Synthetic Evolution of Enzyme Specificity with a Self-Driving Laboratory

Coban Brooks*
Department of Biomedical Engineering, Duke University

Jacob Rapp
Department of Biochemistry, University of Wisconsin-Madison

Pascal Notin
Harvard Medical School

Philip Romero*
Department of Biomedical Engineering, Duke University

Abstract

Engineering enzymes with tailored substrate specificities holds immense promise for biotechnology, but current approaches are slow, labor-intensive, and prone to failure. We present Robot Orchestrated Synthetic Evolution (ROSE), a closed-loop platform that integrates a self-driving laboratory with a transformer-based surrogate model and Bayesian optimization to autonomously explore protein fitness landscapes. ROSE expands beyond traditional directed evolution by exploring non-linear evolutionary trajectories, enabling escape from local optima. We demonstrate this system by engineering β-glucosidase enzymes with altered substrate specificity, achieving up to a 13-fold improvement in specificity on non-native substrates. Beyond this case study, we show that ROSE reliably identifies top-performing variants across diverse protein datasets while experimentally testing only a small fraction of available sequences. Together, ROSE establishes a modular and general-purpose framework for fully autonomous protein engineering.

1 Introduction

For billions of years, enzymes have been engineered by evolution, the universe's most powerful optimization algorithm. They have naturally evolved to be extremely proficient and specialized catalysts, speeding up chemical reactions many orders of magnitude faster than the rate at which they would naturally occur [8], while often exhibiting a high degree of substrate specificity [1]. These traits make enzymes an attractive platform for further engineering, potentially enabling the design of catalysts for processes that nature never encountered and expanding the toolkit of biotechnology far beyond naturally occurring enzymes [13].

Evolution operates on timescales far too slow for human engineering needs. Directed evolution—which exploits the basic mechanisms of evolution in a controlled laboratory setting—has emerged as the primary experimental approach for enzyme optimization; however, these methods are prone to becoming trapped at local fitness optima which they algorithmically cannot escape [2]. More recently, deep learning approaches have become popular tools for protein engineering because of their ability to statistically model unseen regions of protein space [5]. However, the massive and high dimensional nature of this space necessarily forces models to extrapolate far beyond the data

*Correspondence: coban.brooks@duke.edu, philip.romero@duke.edu

Machine Learning for Structural Biology Workshop

they were trained on, a regime where even the most modern deep learning architectures fail [6]. Moreover, regardless of method, all protein engineering requires experimental data, which often requires thousands of pipetting operations and months of time to collect. Current protein engineering methods remain slow, labor-intensive, and often ineffective [26, 4, 11].

Here, we present Robot Orchestrated Synthetic Evolution (ROSE), an autonomous platform for navigating the fitness landscape with Bayesian optimization. ROSE repeatedly mutates and selects variants but is free to promote any previously assayed high performer as a parent, enabling branching trajectories through sequence space that help escape local optima. A transformer surrogate model, retrained after every experimental round, supplies fitness predictions and uncertainties that drive UCB-based acquisition of promising offspring. The algorithm runs end-to-end in a custom autonomous laboratory that assembles genes, expresses proteins, and measures activity without manual intervention. We showcase the system by reprogramming β-glucosidase specificity and highlight its extensibility to other protein engineering campaigns.

2 Methods

We describe methods in the supplementary methods.

3 Results

3.1 Self-driving laboratory

Experiments are conducted by a fully autonomous self-driving laboratory. This experimental environment constitutes half of ROSE, which can be conceptually described as a two-part abstract evolutionary process. The computational agent operates an internal evolutionary algorithm, where it samples variants from a trained deep learning model and selects them according to a Bayesian optimization procedure. The self-driving laboratory acts as an external ground truth generator, which refines the predictive capacity of the computational algorithm as more data is acquired (Figure 1a).

The physical experimental system consists of integrated equipment that can conduct the entire protein engineering workflow (Figure 1b), which is coordinated by a custom software package that generates pipetting worklists, tracks reagent consumption, and directs model training and sampling. Additionally, it conducts data transfer steps between the local machine and GPU server for full agent-environment integration. Robust error handling allows the system to continue to operate smoothly even when sequences are misassembled or erroneously characterized (Figure 1c).

To assess the capabilities of this platform, we selected the enzyme β-glucosidase as our target for protein engineering. Natively, it performs the hydrolysis of terminal non-reducing β-D-glucosyl residues from oligosaccharide chains, like the β-1,4-linked polysaccharide cellulose. Given the abundance of oligosaccharide biomass on Earth and the reliance of cellular function on glucose, these enzymes evolved to be very fast and substrate selective (Figure 1e). Therefore, we pose the engineering of non-native substrate specificity in β-glucosidase as a challenging and relevant protein engineering goal. Specifically, we decided to focus on two non-native substrates: β-D-xylopyranoside and β-D-mannopyranoside, which together present substantial structural and stereochemical engineering challenges (Figure S4).

To generate enzyme variants on-demand, we adopted a chimeric assembly procedure which can be used to generate any of the $6^8 \approx 1.7$ million possible variants in the combinatorial library (Figure 1f and S2). After characterization, data is sent directly to the deep learning model hosted on our GPU server for model training. The end-to-end time, between when the lab receives a sequence query from the GPU server to the production of variant assay data, is approximately 11 hours (Figure 1d). Our automated platform can produce enzyme variant data on-demand rapidly and reproducibly.

3.2 Deep learning integration

The computational evolutionary algorithm is built around ProteinNPT [14], a non-parametric transformer variant [10] with state-of-the-art performance for supervised fitness prediction [16]. The architecture is trained with a hybrid objective combining property prediction and masked amino acid token reconstruction. This dual objective enables the model to learn a joint distribution over

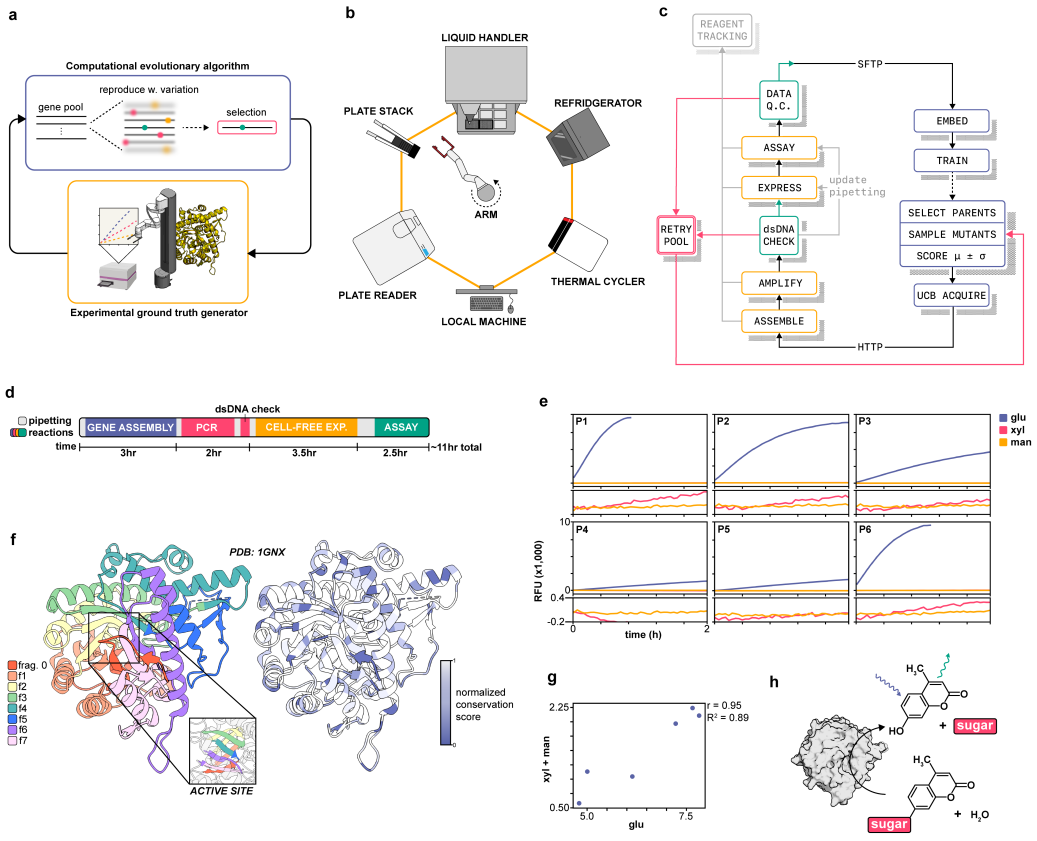

Figure 1: Figure 1. (a) Overview of ROSE. (b) Top-down illustration of the self-driving laboratory. (c) Process diagram of experimental (left) and computational (right) procedures. (d) Timeline of a single experimental acquisition round. (e) Time-course fluorogenic assay data from the six naturally derived β-glucosidases from which the fragment library is constructed. (f) Fragments (indexed 0 through 7) (left) and per-position residue conservation scores (right) overlaid on the β-glucosidase structure. (g) Activity values for all six parents on the native glucose substrate versus both non-native xylose and mannose substrates, showing a natural bias towards promiscuity. (h) β-glucosidase fluorogenic hydrolysis of 4-methylumbelliferyl(4MU)-conjugated sugars.

sequence and property. It also further regularizes the training, making the architecture ideally suited for label-scarce prediction and optimization settings. The model is retrained from scratch at the end of every experimental acquisition cycle with all collected enzyme specificity data, meaning that the model's predictive capacity is iteratively improved as the dataset increases in size.

We repurpose ProteinNPT for variant generation. To begin, the training dataset is ranked according to the functional scores. The top quartile of sequences are selected to serve as "parents", and thousands of 1-fragment-variants are conditionally sampled from the model and scored. From here, sequences that maximize an Upper Confidence Bound (UCB) acquisition objective are sent to the lab for testing. These sequences represent what the model believes to be the most *potentially* high-performing variants, striking a balance between exploitation of the current best parent sequences and exploration to areas of the sequence space the model is most uncertain about (Figure 2a) [19].

3.3 *In-silico* validation

Before leveraging this approach for our self-driving experiments with β-glucosidase, we validated the impressive ability of ProteinNPT for protein engineering by applying our search algorithm to various *in-silico* experiments (Figure 2c and Supplementary Section 5.5). Across four diverse protein datasets, each differing in its size, epistatic topology, protein function, and distribution of fitness

scores [20, 17, 12, 22], it was notably able to identify nearly the top performing sequence while seeing a minute fraction (0.5 - 0.03%) of the corresponding data.

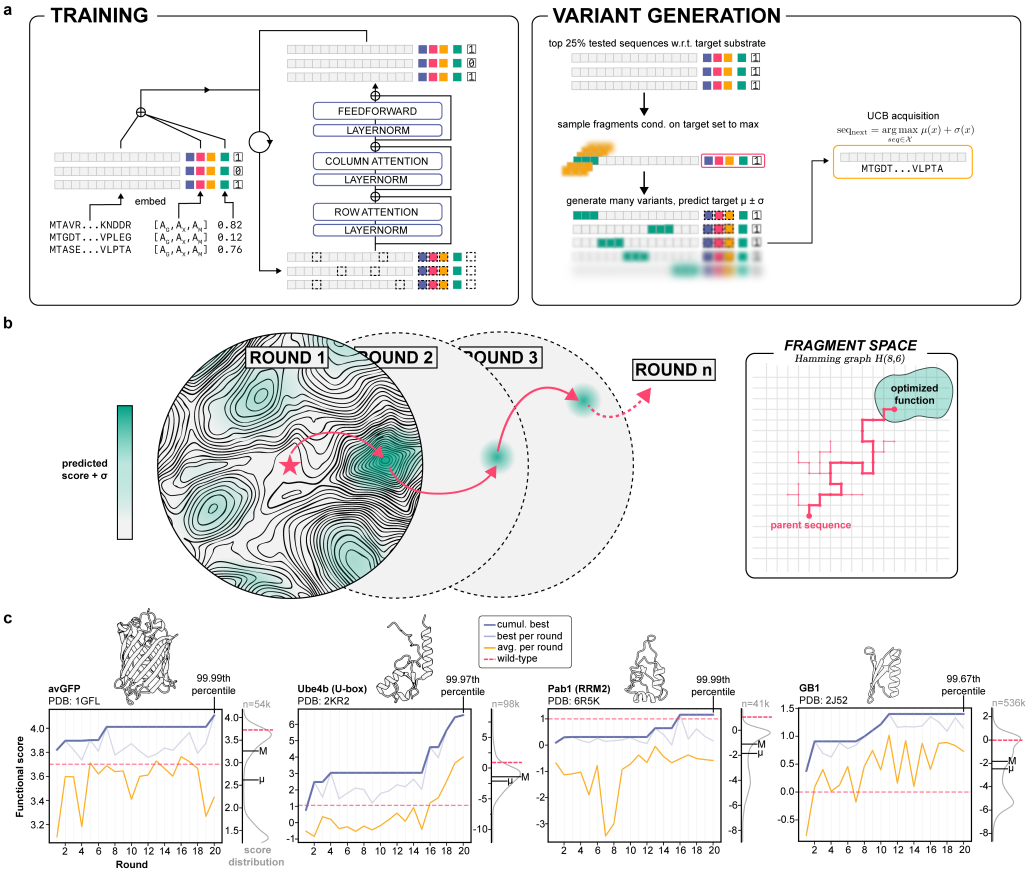

Figure 2: (a) Training and variant generation using ProteinNPT. The model trains by learning to recover masked tokens. It generates 1-fragment-variants by conditionally sampling from the trained model, after which sequences are acquired according to UCB. (b) Procedural overview of the synthetic evolution process. Many sequences within a mutational radius are scored, and the one with the highest mean plus uncertainty is selected for lab acquisition. If it is high performing, it serves as the parent for the next round, where the process repeats. In fragment space, this is equivalent to a directed random walk over a Hamming graph. (c) In-silico analysis of a search over four diverse protein DMS datasets, where the dataset itself is used as the ground truth oracle to simulate lab experiments. The search algorithm finds nearly the best sequence in each dataset within 20 rounds of 10 sequences. The score distributions of the datasets are shown in light gray, with corresponding mean and median values indicated by μ and M, respectively.

3.4 Autonomous enzyme engineering

We implemented our entire integrated system to conduct the specificity engineering campaign of β-glucosidase. We seeded the model with six naturally derived parent sequences, then initiated 3 agents to simultaneously maximize specificity on glucose, xylose, and mannose, respectively. Each agent was given the ability to query the lab with one sequence per round. Over 20 rounds of evolution, we increase the Targeted Activity Score (TAS)—an objective function that combines specificity with activity—on both non-native substrates. The final optimized enzymes show a 5x and 13x increase in TAS for xylose and mannose substrates, respectively (Figure 3a). While we struggle to optimize the enzyme on the native substrate glucose, the two other final variants only display measurable activity on the targets they were optimized for (Figure 3b).

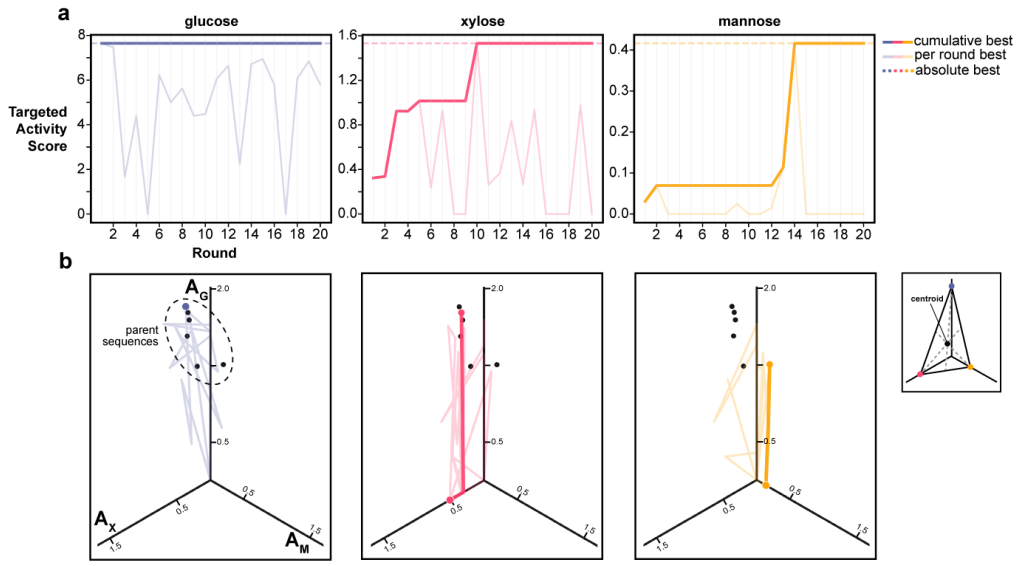

Figure 3: (a) Substrate specificity, measured by TAS, over 20 rounds of synthetic evolution for each substrate. Despite struggling to find a more specific enzyme on the native glucose substrate, we engineer enzymes 5 times as specific on xylose and 13 times as specific on mannose. (b) The centroid of 3-dimensional enzymatic activity, showing the "center of mass" of an enzyme's activity profile. The six parent sequences are clustered high on the glucose axis, reflecting their strong preference for glucose. Over the 20 rounds of evolution, the xylose and mannose trajectories both collapse onto the respective substrate axes, showing that all observable enzymatic activity for these variants are on the optimized substrate.

4 Discussion

Protein engineering remains hampered by slow and costly experimental cycles, the need to build and screen large libraries, low hit rates even with ML guidance, and model brittleness when extrapolating beyond observed data—most methods tackle only a subset of these challenges [2, 29, 25, 9, 6]. While ML- and active-learning–guided evolution improves selection, practical use of increasingly expressive models makes exhaustive scoring intractable, forcing either lightweight surrogate scorers or restricted search spaces. They are intrinsically unable to keep pace with advances in deep learning approaches [28, 27].

ROSE reframes variant design as Bayesian optimization over evolutionary-like trajectories to maximize information gain, constraining proposals to lie near the best empirically validated sequences. This reduces reliance on blind extrapolation, concentrates predictions where uncertainty is actionable, and enables the use of more computationally intensive models. We pair this strategy with a purpose-built autonomous laboratory that executes complete design–build–test–learn cycles without human intervention.

Although self-driving laboratory systems exist in protein engineering, they are often hosted on cloud laboratories [18] or in general-purpose biofoundries [21, 30] and often involve some human involvement during the experimental process. To our knowledge, this is the first purpose-built, in-house, fully closed-loop system for protein engineering that integrates automated experimentation with iterative ML.

We present ROSE as a general and modular framework for protein engineering: the laboratory can incorporate new instruments and assays, and the ML stack can leverage newer, stronger models to continually improve variant generation and selection.

5 Supplementary Methods

5.1 Self-driving laboratory

The physical experimental system consists of integrated equipment that can conduct the entire protein engineering workflow: a Dynamic Devices Lynx LM730i liquid handler, an Analytik Jena Biometra TRobot II thermal cycler, an Agilent BioTek Synergy H1 plate reader, and a custom built automated refrigerator arranged around a central Peak Robotics KX2-750 robotic arm capable of moving labware and reagents between peripheral instruments.

5.2 Automated protein production

We use a chimeric fragment assembly procedure to generate a large library of protein variants. Six homologous β-glucosidase protein sequences from the soil-dwelling bacterial genus *Streptomyces* serve as the initial "parents". We split each sequence into 8 fragments at minimally disruptive sites according to the SCHEMA recombination algorithm [3, 24]. Any 8 fragments (in order) can be assembled together to create a unique protein chimera, resulting in a total possible assembly space of 1,679,616 sequences S2.

Reagents The laboratory system was initialized with a set of reagents in the on-deck refrigerator: All 48 DNA fragments (stored on plasmids at 320 ng/μL), 2X Golden Gate Assembly Master Mix made from the NEBridge® Golden Gate Assembly Kit (E1601L), NEB Phusion Hot-Start Flex DNA Polymerase (M0535L), a 2μM solution of forward and reverse PCR primers, a 2X solution of EvaGreen dsDNA dye, Bioneer AccuRapid Cell Free Protein Expression Kit (Bioneer K-7260) Master Mix diluted in water to 0.66×, AccuRapid E. coli extract with added 40μM fluorescein, and three fluorogenic substrate master mixes (139μM 4-methylumbelliferyl-β-d-glucopyranoside, 2.2mM 4-methylumbelliferyl-β-d-xylopyranoside, and 2.2mM 4-methylumbelliferyl-β-d-mannopyranoside) in 1% vol/vol dimethylsulfoxide (DMSO), 11mM phosphate and 56mM NaCl, pH7.0 and water.

Assembly Experimentally, DNA fragments are pipetted together in a thin-walled PCR plate, and 10 μL of the resulting mixture is added to 10 μL of the 2X Golden Gate Master Mix. This is transferred to the thermal cycler, which performs 60 cycles of alternating 1 minute at 37°C and 1 minute at 16°C.

PCR 5 μL of the Golden Gate product is diluted in 45 μL of the 2μM primer solution, and 10 μL of the resultant mixture is mixed with 10 μL of 2X Hot-Start Phusion Polymerase Master Mix. This is transferred to the thermal cycler, which performs a PCR with the following cycle parameters: 30s initial melt, 30s melt at 98°C, 30s anneal at 59°C, 60s extend at 72°C, 5m final extend at 72°C.

dsDNA check 7.5 μL of the resultant PCR product is mixed with 30 μL water, and 5 μL of the diluted product is diluted again in a new well with 45 μL water. The DNA solution is mixed with 50 μL of a 2X EvaGreen stock, and then pipetted onto a 96-well plate. This is moved into the plate reader, the fluorescence at em:500/ex:530 is measured, and the values are compared to an EvaGreen + dsDNA standard curve for quantification of DNA concentration. If above 100 ng/μL, the assembly is deemed successful and the sequence passes onto the next stage. If not, the sequence is added to a retry pool which the computational agent can select from later.

Cell-free expression 20 μL of the diluted PCR product is added into a cell-free expression mix containing 26.66 μL of E. coli extract and 53.34 μL of Master Mix. A negative control is created at this step by substituting the PCR product for water. The entire expression mix is moved to the thermal cycler for incubation for 3h at 30°C.

Assay The resultant expression is pipetted into 9 wells (3 replicates x 3 substrates) in 10 μL volumes on a 96-well flat-bottom plate. 90 μL of substrate mix is pipetted on top of the expression and mixed. The plate is moved from the liquid handler deck to the plate reader, which measures an initial read at ex:487nm/em:528nm for the fluorescein internal standard, then ex:325nm/em:450nm every 5 minutes over a course of 2 hours at 30°C.

After the time-course assay, the plate is moved into the trash and fresh labware is replaced on the liquid handler deck for the next set of experiments.

5.3 Enzymatic characterization

In order to provide a biochemically robust measure of enzyme specificity without measuring activity over a range of substrate concentrations, we used the initial rate v_0 measured at a single low substrate concentration as a proxy for the specificity constant k_{cat}/K_m.

We start from the Michaelis–Menten equation:

$$v_0 = \frac{k_{\mathrm{cat}}[E][S]}{K_m + [S]}. \tag{1}$$

When the substrate concentration is much smaller than the Michaelis constant ($[S] \ll K_m$), the denominator simplifies to $K_m + [S] \approx K_m$, giving

$$v_0 \approx \frac{k_{\mathrm{cat}}[E][S]}{K_m} = \left(\frac{k_{\mathrm{cat}}}{K_m}\right)[E][S]. \tag{2}$$

Thus, in the low-substrate limit,

$$\frac{v_0}{[E][S]} \approx \frac{k_{\mathrm{cat}}}{K_m}. \tag{3}$$

When $[E]$ and $[S]$ are held constant, the initial rate v_0 is directly proportional to the specificity constant k_{cat}/K_m. Comparing raw v_0 values across substrates and across enzyme variants provides a valid measure of relative specificity.

To normalize replicates against the internal standard, each well's fluorescein measurement was divided by the column(replicate)-average fluorescein intensity to calculate a normalization factor. All subsequent time-course measurements for that well were divided by this factor to correct for pipetting variations and ensure that observed activity differences reflect biological rather than technical variation.

To test whether enzyme activity exceeded that of the negative control, we applied a one-sided Mann–Whitney U test. Given two groups of replicates, enzyme ($n_1 = 3$) and control ($n_2 = 3$), we assign ranks $1, \ldots, 6$ to the pooled data. Let R_1 denote the sum of the ranks for the enzyme group. The test statistic is

$$U = R_1 - \frac{n_1(n_1+1)}{2}, \tag{4}$$

which counts the number of pairwise enzyme–control comparisons for which the enzyme observation exceeds the control. Under the null hypothesis of equal distributions, U follows a known permutation distribution with mean

$$\mathbb{E}[U] = \frac{n_1 n_2}{2} = 4.5, \qquad \mathrm{Var}(U) = \frac{n_1 n_2 (n_1 + n_2 + 1)}{12} = 5.25. \tag{5}$$

For $n_1 = n_2 = 3$ we used the exact permutation distribution ($\binom{6}{3} = 20$ allocations) to compute one-sided p-values. Enzymes were classified as active on a substrate when $p \leq 0.05$. When classified as active, the median of the negative controls was subtracted from the median of the enzyme reads to yield the final raw activity value. When classified as inactive, final raw activity values were set to 0. The raw activity values undergo a $\log(1+x)$ transformation, which we define as the final enzymatic activity used in subsequent analyses.

5.4 Model training and variant generation

Our surrogate model for protein function prediction is ProteinNPT, a variant of a non-parametric transformer. After characterization, sequences are sent with Secure File Transfer Protocol (SFTP) to the GPU server, where they are added to the training database. A thorough explanation of model training can be found in the ProteinNPT manuscript [14]. However, briefly:

ProteinNPT Hyperparameters		
Component	Hyperparameter	Value
Architecture	Protein encoder	ProteinNPT (5 layers)
	Embedding source	ESM2-650M (frozen)
	Internal embedding dim	200
	Self-attention heads	4
	FFN hidden dim	400
Training	Total training steps (per BO round)	5,000
	Batch size (assay sequences / GPU)	425
	Optimizer	AdamW
	LR schedule	Warmup (100 steps) $\rightarrow$ cosine decay
	Max / min learning rate	3×10^{-4} / 1×10^{-5}
	Adam (β_1, β_2) / ϵ	(0.9, 0.999) / 10^{-8}
	Gradient accumulation	1
	Gradient clipping	1.0
	Weight decay	5×10^{-3}
	FP16 training	Enabled
	Indel mode	Enabled
Regularization	Attention dropout	0.1
	Activation dropout	0.1
	Token dropout	0.0
	Label smoothing	0.0
	Seed	2023
Targets & Loss	Primary objective	Specificity-weighted activity (agent-specific)
	Auxiliary labels	Tranception zero-shot score; binary activity flag
	Loss weighting	Annealed AA reconstruction + target MSE
Uncertainty & Acquisition	Uncertainty estimation	MC dropout (5 samples)
	Acquisition rule	UCB: $\mu(x) + \alpha\sigma(x)$, $\alpha = 1$
	Diversity enforcement	ESM2-650M embeddings + k-means clustering
Sampling Loop	Sampling mode	Conditional fragment resampling
	Parents per round	Top quartile by specificity objective
	Offspring per inner batch	30 conditional sequences
	Total samples per iteration	1,000
	BO rounds	20
	Acquisition batch size	1 sequence / agent / round

Table 1: Hyperparameters used in ROSE self-driving runs.

The corresponding zero-shot scores for each sequence are computed using the autoregressive protein language model Tranception [15], which are appended as auxiliary labels and passed as input but not masked during training (and thus do not contribute to the loss). Additionally, a binary activity label is appended, which is passed as a normal input label that is masked and whose loss is backpropagated during training. This data is embedded into ProteinNPT: the sequences with ESM2-650M, and all labels — the 3 specificity targets, the zero-shot scores, and the binary activity — in a learned linear transformation

The concatenated dataset is passed through many successive ProteinNPT layers, where 15% of input tokens are masked. The algorithm applies axial attention along the rows and columns to model the dependencies between all tokens in the input dataset. It does so by optimizing the loss:

$$\mathcal{L}^{\text{total}} = \alpha_t \cdot \mathcal{L}^{\text{AA reconstruction}} + (1 - \alpha_t) \cdot \mathcal{L}^{\text{target prediction}} \tag{6}$$

where $\mathcal{L}^{\text{AA reconstruction}}$ is a cross-entropy loss over amino acid tokens and $\mathcal{L}^{\text{target prediction}}$ is a mean-squared error (MSE) loss over activity target predictions. α_t is a weighting term that is progressively annealed to focus training on target prediction of activity labels. All relevant hyperparameters can be found in Table 1.

At variant generation, the behavior of the three agents (each having the objective to optimize specificity on their respective substrate) diverges. Therefore, each of the three models trains, samples, scores, and acquires sequences independently while sharing the same training data. Since three

models will learn approximately the same function, training could, in theory, be collapsed into one process. However, we elect to conduct each process fully independently to potentially allow for further inference improvements downstream, such as ensembling to get better mean and uncertainty predictions.

At variant generation, each agent begins by first defining an objective function, which we call the Targeted Activity Score (TAS), that combines elements of both activity and specificity. We hypothesized that if we programmed each agent to optimize pure enzyme activity on its corresponding substrate, they would be biased to produce promiscuous enzymes. Similarly, if we programmed each to optimize pure specificity, they would be biased to produce lowly active enzymes. We use TAS to generate enzymes that displayed both activity and specificity.

We represented enzyme activity as a vector in a three-dimensional "activity space," where each enzyme is defined by its activity on the three substrates:

$$\vec{A} = [A_{\text{glu}},\ A_{\text{xyl}},\ A_{\text{man}}].$$

Thus, the basis vectors, corresponding to each substrate, are:

$$\vec{glu} = [1, 0, 0], \quad \vec{xyl} = [0, 1, 0], \quad \vec{man} = [0, 0, 1].$$

We define the specificity as the cosine similarity between the activity vector and the substrate of interest. In the positive octant of 3-dimensional space, this yields a value between 0 and 1. For glucose:

$$\frac{\vec{A} \cdot \vec{glu}}{\|A\| \, \|\vec{glu}\|}$$

Multiplying this specificity by the activity on glucose itself, such that highly active but non-specific variants and lowly active but highly specific variants both have TAS values that approach zero, yields the final objective function:

$$T.A.S = \frac{A_{\text{glu}}^2}{\|\vec{A}\|}$$

After training, all training examples are ranked according to their TAS on the substrate of interest. The top quartile of these sequences serve as the "parents" for the subsequent round. The activity labels of each of these sequences are then set to those corresponding to the max observed TAS in the training dataset; this serves as the conditioning signal for sampling.

Once the conditioning pool is formed, we add the parent sequences (and their segmented representations) into a dataframe and repeatedly generate batched offspring until we reach the desired sample count. Each iteration draws a random permutation over the fragment indices for every parent, so that masking proceeds in a different order across sequences. For a given fragment position k, we replace the corresponding residues in every parent with the special token "`<mask>`" and concatenate the segments to obtain full-length masked sequences. These masked sequences, together with mutation strings relative to the wild type, are passed into the trained ProteinNPT model. The model restores the amino-acid embeddings, produces logits for every position, and we apply a temperature-scaled softmax (default temperature $T = 1$) to obtain log-probabilities over the 20 amino acids.

We then isolate the log-probabilities covering fragment k for each sequence and feed them to a chunk sampler that evaluates all candidate fragments in the precomputed library. The sampler aggregates token-level likelihoods over each fragment, renormalizes them, and returns a replacement fragment $\hat{s}_k$. The sampled fragment is spliced back into the sequence. Cycling through all K fragments across all B parents yields a batch of offspring. Outer iterations repeat until the targeted number of mutants (default = 1000) is produced. We then filter the generated set to remove duplicates and any sequence already present in the acquisition database.

Acquisition Predictive uncertainty is estimated via Monte Carlo dropout [7] with five stochastic forward passes through ProteinNPT with no masking, providing per-target means μ and standard deviations σ for each candidate sampled in the batch.

Acquisition scoring follows an Upper Confidence Bound strategy. For substrate i, the score is

$$\text{UCB}_i = \mu_{s_i} + \alpha \, \sigma_{s_i}$$

where μ_{s_i} and σ_{s_i} denote the posterior mean and predictive uncertainty of specificity toward substrate i. The hyperparameter α (default α=1.0) governs the trade-off between exploration and exploitation; low values of α bias the algorithm toward exploitation, relying heavily on current knowledge at the expense of discovering new optima. Conversely, excessively high values of α drive the algorithm toward exploration, prioritizing highly uncertain regions of the landscape that often yield non-functional variants. Setting $\alpha = 1.0$ places equal weight on the predicted mean and the predictive uncertainty, ensuring the algorithm pursues high-potential novel variants without being misled by excessive uncertainty.

The sequence to acquire for lab characterization is then:

$$\text{seq}_{\text{next}} = \arg\max_{\text{seq}\in\mathcal{X}} [\mu(\text{x}) + \sigma(\text{x})]$$

Each of the three agents identifies one sequence to query the lab for experimental characterization.

One entire cycle of Robot Orchestrated Synthetic Evolution, for three sequences, is outlined in pseudocode in Algorithm 1.

5.5 In-silico analysis

We explore the applicability of our method by conducting in-silico analyses on four diverse protein DMS datasets [22, 12, 17, 23](S1). For each simulated engineering campaign, we seed the algorithm with 9 random 1-mutant sequences plus the wild-type sequence. The model trains according to the same procedures outlined above. At variant generation, we sample a number of mutations from a Poisson distribution $n \sim \text{Poisson}(\lambda = 1)$ with counts 0 and 1 folded into 1 ($n \leftarrow \max(n, 1)$). We enumerate all n-mutant sequences from the top quartile "parents" in the dataset, score them with the trained model, and select 10 new sequences according to UCB acquisition. The dataset then serves as the ground truth oracle to simulate experiments. This continues for 20 total rounds of 10 sequences in each round. For each protein, the algorithm identifies nearly the top performing sequence in the corresponding dataset, reliably finding sequences above the 99.9th percentile of functional scores while only seeing 0.5-0.04% of the available data. We also demonstrate our algorithm's ability to navigate significantly epistatic landscapes; for example, the top performing sequences in the Pab1 dataset display considerable reciprocal sign epistasis where intermediate mutations between WT and the high-fitness sequence are deleterious to protein function.

This simulated task is likely easier than a true protein engineering campaign as the search space (the DMS dataset) is much smaller than the true sequence space for a given protein. To create a more accurate simulation space, we trained an ensemble of 100 multilayer perceptrons (MLPs) on each protein dataset, where the mean predictions of the ensemble serve as the "ground truth" fitness values for the queried sequences. Here, our algorithm is effectively learning the fine-grained surrogate landscape of our trained models. We conducted similar experiments in this search space (here sampling a number n mutations from the same Poisson distribution and randomly generating 1,000 n-mutants) and observed our algorithm was able to identify similarly well-performing protein sequences in the same number of rounds. Notably, we fail to identify >WT sequences in the Pab1 task, likely because the ensemble did not see many high performing sequences during training and never learned how to score them.

In both experimental tasks, tracking the best and average scores per round shows that the search algorithm identifies increasingly functional regions of sequence space to explore; while some of the trajectories venture into regions of low function, they tend towards increased fitness over time.

Algorithm 1 ROSE Pseudocode

Model generates candidate sequence
Lab robot receives three sequence queries through HTTP
Step 1: Gene Assembly
Identify gene fragments, mix
Perform Golden Gate assembly
Perform PCR
repeat
  num_success $\leftarrow 0$
  for each sequence in sequences **do**
    if dsDNA successfully obtained **then**
      Step 2: Expression
      Perform cell-free expression
      Step 3: Assay
      Pipette expression onto assay plate
      Add substrates, mix
      Record activity measurements, process, append to phenotype file
    else
      Report assembly failure
      Add sequence to retry pool, model can sample from later
    end if
  end for
until num_success > 0
Send phenotype file to GPU through SFTP
Step 4: Training
Add sequences and labels to training database
Embed
Train ProteinNPT
Step 5: Variant Generation
for agent in agents **do**
  Calculate TAS for all sequences in dataset w.r.t. substrate of interest
  Select top quartile as parent sequences
  for each parent sequence **do**
    Set labels to activities corresponding to max observed TAS w.r.t. substrate of interest
  end for
  Initialize dataframe with parent sequences and their segmented representations
  while total generated sequences < 1000 **do**
    for each parent in dataframe **do**
      Draw random permutation of fragment indices $1..K$
      for each fragment index k in permutation **do**
        Mask fragment k with `<mask>`
        Concatenate segments to form masked full-length sequence
        Compute logits with ProteinNPT model
        Apply temperature-scaled softmax to obtain log-probabilities over 20 amino acids
        Extract log-probabilities for fragment k
        Use chunk sampler to select replacement fragment $\hat{s}_k$
        Splice $\hat{s}_k$ back into the sequence
      end for
      Append generated sequence to offspring batch
    end for
  end while
  Step 6: Acquisition
  For each candidate sequence, estimate predictive uncertainty via Monte Carlo dropout
  Compute UCB score: $\mathrm{UCB}_i = \mu_{s_i} + \alpha\, \sigma_{s_i}$
  Select sequence with maximum UCB for lab characterization
  Append sequence to sequence query file
end for
Send sequence query to lab through HTTP

6 Supplementary Figures

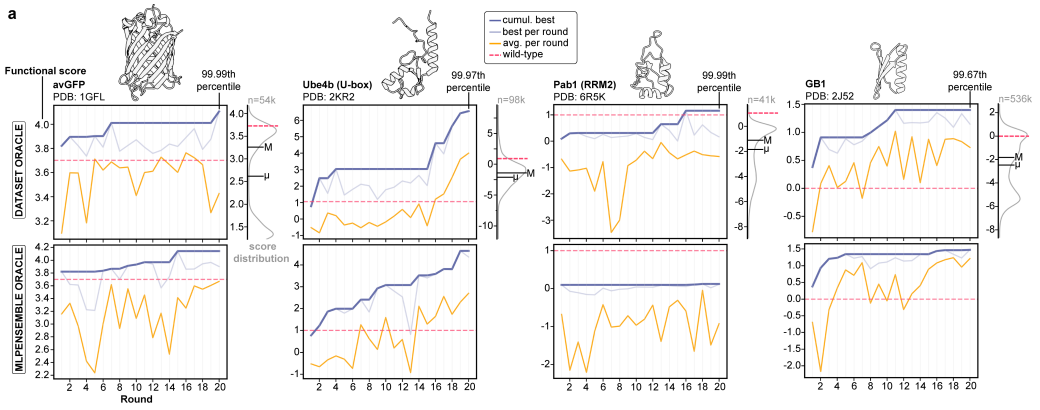

Figure S1: In-silico experiments using the DMS dataset (top) and the ensemble of trained MLPs (bottom) as the ground-truth scoring oracle. The score distributions of the datasets are shown in light gray, with corresponding mean and median values indicated by μ and M, respectively.

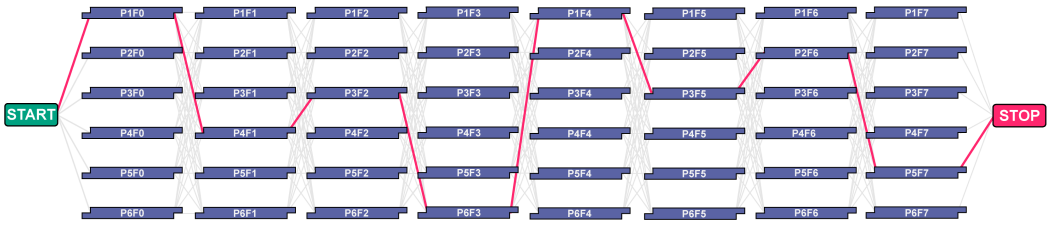

Figure S2: All valid enzyme chimeras can be illustrated as a path through the fragment library. There are approximately 1.7 million possible paths from START to STOP.

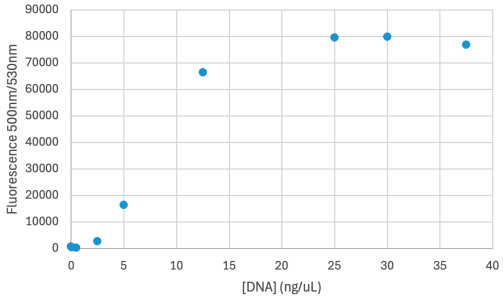

Figure S3: Standard curve of DNA concentration versus EvaGreen fluorescence at ex:500/em:530.

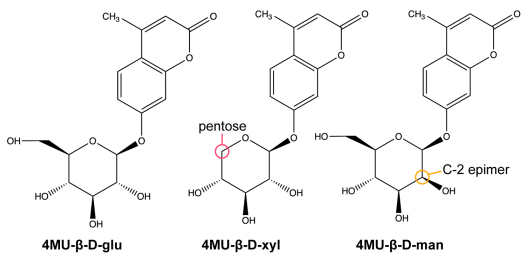

Figure S4: Comparison of the three substrates of interest. 4MU-β-D-xylopyranoside differs in carbon number (pentose vs. hexose) from the native 4MU-β-D-glucopyranoside, while 4MU-β-D-mannopyranoside is a C-2 stereoisomer.

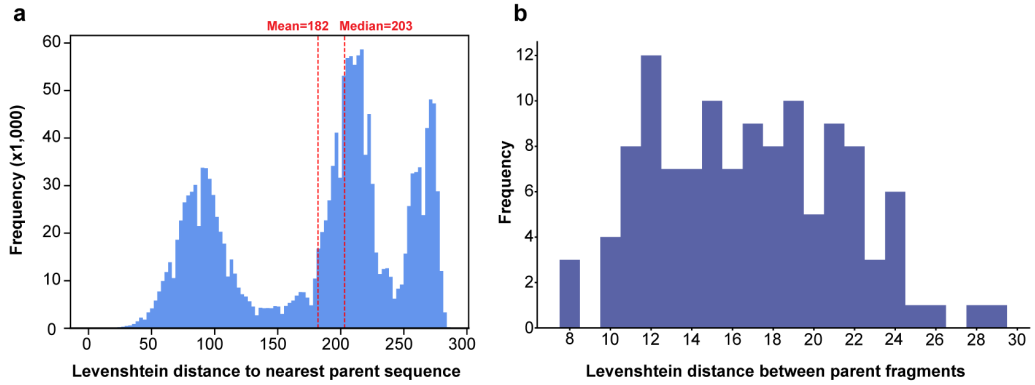

Figure S5: (a) Levenshtein (edit) distance between all 1.7 million sequences in the library and the nearest parent sequence. Chimeric assembly results in a diverse library, where sequences are on average 182 amino acid edits away from the nearest parent. (b) Levenshtein distances between all parent sequences of a given fragment. Fragments themselves are diverse, and 1 fragment mutations constitute a large mutational jump in amino acid sequence space.

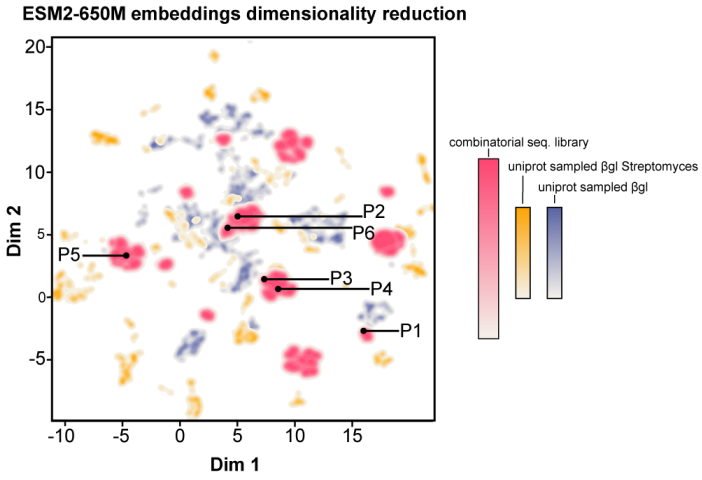

Figure S6: Embedding analysis of all sequences in the combinatorial sequence library (red) with 10,000 *Streptomyces* β-glucosidase sequences sampled from UniProt (yellow) and 10,000 β-glucosidase sequences sampled from Uniprot (purple). Principal Component Analysis (PCA) was performed on the embeddings to 50 dimensions, then Uniform Manifold Approximation and Projection (UMAP) was performed to reduce the dimensions to 2. The combinatorial sequence library is diverse and dissimilar to naturally occurring β-glucosidases. Parent sequences are indicated with P#.

References

[1] Elizabeth L. Bell, William Finnigan, Scott P. France, Anthony P. Green, Martin A. Hayes, Lorna J. Hepworth, Sarah L. Lovelock, Haruka Niikura, Sílvia Osuna, Elvira Romero, Katherine S. Ryan, Nicholas J. Turner, and Sabine L. Flitsch. Biocatalysis. *Nature Reviews Methods Primers*, 1:41, 2021.

[2] R. E. Cobb, R. Chao, and H. Zhao. Directed evolution: Past, present and future. *AIChE J*, 59(5):1432–1440, 2013.

[3] J. B. Endelman, J. J. Silberg, Z.-G. Wang, and F. H. Arnold. Site-directed protein recombination as a shortest-path problem. *Protein Eng Des Sel*, 17(7):589–594, 2004.

[4] M. Ertelt, R. Moretti, J. Meiler, and C. T. Schoeder. Self-supervised machine learning methods for protein design improve sampling but not the identification of high-fitness variants. *Science Advances*, 11(7):eadr7338, 2025.

[5] Roger J. Fox and Guus W. Huisman. Enzyme optimization: moving from blind evolution to statistical exploration of sequence–function space. *Trends in Biotechnology*, 26(3):132–138, 2008.

[6] C. R. Freschlin, S. A. Fahlberg, P. Heinzelman, and P. A. Romero. Neural network extrapolation to distant regions of the protein fitness landscape. *Nat Commun*, 15(1):6405, 2024.

[7] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. *arXiv preprint arXiv:1506.02142*, 2015.

[8] C. M. Horvat and R. V. Wolfenden. A persistent pesticide residue and the unusual catalytic proficiency of a dehalogenating enzyme. *Proceedings of the National Academy of Sciences*, 102(18):6141–6146, 2005.

[9] M. Z. Jawaid, R. W. Yeo, A. Gautam, T. B. Gainous, D. O. Hart, and T. P. Daley. Improving few-shot learning-based protein engineering with evolutionary sampling. *arXiv preprint arXiv:2305.15441*, 2023.

[10] J. Kossen, N. Band, C. Lyle, A. N. Gomez, T. Rainforth, and Y. Gal. Self-attention between datapoints: Going beyond individual input-output pairs in deep learning. *arXiv preprint arXiv:2106.02584*, 2021.

[11] P. Kouba, P. Kohout, F. Haddadi, A. Bushuiev, R. Samusevich, J. Sedlar, J. Damborsky, T. Pluskal, J. Sivic, and S. Mazurenko. Machine learning-guided protein engineering. *ACS Catal.*, 13(21):13863–13895, 2023.

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

[13] Obinna Giles Ndochinwa, Qing-Yan Wang, Oyetugo Chioma Amadi, Tochukwu Nwamaka Nwagu, Chukwudi Innocent Nnamchi, Emmanuel Sunday Okeke, and Anene Nwabu Moneke. Current status and emerging frontiers in enzyme engineering: An industrial perspective. *Heliyon*, 10(11):e32673, 2024.

[14] P. Notin, R. Weitzman, D. S. Marks, and Y. Gal. Proteinnpt: Improving protein property prediction and design with non-parametric transformers. *bioRxiv*, page 2023.12.06.570473, 2023.

[15] Pascal Notin, Mafalda Dias, Jonathan Frazer, Javier Marchena-Hurtado, Aidan Gomez, Debora S Marks, and Yarin Gal. Tranception: protein fitness prediction with autoregressive transformers and inference-time retrieval. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 16990–17017. PMLR, 2022.

[16] Pascal Notin, Aaron W. Kollasch, Daniel Ritter, Lood van Niekerk, Steffanie Paul, Hansen Spinner, Nathan Rollins, Ada Shaw, Ruben Weitzman, Jonathan Frazer, Mafalda Dias, Dinko Franceschi, Rose Orenbuch, Yarin Gal, and Debora S. Marks. Proteingym: Large-scale benchmarks for protein fitness prediction and design. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

[17] C. A. Olson, N. C. Wu, and R. Sun. A comprehensive biophysical description of pairwise epistasis throughout an entire protein domain. *Current Biology*, 24(22):2643–2651, 2014.

[18] J. T. Rapp, B. J. Bremer, and P. A. Romero. Self-driving laboratories to autonomously navigate the protein fitness landscape. *Nat Chem Eng*, 1(1):97–107, 2024.

[19] P. A. Romero, A. Krause, and F. H. Arnold. Navigating the protein fitness landscape with gaussian processes. *Proceedings of the National Academy of Sciences*, 109(3):E174–E183, 2012.

[20] K. S. Sarkisyan, D. A. Bolotin, M. V. Meer, D. R. Usmanova, A. S. Mishin, G. V. Sharonov, D. N. Ivankov, N. G. Bozhanova, M. S. Baranov, O. Soylemez, N. S. Bogatyreva, P. K. Vlasov, E. S. Egorov, M. D. Logacheva, A. S. Kondrashov, D. M. Chudakov, E. V. Putintseva, I. Z. Mamedov, D. S. Tawfik, K. A. Lukyanov, and F. A. Kondrashov. Local fitness landscape of the green fluorescent protein. *Nature*, 533(7603):397–401, 2016.

[21] N. Singh, S. Lane, T. Yu, J. Lu, A. Ramos, H. Cui, and H. Zhao. A generalized platform for artificial intelligence-powered autonomous enzyme engineering. *Nat Commun*, 16(1):5648, 2025.

[22] L. M. Starita, J. N. Pruneda, R. S. Lo, D. M. Fowler, H. J. Kim, J. B. Hiatt, J. Shendure, P. S. Brzovic, S. Fields, and R. E. Klevit. 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.

[23] P. Tian, J. M. Louis, J. L. Baber, A. Aniana, and R. B. Best. Co-evolutionary fitness landscapes for sequence design. *Angewandte Chemie International Edition*, 57(20):5674–5678, 2018.

[24] C. A. Voigt, C. Martinez, Z.-G. Wang, S. L. Mayo, and F. H. Arnold. Protein building blocks preserved by recombination. *Nat Struct Mol Biol*, 9(7):553–558, 2002.

[25] Z. Wu, S. B. J. Kan, R. D. Lewis, B. J. Wittmann, and F. H. Arnold. Machine learning-assisted directed protein evolution with combinatorial libraries. *Proceedings of the National Academy of Sciences*, 116(18):8852–8858, 2019.

[26] H. Xiao, Z. Bao, and H. Zhao. High throughput screening and selection methods for directed enzyme evolution. *Ind Eng Chem Res*, 54(16):4011–4020, 2015.

[27] Jason Yang, Ravi G. Lal, James C. Bowden, Raul Astudillo, Mikhail A. Hameedi, Sukhvinder Kaur, Matthew Hill, Yisong Yue, and Frances H. Arnold. Active learning-assisted directed evolution. *Nature Communications*, 16(1):714, 2025.

[28] K. K. Yang, Z. Wu, and F. H. Arnold. Machine learning-guided directed evolution for protein engineering. *arXiv preprint arXiv:1811.10775*, 2019.

[29] T. Yu, A. G. Boob, N. Singh, Y. Su, and H. Zhao. In vitro continuous protein evolution empowered by machine learning and automation. *Cell Systems*, 14(8):633–644, 2023.

[30] Q. Zhang, W. Chen, M. Qin, Y. Wang, Z. Pu, K. Ding, Y. Liu, Q. Zhang, D. Li, X. Li, Y. Zhao, J. Yao, L. Huang, J. Wu, L. Yang, H. Chen, and H. Yu. Integrating protein language models and automatic biofoundry for enhanced protein evolution. *Nat Commun*, 16(1):1553, 2025.