section 3, the exploration protocol can only interact with the simulated landscape through batch queries. The sequence design algorithm cannot obtain any extra information about the black-box oracle.

We collect several large-scale datasets from previous experimental studies of protein landscape and use TAPE (Rao et al., 2019) as an oracle model to simulate the landscape. These fitness datasets involve extensive applications of protein engineering, including biosensors design, thermosta-

bility improvement, and industrial enzyme renovation. It enables us to formally compare the performance of exploration algorithms in an in-silico sandbox. The source code of our algorithm implementation and oracle landscape simulation models are available at `https://github.com/HeliXonProtein/proximal-exploration`.

Green Fluorescent Proteins (avGFP). Green Fluorescent Proteins from *Aequorea victoria*, which can exhibit bright green fluorescence when exposed to light in the blue to the ultraviolet range, are frequently used as biosensors to detect gene expressions and detect protein locations. Sarkisyan et al. (2016) assayed the fluorescence levels of nearly 52000 genotypes of avGFP obtained by random mutagenesis to the 239-length wild-type sequence, used as our landscape. Our goal is to optimize sequences with higher log-fluorescence intensity values in this 20^{238} search space.

Adeno-associated Viruses (AAV). Engineering a 28-amino acid segment (position 561-588) of the VP1 protein located in the capsid of the Adeno-associated virus has drawn lots of attention in ML-guided design, aiming to remain viable for packaging of a DNA payload for gene therapy (Ogden et al., 2019). We adopt a library with approximately $284,000$ mutant data developed by Bryant et al. (2021) to build the landscape oracle. We aim to design more capable sequences as gene delivery vectors measured by AAV viabilities. The size of search space is 20^{28}.

TEM-1 β-Lactamase (TEM). TEM-1 β-Lactamase protein resisting to penicillin antibiotics in *E.coli* is widely studied to understand mutational effect and fitness landscape (Bershtein et al., 2006; Jacquier et al., 2013). We collect the fitness data from Firnberg et al. (2014) and Gonzalez & Ostermeier (2019), resulting in 17857 sequences as the landscape training data, where fitness is the view as thermodynamic stability. The optimization problem is to propose high thermodynamic-stable sequences upon wild-type TEM-1 in the search space with size 20^{286}.

Ubiquitination Factor Ube4b (E4B). Ubiquitination factor Ube4b plays an important role in the trash degradation process in the cell by interacting with ubiquitin and other proteins. Starita et al. (2013) scored approximately $100,000$ mutation protein sequences by measuring their rates of ubiquitination to the target protein. We focus on designing E4B with higher enzyme activity on the landscape above. The size of search space is 20^{102}.

Aliphatic Amide Hydrolase (AMIE). Amidase encoded by *amiE* is an industrially-relevant enzyme from *from Pseudomonas aeruginosa*. By quantifying the growth rate of each bacterial strain carrying specific amidase mutants, Wrenbeck et al. (2017) measures 6629 sequences with single mutations to model the fitness landscape. We seek for optimizing amidase sequences which lead to great enzyme activities. It defines a search space with 20^{341} sequences.

Levoglucosan Kinase (LGK). Levoglucosan kinase converts LG to the glycolytic intermediate glucose-6-phosphate in an ATP-dependent reaction. A mutant library containing 7891 mutated protein has been established and evaluated using the deep mutational scanning method to recognize mutational effects (Klesmith et al., 2015). Optimizing such protein for improved enzyme activity fitness is our goal. The size of search space is 20^{439}.

Poly(A)-binding Protein (Pab1). Some proteins function by binding to other biomolecules, *e.g.*, RNA or DNA. Pab1 is one of them that binds to multiple adenosine monophosphates (poly-A) using the RNA recognition motif (RRM). Melamed et al. (2013) develop high throughput screening to assay the binding fitness of around 36000 double mutants of Pab1 variants in RRM region. As above, we desire to improve the binding fitness by introducing beneficial mutations to the wild-type sequence. The search space size is 20^{75} on a segment of the wild-type sequence.

SUMO E2 conjugase (UBE2I). Utilizing variants to function mapping of human genomes is substantial for scientific research and clinic treatment. We put to use around 2000 variants of the disease-relevant protein, human SUMO E2 conjugase constructed by Weile et al. (2017) and boost the fitness measured by growth rescue rate at high temperature in a yeast strain. The size of search space is 20^{159}.

5.2. Performance Comparison to Baseline Algorithms

We compare the performance of PEX with several baseline algorithms to design functional proteins by exploring the fitness landscape.

AdaLead (Sinai et al., 2020) is an advanced implementation of model-guided evolution. At each round of batch query, AdaLead first performs a hill-climbing search on the learned landscape model and then queries the sequences with high predicted fitness. Such a simple implementation of model-directed evolution has been demonstrated to achieve competitive performance against many elaborate algorithms.

DyNA PPO (Angermueller et al., 2020b) formulates protein sequence design as a sequential decision-making problem and uses model-based reinforcement learning to perform sequence generation. It applies proximal policy optimization (PPO; Schulman et al., 2017) to search sequences on a learned landscape model. Different from our proximal approach, PPO performs a regularization between the policies of two consecutive rounds. Since a slight alter in policy space may dramatically change the sequence generation distribution, DyNA PPO does not restrict the search space around wild type.

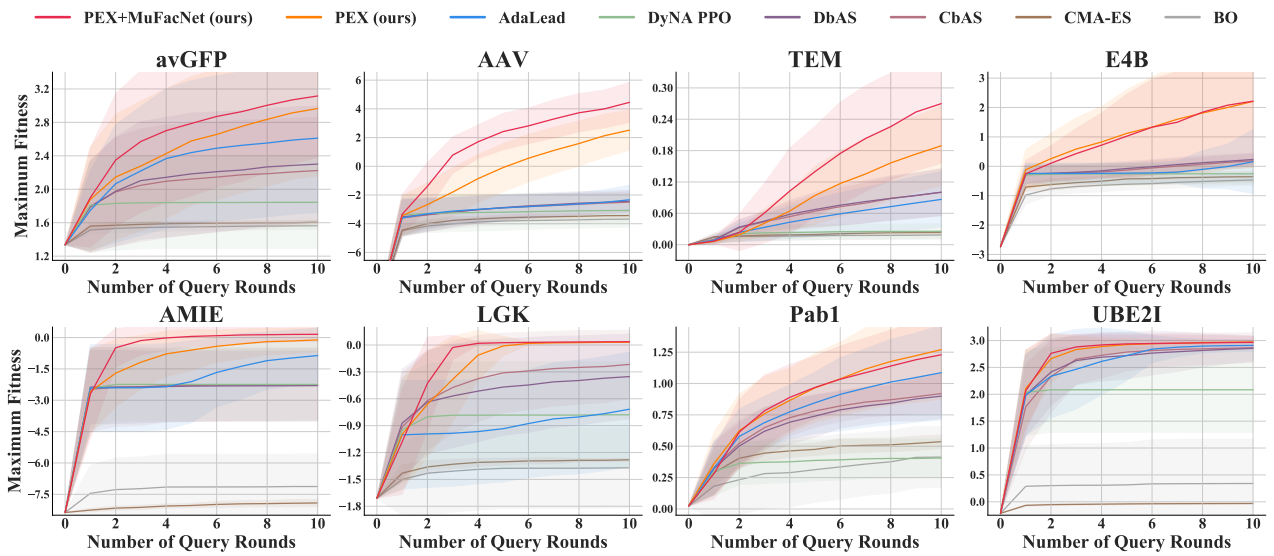


Figure 4. Learning curves on a suite of protein engineering benchmark tasks. Each round of batch black-box query can measure the fitness scores of 100 sequences. The evaluation metric is the cumulative maximum fitness score among queried sequences. All curves are plotted from 80 runs with random network initialization. The shaded region indicates the standard deviation.

DbAS (Brookes & Listgarten, 2018) establishes a probabilistic framework that uses model-based adaptive sampling to explore the fitness landscape. It trains a variational auto-encoder (VAE; Kingma & Welling, 2013) to model the distribution of high-fitness sequences and performs model-guided evolution using this sequence sampler.

CbAS (Brookes et al., 2019) follows the same formulation as DbAS and consider a regularization to stabilize the model-guided search. The motivation and the regularization objective of CbAS are similar to the consideration of DyNA PPO. It restricts the sampling distribution to be supported by the given labeled dataset, which leads to a trust-region search concerning the learned surrogate model. It prevents the exploration algorithm from being trapped in the ill-posed regions with poor model generalization.

CMA-ES (Hansen & Ostermeier, 2001) is a famous algorithm for evolutionary search. It is a second-order approach that estimates the covariance matrix to adaptively adjust the search strategy of the upcoming generations.

Bayesian Optimization (BO; Mockus, 2012) is a classical paradigm for the sequential design problem. We adopt the implementation developed by Sinai et al. (2020), which uses an ensemble of models to estimate the uncertainty and construct the acquisition function for exploration.

We consider an ensemble of three CNN models as the default configuration for these model-guided approaches. Specifically, following the implementation of Angermueller et al. (2020b), DyNA PPO uses a hybrid ensemble and conducts an adaptive selection to pick reliable model candidates. In the evaluation of proximal exploration, **PEX+MuFacNet** and **PEX** refer to using an ensemble of three MuFacNets and the default CNN models, respectively. An ablation study on the model configuration is presented in section 5.3.

Performance Comparison. The protein engineering benchmarks introduced in section 5.1 have much longer sequence length than the proteins considered by previous works, which raises a challenge since the size of search space is exponential to the sequence length. The experiment results are presented in Figure 4. It shows that PEX generally outperforms baseline algorithms with significant improvement. Our approach can find sequences with higher fitness scores using fewer rounds of black-box queries, which demonstrates the improvement of sample efficiency. In Appendix B, we conduct an additional suite of performance evaluation with oracle landscape simulators based on ESM-1b (Rives et al., 2021), and our method also significantly outperforms baselines.

5.3. Effectiveness of MuFacNet

The performance of a model-guided algorithm depends on the quality of its back-end surrogate model. To establish an ablation study on the model configuration, we consider three types of model architectures: CNN, RNN, and MuFacNet, where all of these models are implemented by an ensemble of three network instances. CNN with 1D-convolution is a popular choice for protein fitness prediction (Shanehsazzadeh et al., 2020), and we adopt the implementation developed by AdaLead (Sinai et al., 2020). The RNN architecture is adopted from Gruver et al. (2021), which uses stacked

Table 1. Comparing different model architectures for model-guided sequence design. We present the **maximum fitness scores** obtained in 10 rounds of black-box queries. The **top-3 algorithms** of each task are marked by the bold font. "Overall" refers to the normalized score averaged over 8 tasks. Every entry of this table is averaged over 80 runs. The learning curves are included in Appendix C.

Algorithm + Model	avGFP	AAV	TEM	E4B	AMIE	LGK	Pab1	UBE2I	Overall
PEX + MuFacNet	**3.12**	**4.45**	**0.27**	**2.22**	**0.16**	**0.04**	**1.23**	**2.97**	**0.98**
PEX + CNN	**2.97**	**2.52**	**0.19**	**2.21**	-0.11	0.03	**1.27**	**2.97**	**0.89**
PEX + RNN	**2.64**	1.63	**0.28**	**2.31**	**0.05**	0.03	1.13	**2.97**	**0.88**
AdaLead + MuFacNet	1.88	**3.58**	0.15	1.94	**0.16**	**0.04**	**1.25**	2.96	0.78
AdaLead + CNN	2.61	-2.33	0.09	0.16	-0.86	-0.72	1.09	2.91	0.43
AdaLead + RNN	1.88	-2.09	0.04	0.74	-2.05	-0.41	0.86	2.79	0.31
DyNA PPO + MuFacNet	1.86	-3.33	0.02	-0.23	-2.15	-0.52	0.49	2.08	0.05
DyNA PPO + CNN	1.84	-3.22	0.03	-0.20	-2.13	-0.32	0.42	2.17	0.09
DyNA PPO + RNN	1.86	-3.33	0.02	-0.14	-1.84	-0.34	0.44	2.42	0.15
DbAS + MuFacNet	1.90	2.24	0.09	0.61	-0.61	-0.03	0.96	2.90	0.56
DbAS + CNN	2.30	-2.43	0.10	0.23	-2.30	-0.35	0.90	2.85	0.36
DbAS + RNN	2.13	-2.39	0.10	0.34	-1.91	-0.00	0.79	2.90	0.42
CbAS + MuFacNet	2.08	2.22	0.09	0.47	-0.51	**0.04**	0.83	2.93	0.58
CbAS + CNN	2.22	-2.50	0.10	0.19	-2.26	-0.22	0.92	2.87	0.38
CbAS + RNN	2.07	-2.31	0.11	0.31	-1.55	0.00	0.79	2.87	0.43

GRUs (Chung et al., 2014). The implementation details of MuFacNet are included in Appendix A.3.

The experiment results regarding different model architectures are presented in Table 1. The **Overall** evaluation refers to the normalized scores averaged overall tasks. *i.e.*, we normalize the cumulative maximum fitness score of each task to the unit interval $[0, 1]$ and present the average overall tasks. The results show that **PEX+MuFacNet** provides the most stable performance. It achieves the best average performance in 6 out of 8 benchmark tasks and only marginally underperforms in the remaining 2 tasks. MuFacNet can also improve the performance of AdaLead, DbAS, and CbAS. It shows that MuFacNet is a widely applicable model for sequence design based on the given wild type.

5.4. Encoding Context Information of Mutation Sites

The major characteristic of MuFacNet is its input representation that represents a mutant sequence by a set of amino-acid mutations upon the wild type. To demonstrate the superior performance of MuFacNet is supported by such a set-representation, we conduct several ablation studies to investigate the module design of MuFacNet. First, we vary the size of the context window to locate the mutation site. In the default configuration, we take context windows with radius 10 centered at each mutation site as the inputs of MufacNet. In addition, we consider an alternative way to represent the mutated amino acids. We concatenate the positional encoding (Vaswani et al., 2017) with size 32 to the one-hot encoding of the amino-acid mutation and use an MLP-encoder to produce the feature vector of mutation sites. The experiment results are presented in Table 2. The typical implementation of mutation encoding does not largely affect the performance of the downstream protein sequence design. It indicates that the paradigm of set-representation is critical to the performance improvement. In this paper, since we focus on learning from a small set of protein-fitness data, the encoder and decoder architectures of MuFacNet are implemented by simple modules such as 1D-CNN and MLP layers. A future work is to investigate advanced set models (Lee et al., 2019) on large-scale protein fitness datasets.

Table 2. Ablation studies on the methods to represent the context information of each single amino-acid mutation site. The learning curves of all eight landscapes are included in Appendix D.

Mutation Encoding	avGFP	AAV	TEM	E4B
Window (radius=5)	2.96	4.68	0.23	2.62
Window (radius=10)	3.12	4.45	0.27	2.22
Window (radius=15)	2.86	4.58	0.25	2.40
Positional Encoding	3.15	4.32	0.27	2.61
without MuFacNet	2.97	2.52	0.19	2.21

5.5. Discovering Low-Order Mutants with High Fitness

In addition to the ability to discover high-fitness proteins, we present another strength of PEX, *i.e.*, the sequences designed by our method have a low mutation count upon the wild-type sequence. To demonstrate this algorithmic

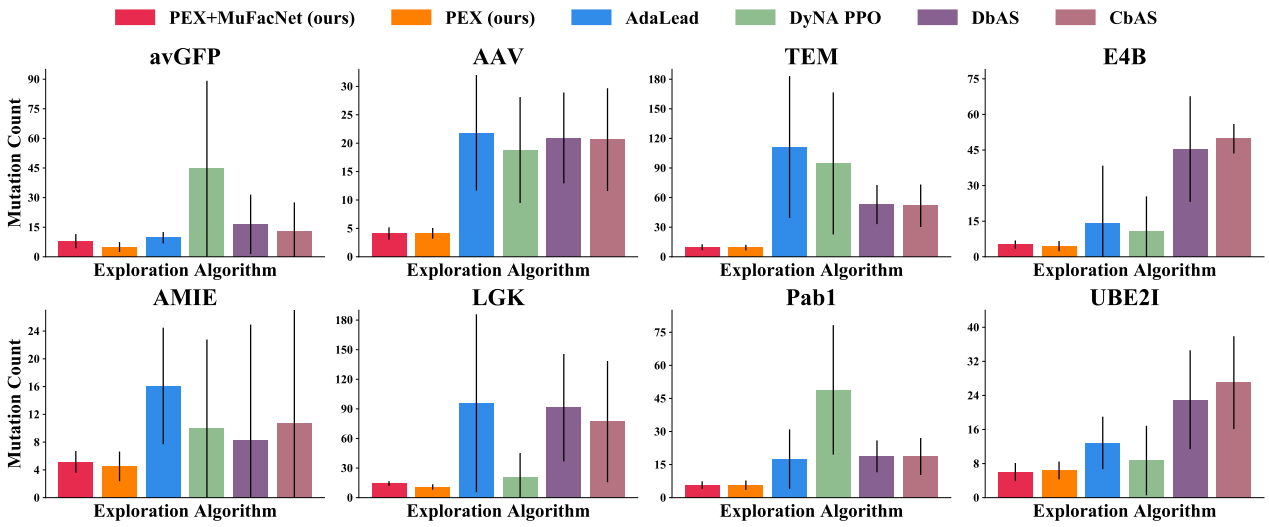


Figure 5. The mutation count of the best-designed sequence for each method in multiple benchmarks. Our methods achieve fewer mutation count while maintaining high fitness compared to other algorithms. The error bar has also been shown in each panel.

property, we collect the best sequence designed by every algorithm and evaluate their mutation counts, *i.e.*, the number of amino-acid mutations required to synthesize these sequences from the wild type sequence. The results are presented in Figure 5. In comparison to baseline algorithms, PEX can discover high-fitness sequences with far less number of mutations upon the wild type. Since synthesizing sequences with a large number of mutation sites is time- and labor-consuming for modern site-directed mutagenesis toolkit (Hogrefe et al., 2002; Carrigan et al., 2011), and mutants with fewer mutation count may also have good performance in other biochemical aspects (Klesmith et al., 2017; Araya & Fowler, 2011), our framework is a more powerful tool for efficient high-throughput protein design.

To demonstrate PEX is an outstanding approach to search in the local landscape around the wild-type sequence, we consider two simple alternative methods to restrict the exploration scope:

1. **Reject by Distance (RD):** Rejecting sequences with more than 10 mutations since Figure 5 has shown that 10 mutation sites are sufficient to cover the high-fitness sequences found by PEX. This simple heuristics is previously used by Biswas et al. (2021) where the distance threshold is named by the "trust radius" of the wild type sequence.
2. **Lower Confidence Bound (LCB):** Using lower confidence bound that minuses the standard deviation of the ensemble predictions from the average to make the exploration algorithm be conservative on landscape regions with high model uncertainty. Similar approaches are used in the literature of model-based reinforcement learning (Kidambi et al., 2020; Yu et al., 2020).

Table 3. Comparison to two simple alternative methods to restrict the exploration scope. The learning curves of all eight landscapes are included in Appendix E.

Algorithm	avGFP	AAV	TEM	E4B
PEX	**2.97**	**2.52**	0.19	**2.21**
AdaLead + RD	2.53	-0.12	**0.20**	1.23
AdaLead + LCB	2.67	-2.30	0.09	-0.06
AdaLead + RD + LCB	2.59	1.25	**0.20**	1.18
CbAS + RD	2.28	-1.51	0.12	0.45
CbAS + LCB	2.30	-2.37	0.13	0.24
CbAS + RD + LCB	2.27	-0.60	0.15	0.36

The experiment results are presented in Table 3. These simple tricks work well in a few cases but cannot fundamentally improve the performance as what is achieved by PEX.

6. Conclusion

This paper propose a novel model-guided directed evolution algorithm, called proximal exploration (PEX), that models the protein sequence design problem by finding the proximal maximization point near the wild-type sequence. It derives a regularized objective function to locally search for sequences with low mutation count upon the wild-type sequence. Based on the localized property of PEX, we propose MuFacNet to specialize in the landscape modeling near the wild type, which factorizes the epistasis effects of mutations to the interactions among the mutated amino acids. In experiments, the integration of PEX and MuFacNet shows great performance improvement compared to prior model-guided protein design baselines in terms of both sample efficiency and the quality of designed sequences.

References

Alley, E. C., Khimulya, G., Biswas, S., AlQuraishi, M., and Church, G. M. Unified rational protein engineering with sequence-based deep representation learning. *Nature methods*, 16(12):1315–1322, 2019.

Amari, S.-I. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

Andrew, A. M. Another efficient algorithm for convex hulls in two dimensions. *Information Processing Letters*, 9(5): 216–219, 1979.

Angermueller, C., Belanger, D., Gane, A., Mariet, Z., Dohan, D., Murphy, K., Colwell, L., and Sculley, D. Population-based black-box optimization for biological sequence design. In *International Conference on Machine Learning*, pp. 324–334. PMLR, 2020a.

Angermueller, C., Dohan, D., Belanger, D., Deshpande, R., Murphy, K., and Colwell, L. Model-based reinforcement learning for biological sequence design. In *International conference on learning representations*, 2020b.

Araya, C. L. and Fowler, D. M. Deep mutational scanning: assessing protein function on a massive scale. *Trends in biotechnology*, 29(9):435–442, 2011.

Arnold, F. H. Design by directed evolution. *Accounts of chemical research*, 31(3):125–131, 1998.

Arnold, F. H. Directed evolution: bringing new chemistry to life. *Angewandte Chemie International Edition*, 57(16): 4143–4148, 2018.

Bershtein, S., Segal, M., Bekerman, R., Tokuriki, N., and Tawfik, D. S. Robustness–epistasis link shapes the fitness landscape of a randomly drifting protein. *Nature*, 444 (7121):929–932, 2006.

Biswas, S., Kuznetsov, G., Ogden, P. J., Conway, N. J., Adams, R. P., and Church, G. M. Toward machine-guided design of proteins. *bioRxiv*, pp. 337154, 2018.

Biswas, S., Khimulya, G., Alley, E. C., Esvelt, K. M., and Church, G. M. Low-N protein engineering with data-efficient deep learning. *Nature Methods*, 18(4):389–396, 2021.

Bloom, J. D. and Arnold, F. H. In the light of directed evolution: pathways of adaptive protein evolution. *Proceedings of the National Academy of Sciences*, 106(Supplement 1): 9995–10000, 2009.

Brookes, D., Park, H., and Listgarten, J. Conditioning by adaptive sampling for robust design. In *International conference on machine learning*, pp. 773–782. PMLR, 2019.

Brookes, D. H. and Listgarten, J. Design by adaptive sampling. *arXiv preprint arXiv:1810.03714*, 2018.

Bryant, D. H., Bashir, A., Sinai, S., Jain, N. K., Ogden, P. J., Riley, P. F., Church, G. M., Colwell, L. J., and Kelsic, E. D. Deep diversification of an AAV capsid protein by machine learning. *Nature Biotechnology*, 39(6):691–696, 2021.

Carrigan, P. E., Ballar, P., and Tuzmen, S. Site-directed mutagenesis. In *Disease Gene Identification*, pp. 107–124. Springer, 2011.

Chari, S. and Dworkin, I. The conditional nature of genetic interactions: the consequences of wild-type backgrounds on mutational interactions in a genome-wide modifier screen. *PLoS genetics*, 9(8):e1003661, 2013.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Combettes, P. L. and Wajs, V. R. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005.

Crick, F. H. On protein synthesis. In *Symp Soc Exp Biol*, volume 12, pp. 8, 1958.

Dalby, P. A. Strategy and success for the directed evolution of enzymes. *Current opinion in structural biology*, 21(4): 473–480, 2011.

Firnberg, E., Labonte, J. W., Gray, J. J., and Ostermeier, M. A comprehensive, high-resolution map of a gene's fitness landscape. *Molecular biology and evolution*, 31 (6):1581–1592, 2014.

Fisher, R. A. *The genetical theory of natural selection*. 1958.

Fowler, D. M. and Fields, S. Deep mutational scanning: a new style of protein science. *Nature methods*, 11(8): 801–807, 2014.

Fowler, D. M., Stephany, J. J., and Fields, S. Measuring the activity of protein variants on a large scale using deep mutational scanning. *Nature protocols*, 9(9):2267–2284, 2014.

Fox, R. J., Davis, S. C., Mundorff, E. C., Newman, L. M., Gavrilovic, V., Ma, S. K., Chung, L. M., Ching, C., Tam, S., Muley, S., et al. Improving catalytic function by ProSAR-driven enzyme evolution. *Nature biotechnology*, 25(3):338–344, 2007.

Gonzalez, C. E. and Ostermeier, M. Pervasive pairwise intragenic epistasis among sequential mutations in TEM-1 β-lactamase. *Journal of molecular biology*, 431(10): 1981–1992, 2019.

Gruver, N., Stanton, S., Kirichenko, P., Finzi, M., Maffettone, P., Myers, V., Delaney, E., Greenside, P., and Wilson, A. G. Effective surrogate models for protein design with bayesian optimization. *ICML Workshop on Computational Biology*, 2021.

Hansen, N. and Ostermeier, A. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation*, 9(2):159–195, 2001.

Hogrefe, H. H., Cline, J., Youngblood, G. L., and Allen, R. M. Creating randomized amino acid libraries with the QuikChange® multi site-directed mutagenesis kit. *Biotechniques*, 33(5):1158–1165, 2002.

Hopf, T. A., Ingraham, J. B., Poelwijk, F. J., Schärfe, C. P., Springer, M., Sander, C., and Marks, D. S. Mutation effects predicted from sequence co-variation. *Nature biotechnology*, 35(2):128–135, 2017.

Hsu, C., Nisonoff, H., Fannjiang, C., and Listgarten, J. Learning protein fitness models from evolutionary and assay-labeled data. *Nature Biotechnology*, pp. 1–9, 2022.

Jacquier, H., Birgy, A., Le Nagard, H., Mechulam, Y., Schmitt, E., Glodt, J., Bercot, B., Petit, E., Poulain, J., Barnaud, G., et al. Capturing the mutational landscape of the beta-lactamase TEM-1. *Proceedings of the National Academy of Sciences*, 110(32):13067–13072, 2013.

Kidambi, R., Rajeswaran, A., Netrapalli, P., and Joachims, T. Morel: Model-based offline reinforcement learning. In *Advances in neural information processing systems*, volume 33, pp. 21810–21823, 2020.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Klesmith, J. R., Bacik, J.-P., Michalczyk, R., and Whitehead, T. A. Comprehensive sequence-flux mapping of a levoglucosan utilization pathway in E. coli. *ACS synthetic biology*, 4(11):1235–1243, 2015.

Klesmith, J. R., Bacik, J.-P., Wrenbeck, E. E., Michalczyk, R., and Whitehead, T. A. Trade-offs between enzyme fitness and solubility illuminated by deep mutational scanning. *Proceedings of the National Academy of Sciences*, 114(9):2265–2270, 2017.

Kosuri, S., Goodman, D. B., Cambray, G., Mutalik, V. K., Gao, Y., Arkin, A. P., Endy, D., and Church, G. M. Composability of regulatory sequences controlling transcription and translation in escherichia coli. *Proceedings of the National Academy of Sciences*, 110(34):14024–14029, 2013.

Lagassé, H. D., Alexaki, A., Simhadri, V. L., Katagiri, N. H., Jankowski, W., Sauna, Z. E., and Kimchi-Sarfaty, C. Recent advances in (therapeutic protein) drug development. *F1000Research*, 6, 2017.

Lee, J., Lee, Y., Kim, J., Kosiorek, A., Choi, S., and Teh, Y. W. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pp. 3744–3753. PMLR, 2019.

Luo, Y., Jiang, G., Yu, T., Liu, Y., Vo, L., Ding, H., Su, Y., Qian, W. W., Zhao, H., and Peng, J. ECNet is an evolutionary context-integrated deep learning framework for protein engineering. *Nature communications*, 12(1): 1–14, 2021.

Meier, J., Rao, R., Verkuil, R., Liu, J., Sercu, T., and Rives, A. Language models enable zero-shot prediction of the effects of mutations on protein function. *Advances in Neural Information Processing Systems*, 34, 2021.

Melamed, D., Young, D. L., Gamble, C. E., Miller, C. R., and Fields, S. Deep mutational scanning of an rrm domain of the saccharomyces cerevisiae poly (A)-binding protein. *Rna*, 19(11):1537–1551, 2013.

Mockus, J. *Bayesian approach to global optimization: theory and applications*, volume 37. Springer Science & Business Media, 2012.

Nelson, D. L., Lehninger, A. L., and Cox, M. M. *Lehninger principles of biochemistry*. Macmillan, 2008.

Ogden, P. J., Kelsic, E. D., Sinai, S., and Church, G. M. Comprehensive AAV capsid fitness landscape reveals a viral gene and enables machine-guided design. *Science*, 366(6469):1139–1143, 2019.

Parikh, N. and Boyd, S. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):127–239, 2014.

Peterman, N. and Levine, E. Sort-seq under the hood: implications of design choices on large-scale characterization of sequence-function relations. *BMC genomics*, 17(1): 1–17, 2016.

Rao, R., Bhattacharya, N., Thomas, N., Duan, Y., Chen, X., Canny, J., Abbeel, P., and Song, Y. S. Evaluating protein transfer learning with TAPE. *Advances in neural information processing systems*, 32:9689, 2019.

Riesselman, A. J., Ingraham, J. B., and Marks, D. S. Deep generative models of genetic variation capture the effects of mutations. *Nature methods*, 15(10):816–822, 2018.

Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., Guo, D., Ott, M., Zitnick, C. L., Ma, J., et al. Biological

structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15):e2016239118, 2021.

Rockafellar, R. T. Monotone operators and the proximal point algorithm. *SIAM journal on control and optimization*, 14(5):877–898, 1976.

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

Sarkisyan, K. S., Bolotin, D. A., Meer, M. V., Usmanova, D. R., Mishin, A. S., Sharonov, G. V., Ivankov, D. N., Bozhanova, N. G., Baranov, M. S., Soylemez, O., et al. Local fitness landscape of the green fluorescent protein. *Nature*, 533(7603):397–401, 2016.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Shan, S., Luo, S., Yang, Z., Hong, J., Su, Y., Ding, F., Fu, L., Li, C., Chen, P., Ma, J., Shi, X., Zhang, Q., Berger, B., Zhang, L., and Peng, J. Deep learning guided optimization of human antibody against SARS-CoV-2 variants with broad neutralization. *Proceedings of the National Academy of Sciences*, 119(11):e2122954119, 2022.

Shanehsazzadeh, A., Belanger, D., and Dohan, D. Is transfer learning necessary for protein landscape prediction? *arXiv preprint arXiv:2011.03443*, 2020.

Sinai, S., Wang, R., Whatley, A., Slocum, S., Locane, E., and Kelsic, E. D. AdaLead: A simple and robust adaptive greedy search algorithm for sequence design. *arXiv preprint arXiv:2010.02141*, 2020.

Smith, J. M. Natural selection and the concept of a protein space. *Nature*, 225(5232):563–564, 1970.

Sorensen, D. C. Newton's method with a model trust region modification. *SIAM Journal on Numerical Analysis*, 19 (2):409–426, 1982.

Starita, L. M., Pruneda, J. N., Lo, R. S., Fowler, D. M., Kim, H. J., Hiatt, J. B., Shendure, J., Brzovic, P. S., Fields, S., and Klevit, R. E. Activity-enhancing mutations in an E3 ubiquitin ligase identified by high-throughput mutagenesis. *Proceedings of the National Academy of Sciences*, 110(14):E1263–E1272, 2013.

Storici, F. and Resnick, M. A. The delitto perfetto approach to in vivo site-directed mutagenesis and chromosome rearrangements with synthetic oligonucleotides in yeast. *Methods in enzymology*, 409:329–345, 2006.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in neural information processing systems*, volume 30, 2017.

Weile, J., Sun, S., Cote, A. G., Knapp, J., Verby, M., Mellor, J. C., Wu, Y., Pons, C., Wong, C., van Lieshout, N., et al. A framework for exhaustively mapping functional missense variants. *Molecular systems biology*, 13(12): 957, 2017.

Weinreich, D. M., Watson, R. A., and Chao, L. Perspective: sign epistasis and genetic costraint on evolutionary trajectories. *Evolution*, 59(6):1165–1174, 2005.

Wittmann, B. J., Johnston, K. E., Wu, Z., and Arnold, F. H. Advances in machine learning for directed evolution. *Current Opinion in Structural Biology*, 69:11–18, 2021.

Wrenbeck, E. E., Azouz, L. R., and Whitehead, T. A. Single-mutation fitness landscapes for an enzyme on multiple substrates reveal specificity is globally encoded. *Nature communications*, 8(1):1–10, 2017.

Wright, S. The roles of mutation, inbreeding, crossbreeding and selection in evolution. In *Proceedings of the sixth international congress of Genetics*, volume 1, pp. 356–366, 1932.

Yang, K. K., Wu, Z., and Arnold, F. H. Machine-learning-guided directed evolution for protein engineering. *Nature methods*, 16(8):687–694, 2019.

Yu, T., Thomas, G., Yu, L., Ermon, S., Zou, J. Y., Levine, S., Finn, C., and Ma, T. Mopo: Model-based offline policy optimization. In *Advances in Neural Information Processing Systems*, volume 33, pp. 14129–14142, 2020.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep sets. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Zhang, D., Fu, J., Bengio, Y., and Courville, A. Unifying likelihood-free inference with black-box sequence design and beyond. *arXiv preprint arXiv:2110.03372*, 2021.

A. Experiment Setting and Implementation Details

A.1. Experiment Setting

As discussed in section 5.1, we collect eight large-scale datasets from previous experimental studies of protein landscape and use TAPE (Rao et al., 2019) as an oracle model to simulate the landscape. Since the wild-type sequences in most datasets have already achieved a high fitness score, we relocate the starting sequence of the search algorithm to the sequence with the lowest fitness in the dataset. It means the starting sequences of these benchmarks are not the true wild-type sequences in the natural environment.

The exploration protocol can only interact with the simulated landscape through batch queries and cannot obtain any extra information about the black-box oracle. The exploration procedure contains 10 rounds of black-box queries. Each batch contains 100 sequences. In Figure 4, we plot the cumulative maximum fitness scores achieved by every algorithm, i.e, the score at round t is the maximum fitness scores overall sequences queried in round 1 to t.

A.2. Implementation Details of PEX

The query generation procedure of PEX contains several steps at each round:

1. As a special case, the first round of query generation is completed by randomly mutating the wild-type sequence without accessing to the surrogate model, since the landscape exploration algorithm has not accumulated any protein-fitness data from the interaction with the black-box oracle. The main procedure of PEX is launched at the second round of query generation.

2. To reduce the computation burden of the surrogate model, we perform a heuristics to make the size of candidate pool $\mathcal{D}^{\text{pool}}$ manageable. We first compute the proximal frontier of the measured sequence buffer $\mathcal{D}$. *i.e.*, we compute the upper convex hull of $\mathcal{D}$ on the coordinate system constructed by (1) the hamming distance $d(s, s^{\texttt{wt}})$ to the wild-type sequence $s^{\texttt{wt}}$ and (2) the fitness score $f(s)$ obtained from the landscape oracle. We use Andrew's monotone chain convex hull algorithm (Andrew, 1979) as follows:

   (a) First, we sort the sequences in buffer $\mathcal{D}$ in the increasing order of $d(s, s^{\texttt{wt}})$. For sequences with the same value of $d(s, s^{\texttt{wt}})$, we only preserve the one with largest fitness score $f(s)$.

   (b) We maintain a stack whose the elements constructs a convex hull. Initially, the stack is set to empty.

   (c) Then, we sweep the sorted sequences. For each sequence, we will intend to push it to the top of the stack. Before the push operation, we will continually pop the elements on the top of the stack that would violate the convex condition when pushing the new element. The convex condition is checked by computing the cross product.

   (d) The prefix of the final stack with monotonic increasing values of $f(s)$ refer to the proximal frontier $\text{prox}(s^{\texttt{wt}}, f)$.

   The above procedure can be completed in $O(n \log n)$ time with $n = |\mathcal{D}^{\text{pool}}|$. We screen the measured sequences near the proximal frontier and only perform random mutations upon these outstanding sequences to construct the candidate pool $\mathcal{D}^{\text{pool}}$, since the mutants based on the known proximal frontier are more likely to be selected in the further steps of PEX algorithm.

3. Following line 4-7 of Algorithm 2, we iteratively compute the proximal frontier of the candidate pool, add the sequences lying on the frontier to the query batch, and remove them from the candidate pool. This procedure is repeated until we gather sufficient sequences in the query batch. The proximal frontier is also computed by Andrew's monotone chain convex hull algorithm.

A.3. Architecture of MuFacNet

The input of MuFacNet is a set of context window of the mutation sites. Each context window is a one-hot encoding for the sequence fragment centered at the corresponding mutation site. The context window size is $L_c = 21$ (*i.e.*, radius$= 10$).

Table 4. The network architecture of MuFacNet.

Input:	K fragments with shape $L_c \times 20$
1D-CNN	number of filters: 32, kernel size: 5 ReLU activation
1D-CNN	number of filters: 32, kernel size: 5 ReLU activation
	Global Max-Pooling
Dense	output size: 128 ReLU activation
Dense	output size: 32
Local Features:	K feature vectors with length 32
	Sum-Pooling
Joint Features:	a feature vector with length 32
Dense	output size: 128 ReLU activation
Dense	output size: 128 ReLU activation
Dense	output size: 1
Fitness Prediction:	a scalar value

We use Adam optimizer with a learning rate 10^{-3} to train the fitness prediction. The loss function is the mean squared error. We stop network training when the training loss does not decrease in 10 epochs. This training procedure is applied to all models considered in this paper.

B. Experiments based on ESM-1b

We consider an alternative oracle model built on the features produced by ESM-1b (Rives et al., 2021). We adopt the ESM-1b feature with dimension 1280 and train an Attention1D decoder to predict the fitness value.

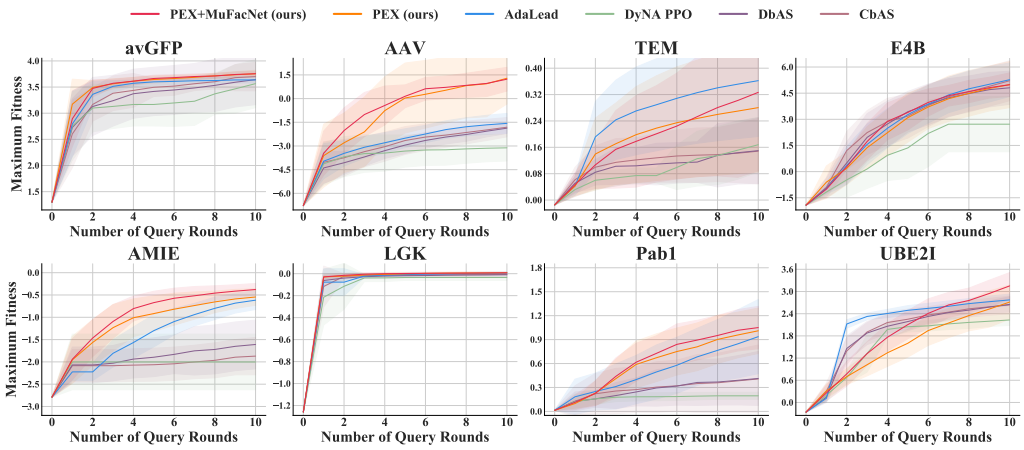


Figure 6. Learning curves of PEX and baselines on a suite of protein engineering tasks simulated by oracle models based on ESM-1b.

C. Learning Curves of the Experiments in Table 1

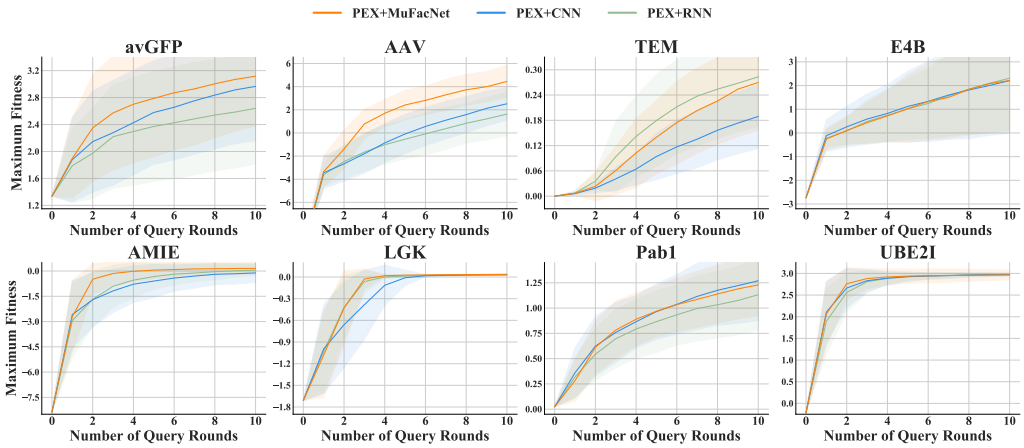


Figure 7. Learning curves of PEX with MuFacNet, CNN, and RNN.

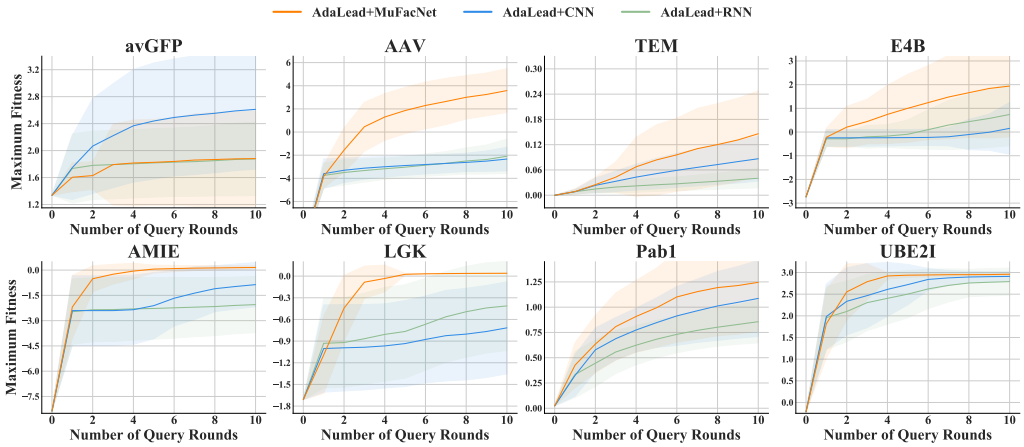


Figure 8. Learning curves of AdaLead with MuFacNet, CNN, and RNN.

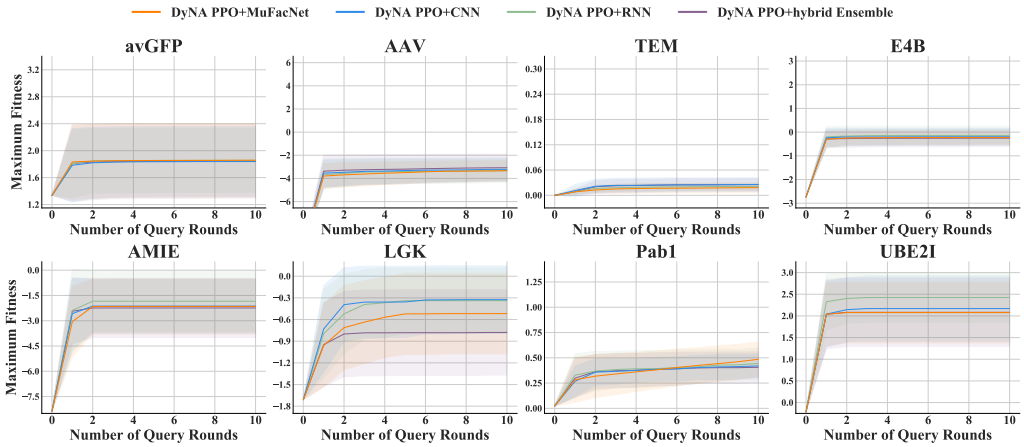


Figure 9. Learning curves of DyNA PPO with MuFacNet, CNN, RNN, and its default hybrid ensemble.

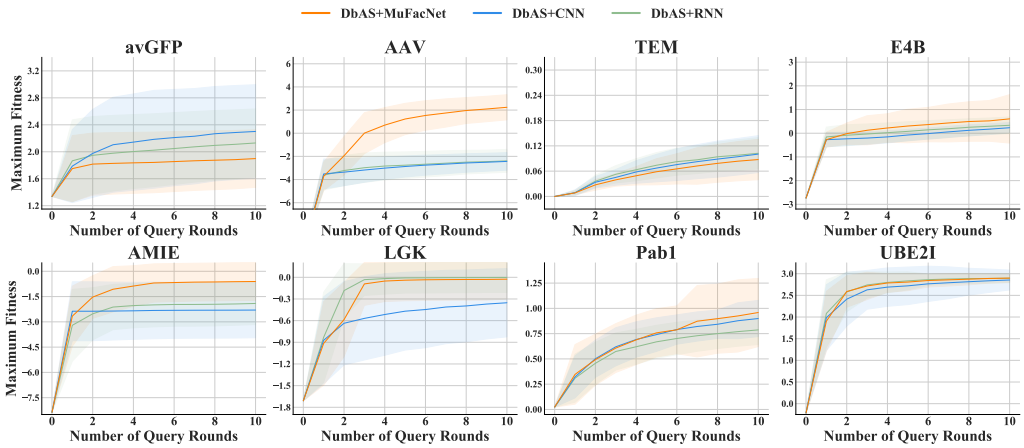


Figure 10. Learning curves of DbAS with MuFacNet, CNN, and RNN.

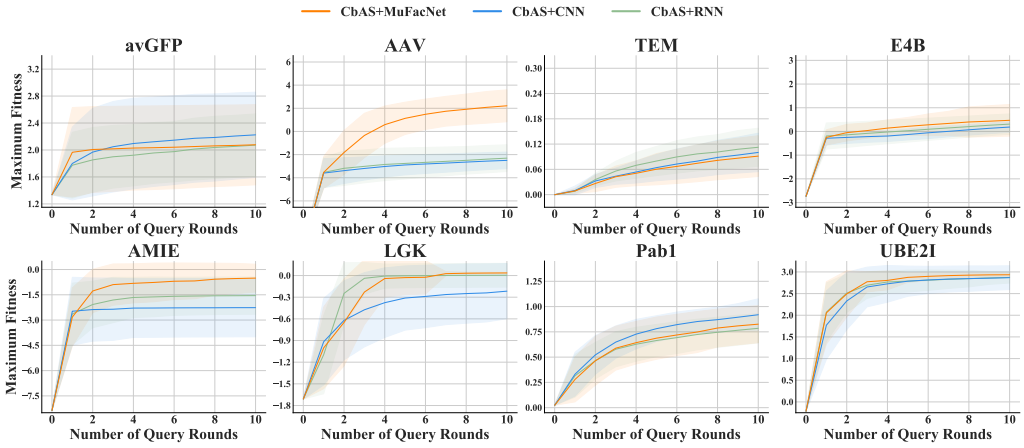


Figure 11. Learning curves of CbAS with MuFacNet, CNN, and RNN.

D. Learning Curves of the Experiments in Table 2

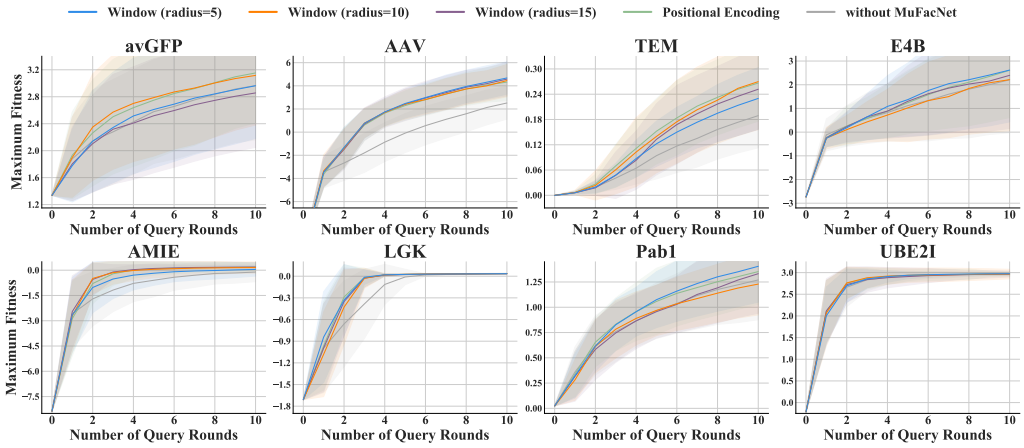


Figure 12. Learning curves of ablation studies on the representation of the context information of each single amino-acid mutation site.

E. Learning Curves of the Experiments in Table 3

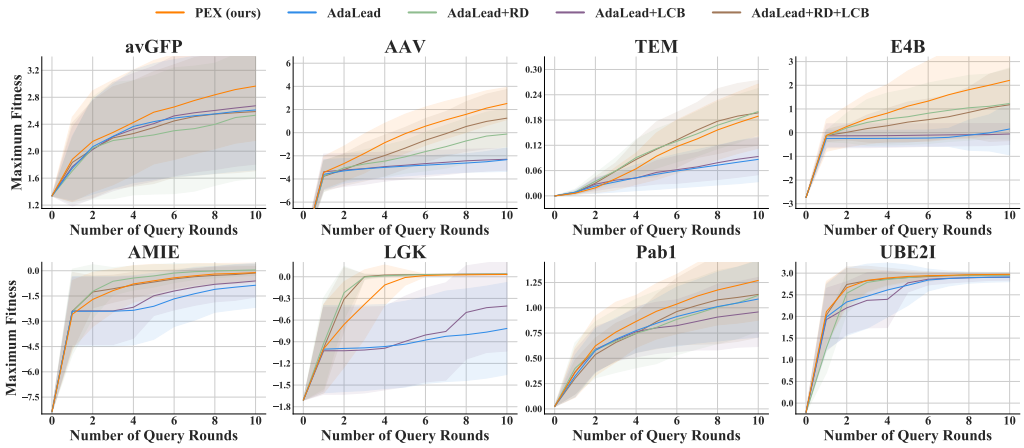


Figure 13. Learning curves of AdaLead with two simple methods to restrict the exploration scope.

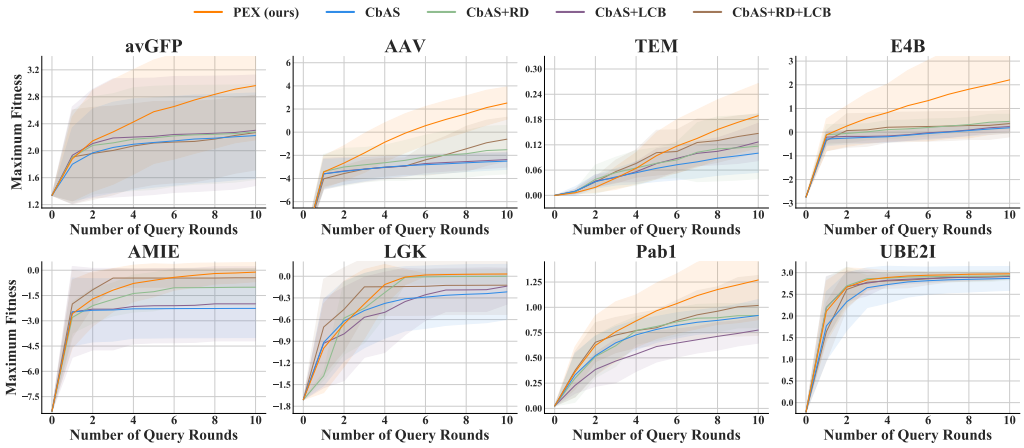


Figure 14. Learning curves of CbAS with two simple methods to restrict the exploration scope.