Spectral Feedback for Test-Time Alignment of Protein Diffusion Models

Shai Dickman **Mert Cemri** **Landon Butler** **Kannan Ramchandran**
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley

Abstract

Discrete diffusion models have become leading architectures for inverse protein folding, which involves complex non-causal interactions. Test-time alignment is a useful technique in protein design to guide these models at inference to generate proteins with desirable attributes. Alignment methods for discrete diffusion models have primarily focused on steering the diffusion reverse process, either by influencing token logits or by selecting favorable sequences at intermediate steps to eventually yield high-reward samples. These approaches largely treat inference as a unidirectional process, lacking effective mechanisms for revisiting undesirable token selections. We introduce **Spectral Feedback**, an algorithm that selects edit-positions in a feedback loop, allowing the model to iteratively correct its own generations. This approach leverages the masking structure of discrete diffusion models by re-masking and re-sampling tokens, analogous to image editing methods that reintroduce noisy latents and re-run the reverse diffusion process. While prior alignment methods focus on *what token labels* to assign to maximize a target reward, we instead treat *which tokens* to revisit as the central alignment problem. Selecting edit-positions is challenging because edit effects are interdependent: the impact of modifying one token depends on which others are edited simultaneously. We define an edit-set as a set of positions to edit by re-masking and re-sampling the corresponding tokens in a sequence. Motivated by prior work on sparse interactions in biological systems, we find empirically that edit-set value functions for protein inverse folding admit sparse Fourier representations. This structure enables Spectral Feedback to efficiently learn and optimize the value functions for edit-position selection. Spectral Feedback is model-agnostic and can be applied to pre-trained, test-time aligned, and fine-tuned diffusion models without modifying the underlying generative processes. Applied to inverse folding with a protein stability reward oracle, Spectral Feedback increases the proportion of stable proteins by 32.3 percentage points over a pre-trained model, 24.8 over Best-of-10, and 5.8 over a state-of-the-art RL fine-tuned diffusion model.

1 Introduction

Inverse protein folding is the task of designing amino acid sequences that fold into a target protein backbone. Discrete diffusion models have emerged as a leading architecture for this task [Yi et al., 2023, Gruver et al., 2023, Cemri et al., 2024, Wang et al., 2025]. While these models can successfully generate sequences that fold into target backbones, scientists may also target additional attributes such as stability [Wang et al., 2025] or β-sheets [Gruver et al., 2023]. Many sequences can fold into similar structures, yet only a few may satisfy target criteria [Xiong et al., 2026]. Given an arbitrary protein model, our goal is to generate proteins with desirable attributes by efficiently using a reward model that quantifies those attributes. Test-time alignment is a category of methods that use additional compute during inference to generate desirable samples. Existing test-time alignment of discrete

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

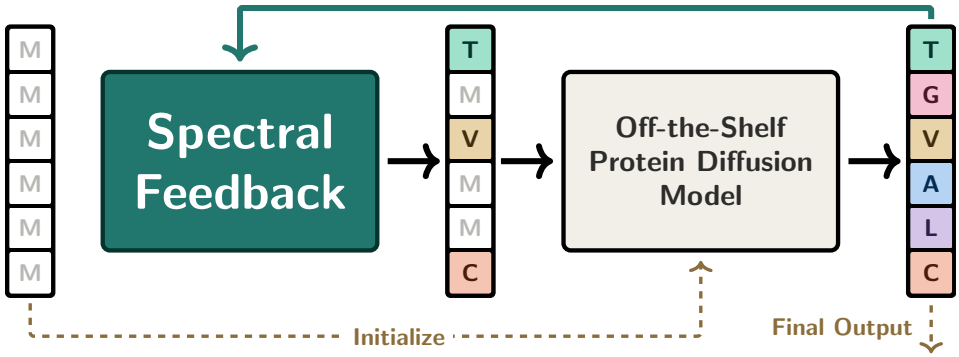


Figure 1: Spectral Feedback loop using targeted edits to iteratively improve attributes of a protein sequence with discrete diffusion.

diffusion models can be organized into three groups: token logit alignment, tree-search, and re-noising feedback. Token logit alignment skews probabilities of tokens during the reverse process such as in [Xiong et al., 2026] and [Gruver et al., 2023]. Tree-search techniques, like those discussed in [Huang et al., 2025, Darmawan et al., 2025, Uehara et al., 2025], explore different denoising trajectories in the reverse process and select the highest-reward sample. We focus on the third and less studied group of methods: re-noising feedback. This method, discussed in Wang et al. [2026], naturally exploits the masking structure of discrete diffusion models in the following sequence: generate a candidate sequence, re-mask a subset $S \subseteq [L]$ of positions to edit, and re-sample those positions. This type of method uniquely treats the model as a black box instead of modifying its denoising trajectory. Previous sequence editing methods have used independent random re-masking; however, the actual alignment comes from other techniques such as importance sampling or classifier-free guidance rather than solely re-masking [Lee et al., 2025, Reid et al., 2022, Wang et al., 2026, Gruver et al., 2023]. The addition and removal of noise also have parallels with continuous diffusion model image editing methods, which transform images into partially noisy latents and then re-sample a segment of the diffusion reverse process [Hertz et al., 2022, Mokady et al., 2022, Meng et al., 2022]. Similar to the discrete case, alignment in these methods comes from techniques other than noise generation.

We claim that re-masking without independence assumptions can be an effective alignment method on its own. This approach is useful because its modularity allows any other alignment method to be improved within a re-masking feedback loop. Formally, we focus on selecting an edit-set S that maximizes a value function f which relates edit-sets to alignment rewards. For an arbitrary value function $f : 2^{[L]} \to \mathbb{R}$, maximization over 2^L candidates is intractable, which supports the use of independence assumptions in past works. Our work identifies structure in edit-set value functions which enables approximations that avoid combinatorial challenges during learning and optimization.

The expansion of f in the Fourier basis assigns every subset $T \subseteq [L]$ a coefficient $F(T)$ that measures the strength of the interaction among positions in T [O'Donnell, 2014, Li et al., 2014, Butler et al., 2025]. A sparse Fourier expansion contains a small number of coefficients that capture most of the variation in f. Sparsity of this form is consistent with the epistatic structure of biological sequences [Poelwijk et al., 2019, Aghazadeh et al., 2021, 2020], in which the joint effect of mutating multiple amino acids is dominated by a small number of interacting groups. Because the interactions that determine the physical properties of a protein are sparse, we expect that the interactions between edits of these very same amino acids will also reflect a sparse structure. We verify this empirically for protein sequence edit-set value functions.

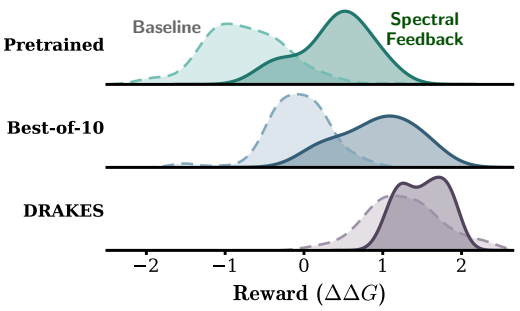


Figure 2: Spectral Feedback in different protein model inference settings, aligning for stability ($\Delta\Delta G$ - kcal/mol). These are reward evaluation distributions from experiments in Section 5.

Importantly, Fourier sparsity makes it tractable to approximate f from a limited number of queries. Instead of requiring 2^L reward evaluations, the number of samples needed for sparse Fourier recovery grows only with the sparsity level [Scheibler et al., 2015, Li et al., 2014], which we find to be small in practice for edit-set value functions [Kang et al., 2025, Butler et al., 2025]. We also find that most spectral energy is concentrated in lower-order terms, with some value function instances being nearly linear.

Motivated by our observations of sparsity, we introduce **Spectral Feedback**, a feedback-based test-time alignment method for discrete diffusion. Our method leverages sparse Fourier recovery to approximate the combinatorial edit-selection problem in a tractable form. Spectral Feedback treats the underlying diffusion model as a black box: it requires only the ability to sample from the model conditioned on a partially masked input, and never modifies its weights or its denoising trajectory. This makes it compatible with any pre-trained discrete diffusion model and composable with other alignment techniques. We apply Spectral Feedback on top of a pre-trained model, Best-of-N sampling, and an RL fine-tuned model (DRAKES [Wang et al., 2025]).

We evaluate Spectral Feedback on protein backbones from the Megascale dataset [Tsuboyama et al., 2023], aligning sequences to a stability oracle that predicts $\Delta\Delta G$, the difference in Gibbs free energy between a generated sequence and its wild-type variant. To detect reward over-optimization, we separately track self-consistency RMSD ($scRMSD$) between the target backbone and the ESMFold-predicted structure of each sample [Lin et al., 2023] as well as naturalness with ProtGPT2 log-likelihoods [Ferruz et al., 2022].

Contributions.

1. **Formulation.** We formalize a feedback loop that relates the selection of edit-positions at each iteration to an alignment reward oracle by defining an edit-set value function. The test-time alignment problem is then to maximize this intermediate value function rather than the reward oracle directly. This creates a combinatorial challenge that prior refinement methods largely avoid, since the value of editing one position can depend on which other positions are edited simultaneously.
2. **Sparsity analysis.** We characterize the Fourier spectra and the sparsity structure of edit-set value functions across protein targets. Our analysis suggests that some proteins are dominated by first-order effects, while others involve high-order interactions, giving an empirical explanation for performance differences between simpler position-wise edit methods and high-order sparse recovery methods.
3. **Method.** We introduce Spectral Feedback, which learns a sparse Fourier approximation of an edit-set value function, and then uses the approximation to efficiently identify promising edits at each iteration of a diffusion feedback loop. Spectral Feedback is model-agnostic, which allows it to improve the performance of any pre-trained, test-time aligned, or fine-tuned model.
4. **Empirical results.** On inverse folding of protein backbones in the Megascale dataset, Spectral Feedback improves alignment rewards for several different underlying protein models. Compared with several greedy or gradient-based baseline edit-set selection methods, Spectral Feedback offers a favorable reward–latency trade-off.

2 Preliminaries

Discrete Diffusion Models. We establish the notation here, while a complete treatment appears in Appendix B. Consider a vocabulary V, sequence length L, and sample space $\mathcal{X} := V^L$. We consider a masked discrete diffusion model whose forward process progressively masks tokens via a Continuous Time Markov Chain (CTMC) with rate matrices Q_t. Its probability mass vector evolves according to $dp_t/dt = Q_t p_t$, with $p_0 = p_{\text{data}}$, where $Q_t(y, x)$ denotes the transition rate from x to y [Liang et al., 2025]. The reverse process, initialized from a fully masked sequence x_T, progressively unmasks tokens by learning a score function $p_t(y)/p_t(x)$ from training data. A Hamming distance constraint ($Q_t(x, y) = 0$ for $d(x, y) > 1$) restricts each transition to a single-token flip. Practical samplers can update multiple tokens per time step by approximating their transitions as conditionally independent given the current sequence.

Feedback with Edit-Positions. Given a protein sequence x of length L, feedback selects a subset of positions to re-mask and then re-samples them from the diffusion model. Let $S \subseteq [L]$ denote a set of edit-positions. Applying S constructs a partially masked sequence $\tilde{x}$:

$$\tilde{x}_i = \begin{cases} MASK & \text{if } i \in S \\ x_i & \text{if } i \notin S \end{cases} \tag{1}$$

A new sequence $z \sim p_{\text{pre}}(\cdot|\tilde{x})$ is generated by running the reverse process initialized with $\tilde{x}$. The refinement scheme alternates between selecting S and re-sampling, seeking edit-positions that maximize a value function: $S^* = \underset{S \subseteq [L], |S| \leq k}{\arg\max} f(S)$. The cardinality bound k balances alignment performance with the computational cost of solving the maximization problem and re-sampling at the edit positions. Prior methods select edit-positions randomly or independently [Lee et al., 2025, Reid et al., 2022, Gruver et al., 2023]. Our method instead searches for optimal *sets* of positions by exploiting their joint reward structure, as described in Section 4.

We define the total number of steps N of a feedback alignment process as the total number of proteins generated during the process. Our algorithm generates an initial protein sequence before executing the feedback loop, so it has $N - 1$ feedback iterations. At each step, we initialize the reverse process with $\tilde{x}$. We then sample z by running the reverse process from the first time-step to completion. In future work, it could be natural and more efficient to use a later time-step that relates to the masking pattern of $\tilde{x}$ and the noise schedule of the diffusion model.

3 Related Work

Protein Sequence Design and Inverse Folding. Recent work has produced strong generative models for inverse folding. ProteinMPNN [Dauparas et al., 2022] introduced a graph neural network conditioned on backbone geometry that autoregressively generates sequences. Discrete diffusion variants extend the structured methods to non-autoregressive masked generation [Yi et al., 2023, Gruver et al., 2023, Cemri et al., 2024, Wang et al., 2025]. DRAKES [Wang et al., 2025] fine-tunes a discrete diffusion inverse folding model with a differentiable reward signal, improving alignment to a stability oracle at the cost of model-specific training.

Inference-Time Alignment for Discrete Generative Models. Inference-time alignment methods spend additional compute at sampling time to improve reward without retraining. Best-of-N draws N independent samples and returns the highest-reward one [Huang et al., 2025]; it is the dominant approach in protein design due to its simplicity [Yeh et al., 2023, Bennett et al., 2026, Darmawan et al., 2025], but its reward improvement grows slowly with N. Beam Search retains only the highest-reward partial trajectories at each reverse-process step [Uehara et al., 2025]. Token-level guidance methods modify the score function during the reverse process: ProteinGuide uses Bayesian conditioning with classifiers trained on partially masked sequences [Xiong et al., 2026], and NOS performs Langevin updates in the protein embedding space [Gruver et al., 2023].

Sequence and Image Editing Methods. Sequence editing techniques often progressively transform an existing sample toward a target objective. Lee et al. [2025] use Multiple-Try Metropolis with uniform random noise to refine intermediate reverse-process time-steps. Wang et al. [2026] design re-masking schedules that facilitate better sample quality and can benefit existing reward guidance methods, but do not themselves provide reward guidance. Gruver et al. [2023] apply embedding-space Langevin updates at positions sampled with probabilities proportional to reward-gradient magnitudes, and DiffusER uses random Levenshtein edits as the forward process [Reid et al., 2022]. In the language model setting, Self-Refine [Madaan et al., 2023] and multi-agent pipeline optimization [Xue et al., 2025] use LLM-generated feedback rather than reward oracles.

In parallel to sequence editing, several image editing methods re-noise and re-sample image latents using the continuous diffusion forward and reverse processes which involve independent Gaussian noise [Hertz et al., 2022, Mokady et al., 2022, Meng et al., 2022]. A common pattern across both discrete and continuous diffusion editing methods is that noise is generated independently. The discrete case offers a unique opportunity for targeted edit selection since the noise is made up of masks that directly correspond to individual tokens. Spectral Feedback uses this structure during test-time alignment by selecting tokens to re-mask and re-sample.

Sparse Boolean Function Recovery and Epistasis. The combinatorial structure of edit-set selection connects to a long line of work on sparse Boolean function recovery. Epistasis, in which genes or amino acids interact to determine biological traits [Cordell, 2002], is documented to exhibit sparse high-order structure [Poelwijk et al., 2019, Aghazadeh et al., 2021, 2020], and similar sparsity patterns have been observed in language and vision data [Butler et al., 2025]. LASSO recovers sparse linear models efficiently [Tibshirani, 1996] but requires explicit enumeration of interaction features, which scales exponentially with order. Recent algorithms [Scheibler et al., 2015, Li et al., 2014, Amrollahi et al., 2019, Kang et al., 2025, Butler et al., 2025] address this by recovering high-order Fourier coefficients with sub-exponential sample complexity. We use this line of work to approximate edit-set value functions and to study how the sparsity structure of these functions varies across proteins.

4 Spectral Feedback

We introduce Spectral Feedback, a feedback loop that iteratively selects and applies edits to a sequence with a discrete diffusion model. Our algorithm leverages sparse interactions in edit-set value functions to efficiently select promising edits. In this section, we study these sparse interactions and then describe the details of our algorithm.

4.1 Spectral Function Approximation

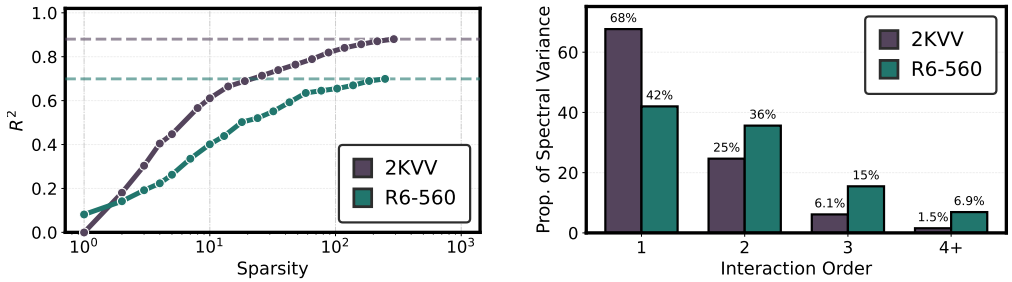


Figure 3: Spectral analysis of the value function f_{avg} with the pre-trained diffusion model. The plots show results for two generated protein sequences conditioned on the 2KVV and R6-560 (r6_560_TrROS_Hall) backbones from the Megascale dataset. Experimental details and additional plots are provided in Appendix D.2.

The goal of edit-set selection is, given a protein sequence x, target backbone y, and protein model $p(\cdot|S, x, y)$, to find a set S^* that maximizes a value function $f : 2^{[L]} \to \mathbb{R}$. In terms of edits, this value function should reward more promising edit selections (i.e., those with higher expected rewards after re-sampling). Given a reward oracle $r(\cdot)$, we use the following value functions depending on the sampling techniques used for the underlying protein model:

$$f_{avg}(S, x_t, y) = \frac{1}{n}\sum_{i=1}^{n} r(z_i), \quad f_{max}(S, x_t, y) = \max\{r(z_1), \ldots, r(z_n)\} \tag{2}$$

where $z_i \sim p(x_{t-1}|S, x_t, y$ and x_{t-1} is unmasked) for a discrete diffusion model p. Other value functions exist—such as trained predictors for partially masked sequences [Gruver et al., 2023, Xiong et al., 2026] or posterior mean approximations [Uehara et al., 2025]—though we selected this method for computational feasibility. Still, we find that this single-step approximation of the reverse process leads to strong alignment performance.

A value function f admits a Fourier transform $F : 2^{[L]} \to \mathbb{R}$ as follows [O'Donnell, 2014]:

$$\text{Transform: } F(T) = \frac{1}{2^L}\sum_{S\subseteq[L]} (-1)^{|S\cap T|} f(S), \quad \text{Inverse: } f(S) = \sum_{T\subseteq[L]} (-1)^{|T\cap S|} F(T). \tag{3}$$

While there are 2^L coefficients that fully describe f, if we can use an approximation $\hat{f} \approx f$ with significantly fewer coefficients, then learning $\hat{f}$ can become tractable. The sparse support also allows for optimizing via a small integer program. Parseval's theorem gives us

$$\frac{1}{2^L}\sum_{S\subseteq[L]} \left(f(S) - \hat{f}(S)\right)^2 = \sum_{T\subseteq[L]} \left(F(T) - \hat{F}(T)\right)^2. \tag{4}$$

Consequently, if f admits a *sparse* Fourier representation—i.e., most of its energy is concentrated in a small number of coefficients—then accurately recovering only those dominant coefficients suffices to approximate f well. This motivates our use of sparse recovery methods to estimate F from a limited number of evaluations of f.

$$R^2 = 1 - \frac{||\hat{f} - f||^2}{||f - \bar{f}||^2}, \quad \text{where } ||f||^2 = \sum_{S \subseteq [L]} f(S)^2, \bar{f} = \frac{1}{2^L} \sum_{S \subseteq [L]} f(S). \tag{5}$$

We use the sparse recovery algorithm SPEX [Kang et al., 2025] to accurately compute Fourier coefficients that describe a chosen edit-set value function during a single feedback iteration. Figure 3 shows results from this spectral analysis of f_{avg} for two distinct protein backbones. Most spectral variance is captured by coefficients with order at most three and value function approximations for both proteins reach an R^2 of at least 0.7. These results indicate that the Fourier coefficients representing f_{avg} are sparse and concentrated at low orders. Similar results are shown in Appendix D.2 for the whole test set with f_{avg} and f_{max}.

An interesting difference between the two held-out protein examples is that first-order coefficients dominate for *2KVV* while high-order interactions are more significant for *R6-560*. These differences suggest that interactions between edit-positions depend on the properties of the underlying proteins, which are strongly influenced by interactions among their respective amino acids. When mainly first-order coefficients describe the value function, like for *2KVV*, a first-order sparse recovery method will likely be sufficient to learn a useful $\hat{f}$. We further investigate this in Appendix G.

4.2 Spectral Feedback Algorithm

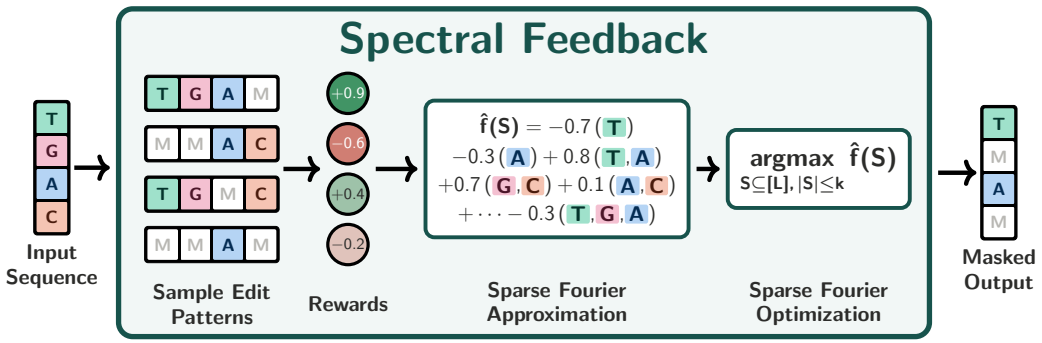


Figure 4: Spectral Feedback learns a sparse Fourier representation of the edit-set value function, then solves for the max-reward set via integer optimization over the sparse coefficients.

The Spectral Feedback algorithm alternates edit-position selection and model sampling, using any off-the-shelf masked diffusion model as shown in Figure 1. The algorithm is described below.

Step 1 - Select edit-positions (Algorithm 1). Given an amino acid sequence x and an edit budget k, the first step selects a set of edit-positions $S \subseteq [L]$ such that $|S| \leq k$. This step, depicted in Figure 4, is done in the following sub-steps: sample sets of edit-positions, evaluate corresponding rewards, fit a sparse Fourier approximation, and solve for the optimal positions. The sample sets are assigned rewards using Equation 2. A sparse Fourier approximation is then fit with the sampled set-reward pairs. This can be done with any sparse recovery method. Finally, edit-positions are selected by solving an optimization problem over the learned sparse Fourier approximation. The setup of this optimization problem is described in Appendix D.1.

Step 2 - Sample the Protein Model (Algorithm 2). After finding a set S of edit-positions, the corresponding amino acids at each position are re-masked as described in Equation 1. The partially masked sequence is then passed back through the discrete diffusion reverse process.

Sparse Recovery Method. While any sparse recovery method can be used in Spectral Feedback (e.g., Kang et al. [2025], Li et al. [2014], Amrollahi et al. [2019]), we use ProxySPEX [Butler et al., 2025], a method that uses Gradient Boosted Trees to learn a sparse Fourier representation due to its sample efficiency. To sample edit-sets for ProxySPEX, we randomly generate binary masks in which each element is an independent Bernoulli random variable that indicates inclusion in the set.

Require: $L, D, k, N \in \mathbb{N}, 0 < \gamma < 1$. Vocabulary V with mask token M. Protein backbone $y \in Y$.

Algorithm 1 Spectral Edit-Set Selection

Input: $x \in V^L$
$S_j \subseteq [L]\ \forall j \in [D]$
$i \in S_j$ with prob. γ ▷ Sample edit sets
$\tilde{r}_j \leftarrow f(S_j, x, y)$ ▷ Equation 2
$\hat{f} \leftarrow \text{ProxySPEX}(\tilde{r}, S)$ ▷ Sparse Recovery
$S^* \leftarrow \underset{S \subseteq [L],\ |S| \leq k}{\arg\max}\ \hat{f}(S)$ ▷ Appendix D.1
return S^*

Algorithm 2 Protein Model Feedback

Input: $x \in V^L$
$x_i \leftarrow M\ \forall i \in [L]$ ▷ Full mask initialization
$S \leftarrow [L]$ ▷ Edit all tokens initially
for i in $1, \ldots, N-1$ **do**
  $x \sim p(\cdot|S, x, y)$ ▷ Protein model
  $S \leftarrow \text{EditSelection}(x)$ ▷ Algorithm 1
end for
$x \sim p(\cdot|S, x, y)$ ▷ Last feedback iteration
return x

Value Function. The choice of the value function from Equation 2 depends on how the underlying diffusion model is sampled. When Spectral Feedback is applied directly to a diffusion model such as a pre-trained or RL fine-tuned model like DRAKES, we use f_{avg} to approximate the expected reward of the resulting proteins. However, additional test-time alignment methods can be applied on top of a diffusion model within the feedback loop. We demonstrate this by applying Spectral Feedback to Best-of-N with a pre-trained diffusion model. To more naturally follow the dynamics of Best-of-N, we use f_{max} for the edit-set value function. The value function f_{max} should be used instead of f_{avg} whenever there is an underlying reward-maximizing test-time alignment method, such as Best-of-N. In these methods, only the highest-reward sample matters, so we should not penalize an edit-set for low-reward samples by using an average.

5 Experiments

5.1 Experimental Setup

We use protein backbones from the Megascale dataset to evaluate our algorithm. The evaluation metrics are *stability*, $scRMSD$, and *naturalness*.

- The *stability* oracle represents the quantity $\Delta\Delta G = \Delta G_{wild} - \Delta G_{align}$ where ΔG is the Gibbs free energy of the corresponding protein. A positive $\Delta\Delta G$ means a lower ΔG than the corresponding wild-type sequence, and thus a more stable protein.
- The self-consistency RMSD ($scRMSD$) oracle represents the RMSD between the target protein backbone of the inverse folding process and the predicted folded structure of a generated amino acid sequence. We use ESMFold [Lin et al., 2023] to predict the structure of a sequence and then calculate the RMSD with the given backbone.
- The *naturalness* oracle represents the log-likelihood of an amino acid sequence. We use ProtGPT2 [Ferruz et al., 2022], an autoregressive language model that captures the distribution of protein sequences. This metric complements $scRMSD$ in identifying overfitting to the stability oracle.

We use the $\Delta\Delta G$ metric as the primary target reward for aligning the underlying protein model, though we also include results for alignment with the ProtGPT2 oracle and a multi-objective oracle in Appendix F.2. DRAKES trained two $\Delta\Delta G$ oracles with the Megascale dataset: one for alignment and one for evaluation [Wang et al., 2025]. To fairly compare with their work, we also align and evaluate $\Delta\Delta G$ with these oracles. We also use the same validation/test dataset split. Following previous works [Campbell et al., 2024, Nisonoff et al., 2025, Wang et al., 2025], we classify a *successful* protein as one that satisfies $\Delta\Delta G > 0$ and $scRMSD < 2$. These constraints ensure that proteins are at least as stable as their wild-type variants and that their folded structure resembles the target structure.

We use both the pre-trained and RL models from DRAKES to test our method. These models are built with the ProteinMPNN architecture which uses a graph neural network to capture local interactions between amino acids and incorporates extracted features from the backbone structure [Dauparas et al., 2022, Wang et al., 2025]. We also test Spectral Feedback alignment of Best-of-N with the pre-trained model. Following our discussion in Section 4.2, we use f_{avg} for the pre-trained and DRAKES models, and f_{max} for the pre-trained model with Best-of-N.

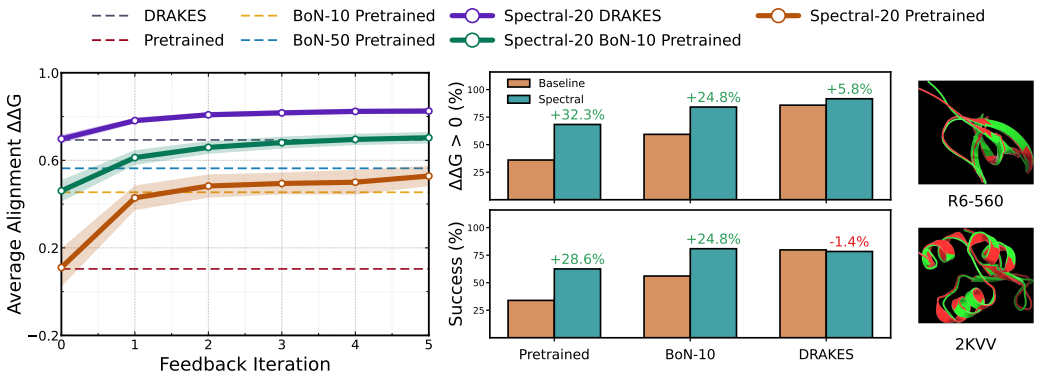


Figure 5: **(left)** Spectral Feedback with $k = 20$ is applied to several underlying protein models: pre-trained, Best-of-10 with pre-trained, and DRAKES. Spectral Feedback improves the alignment reward of each algorithm in only a few iterations. **(middle)** Percentage of stable proteins (evaluation $\Delta\Delta G > 0$) and the resulting success rate when $scRMSD < 2$ are shown. The $scRMSD$ constraint and metric averages are in Table 1. (**right**) ESMFold protein structures comparing the wild-type (*red*) and Spectral Feedback (*green*) sequences demonstrate that Spectral Feedback preserves the underlying inverse protein folding model's capabilities.

5.2 Baseline Edit Selection Methods

Spectral Feedback is unique in that it chooses edit-positions directly based on interactions between tokens. To establish that these interactions are important, we compare the algorithm's performance with methods that do not model interactions. These are described below.

Gradient-Weighting. This method selects edit-positions based on the reward function's gradient at those positions. In particular, for a sequence x, the i^{th} token position is assigned a weight $w(i) = h_i(x)^T \nabla_i r(h_i(x))$ where $h_i(x)$ is the embedding for the i^{th} token. This method is meant to mirror the use of saliency maps in Gruver et al. [2023] for edit-position selection in Langevin dynamics updates, but with signed importance scores. For edit-positions, we select the tokens with the most negative scores since these correspond to tokens that lead to lower reward.

Token Exclusion. Whereas Spectral Feedback directly selects an optimal set of edit-positions, Token Exclusion looks at individual tokens to determine edit-positions. It calculates the value function when masking each token in isolation and then selects the highest-value positions.

Argmax Selection. To show that the sparse recovery methods successfully use observed edit sets to identify a better edit set, we compare against the argmax method which selects the best observed sample edit set. While this method will suffer if sample edit-sets are suboptimal, a sparse recovery method still has hope of finding a rare but optimal edit-set.

Hill Climbing. We implement a randomized 1-flip local search, as described in Szeider [2011], for an alternative greedy baseline. This method iteratively improves its edit selection by uniformly randomly adding or removing an edit-position from the current selection and accepting the change if it improves the reward. A drawback is that the algorithm is inherently serial so parallelization is limited to single-set reward calculations.

5.3 Results on Protein Inverse Folding

Modular Integrations. We first evaluate Spectral Feedback's performance as a modular alignment technique. The reward trajectories in Figure 5 are of the alignment oracle whereas the bar plot shows the evaluation oracle results. The evaluation oracle distributions are also visualized in Figure 2. These results show that Spectral Feedback improves the performance of whichever underlying protein model it samples from. In particular, in Table 1, the reward increases for both the alignment and evaluation oracles. The relatively stable log-likelihoods and the high success rates in Figure 5 further support Spectral Feedback's robustness. However, the average $scRMSD$ increases when Spectral Feedback is applied to DRAKES, thus limiting the gains that can be made in protein success rate. This trade-off is consistent with the reward–structure trade-off already present in high-stability regions within the DRAKES protein distribution. This trade-off is discussed in detail in Appendix E.

Method	Align $\Delta\Delta G \uparrow$	Eval $\Delta\Delta G$ (avg) $\uparrow$	scRMSD (avg) $\downarrow$	scRMSD < 2 (%) $\uparrow$	Log-Likelihood $\uparrow$
pre-trained	0.10	-0.49	1.10	91.8	-147.8
+ **Spectral**	**0.53** (+0.43)	**0.33** (+0.82)	**1.06** (-0.04)	**94.2** (+2.4)	**-146.5** (+1.3)
BoN-10	0.45	0.09	1.09	92.4	-148.1
+ **Spectral**	**0.70** (+0.25)	**0.86** (+0.77)	**1.07** (-0.02)	**96.7** (+4.3)	**-147.5** (+0.6)
DRAKES	0.69	0.99	**1.17**	**93.4**	-157.5
+ **Spectral**	**0.83** (+0.14)	**1.38** (+0.39)	1.53 (+0.36)	85.8 (-7.6)	**-157.7** (-0.2)

Table 1: Spectral Feedback improves alignment and evaluation oracle rewards across all three base methods, with the largest gains on the pre-trained and Best-of-10 baselines. DRAKES + Spectral achieves the highest evaluation $\Delta\Delta G$ overall (1.38) but trades off structural quality ($scRMSD$). Each pair shows baseline followed by its Spectral-enhanced counterpart, using $k = 20$.

Figure 5 shows that Spectral Feedback surpasses Best-of-10's alignment reward after two feedback iterations, though not at a lower computational cost: edit-position selection requires many diffusion model calls to evaluate Equation 2. A lightweight learned value model, as used in Gruver et al. [2023] and Xiong et al. [2026], could reduce this overhead. Despite this cost, Spectral Feedback improves on Best-of-N beyond its diminishing returns. A detailed discussion of computational costs and scaling beyond diminishing returns in Best-of-N is provided in Appendix C.

Many existing alignment methods target either an argmax or a weighted Gibbs distribution, $p_a(x) \propto p_{pre}(x)\exp(\alpha r(x))$. Techniques such as Best-of-N, Beam Search, and MCTS all directly approximate an argmax by searching the sequence space for the highest-reward sample. Common RL objectives with KL regularization, like the objective used in DRAKES [Wang et al., 2025], are maximized by weighted Gibbs distributions. Our experiments suggest that Spectral Feedback could improve additional alignment methods because of the similarity of the target distributions across alignment techniques and the modularity of the feedback mechanism.

Baseline Edit-Position Selection. We compare Spectral Feedback with baseline edit-set selection methods, evaluating how alignment reward scales with the number of edit-set samples and total feedback iteration time. All results are from a single feedback iteration. Exclusion and Gradient are faster but achieve lower rewards than Spectral Feedback, Argmax Selection, and Hill Climbing. Spectral Feedback and Hill Climbing exhibit similar sample-scaling, but Spectral Feedback achieves a better latency–reward trade-off because Hill Climbing's sequential search limits batching across edit-set samples. The batched sampling and reward oracle evaluation required by Equation 2 limit the number of edit-set values that Spectral Feedback and Argmax Selection can compute in parallel. A lightweight value function that avoids these operations could support larger batches which would increase the computational advantage over the sequential search in Hill Climbing. Argmax Selection and Spectral Feedback use identical samples in these experiments, but Spectral Feedback incurs additional overhead from ProxySPEX fitting and optimization. Still, the latency trade-off plot shows that this overhead is insignificant as the number of samples increases.

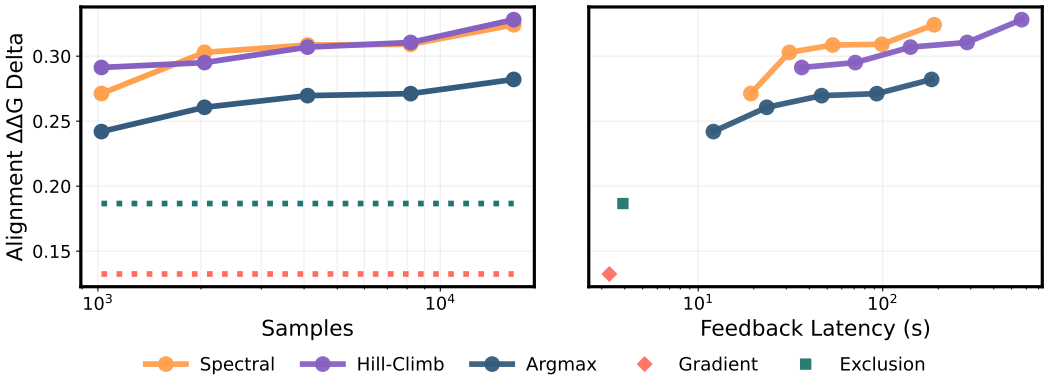


Figure 6: Spectral Feedback shows strong performance in scaling of edit-set samples and latency.

6 Discussion

Conclusion. In this work, we study test-time alignment of discrete diffusion models for inverse folding with access to protein sequence reward oracles. We introduce Spectral Feedback, an algorithm that iteratively identifies a promising subset of edit-positions for a generated sequence and re-samples the corresponding amino acids through the diffusion reverse process. The value of editing an amino acid depends on which other amino acids are edited alongside it, making edit-set selection a challenging combinatorial problem. We address this challenge by empirically showing that edit-set value functions can have sparse Fourier representations. We use sparse recovery methods to approximate these functions and then optimize the approximations to select promising edit-sets. On the Megascale dataset, Spectral Feedback improves stability alignment rewards across three settings: the pre-trained model, Best-of-N, and DRAKES. It also preserves structural self-consistency and increases protein success rates for the pre-trained and Best-of-N models, while applying Spectral Feedback to DRAKES introduces a stability–structure trade-off.

Limitations & Future Work. A limitation of our sparse recovery method is that high global R^2 does not necessarily correspond to accurate argmax recovery. We explore using an exponentially tilted objective to address this in Appendix D.3. Another sparse recovery limitation is that a large number of edit-set samples is required to successfully learn a sparse Fourier representation. However, once in this high-sample regime, our algorithm scales better than competing methods. In the future, the scaling of alternative sparse recovery methods such as SPEX [Kang et al., 2025] could be explored.

Acknowledgments and Disclosure of Funding

We thank Jennifer Listgarten and her lab group for fruitful discussions and words of advice. We also thank Justin Kang for helpful discussions and his insights. This research used both the DeltaAI advanced computing and data resource, which is supported by the National Science Foundation (award OAC 2320345) and the State of Illinois, and the Delta advanced computing and data resource which is supported by the National Science Foundation (award OAC 2005572) and the State of Illinois [Gropp et al., 2023]. Delta and DeltaAI are joint efforts of the University of Illinois Urbana-Champaign and its National Center for Supercomputing Applications. Additionally, this research used the Anvil supercomputer at Purdue University which is also supported by the National Science Foundation (award OAC 2005632) [Song et al., 2022]. Lastly, part of this work was funded by NSF fund CIF-2507523 and ARO fund 062982.

References

Amirali Aghazadeh, Orhan Ocal, and Kannan Ramchandran. CRISPRLand: Interpretable large-scale inference of dna repair landscape based on a spectral approach. *Bioinformatics*, 36(Supplement 1): i560–i568, 2020. doi: 10.1093/bioinformatics/btaa505.

Amirali Aghazadeh, Hunter Nisonoff, Orhan Ocal, David H. Brookes, Yijie Huang, O. Ozan Koyluoglu, Jennifer Listgarten, and Kannan Ramchandran. Epistatic net allows the sparse spectral regularization of deep neural networks for inferring fitness functions. *Nature Communications*, 12(1):5225, September 2021. doi: 10.1038/s41467-021-25371-3. URL https://www.nature.com/articles/s41467-021-25371-3.

Andisheh Amrollahi, Amir Zandieh, Michael Kapralov, and Andreas Krause. Efficiently learning fourier sparse set functions. *Advances in Neural Information Processing Systems*, 32, 2019.

Nathaniel R. Bennett, Joseph L. Watson, Robert J. Ragotte, Andrew J. Borst, DéJenaé L. See, Connor Weidle, Riti Biswas, Yutong Yu, Ellen L. Shrock, Russell Ault, Philip J. Y. Leung, Buwei Huang, Inna Goreshnik, John Tam, Kenneth D. Carr, Benedikt Singer, Cameron Criswell, Basile I. M. Wicky, Dionne Vafeados, Mariana Garcia Sanchez, Ho Min Kim, Susana Vázquez Torres, Sidney Chan, Shirley M. Sun, Timothy T. Spear, Yi Sun, Keelan O'Reilly, John M. Maris, Nikolaos G. Sgourakis, Roman A. Melnyk, Chang C. Liu, and David Baker. Atomically accurate de novo design of antibodies with rfdiffusion. *Nature*, 649(8095):183–193, January 2026. doi: 10.1038/s41586-025-09721-5. URL https://www.nature.com/articles/s41586-025-09721-5.

Landon Butler, Abhineet Agarwal, Justin Singh Kang, Yigit Efe Erginbas, Bin Yu, and Kannan Ramchandran. Proxyspex: Inference-efficient interpretability via sparse feature interactions in llms, 2025. URL https://arxiv.org/abs/2505.17495.

Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design, 2024. URL https://arxiv.org/abs/2402.04997.

Mert Cemri, Ajil Jalal, and Kannan Ramchandran. Discrete diffusion posterior sampling for protein design. In *ICML 2024 Workshop on Structured Probabilistic Inference & Generative Modeling*, 2024.

Heather J. Cordell. Epistasis: What it means, what it doesn't mean, and statistical methods to detect it in humans. *Human Molecular Genetics*, 11(20):2463–2468, October 2002. doi: 10.1093/hmg/11.20.2463. URL https://doi.org/10.1093/hmg/11.20.2463.

Jeremie Theddy Darmawan, Yarin Gal, and Pascal Notin. Sampling protein language models for functional protein design. *bioRxiv*, 2025. doi: 10.1101/2025.09.14.676087. URL https://www.biorxiv.org/content/early/2025/09/17/2025.09.14.676087.

J. Dauparas, I. Anishchenko, N. Bennett, H. Bai, R. J. Ragotte, L. F. Milles, B. I. M. Wicky, A. Courbet, R. J. de Haas, N. Bethel, P. J. Y. Leung, T. F. Huddy, S. Pellock, D. Tischer, F. Chan, B. Koepnick, H. Nguyen, A. Kang, B. Sankaran, A. K. Bera, N. P. King, and D. Baker. Robust deep learning–based protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022. doi: 10.1126/science.add2187. URL https://www.science.org/doi/abs/10.1126/science.add2187.

Noelia Ferruz, Steffen Schmidt, and Birte Höcker. Protgpt2 is a deep unsupervised language model for protein design. *Nature Communications*, 13(1):4348, July 2022. doi: 10.1038/s41467-022-32007-7. URL https://doi.org/10.1038/s41467-022-32007-7.

William Gropp, Tim Boerner, Brett Bode, and Greg Bauer. Delta: Balancing GPU Performance with Advanced System Interfaces. 2023. URL https://hdl.handle.net/2142/117179.

Nate Gruver, Samuel Stanton, Nathan C. Frey, Tim G. J. Rudner, Isidro Hotzel, Julien Lafrance-Vanasse, Arvind Rajpal, Kyunghyun Cho, and Andrew Gordon Wilson. Protein design with guided discrete diffusion, 2023. URL https://arxiv.org/abs/2305.20009.

Tejas M. Gupte, Michael Ritt, and Sivaraj Sivaramakrishnan. Chapter Seven - ER/K-link—Leveraging a native protein linker to probe dynamic cellular interactions. In Maarten Merkx, editor, *Linkers in Biomacromolecules*, volume 647 of *Methods in Enzymology*, pages 173–208. Academic Press, January 2021. doi: 10.1016/bs.mie.2020.10.002. URL https://www.sciencedirect.com/science/article/pii/S0076687920303487.

Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2026. URL https://www.gurobi.com.

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control, 2022. URL https://arxiv.org/abs/2208.01626.

Audrey Huang, Adam Block, Qinghua Liu, Nan Jiang, Akshay Krishnamurthy, and Dylan J. Foster. Is best-of-n the best of them? coverage, scaling, and optimality in inference-time alignment, 2025. URL https://arxiv.org/abs/2503.21878.

Justin Singh Kang, Landon Butler, Abhineet Agarwal, Yigit Efe Erginbas, Ramtin Pedarsani, Kannan Ramchandran, and Bin Yu. Spex: Scaling feature interaction explanations for llms, 2025. URL https://arxiv.org/abs/2502.13870.

Sanghyun Lee, Sunwoo Kim, Seungryong Kim, Jongho Park, and Dongmin Park. Effective test-time scaling of discrete diffusion through iterative refinement, 2025. URL https://arxiv.org/abs/2511.05562.

Xiao Li, Joseph Kurata Bradley, Sameer Pawar, and Kannan Ramchandran. The spright algorithm for robust sparse hadamard transforms. In *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, pages 1857–1861, Honolulu, HI, USA, June 2014. IEEE. doi: 10.1109/ISIT.2014.6875155. URL https://ieeexplore.ieee.org/document/6875155.

Yuchen Liang, Yingbin Liang, Lifeng Lai, and Ness Shroff. Discrete diffusion models: Novel analysis and new sampler guarantees, 2025. URL https://arxiv.org/abs/2509.16756.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023. doi: 10.1126/science.ade2574. URL https://www.science.org/doi/abs/10.1126/science.ade2574.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback, 2023. URL https://arxiv.org/abs/2303.17651.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations, 2022. URL https://arxiv.org/abs/2108.01073.

Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models, 2022. URL https://arxiv.org/abs/2211.09794.

Hunter Nisonoff, Junhao Xiong, Stephan Allenspach, and Jennifer Listgarten. Unlocking guidance for discrete state-space diffusion and flow models, 2025. URL https://arxiv.org/abs/2406.01572.

Ryan O'Donnell. *Analysis of Boolean functions*. Cambridge University Press, 2014.

Frank J. Poelwijk, Michael Socolich, and Rama Ranganathan. Learning the pattern of epistasis linking genotype and phenotype in a protein. *Nature Communications*, 10(1):4213, September 2019. doi: 10.1038/s41467-019-12130-8. URL https://www.nature.com/articles/s41467-019-12130-8.

Machel Reid, Vincent J. Hellendoorn, and Graham Neubig. Diffuser: Discrete diffusion via edit-based reconstruction, 2022. URL https://arxiv.org/abs/2210.16886.

Robin Scheibler, Saeid Haghighatshoar, and Martin Vetterli. A fast hadamard transform for signals with sublinear sparsity in the transform domain. *IEEE Transactions on Information Theory*, 61(4): 2115–2132, 2015. doi: 10.1109/TIT.2015.2404441.

X. Carol Song, Preston Smith, Rajesh Kalyanam, Xiao Zhu, Eric Adams, Kevin Colby, Patrick Finnegan, Erik Gough, Elizabett Hillery, Rick Irvine, Amiya Maji, and Jason St. John. Anvil - system architecture and experiences from deployment and early user operations. In *Practice and Experience in Advanced Research Computing 2022: Revolutionary: Computing, Connections, You*, PEARC '22, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450391610. doi: 10.1145/3491418.3530766. URL https://doi.org/10.1145/3491418.3530766.

Haoran Sun, Lijun Yu, Bo Dai, Dale Schuurmans, and Hanjun Dai. Score-based continuous-time discrete diffusion models, 2023. URL https://arxiv.org/abs/2211.16750.

Stefan Szeider. The parameterized complexity of k-flip local search for sat and max sat. *Discrete Optimization*, 8(1):139–145, 2011. ISSN 1572-5286. doi: https://doi.org/10.1016/j.disopt.2010.07.003. URL https://www.sciencedirect.com/science/article/pii/S1572528610000526. Parameterized Complexity of Discrete Optimization.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996. URL https://www.jstor.org/stable/2346178.

Kotaro Tsuboyama, Justas Dauparas, Jonathan Chen, Elodie Laine, Yasser Mohseni Behbahani, Jonathan J. Weinstein, Niall M. Mangan, Sergey Ovchinnikov, and Gabriel J. Rocklin. Mega-scale experimental analysis of protein folding stability in biology and design. *Nature*, 620(7973):434–444, 2023. doi: 10.1038/s41586-023-06328-6. URL https://www.nature.com/articles/s41586-023-06328-6.

Masatoshi Uehara, Yulai Zhao, Chenyu Wang, Xiner Li, Aviv Regev, Sergey Levine, and Tommaso Biancalani. Inference-time alignment in diffusion models with reward-guided generation: Tutorial and review, 2025. URL https://arxiv.org/abs/2501.09685.

Chenyu Wang, Masatoshi Uehara, Yichun He, Amy Wang, Tommaso Biancalani, Avantika Lal, Tommi Jaakkola, Sergey Levine, Hanchen Wang, and Aviv Regev. Fine-tuning discrete diffusion models via reward optimization with applications to dna and protein design. *arXiv preprint arXiv:2410.13643*, 2025. URL https://arxiv.org/abs/2410.13643.

Guanghan Wang, Yair Schiff, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Remasking discrete diffusion models with inference-time scaling, 2026. URL https://arxiv.org/abs/2503.00307.

Junhao Xiong, Ishan Gaur, Maria Lukarska, Hunter Nisonoff, Luke M. Oltrogge, David F. Savage, and Jennifer Listgarten. Proteinguide: On-the-fly property guidance for protein sequence generative models, 2026. URL https://arxiv.org/abs/2505.04823.

Eric Xue, Ke Chen, Zeyi Huang, Yuyang Ji, and Haohan Wang. Improve: Iterative model pipeline refinement and optimization leveraging llm experts, 2025. URL https://arxiv.org/abs/2502.18530.

Andy Hsien-Wei Yeh, Christoffer Norn, Yakov Kipnis, Doug Tischer, Samuel J. Pellock, Declan Evans, Pengchen Ma, Gyu Rie Lee, Jason Z. Zhang, Ivan Anishchenko, Brian Coventry, Longxing Cao, Justas Dauparas, Samer Halabiya, Michelle DeWitt, Lauren Carter, K. N. Houk, and David Baker. De novo design of luciferases using deep learning. *Nature*, 614(7949):774–780, February 2023. doi: 10.1038/s41586-023-05696-3. URL https://www.nature.com/articles/s41586-023-05696-3.

Kai Yi, Bingxin Zhou, Yiqing Shen, Pietro Liò, and Yu Guang Wang. Graph denoising diffusion for inverse protein folding, 2023. URL https://arxiv.org/abs/2306.16819.

A Hyperparameter Settings

A.1 Computational Resources

The experiments used NVIDIA GH200, H100, and L40S GPUs. All timing experiments were run on GH200s. The timing experiments also used 32 GB of RAM and 16 CPU cores. The CPU parallelism is useful for speeding up the Gurobi solver used during sparse Fourier optimization.

A.2 Protein Design Methods

pre-trained We use the ProteinMPNN discrete diffusion model that was developed in the original DRAKES paper.

DRAKES The KL-penalty is $\beta = 0.001$. As with the pre-trained model, we use the RL fine-tuned ProteinMPNN model developed in the original DRAKES paper.

Best-of-N The performance of Best-of-N will increase monotonically as N increases, but with diminishing returns. We use Spectral Feedback in conjunction with Best-of-N to improve alignment beyond these limitations.

A.3 Feedback Mechanisms

Unless otherwise stated, we use $n = 64$ samples for the value functions in Equation 2.

Feedback Loop We select $k = 20$ edit-positions and $D = 8192$ mask samples as our experimental parameters by following the results in the validation curves below. The alignment reward continues to improve with more edit-positions and more mask samples, though with diminishing returns. Increasing k and D also increases the computational cost so we found our selected parameters to be a reasonable balance. The mask sampling experiment does not use cross-validation during edit-selection, resulting in worse relative performance than the edit-positions experiment, which does use cross-validation. We used pre-selected settings (Max Depth=None, Leaves=50, Rate=0.01, λ=0.0001) because we faced compute resource limitations when increasing D to 32768. We used 3 feedback iterations in these experiments for similar reasons.

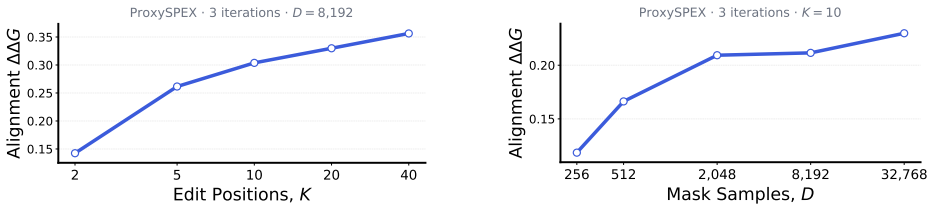


Figure 7: Validation curves for feedback loop parameters. Log-scale shows diminishing returns.

Spectral Feedback (ProxySPEX). The ProxySPEX subroutine used in Spectral Feedback fits a GBT ensemble. We used the validation protein set, optimizing for each hyperparameter in isolation.

Parameter	Validation Values
Max. Tree Depth	$3, 5,$ None
Number of leaves	$30, 50$
Learning Rate	$0.01, 0.1$
λ	5 values with $\lambda_{min} = 0.00001, \lambda_{max} = 0.1, \lambda = 0$

First-order LASSO. We use LASSO to implement a first-order sparse recovery method. In Spectral Feedback, we are given training sets $S_j \subseteq [L] \forall j \in [D]$. We define corresponding bit masks b_j where $b_j[i] = 1 \iff i \in S_j$. A first-order sparse recovery method models the value function as a linear function $f(S) = \beta + \sum_i c_i \cdot \mathbb{1}\{i \in S\}$ where $\beta \in \mathbb{R}$ and $c \in \mathbb{R}^L$. We can then define the value function for the bit masks as $f(b) = \beta + c^T b$. Each S_j and b_j have a corresponding training reward r_j. LASSO is natural for this setting because we assume a sparse Fourier transform and the

ℓ_1 norm induces sparsity. Applying LASSO with the penalty coefficient $\lambda \geq 0$ yields the following optimization problem:

$$(\beta, \tilde{c}) = \underset{\beta \in \mathbb{R},\ c \in \mathbb{R}^L}{\arg\min} \frac{1}{D} \sum_{j=1}^{D} (\beta + c^T b_j - r_j)^2 + \lambda ||c||_1$$

We can then write the maximization problem of $f(b)$ like so:

$$b^* = \underset{b \in \{0,1\}^L,\ \sum_i b[i] \leq k}{\arg\max} \tilde{c}^T b$$

The solution is then the top-k coefficients of $\tilde{c}$ that are also positive:

$$b^*[i] = \begin{cases} 1 \text{ if } \tilde{c}_i > 0 \text{ and } \tilde{c}_i \in \text{top } k \text{ coefficients} \\ 0 \text{ else} \end{cases}$$

As with Spectral Feedback, when using LASSO for edit-position selection, we run cross-validation to be consistent with the original ProxySPEX paper [Butler et al., 2025].

Parameter	**Validation Values**
λ	5 values with $\lambda_{min} = 0.00001, \lambda_{max} = 0.1$ and $\lambda = 0.0$

Exclusion & Gradient. The only hyperparameter for these edit-position methods is the number of edit-positions, which we already select for Spectral Feedback on the validation set. We use the same number of positions (20 in our experiments) to make them fair baselines.

B Discrete Diffusion Models

The discrete diffusion model is a type of generative language model that has been popularized for biology-related tasks such as inverse folding. We describe these models formally and explain how to sample from them. First, consider a finite vocabulary V, a sequence length L, and a corresponding sample space $\mathcal{X} := V^L$. We then model a corresponding probability mass trajectory $p_t \in \mathbb{R}^N$ where $N := |\mathcal{X}| = |V|^L$. Like in continuous diffusion, the discrete diffusion process involves evolving p_t by a differential equation. The evolution of p_t forward in time is called the forward process. In the continuous case, it is common to use the Fokker-Planck equation, a partial differential equation related to the Itô diffusion process. In the discrete setting, the following linear ordinary differential equation is used:

$$\frac{dp_t}{dt} = Q_t p_t, \quad p_0 = p_{data}$$

Here, $Q_t \in \mathbb{R}^{N \times N}$ are the rate matrices of a Continuous Time Markov Chain (CTMC) and correspondingly must satisfy

$$Q_t(i,j) \geq 0 \quad \forall i \neq j$$
$$\sum_i Q_t(i,j) = 0 \quad \forall j.$$

The first constraint is due to the non-negativity of probability masses, whereas the second constraint ensures that the total probability mass of the system does not change. Like for continuous diffusion models, to make such a process useful for inference, we need a way to reverse the process. In particular, if we structure the process to converge to a probability distribution p_T from which we know how to sample, and then sample from the reverse process initialized by p_T, we will converge to the target distribution p_0. The following is a commonly used reverse process [Liang et al., 2025]:

$$\frac{dp_{T-t}}{dt} = \bar{Q}_{T-t} p_{T-t}, \quad \bar{Q}_t(x,y) = \begin{cases} \dfrac{p_t(x)}{p_t(y)} Q_t(y,x) & x \neq y \\ -\displaystyle\sum_{x' \neq x} \bar{Q}_t(x',x) & x = y \end{cases}$$

Like before, the reverse process is a CTMC and $\bar{Q}_t$ are the corresponding rate matrices. We can design the process to converge to a distribution of our choosing by selecting Q_t accordingly. Thus, to learn $\bar{Q}_t$ so that we can sample from the reverse process, we just need to learn $p_t(y)/p_t(x)$. This ratio is commonly called the score function [Sun et al., 2023].

In its most abstract form, designing a discrete diffusion model requires defining a forward process with some generator Q_t and then learning a corresponding score function $p_t(y)/p_t(x)$ from the training data. A popular choice for the forward process is random masking. The CTMC has some probability of transitioning tokens into masks, and once a token is masked, it stays masked. The reverse process is initialized with a fully masked sequence x_T, and then progressively flips tokens from masks to unmasked elements in the vocabulary. Analogous to the forward process, a token will stay unmasked once it leaves its masked state.

The Euler-Maruyama scheme, a first-order accurate iterative method, is commonly used to approximate the solution of continuous diffusion models. It is then natural to apply Euler's method to approximate the reverse discrete diffusion evolution:

$$p_{n+1} \leftarrow p_n + h\bar{Q}_{T-t_n} p_n$$

where h is the step size. This method is unfortunately impractical since it would require keeping track of the scores for an exponential number of state transitions $x \to y$. Instead, the following Hamming distance constraint is often used: $Q_t(x, y) = 0$ if $d(x, y) > 1$ [Liang et al., 2025]. This allows only state transitions that flip one token. Still, to support multiple token updates during a single step for faster inference, the transitions of each token are assumed to be independent.

C Compute Complexity

The Spectral Feedback pipeline involves calls to several models with varying computational costs. The protein diffusion model is called at each iteration of the reverse process. In the first iteration, a protein sequence is generated from a fully masked sequence. Subsequent iterations initialize the reverse process with partially noisy sequences via the feedback process. Recall that N is the total number of generated proteins so the number of feedback iterations is $N-1$. In each feedback iteration Spectral Feedback makes D calls to the mask value function. Let M be the number of reverse process steps. Then Spectral Feedback makes $M \times N$ calls to the diffusion model and $D \times (N-1)$ calls to the mask value function. The value functions in Equation 2 call the diffusion model once to get state change probabilities of a single reverse process step, then sample n state changes and pass each generated sequence to the alignment reward oracle. With these value functions, Spectral Feedback makes $M \times N + D \times (N-1)$ diffusion model calls and $n \times D \times (N-1)$ alignment reward oracle calls. Best-of-N is cheaper in compute since it makes $M \times N$ calls to the diffusion model and N calls to the alignment reward oracle. A key limitation of Best-of-N, however, is that it has diminishing returns as N increases and levels off in reward. With protein design, desirable proteins may be rare as discussed in Xiong et al. [2026]. Figure 8 demonstrates that Spectral Feedback can improve Best-of-N beyond its diminishing returns and identify these rare yet more stable proteins. The modularity of the algorithm allows it to also improve other alignment methods such as Beam Search and RL as shown in Figure 12.

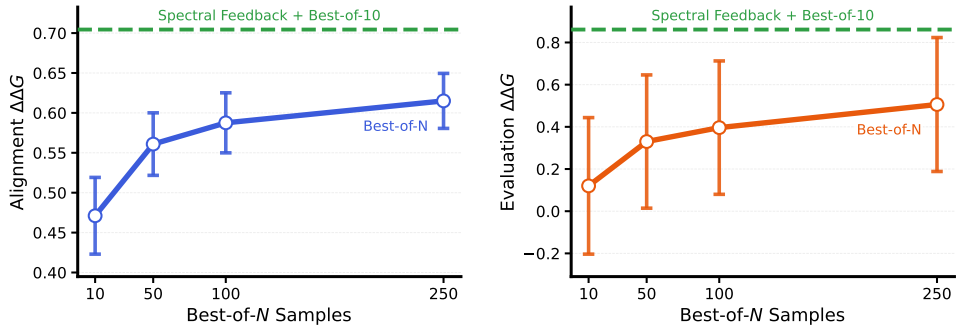


Figure 8: Spectral Feedback with Best-of-10 reaches better performance than scaling Best-of-N. Hyperparameters: $D = 8192$, $k = 20$, and 5 feedback iterations.

D Sparse Fourier Approximation

D.1 Sparse Fourier Optimization

The derivation in this section originally appeared in Appendix A.3 of Butler et al. [2025]. We restate it here.

Assume $\hat{f} : 2^{[n]} \to \mathbb{R}$ has a sparse, low-degree Fourier expansion with support

$$\mathcal{A} \;=\; \big\{\, T \subseteq [n] \;:\; \hat{F}(T) \neq 0,\; |T| \leq d \,\big\}, \qquad |\mathcal{A}| \ll 2^n,$$

such that

$$\hat{f}(S) \;=\; \sum_{T \in \mathcal{A}} (-1)^{|S \cap T|}\, \hat{F}(T).$$

To formulate the optimization as an integer program, we re-express $\hat{f}$ in the Möbius basis, which replaces the parity functions $(-1)^{|S\cap T|} \in \{-1, +1\}$ with subset-indicator functions $\mathbb{1}[T \subseteq S] \in \{0, 1\}$. Through a change of variable, the Möbius coefficients are

$$\hat{M}(T) \;=\; (-2)^{|T|} \sum_{\substack{S \in \mathcal{A}, \\ S \supseteq T}} \hat{F}(S).$$

Letting $\mathcal{A}^+ = \big\{\, R \subseteq T \;\big|\; T \in \mathcal{A} \,\big\}$ denote the downward closure of $\mathcal{A}$, the inverse Möbius transform gives

$$\hat{f}(S) \;=\; \sum_{\substack{R \in \mathcal{A}^+, \\ R \subseteq S}} \hat{M}(R).$$

The optimization problem can then be expressed as a polynomial over $\{0, 1\}^n$. Let $\mathbf{x} \in \{0, 1\}^n$ be the characteristic vector of S, so that $x_i = 1$ if and only if $i \in S$. We focus on the maximization problem (minimization follows analogously):

$$\max_{\substack{S \subseteq [n], \\ |S| \leq k}} \hat{f}(S) \;=\; \max_{\substack{\mathbf{x} \in \{0,1\}^n, \\ \sum_i x_i \leq k}} \sum_{R \in \mathcal{A}^+} \hat{M}(R) \prod_{i \in R} x_i.$$

To reduce the problem to a linear integer program, each monomial $\prod_{i \in R} x_i$ is replaced with a binary decision variable $y_R \in \{0, 1\}$. We augment $\mathcal{A}^+$ to include all singletons $\{i\}$ for $i \in [n]$, setting $\hat{M}(\{i\}) = 0$ for any newly added singletons, so that $y_{\{i\}}$ exists for every i. Linking constraints enforce $y_R = \prod_{i \in R} x_i$:

$$\begin{aligned}
\max_{\mathbf{y} \in \{0,1\}^{|\mathcal{A}^+|}} \quad & \sum_{R \in \mathcal{A}^+} \hat{M}(R)\, y_R \\
\text{s.t.} \quad & y_R \;\leq\; y_Q && \forall\, Q \subset R,\; R \in \mathcal{A}^+ \\
& \sum_{i \in R} y_{\{i\}} \;\leq\; |R| - 1 + y_R && \forall\, R \in \mathcal{A}^+ \\
& \sum_{i \in [n]} y_{\{i\}} \;\leq\; k.
\end{aligned}$$

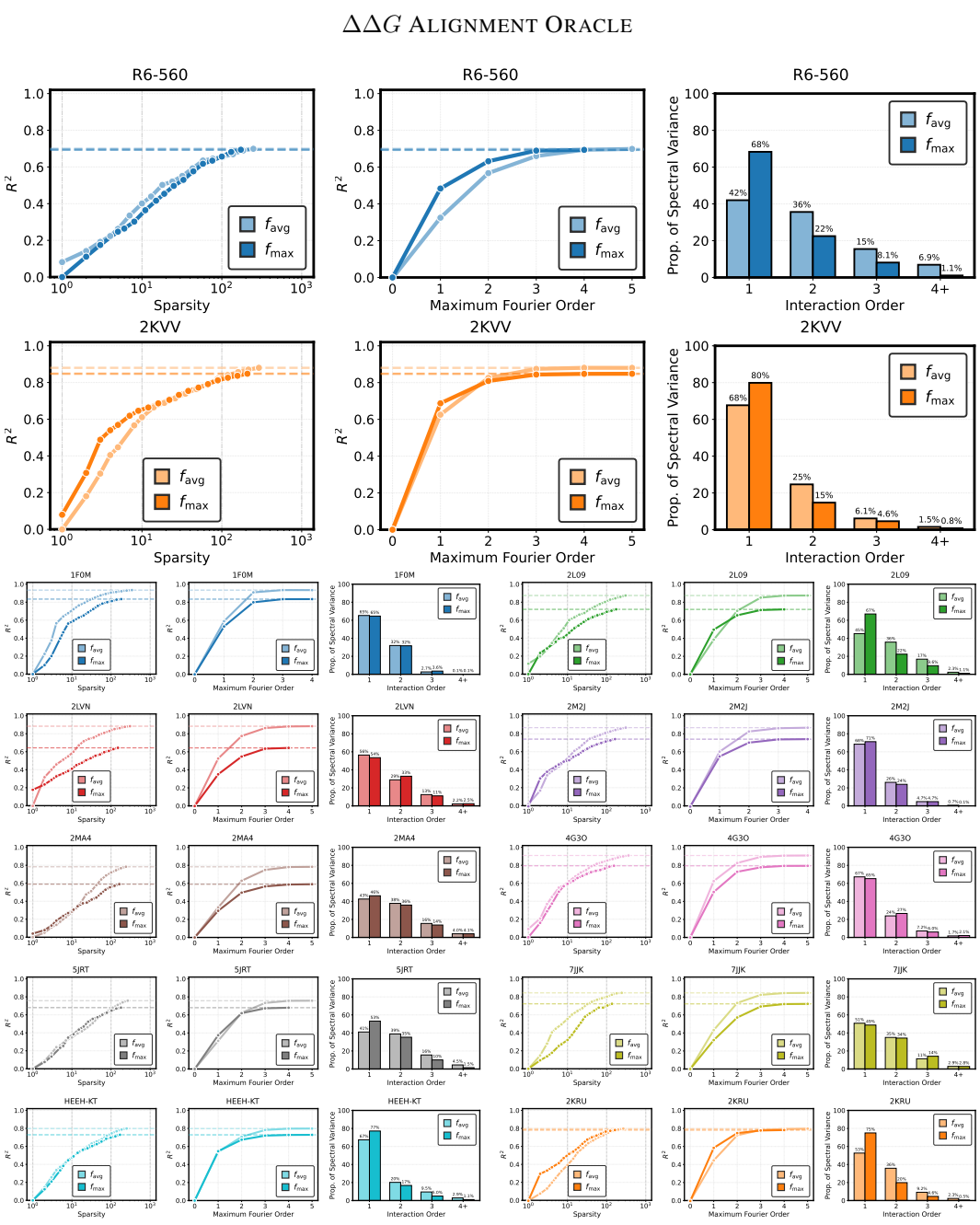
The first constraint guarantees that whenever a monomial is activated (i.e., $x_i = 1$ for all $i \in R$), all of its subsets are also activated. The second ensures that if a monomial is deactivated (i.e., $x_i = 0$ for some $i \in R$), then at least one of its constituent singletons $y_{\{i\}}$ is likewise deactivated. The third imposes the cardinality constraint, and after solving the program, the solution $\mathbf{x}$ is read off from $y_{\{i\}}$.

Let $s = |\mathcal{A}|$ be the sparsity of $\hat{f}$. The resulting program has at most $s \cdot 2^d + n$ decision variables: up to 2^d subsets per element of $\mathcal{A}$, plus the n augmented singletons. The constraints decompose as at most $(s \cdot 2^d + n)(2^d - 1)$ subset-activation constraints (one per pair $Q \subset R$), $s \cdot 2^d + n$ deactivation constraints, and a single cardinality constraint, giving at most $s \cdot 4^d + n \cdot 2^d + 1$ constraints in total. The program is therefore tractable when $\hat{f}$ is both sparse (s small) and low-degree (d small), even for large n.

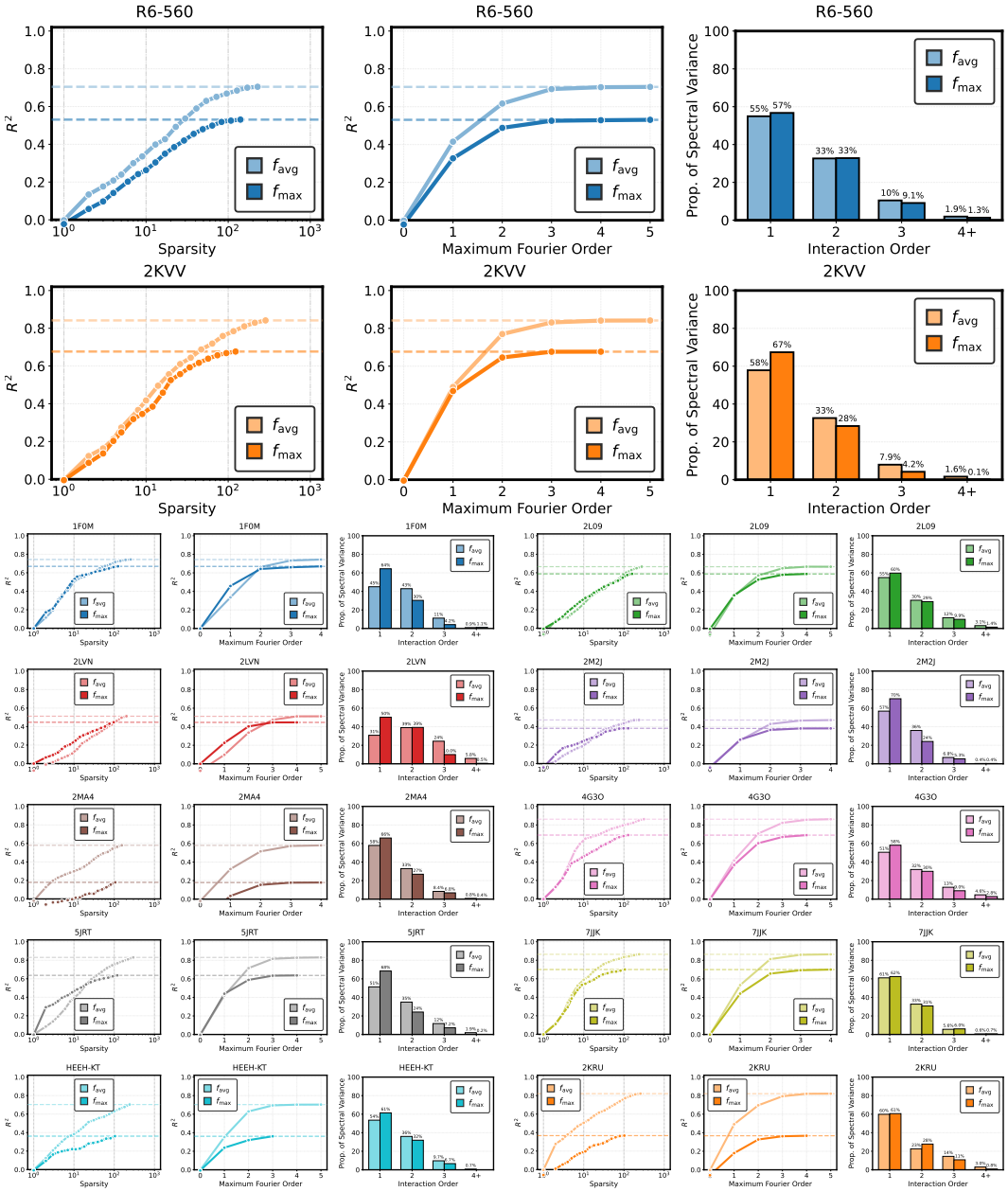
We solve the program using Gurobi's default branch-and-cut algorithm via the `gurobipy` interface [Gurobi Optimization, LLC, 2026].

D.2 Sparsity Experiments

The data in Figure 3 were computed with SPEX under a compute budget of 100k and a max interaction order of 5. The value functions use $n = 64$ empirical samples for each evaluation. The underlying protein model is the pre-trained model. We show additional spectral analysis results below for each protein in the test set, using both the f_{max} and the f_{avg} value functions. We evaluate the sparsity of the functions as well as the necessity of high-order terms. We first study the $\Delta\Delta G$ alignment oracle that was used in the paper's main results. We repeat this study with the ProtGPT2 Log-Likelihood oracle as the target alignment oracle. The results show that both value functions exhibit sparse Fourier transforms as they can be approximated with high faithfulness (R^2) by using relatively few Fourier coefficients. Additionally, most spectral variance is concentrated at lower interaction orders. Interestingly, f_{avg} tends to achieve a higher R^2 than f_{max} under the same compute budget. This may reflect greater variance in f_{max} than in f_{avg}.



PROTGPT2 LOG-LIKELIHOOD ORACLE



D.3 High-Sensitivity Value Function Ablation

High global R^2 is not by itself a sufficient certificate for argmax recovery. One can easily construct counterexamples where an approximation has arbitrarily high global R^2, yet selects a poor maximizer. Given our strong experimental results using the global-R^2-driven approximation, we do not believe our value functions fall into this pathological regime. Nevertheless, this is an important distinction. Alternative objectives can be used to more directly align with maximization. In particular, a value function $f(S)$ can be replaced with a strictly increasing transformation

$$g(S) = \phi(f(S)),$$

which preserves the optimizer but can be used to make an L_2 approximation increasingly sensitive to high-value subsets. One concrete choice is an exponentially tilted objective,

$$g_\beta(S) = \exp\left(\beta \frac{f(S) - f_{\max}}{f_{\max} - f_{\min}}\right),$$

where $f_{\max}$ and $f_{\min}$ denote the maximum and minimum values of f over the sampled sets. Under this normalization, the maximizer has value 1, while lower-value subsets are exponentially downweighted toward $\exp(-\beta)$. Larger β therefore places more emphasis on accurately approximating the top of the value function. We use the convention that for $\beta = 0$, $g_0(S) = f(S)$.

This transformation has interesting implications for spectral sparsity. For normalization with unique global extrema, as $\beta \to \infty$, g_β approaches a Dirac delta at the maximizer $S^\star$. A Dirac delta on the Boolean cube is spectrally dense. We have repeated our experiments above by analyzing spectral sparsity under different severities of exponential tilt. We find that for modest choices of β (0.25-8), the sparsity levels of the function remain similar to the untransformed $f(S)$, and the proportion of spectral variance in higher-degree terms remains high. For larger values of β, the spectrum places more energy on high-order terms and becomes increasingly dense, making it difficult for spectral recovery techniques to achieve high R^2. Thus, increasing β can trade better argmax alignment for weaker Fourier sparsity. An improved version of Spectral Feedback could try learning g_β for a range of β using the same samples, and then cross-validate to select the highest β for a pre-specified approximation quality.

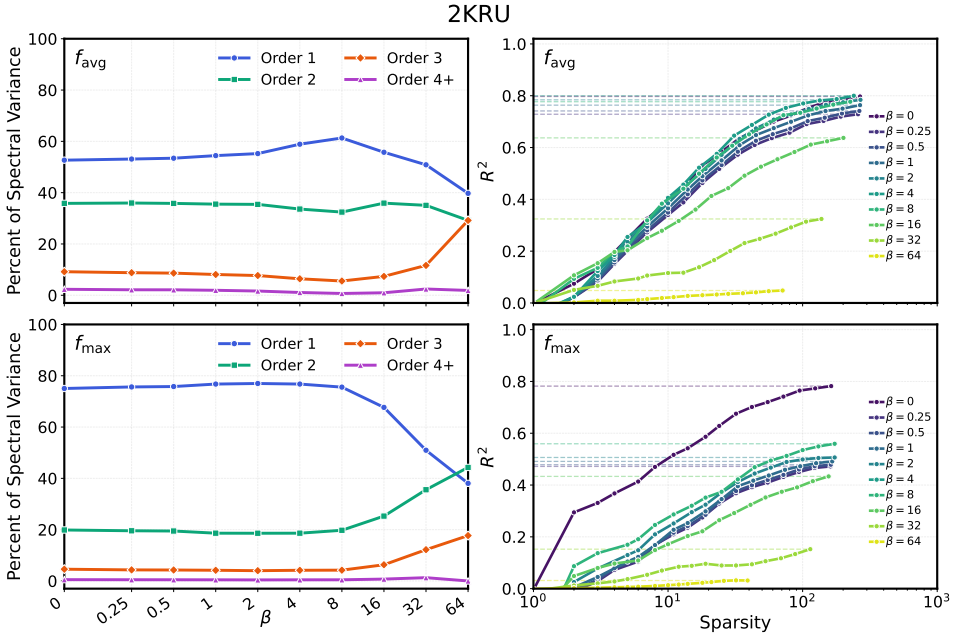


Figure 9: Exponential tilt function Fourier analysis for 2KRU with the pre-trained model.

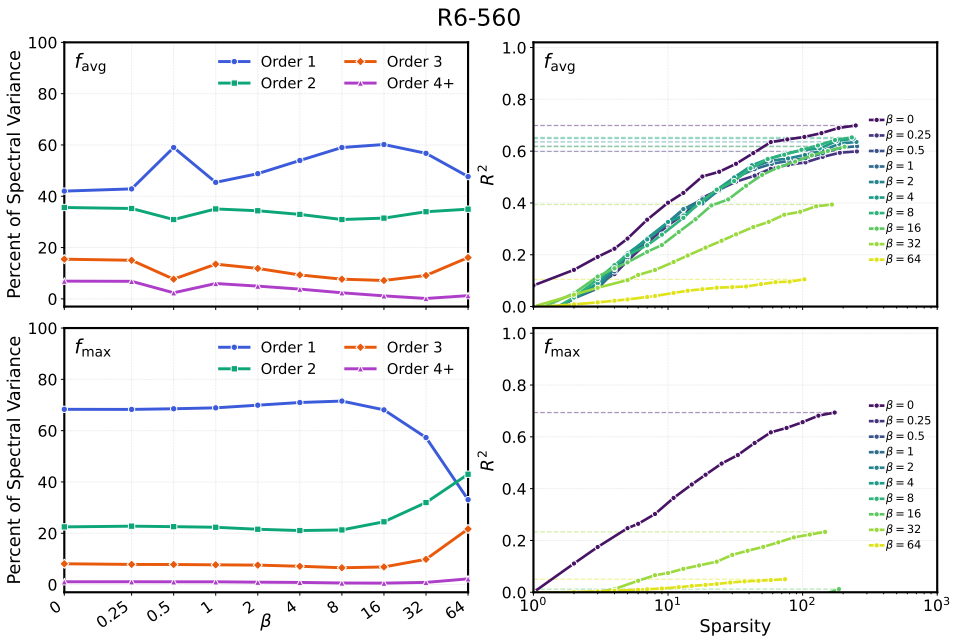


Figure 10: Exponential tilt function Fourier analysis for r6_560_TrROS_Hall with pre-trained model.

E Spectral Feedback $scRMSD$ Discussion

Table 1 shows an increase in $scRMSD$ when Spectral Feedback is applied to DRAKES. This is an undesired result since this means the inverse folding is less accurate. Recall that one of the two conditions for a protein to be classified as successful is that $scRMSD < 2$. An increase in $scRMSD$ then limits the gains that can be made in protein success rate, as seen in Figure 5.

To better understand this behavior, we analyzed scatter and contour plots relating $scRMSD$ to alignment $\Delta\Delta G$. The results in the corresponding figure below indicate that Spectral Feedback inherits the properties of both the underlying model and the reward objective. In the DRAKES distribution, the highest-reward region contains structurally inconsistent samples, including sequences with $scRMSD$ values between 8 and 10, well above the desired threshold of 2. In contrast, the pre-trained model has fewer of these high-$scRMSD$ samples, and its highest-reward samples generally retain lower $scRMSD$. As a result, the average $scRMSD$ is greater for DRAKES than for the pre-trained model: 1.17 vs 1.10. This difference increases with the addition of Spectral Feedback as reported in the paper. DRAKES also has a lower average log-likelihood than the pre-trained model, as seen in Table 1, which further indicates overfitting to the alignment oracle that leads to more out-of-distribution proteins.

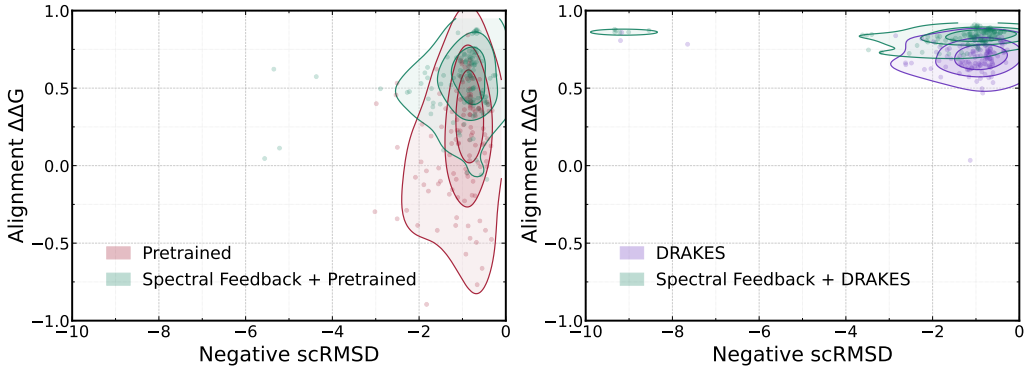


Figure 11: $scRMSD$ vs $\Delta\Delta G$ distributions show a worse trade-off with $scRMSD$ for DRAKES when applying Spectral Feedback.

Spectral Feedback further optimizes the stability reward and therefore can move samples toward regions where stability reward improves but structural consistency degrades. However, this degradation is likely a reflection of the DRAKES distribution in high-reward regions. We conclude that Spectral Feedback improves a specified alignment objective, while preservation of auxiliary properties depends on the quality of the reward and the distribution induced by the underlying model. In settings such as DRAKES, a regularized or multi-objective reward that explicitly includes structural consistency or correlated statistics would be needed to prevent this trade-off. We discuss using a multi-objective reward in Appendix F.2.

F Additional Scaling and Alignment Studies

F.1 Spectral Feedback Applications

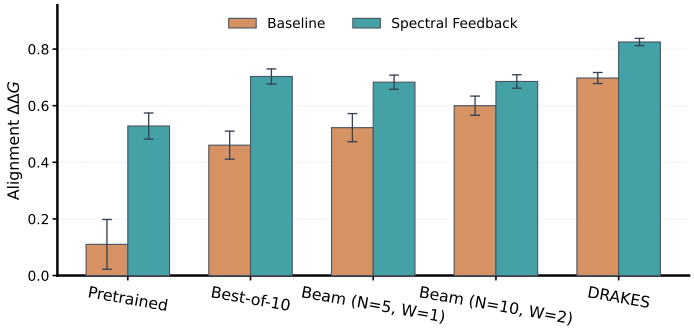


Figure 12: Spectral Feedback with $k = 20$, $D = 8192$, and 5 feedback iterations.

In general, feedback can use substantial compute resources when many samples are used to make edit-selections; however, it is useful when a target alignment model reaches insufficient reward values. Still, it is important to make each feedback step as efficient as possible and Figure 6 shows that Spectral Feedback with ProxySPEX scales well in alignment performance and latency when compared to competing edit-position selection methods.

F.2 Additional Alignment Oracles

Most of this paper focuses on aligning inverse protein folding diffusion models with a $\Delta\Delta G$ reward oracle, though it is important to study the flexibility of Spectral Feedback across not just different protein models but also different reward oracles. We study its performance with two additional alignment oracles: ProtGPT2 Log-Likelihood and a multi-objective reward that balances log-likelihood with $\Delta\Delta G$.

ProtGPT2 Log-Likelihood Oracle

The ProtGPT2 Log-Likelihood oracle computes log-likelihoods of diffusion model sample sequences by using the ProtGPT2 autoregressive language model. This has been done in previous works such as Gruver et al. [2023] and Cemri et al. [2024]. These log-likelihoods can be used as a measure of naturalness of protein sequences because ProtGPT2 was trained to capture the distribution of protein amino acid sequences [Ferruz et al., 2022]. The figure below shows Spectral Feedback applied to the pre-trained model with the ProtGPT2 oracle, using first-order LASSO as the edit-selection strategy.

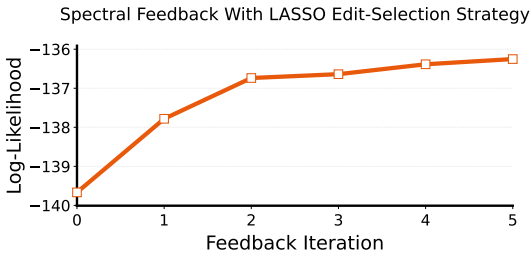


Figure 13: ProtGPT2 Log-Likelihood alignment trajectory with Spectral Feedback. Hyperparameters are $k = 10$, $D = 2048$, $n = 16$ (reward oracle calls per value function call).

Balanced Oracle

We define the multi-objective reward oracle as $r(x) = \alpha \cdot \Delta\Delta G(x) + 0.005 \cdot (1-\alpha) \cdot ProtGPT2(x)$. The weight α balances between aligning for $\Delta\Delta G$ and aligning for log-likelihood. The constant 0.005 rescales the ProtGPT2 oracle to make the choice of α more linear for weighting the importance of the two oracles. We show results for two proteins in the test set which demonstrate differing trade-offs. R6-560 has a direct trade-off between the oracles whereas 2KRU has higher log-likelihoods when primarily aligning for $\Delta\Delta G$.

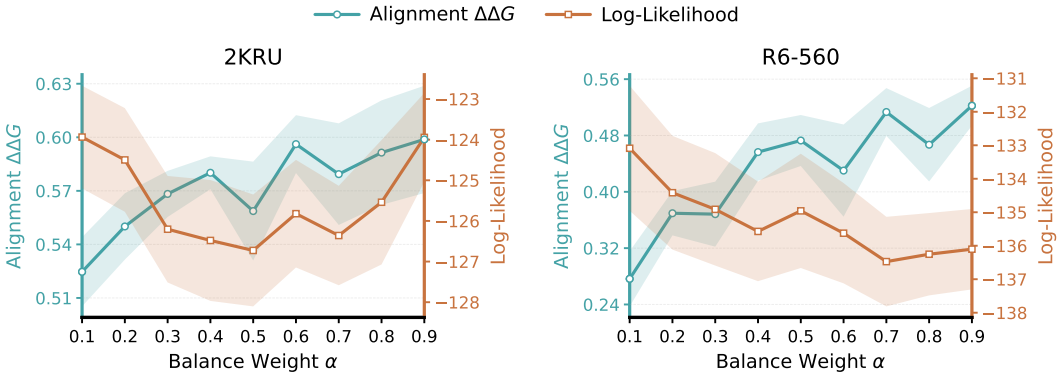


Figure 14: An example multi-objective oracle balances between $\Delta\Delta G$ and ProtGPT2 Log-Likelihood.

These results show that a multi-objective oracle can be useful for targeting and balancing between multiple objectives with Spectral Feedback. This can be useful as a form of regularization to avoid overfitting, which is typical for alignment methods with a maximizing objective. For example, one could make an oracle that balances between $\Delta\Delta G$ and $scRMSD$ to avoid the high $scRMSD$ proteins identified in Figure 11.

G High-Order Interactions Case Study ($\Delta\Delta G$ Alignment)

G.1 Comparing ProxySPEX with LASSO

In our work, we observed that the necessity of multivariate interactions for determining edit-positions is highly dependent on the target protein. While Spectral Feedback performs well when paired with ProxySPEX, a high-order sparse recovery method such as ProxySPEX may not always be necessary. We study how LASSO performs as a first-order sparse recovery method to better understand the impact of high-order sparse recovery. We first compare using ProxySPEX and LASSO for Spectral Feedback with two protein backbones: R6-560 and 2KVV. We run a single Spectral Feedback process for each backbone and initialize each process with the same corresponding protein sequences that were studied in Figure 3. We then compare ProxySPEX with LASSO across the entire test set.

In the first experiment, ProxySPEX had an 8.3% larger final reward than LASSO for *2KVV* and a 24% larger reward for *R6-560*. A possible explanation for these results lies in the structural differences

between the protein backbones. Figure 5 shows that the *2KVV* backbone contains notable α-helices while this structure is less prominent in *R6-560*. An α-helix is a secondary protein structure that is formed by structured local interactions between the i^{th} and $(i+4)^{th}$ amino acids in a protein sequence [Gupte et al., 2021]. The primarily local interactions that create this structure may then lead to less complex interactions between edit-positions. In Figure 3, for f_{avg}, *2KVV* has strong first-order interactions while *R6-560* has more significant high-order interactions that ProxySPEX can leverage. Differences in backbone structure may influence differences in the value function Fourier spectra, which then relate to the performance of ProxySPEX and LASSO for edit-selection. We show detailed results of the edit-selections and amino-acid updates for this experiment in Appendices G.2 and G.3.

In the second experiment, ProxySPEX slightly outperforms LASSO for the pre-trained model and Best-of-10. Additionally, LASSO slightly outperforms ProxySPEX for DRAKES. The close performance may be because most proteins in the test set have dominant first-order interactions for both value functions (see Appendix D.2). It would be interesting to see if the corresponding spectral profiles of the value functions for DRAKES are also more dominantly first-order, given the better performance of LASSO. It would also be interesting to find cases where high-order interactions are more dominant and where ProxySPEX would stand out more. Perhaps better performance could be reached in cases where the value function is deterministic since then larger R^2 values would be achievable and the sparse recovery from ProxySPEX would be more accurate.

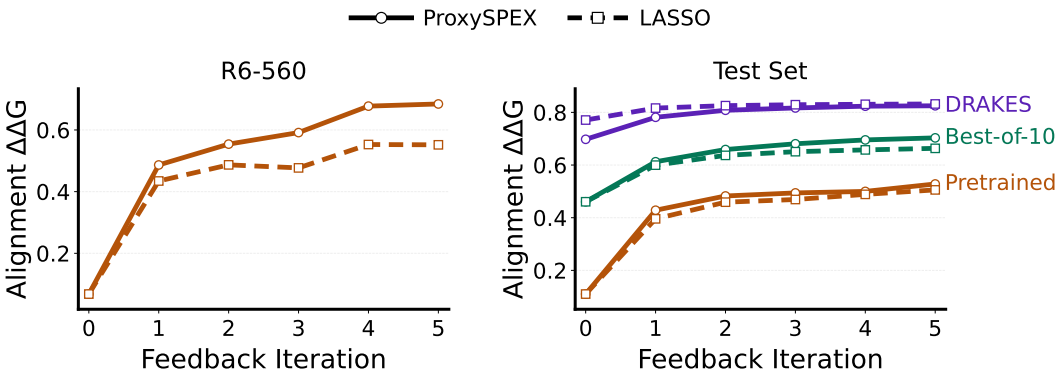


Figure 15: Spectral Feedback performance comparison with ProxySPEX vs first-order LASSO as the edit selection method. We constrain edit-selections to at most $k = 20$ positions. **(left)** Visualization of one held-out feedback trajectory for the R6-560 protein backbone with the pre-trained model. **(right)** Average Spectral Feedback alignment across the test set, comparing ProxySPEX and LASSO as the edit-selection strategies for the pre-trained model, Best-of-10, and DRAKES.

G.2 Target Protein: R6-560 (r6_560_TrROS_Hall)

Initial: SKPPKVVTVEVAVTKPDGKTELVKVTFTNLPRELKPGDTVTIPETGQKATVVKIIP

ProxySPEX - f_{avg}

Iteration 1

Targets: 4, 6, 9, 12, 15, 18, 19, 20, 24, 25, 26, 28, 34, 40, 41, 43, 48, 49, 50, 52

Changes: 4 (K → R), 9 (E → V), 15 (P → A), 26 (F → L), 41 (I → L), 50 (V → I), 52 (K → E)

Result: SKPPRVVTVVVAVTKADGKTELVKVTLTNLPRELKPGDTVTLPETGQKATIVEIIP

Iteration 2

Targets: 0, 1, 9, 13, 14, 17, 18, 19, 25, 27, 28, 32, 34, 36, 38, 43, 46, 47, 54

Changes: 0 (S → A), 1 (K → P), 9 (V → L), 14 (K → R), 28 (N → G), 34 (K → R), 38 (T → V), 47 (K → E), 54 (I → L)

Result: APPPRVVTVLVAVTRADGKTELVKVTLTGLPRELRPGDVVTLPETGQEATIVEILP

Iteration 3

Targets: 0, 8, 11, 12, 17, 18, 20, 22, 23, 28, 30, 36, 42, 43, 44, 45, 47, 49, 55

Changes: 18 (K → R), 22 (V → R), 23 (K → R), 47 (E → R)

Result: APPPRVVTVLVAVTRADGRTELRRVTLTGLPRELRPGDVVTLPETGQRATIVEILP

Iteration 4

Targets: 0, 7, 8, 13, 14, 16, 19, 20, 28, 29, 30, 34, 35, 38, 40, 45, 46, 47, 48, 54

Changes: 7 (T → E), 14 (R → E), 20 (E → V), 46 (Q → E), 47 (R → E)

Result: APPPRVVEVLVAVTEADGRTVLRRVTLTGLPRELRPGDVVTLPETGEEATIVEILP

Iteration 5

Targets: 0, 1, 6, 10, 16, 17, 24, 26, 29, 30, 31, 35, 36, 39, 42, 47, 51, 52, 54, 55

Changes: 47 (E → R)

Result: APPPRVVEVLVAVTEADGRTVLRRVTLTGLPRELRPGDVVTLPETGERATIVEILP

Reward Trajectory: [0.0679, 0.4864, 0.5541, 0.5915, 0.6773, 0.6843]

LASSO - f_{avg}

Iteration 1

Targets: 0, 1, 4, 6, 11, 13, 15, 18, 24, 25, 26, 27, 28, 29, 34, 38, 41, 48, 50, 52

Changes: 0 (S → A), 4 (K → R), 26 (F → L), 38 (T → V), 41 (I → L), 50 (V → I), 52 (K → R)

Result: AKPPRVVTVEVAVTKPDGKTELVKVTLTNLPRELKPGDVVTLPETGQKATIVRIIP

Iteration 2

Targets: 1, 2, 6, 10, 12, 13, 15, 18, 23, 24, 27, 28, 29, 34, 35, 36, 43, 44, 48

Changes: 1 (K → P), 15 (P → A), 23 (K → T), 27 (T → E), 28 (N → G)

Result: APPPRVVTVEVAVTKADGKTELVTVTLEGLPRELKPGDVVTLPETGQKATIVRIIP

Iteration 3

Targets: 6, 12, 14, 17, 18, 24, 27, 28, 29, 34, 35, 36, 43, 44, 48, 52

Changes: 14 (K → R), 27 (E → R), 28 (G → D), 29 (L → T), 34 (K → R)

Result: APPPRVVTVEVAVTRADGKTELVTVTLRDTPRELRPGDVVTLPETGQKATIVRIIP

Iteration 4

Targets: 1, 2, 5, 6, 8, 11, 12, 13, 18, 22, 24, 27, 29, 34, 36, 43, 44, 48, 52, 55

Changes: 1 (P → A), 27 (R → T), 29 (T → L), 34 (R → K)

Result: AAPPRVVTVEVAVTRADGKTELVTVTLTDLPRELKPGDVVTLPETGQKATIVRIIP

Iteration 5

Targets: 3, 6, 10, 12, 13, 17, 24, 29, 30, 34, 36, 43, 44, 45, 48, 49

Changes: 34 (K → R)

Result: AAPPRVVTVEVAVTRADGKTELVTVTLTDLPRELRPGDVVTLPETGQKATIVRIIP

Reward Trajectory: [0.0679, 0.4345, 0.4868, 0.4770, 0.5528, 0.5518]

G.3 Target Protein: 2KVV (v2K43S_2KVV)

Initial: EKWIEQNELMKETGLKRSTITKLRKTKLKEGEHYKRVSKDGKPSKDATILYNLEKIKKLLK

ProxySPEX - f_{avg}

Iteration 1

Targets: 1, 5, 6, 10, 19, 26, 28, 31, 34, 38, 41, 43, 44, 50, 53, 54, 56, 57, 59, 60

Changes: 1 (K → R), 5 (Q → E), 34 (K → R), 44 (K → P), 57 (K → E)

Result: ERWIEENELMKETGLKRSTITKLRKTKLKEGEHYRRVSKDGKPSPDATILYNLEKIKELLK

Iteration 2

Targets: 3, 6, 9, 10, 11, 15, 16, 21, 24, 26, 28, 31, 34, 41, 42, 48, 54, 55, 56, 60

Changes: 21 (K → R), 42 (P → D), 56 (K → L)

Result: ERWIEENELMKETGLKRSTITRLRKTKLKEGEHYRRVSKDGKDSPDATILYNLEKILELLK

Iteration 3

Targets: 3, 6, 9, 10, 11, 15, 16, 17, 24, 26, 28, 31, 40, 41, 48, 51, 53, 54, 55, 60

Changes: 6 (N → R), 10 (K → A), 11 (E → A), 15 (K → A), 24 (K → R), 26 (K → R), 28 (K → E), 31 (E → R), 41 (K → R), 54 (K → A), 60 (K → A)

Result: ERWIEERELMAATGLARSTITRLRRTRLEEGRHYRRVSKDGRDSPDATILYNLEAILELLA

Iteration 4

Targets: 1, 2, 8, 9, 14, 15, 19, 20, 21, 25, 26, 35, 36, 38, 47, 51, 52, 55, 57, 58

Changes: 15 (A → R), 20 (T → A), 26 (R → A), 38 (K → A), 47 (T → R), 57 (E → A)

Result: ERWIEERELMAATGLRRSTIARLRRTALEEGRHYRRVSADGRDSPDARILYNLEAILALLA

Iteration 5

Targets: 0, 2, 8, 13, 14, 15, 16, 19, 21, 24, 25, 29, 36, 39, 43, 49, 52, 53, 58, 60

Changes: 15 (R → A), 19 (I → L), 21 (R → A)

Result: ERWIEERELMAATGLARSTLAALRRTALEEGRHYRRVSADGRDSPDARILYNLEAILALLA

Reward Trajectory: [-0.1766, 0.3299, 0.3596, 0.5910, 0.6261, 0.6717]

LASSO - f_{avg}

Iteration 1

Targets: 5, 6, 10, 12, 19, 25, 28, 29, 31, 34, 38, 40, 41, 44, 50, 54, 55, 56, 57, 60

Changes: 5 (Q → E), 34 (K → R), 57 (K → E)

Result: EKWIEENELMKETGLKRSTITKLRKTKLKEGEHYRRVSKDGKPSKDATILYNLEKIKELLK

Iteration 2

Targets: 1, 6, 9, 10, 15, 17, 19, 24, 26, 28, 30, 31, 34, 38, 44, 54, 55, 56, 57, 60

Changes: 1 (K → R), 31 (E → I), 38 (K → A), 44 (K → P)

Result: ERWIEENELMKETGLKRSTITKLRKTKLKEGIHYRRVSADGKPSPDATILYNLEKIKELLK

Iteration 3

Targets: 3, 6, 9, 10, 15, 19, 21, 24, 26, 28, 29, 39, 43, 44, 54, 55, 56, 60

Changes: 6 (N → R), 15 (K → A), 21 (K → R), 24 (K → R), 54 (K → A), 60 (K → A)

Result: ERWIEERELMKETGLARSTITRLRRTKLKEGIHYRRVSADGKPSPDATILYNLEAIKELLA

Iteration 4

Targets: 0, 3, 9, 10, 17, 19, 22, 26, 28, 29, 42, 44, 55, 57

Changes: 26 (K → R), 42 (P → D), 57 (E → A)

Result: ERWIEERELMKETGLARSTITRLRRTRLKEGIHYRRVSADGKDSPDATILYNLEAIKALLA

Iteration 5

Targets: 0, 3, 9, 10, 17, 19, 28, 39, 42, 45, 55, 60

Changes: 10 (K → R)

Result: ERWIEERELMRETGLARSTITRLRRTRLKEGIHYRRVSADGKDSPDATILYNLEAIKALLA

Reward Trajectory: [-0.1766, 0.1106, 0.3755, 0.5710, 0.6129, 0.6203]