MULTI-OBJECTIVE PROTEIN OPTIMIZATION VIA MODULAR DIFFUSION STEERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Protein engineering often requires improving several properties simultaneously, motivating the search for high-performing sequences that represent useful fitness tradeoffs. Discovering these sequences is challenging because the design space is combinatorially large and experimental campaigns often permit only a few rounds of batched measurements. We introduce *MODS*, a modular framework for multi-objective protein optimization via inference-time diffusion steering. For each proposal, MODS samples augmented Chebyshev scalarization weights and a fitness-surrogate ensemble member to define a scalar steering reward. Varying the weights directs generation toward different objective tradeoffs, while varying ensemble members diversifies the predictions used for guidance. Because MODS expresses multi-objective acquisition through scalar rewards, it can be paired with different diffusion models and compatible steering procedures. After each evaluation batch, MODS updates the surrogates while keeping the generative model fixed. We evaluate MODS on six protein optimization tasks using predictive oracles trained on experimental assay data, with two diffusion models and two steering procedures. On every task, at least one MODS configuration substantially outperforms all evaluated baselines in final hypervolume, while the other configurations remain competitive. Our results demonstrate a flexible and effective approach to multi-objective protein optimization with limited feedback.

1 INTRODUCTION

Protein engineering often requires improving several properties simultaneously. Because improving one property can degrade another, the goal is not necessarily a single best sequence but a set of high-performing sequences representing useful fitness tradeoffs (Gruver et al., 2023; Luo et al., 2025). Multi-objective optimization formalizes this goal through the Pareto front, which represents solutions for which no objective can be improved without worsening at least one other. Finding these

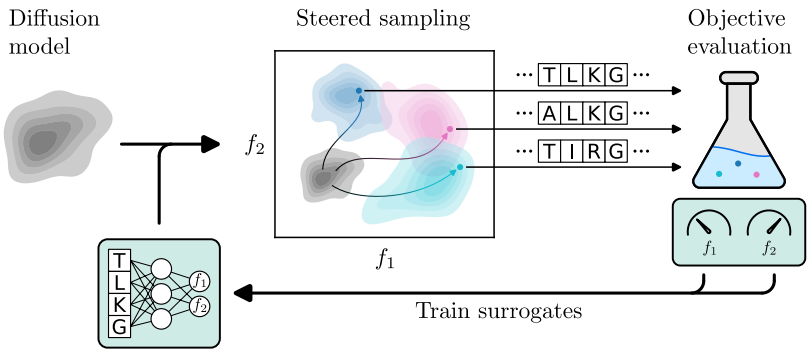


Figure 1: **Overview of MODS.** At each optimization round, MODS samples Chebyshev scalarization weights and a surrogate-ensemble member to construct rewards targeting different objective tradeoffs under predictive uncertainty. These rewards steer a fixed diffusion model to generate the next evaluation batch. The generated sequences are evaluated, and their fitness observations are used to update the surrogate ensemble for the following round.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

tradeoffs is difficult because the protein sequence space grows exponentially with sequence length, while experimental costs severely limit the number of variants that can be evaluated (Yang et al., 2019). Protein engineering campaigns therefore often permit only a few rounds of batched measurements, and each batch must be selected with incomplete knowledge of the fitness landscapes.

Generative protein models make this search more tractable by learning patterns from unlabeled sequences (Riesselman et al., 2018; Alamdari et al., 2024). The resulting generative models focus generation on plausible regions of sequence space but do not encode the properties required for a particular engineering task. For single-objective optimization, prior work has used task-specific fitness predictions to steer fixed generative models, refine samples at inference time, or adapt generators toward high-reward sequences (Yang et al., 2025a; Li et al., 2025; Uehara et al., 2025a; Wang et al., 2025). These approaches combine sequence knowledge with task-specific fitness signals.

Multi-objective generative optimization must produce candidates spanning a range of Pareto tradeoffs rather than concentrate generation on a single objective or fixed preference. Prior work has addressed this challenge by combining learned sequence representations with Bayesian optimization, guiding protein generation through hypervolume-based acquisition or gradients from multiple property predictors, and learning generative distributions over different objective preferences (Stanton et al., 2022; Gruver et al., 2023; Luo et al., 2025; Jain et al., 2023; Yuan et al., 2025; Annadani et al., 2025). These methods provide effective solutions but generally instantiate the acquisition strategy together with a particular generative model or optimization procedure. This motivates the need for a principled, modular approach to multi-objective generative optimization that can be applied to any compatible combination of generative model and steering mechanism.

We introduce *MODS*, an iterative, modular approach to multi-objective protein optimization through inference-time diffusion steering. MODS combines randomized augmented Chebyshev scalarizations with surrogate ensemble sampling. Scalarization decomposes the multi-objective problem into a family of scalar steering problems targeting different objective tradeoffs (Knowles, 2006; Paria et al., 2020). For each proposal, MODS samples both scalarization weights and a fitness-surrogate ensemble member. The weights determine the targeted tradeoff, while the ensemble member selects a plausible predicted fitness landscape. Varying both across proposals couples exploration over objective tradeoffs with exploration under predictive uncertainty. Because each pair defines a scalar reward, MODS separates multi-objective acquisition from the implementation of generation, allowing it to be paired with different diffusion models and compatible steering procedures.

Figure 1 illustrates the resulting optimization loop. Each campaign begins with an initial set of evaluated sequences. At every subsequent round, MODS trains a surrogate ensemble on all accumulated observations, constructs rewards from sampled scalarization weights and ensemble members, and uses those rewards to steer the fixed generative model. The generated sequences are evaluated to expand the dataset for the next round. Thus, new feedback progressively refines the task-specific steering without retraining the generative model.

We evaluate MODS on six protein optimization tasks, five with two objectives and one with three, using fixed predictive oracles trained on experimental assay data. Each simulated campaign begins with 100 evaluated sequences and acquires eight additional batches of 100. We instantiate MODS with two diffusion models and two steering procedures, compare it with alternative optimization methods, and examine its sensitivity to the diffusion model, steering procedure, and steering strength. On every task, at least one MODS configuration substantially outperforms all evaluated baselines in final hypervolume, while the other configurations remain competitive.

Our main contributions are as follows:

- **A modular framework for multi-objective generative optimization:** MODS represents acquisition under competing objectives and uncertain feedback as a mixture of reward-tilted sequence distributions, defined by scalarization weights and surrogate-ensemble members. Because each component is specified through a scalar reward, the framework can combine different pretrained diffusion models with compatible steering procedures.
- **A systematic evaluation under limited feedback:** Across six assay-derived oracle benchmarks, MODS consistently delivers substantial gains in final hypervolume over the evaluated baselines. Experiments with two diffusion models and two steering procedures further characterize sensitivity to the diffusion model, steering procedure, and steering strength.

2 RELATED WORK

Protein generative models and steering. Deep generative models learn patterns of variation from protein sequences and thereby provide priors that concentrate generation on plausible regions of sequence space (Riesselman et al., 2018; Alamdari et al., 2024). These generative models can be directed toward task-specific properties using gradients, value estimates, or iterative refinement during sampling (Uehara et al., 2025b; Li et al., 2025; Uehara et al., 2025a). Related approaches instead adapt the generative model itself by optimizing it with respect to a learned reward (Wang et al., 2025). MODS adopts the inference-time paradigm: it updates task-specific fitness surrogates as feedback accumulates while keeping the generative model fixed.

Protein optimization with limited feedback. Feedback-efficient protein optimization commonly alternates between surrogate-guided batch selection and experimental measurements or prioritizes variants near known functional sequences (Yang et al., 2025b; Ren et al., 2022). These approaches typically operate in localized mutation spaces, considering either a small set of mutable positions or variants that are separated from known sequences by relatively few mutations. Generative protein models help extend optimization to larger combinatorial spaces by using learned models to focus search on plausible proteins while permitting coordinated changes across many positions. Under limited feedback, prior work uses surrogate ensembles to steer a diffusion model toward high-fitness sequences for single-objective design (Yang et al., 2025a). MODS builds on this prior work while addressing the additional challenges posed by multiple, potentially competing objectives.

Multi-objective acquisition strategies. Multi-objective acquisition must promote high objective values while covering useful tradeoffs. Hypervolume-based methods address both goals by evaluating the expected increase in dominated hypervolume (Daulton et al., 2020). Scalarization takes a different approach, converting the multi-objective problem into a family of scalar optimization problems. Randomized scalarizations have been studied extensively in Bayesian optimization, including in combination with uncertainty-aware acquisition (Knowles, 2006; Paria et al., 2020). Chebyshev scalarizations are particularly useful because they can recover tradeoffs on nonconvex Pareto fronts that weighted sums cannot (Miettinen, 1998; Lin et al., 2024). In MODS, the scalarization weights select an objective tradeoff, while the sampled surrogate member selects a predicted fitness landscape. Because both choices enter only through the scalar reward, they can vary between proposals and optimization rounds while the generative model remains fixed.

Multi-objective generative protein optimization. Multi-objective protein-design methods have combined learned sequence representations with Bayesian optimization, steered discrete diffusion with hypervolume-based acquisition, and composed gradients from fixed property predictors to guide mutation sampling toward Pareto-optimal sequences (Stanton et al., 2022; Gruver et al., 2023; Luo et al., 2025). Other studies investigate multi-property steering, tree-guided peptide generation, and iterative adaptation of protein generators (Xiong et al., 2026; Tang et al., 2025; Lim et al., 2025). More generally, reward-conditioned generative models can produce solutions spanning different objective tradeoffs, including in sequential evaluation settings (Jain et al., 2023; Steinberg et al., 2025). Guided flows and diffusion models have also been developed for offline multi-objective optimization with fixed datasets (Yuan et al., 2025; Annadani et al., 2025; Shrestha et al., 2026). Like prior online approaches, MODS performs adaptive acquisition under limited feedback, updating its surrogate ensemble after each evaluation batch. A key contribution of MODS is its modular design, which allows its multi-objective acquisition strategy to be paired with different diffusion models and compatible steering mechanisms.

3 PROBLEM SETUP

We consider a task-specific protein design problem over sequences of fixed length L and an amino acid alphabet $\mathcal{A}$. The sequence space is $\mathcal{X} = \mathcal{A}^L$. Each sequence $x \in \mathcal{X}$ has $M \geq 2$ unknown fitness objectives, summarized by

$$\mathbf{f}(x) = \big(f_1(x), \ldots, f_M(x)\big) \in \mathbb{R}^M,$$

and all objectives are to be maximized. Because fitness evaluations are expensive, the optimizer can query only a limited number of sequences in a small number of batched rounds.

Because the objectives may conflict, no single sequence needs to maximize all of them. We therefore seek Pareto-optimal tradeoffs. A sequence x dominates x', written $x \succ_{\mathbf{f}} x'$, if $f_m(x) \geq f_m(x')$ for every objective m, with strict inequality for at least one. The Pareto set consists of the sequences that are not dominated by any other sequence:

$$\mathcal{P}^\star = \{x \in \mathcal{X} \mid \nexists\, x' \in \mathcal{X} : x' \succ_{\mathbf{f}} x\}\,.$$

The corresponding Pareto front is the image of this set in objective space, $\mathcal{F}^\star = \{\mathbf{f}(x) \mid x \in \mathcal{P}^\star\}$.

To guide the search beyond the available fitness observations, we assume access to a pretrained diffusion model p_0, trained independently on unlabeled protein sequences. The diffusion model provides a distribution over sequence space and remains fixed throughout optimization.

Optimization begins with an initial evaluated dataset $\mathcal{D}_0$. At each round $t = 1, \ldots, T$, the optimizer uses the diffusion model and accumulated observations in $\mathcal{D}_{t-1}$ to generate a batch $\mathcal{B}_t$ of B sequences. Evaluating a sequence $x \in \mathcal{B}_t$ returns a possibly noisy observation $\mathbf{y}_x$ of its fitness vector $\mathbf{f}(x)$. After evaluating the batch, the dataset becomes

$$\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{(x, \mathbf{y}_x) \mid x \in \mathcal{B}_t\}.$$

After T rounds, the campaign has a total budget of $|\mathcal{D}_0| + TB$ sequence evaluations. The goal is to use the pretrained diffusion model and limited feedback to identify evaluated sequences whose fitness vectors approach $\mathcal{F}^\star$ while representing diverse objective tradeoffs.

4 The MODS Framework

MODS is an iterative, modular framework for multi-objective protein optimization via inference-time diffusion steering. At each round, it fits an ensemble of fitness surrogates and applies randomized augmented Chebyshev scalarizations to sampled surrogate predictions, producing rewards that target different objective tradeoffs. Applied to the same fixed generative model, these rewards direct generation toward different regions of objective space, as illustrated in Figure 2. The resulting sequences form the evaluation batch, and their observed fitness values are used to update the surrogate ensemble for the next round.

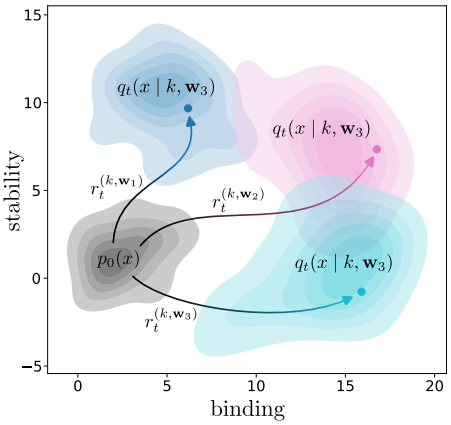


Figure 2: **Illustration of multi-objective steering in MODS.** Shown are the diffusion model p_0 and three steered distributions from the sixth optimization round for the GB1 problem. To isolate the effect of scalarization weights, all three distributions use the same surrogate $\widehat{\mathbf{f}}_6^{(k)}$ but different weight vectors $\mathbf{w}_1$, $\mathbf{w}_2$, and $\mathbf{w}_3$, thereby targeting different regions of objective space.

4.1 Surrogates and scalarized rewards

Our goal is to generate sequences spanning different Pareto-optimal tradeoffs. The steering procedures considered here assign a scalar reward to each candidate, whereas the fitness surrogates predict a vector of objective values. Scalarization connects these two representations by converting the vector-valued predictions into a parametric family of per-sequence scalar rewards. Each weight vector specifies an objective tradeoff and thus defines a reward targeting a different region of the Pareto front. By varying the weights across candidates, MODS can pursue multiple tradeoffs in parallel.

At round t, we train an ensemble of K fitness surrogates $\{\widehat{\mathbf{f}}_t^{(k)}\}_{k=1}^K$ on the observations in $\mathcal{D}_{t-1}$. Each ensemble member is a small multilayer perceptron that predicts all M objectives, enabling repeated fitness prediction during generation without additional objective evaluations. Variation across ensemble predictions serves as a practical proxy for predictive uncertainty (Yang et al., 2025a). Architecture and training details are provided in Appendix D.

Before scalarization, we normalize each predicted objective using the smallest and largest values of that objective observed in $\mathcal{D}_{t-1}$. For a normalized prediction vector $\mathbf{v} \in \mathbb{R}^M$ and a positive weight

vector $\mathbf{w} \in \mathbb{R}^M$, we use the augmented Chebyshev scalarization

$$s_{\mathbf{w}}(\mathbf{v}) = \min_{m=1,\dots,M} w_m v_m + \rho \sum_{m=1}^{M} w_m v_m, \tag{1}$$

where $w_m > 0$, $\sum_{m=1}^{M} w_m = 1$, and $\rho > 0$ is a small augmentation coefficient. The minimum term emphasizes the smallest weighted objective, while the augmentation term also rewards improvements in the remaining objectives. Changing $\mathbf{w}$ changes the favored tradeoff. Unlike linear scalarizations, augmented Chebyshev scalarizations can recover optimal tradeoffs on nonconvex Pareto fronts (Miettinen, 1998).

We sample a surrogate index $k \sim \mathrm{Uniform}(\{1,\dots,K\})$ and scalarization weights $\mathbf{w} \sim \mathrm{Dirichlet}(\mathbf{1})$, and define the steering reward

$$r_t^{(k,\mathbf{w})}(x) = s_{\mathbf{w}}(\widehat{\mathbf{f}}_t^{(k)}(x)) \tag{2}$$

The weights specify the targeted objective tradeoff, while the sampled ensemble member selects one plausible predicted fitness landscape. Varying both across the evaluation batch promotes coverage of different tradeoffs and exploration under surrogate uncertainty.

This construction fits naturally with inference-time steering. For any fixed weight vector and surrogate prediction, the resulting scalar reward can be supplied to a compatible single-objective steering procedure without modifying or retraining the diffusion model. Changing the weight vector redirects generation toward a different tradeoff while reusing the same generative model and steering mechanism. Scalarization thus provides a modular interface between multi-objective acquisition and single-objective generative steering.

4.2 Generating candidates with diffusion steering

For a fixed pair $(k, \mathbf{w})$, MODS seeks a sampling distribution that attains high expected reward $r_t^{(k,\mathbf{w})}$ while remaining close to the pretrained diffusion model. This tradeoff is expressed by the standard KL-regularized objective (Rafailov et al., 2023):

$$q_t(\cdot \mid k, \mathbf{w}) = \arg\max_{q \ll p_0} \Big(\mathbb{E}_{x\sim q}\big(r_t^{(k,\mathbf{w})}(x)\big) - \alpha^{-1}\,\mathrm{KL}(q \,\|\, p_0) \Big). \tag{3}$$

The optimizer has the analytic form

$$q_t(x \mid k, \mathbf{w}) = \frac{p_0(x)\exp\big(\alpha r_t^{(k,\mathbf{w})}(x)\big)}{Z_t^{(k,\mathbf{w})}}, \tag{4}$$

where $Z_t^{(k,\mathbf{w})}$ is the normalizing constant. Larger $\alpha > 0$ places greater emphasis on predicted reward, whereas smaller values keep the distribution closer to the diffusion model.

MODS uses inference-time diffusion steering (Uehara et al., 2025b) to approximately sample from these reward-tilted distributions. Sampling different pairs $(k, \mathbf{w})$ across proposals yields the mixture $q_t(x) = \mathbb{E}_{k,\mathbf{w}}[q_t(x \mid k, \mathbf{w})]$, whose components target different objective tradeoffs under different surrogate predictions. The sampled pair remains fixed throughout a candidate's diffusion trajectory.

We instantiate this procedure with classifier guidance (CG) (Dhariwal & Nichol, 2021) and decoupled annealing posterior sampling (DAPS) (Zhang et al., 2025). For CG, the surrogates predict fitness from corrupted sequences together with their noise levels; for DAPS, they are trained on clean sequences. For both steering procedures, the surrogates are trained only on fitness observations collected during the campaign rounds.

We generate the evaluation batch $\mathcal{B}_t$ in computational minibatches. Each minibatch shares a sampled surrogate member, while each candidate receives its own scalarization weights. This structure enables efficient batched surrogate evaluation. The surrogate member and weights remain fixed throughout each candidate's diffusion trajectory.

4.3 The complete MODS procedure

We form the initial dataset $\mathcal{D}_0$ by sampling n_0 sequences from the diffusion model without steering and evaluating their fitness. Each subsequent round consists of three steps, as illustrated in Figure 1:

1. **Fit the surrogate ensemble.** Train fitness surrogates on all observations in $\mathcal{D}_{t-1}$, obtaining updated estimates of the objectives and their uncertainty.
2. **Generate the evaluation batch.** Sample surrogate members and scalarization weights to construct steering rewards $r_t^{(k,\mathbf{w})}$. Use these rewards to steer the diffusion model and generate an evaluation batch $\mathcal{B}_t$ of B sequences.
3. **Evaluate and update.** Evaluate all objectives for the sampled sequences and update $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{(x, \mathbf{y}_x) : x \in \mathcal{B}_t\}$, where $\mathbf{y}_x$ denotes the possibly noisy objective observation.

After T rounds, the campaign has used $n_0 + TB$ evaluations. We return the evaluated sequences that are nondominated according to their observed objective vectors.

5 Experiments

We evaluate MODS on six protein optimization tasks. We compare our framework with three iterative optimization baselines and study how performance depends on the diffusion model, steering procedure, and steering strength.

5.1 Experimental setup

Tasks and evaluation oracles. The six tasks span different proteins and fitness objectives: five have two objectives, and one has three, as summarized in Table A.1. To evaluate sequences without expensive laboratory measurements, we train an oracle for each objective using existing experimental assay data. Each fixed oracle averages the predictions of a CNN ensemble and applies a task-scaled penalty to sequences beyond the labeled mutational neighborhood. Appendix A describes the datasets and their preprocessing. Appendix C describes model architecture, reports held-out predictive accuracy and defines the distance penalty.

MODS configurations. We train two diffusion models, MDLM (Sahoo et al., 2024) and D3PM (Austin et al., 2021), on a dataset of unlabeled sequences. We pair each diffusion model with classifier guidance (CG) and decoupled annealing posterior sampling (DAPS), yielding four MODS configurations. At each round, each configuration trains an ensemble of multilayer perceptron surrogates using only observations acquired during the campaign, without access to the oracle training data. Appendices B and D provide details on the diffusion models and surrogates, respectively.

Baselines. We compare MODS against three alternative approaches for iterative protein optimization. ARLM + DPO fine-tunes an autoregressive model based on ProGen2-small (Nijkamp et al., 2023) using direct preference optimization (Rafailov et al., 2023). Its preference pairs are constructed from observed fitness values, and a KL penalty regularizes the model toward a fixed reference. LaMBO-2 (Gruver et al., 2023) couples a diffusion model and fitness predictors through a jointly trained, shared encoder. It uses diffusion optimized sampling (NOS) to guide generation with expected hypervolume improvement (Daulton et al., 2020) and saliency maps to select positions for mutation. SAGE-Prot (Lim et al., 2025) generates candidate pools with pretrained LSTM and transformer-decoder models, diversifies them through mutation, crossover, and homologous search, and fine-tunes the generator on high-fitness sequences. Unlike these baselines, MODS keeps the generative model fixed throughout optimization. Only its fitness surrogates are updated from new observations, and unlike LaMBO-2, these surrogates are decoupled from the generative model.

Optimization protocol and metrics. Each campaign begins with $n_0 = 100$ sequences sampled from the generative model, followed by eight rounds of 100 additional evaluations. All methods therefore receive the same budget of 900 sequences, with every objective evaluated for every sequence. We repeat each configuration five times and report means and standard errors. We measure performance by the hypervolume of all sequences evaluated up to each round. For each task, the reference point is the component-wise minimum across all sequences evaluated by all methods.

Hyperparameter selection. We sweep the steering parameters for CG and DAPS and the adaptation coefficient for ARLM + DPO. For every task, the best DAPS setting achieves a higher final mean hypervolume than the best CG setting. We therefore show the two DAPS variants in the main

Table 1: **Hypervolume across six protein tasks.** Entries report mean ± one standard error of the mean over five repetitions. Bold identifies the highest mean among all the shown methods for each problem.

Method	GB1	ERBIN PDZ1	NHERF3 PDZ2	KRAS	Src	NHERF3 PDZ1
MDLM (unsteered)	231.6 ± 11.4	231.4 ± 2.0	145.4 ± 5.0	70.0 ± 6.1	64.0 ± 2.4	2244.8 ± 78.8
D3PM (unsteered)	225.0 ± 4.8	221.0 ± 5.5	95.7 ± 3.4	40.7 ± 2.1	44.7 ± 1.0	1004.0 ± 28.4
MODS (MDLM + DAPS)	685.3 ± 10.4	391.4 ± 3.7	254.9 ± 5.3	**128.2** ± 3.6	66.4 ± 2.8	**4403.5** ± 97.0
MODS (D3PM + DAPS)	**781.4** ± 5.9	**413.8** ± 4.2	**271.5** ± 4.1	106.5 ± 2.3	**85.2** ± 0.3	4278.9 ± 72.5
ARLM + DPO	328.4 ± 6.8	252.4 ± 5.3	165.7 ± 2.1	98.5 ± 2.5	67.5 ± 2.4	2272.7 ± 39.7
SAGE-Prot	210.8 ± 2.5	230.5 ± 2.6	130.2 ± 9.1	95.6 ± 1.3	66.2 ± 3.2	1960.6 ± 126.3
LaMBO-2	234.2 ± 13.5	216.2 ± 12.8	121.4 ± 6.4	44.1 ± 2.2	45.7 ± 1.2	1296.8 ± 136.3

comparisons. For MODS (MDLM + DAPS), the main results report the results for $\alpha = 2^{12}$ and for MODS (MDLM + DAPS) for $\alpha = 2^{14}$, both of which performed well across tasks. For ARLM + DPO, we report the results for the hyperparameters that achieved the highest hypervolume on each task. Figures E.1 and E.2 and Appendices E and F report complete results for both steering procedures and all tested steering strengths.

5.2 Results

Overall optimization performance. Figure 3 shows the hypervolume of evaluated sequences over the optimization rounds. On every task, at least one MODS configuration achieves the highest final mean hypervolume among the evaluated methods. On GB1, ERBIN-PDZ1, NHERF3-PDZ2, KRAS, and NHERF3-PDZ1, both DAPS variants achieve higher final mean hypervolume than ARLM + DPO, LaMBO-2, and SAGE-Prot. On Src, MODS with D3PM + DAPS achieves the highest final mean hypervolume, while MDLM + DAPS achieves 66.4 ± 2.8, compared with 67.5 ± 2.4 for ARLM + DPO, 66.2 ± 3.2 for SAGE-Prot, and 45.7 ± 1.2 for LaMBO-2. Table F.1 reports Holm-adjusted p-values from Welch's t-tests where all reported comparisons have adjusted $p < 0.05$. Table 1 also shows that each DAPS variant improves over unsteered sampling from the same diffusion model under the common evaluation budget. Together, these results show that the gains from MODS hold across both diffusion models, although the better model depends on the task.

Coverage of objective tradeoffs. For the five two-objective tasks, Figure 4 provides a qualitative view of the final tradeoffs recovered by each method. The three-objective NHERF3-PDZ1 task is omitted because its Pareto front cannot be represented by a single two-dimensional plot. The differences are most visible on GB1, ERBIN-PDZ1, KRAS, and Src, where the highest-hypervolume fronts extend farther along complementary objective directions.

Effect of steering strength. Figures E.1 and E.2 show how the KL-regularization coefficient α^{-1} in Equation 3 affects optimization performance. Reducing α^{-1} strengthens steering by allowing the sampling distributions q_t to deviate more from the predtrained diffusion model. With weak regularization, sampling relies more heavily on surrogate predictions far from the observed data, where those predictions may be unreliable. The resulting performance decline in some settings is consistent with oversteering, although we do not directly test this mechanism. For the MDLM model, performance peaks at an intermediate steering strength on KRAS and Src with DAPS and on NHERF3-PDZ2 and KRAS with CG. Further strengthening steering reduces final hypervolume in these settings. In most other diffusion model–task combinations, stronger steering improves performance or eventually reaches a plateau. These results show that the appropriate steering strength depends jointly on the prior, steering procedure, and task.

6 Conclusion

We introduced MODS, a modular framework for multi-objective protein optimization with limited feedback. MODS samples augmented Chebyshev scalarization weights and surrogate ensemble

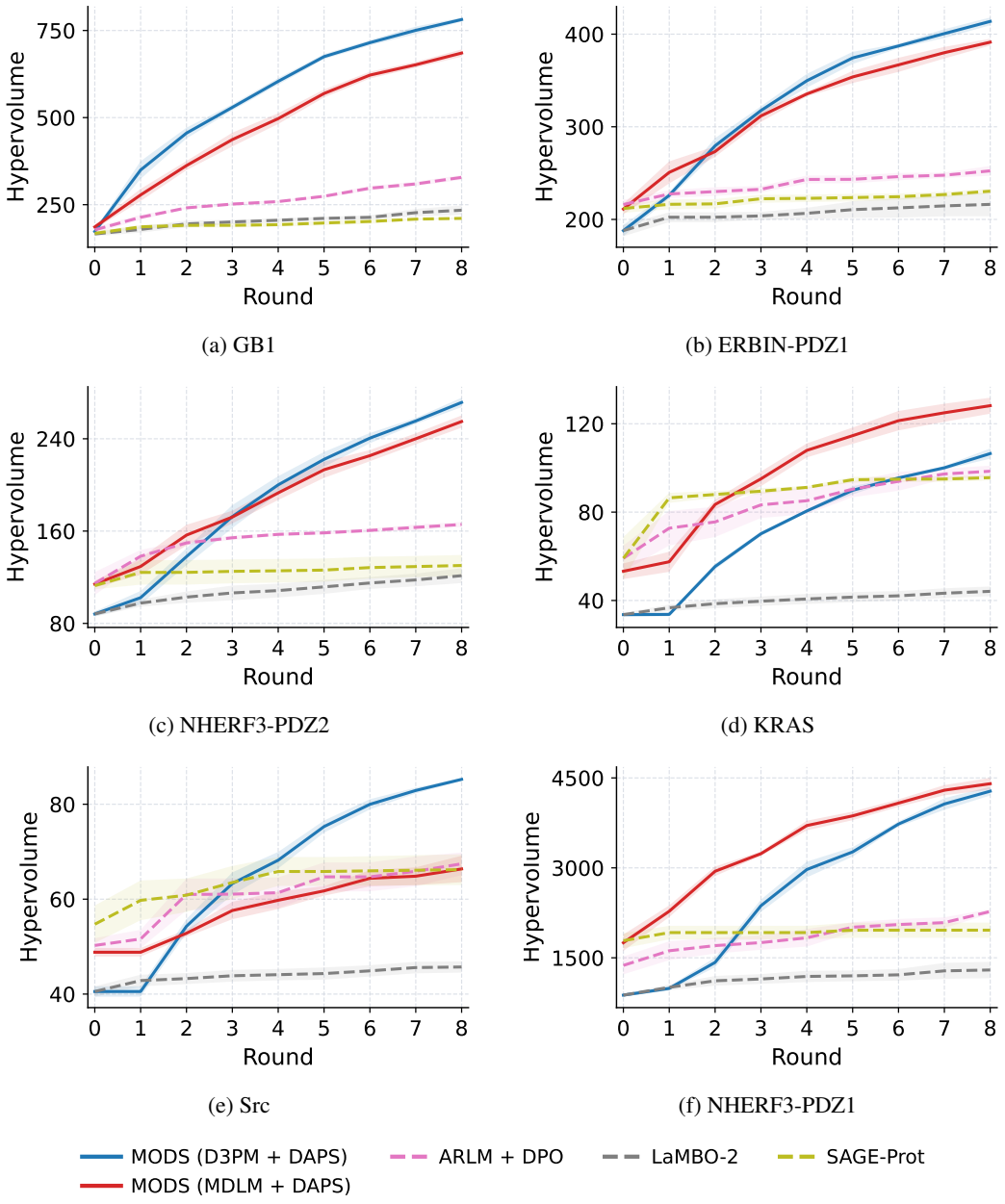


Figure 3: **Hypervolume over optimization rounds.** Hypervolume over all evaluated sequences up to each round for each task. Lines show means; shaded regions indicate $\pm$ one standard error of the mean over five repetitions.

members to construct rewards targeting different objective tradeoffs. These rewards provide a modular interface between multi-objective acquisition and generative steering: they can be used with compatible inference-time steering procedures without modifying or retraining the generative model. As new observations are collected, MODS updates the fitness surrogates and redirects generation while keeping the generative model fixed.

We evaluated MODS with two diffusion models and two steering procedures across six protein optimization tasks. On every task, at least one MODS configuration achieved the highest mean final hypervolume among the evaluated methods. MODS performed strongly with both diffusion models, although performance depended on the choice of generative model and steering procedure. These

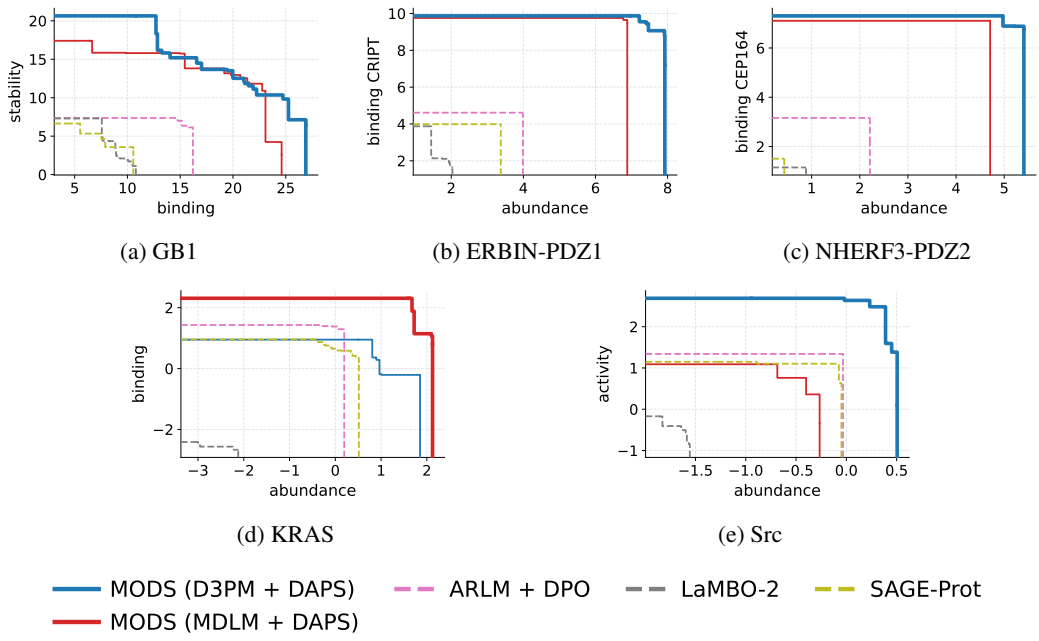


Figure 4: **Final Pareto fronts.** Each panel shows the nondominated objective vectors after the final evaluation round. For each method, we plot the repetition with median final hypervolume; the front with the largest hypervolume is highlighted in bold.

results show that MODS can discover strong and diverse objective tradeoffs within a few rounds of batched feedback while supporting different model–steering combinations.

Scope and limitations. Our evaluation uses predictive oracles trained and validated on experimental assay data, enabling controlled comparisons across tasks and methods. The oracles include a distance-based penalty to reduce unreliable extrapolation beyond the labeled mutational neighborhoods, but laboratory studies remain necessary to determine whether the observed gains translate to experimentally measured fitness. We evaluated MODS on a limited set of protein tasks, diffusion priors, and steering procedures. Experiments spanning additional protein families, generative models, and steering procedures would further establish the framework's generality.

Future work. Beyond laboratory validation, a natural next step is to combine feedback at different fidelities, such as inexpensive computational predictions, screening assays, and more costly measurements. Doing so would require deciding both which sequences to evaluate and at what fidelity. Another promising direction is to transfer fitness information across related proteins, targets, or assay conditions. Extending reuse from generative sequence models to task-specific fitness models could further reduce the number of measurements required for new optimization campaigns.

AI USE STATEMENT

In this work, we used generative AI tools for providing feedback on research methodology or experiments, and assisting with translation. We have not used generative AI tools to generate synthetic data sets, help develop theoretical models or conceptual frameworks, propose or refine hypotheses, implement methods, clean and reformat dataset, support qualitative and thematic data analysis, interpret results, and formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs are not applicable to this work. Additionally, we used generative AI tools for editing non-methodological parts of software code, sourcing/searching for information, and editing the research paper to improve readability. We have reviewed all AI-assisted work. We checked, verified and tested all LLM-generated outputs including text, code, summaries, analysis and edits before incorporating them into work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Ethics statement

Our study does not involve human subjects, personally identifiable information, or sensitive personal data. All datasets used are publicly available, and we comply with their original licenses and terms of use. The pretrained generative-prior and fitness-oracle checkpoints are trained solely on these datasets, no private or proprietary user data were used. Like other generative-modeling research, it may inherit general risks associated with the generation of misleading, biased, or otherwise harmful content. Because our work involves protein fitness prediction and generative design, we considered biosafety and dual-use concerns. We do not intentionally generate or release protein sequences with known harmful function. This work is intended for methodological research on optimization with learned generative models and oracles.

Reproducibility statement

Dataset sources and processing, model architectures, optimization settings, evaluation protocols, checkpoint selection, and additional details are documented in Section 5.1 and Appendices A, B, C, and D. We intend to release the source code, processed datasets, data splits, pretrained generative-prior and fitness-oracle checkpoints, experiment configurations, random seeds, environment setup instructions, saved experimental outputs, and analysis scripts that support regeneration of the reported tables and figures. At the time of submission, these are not yet publicly available. An anonymized reproducibility package will be made available to reviewers during the rebuttal.

References

Sarah Alamdari, Nitya Thakkar, Rianne van den Berg, Neil Tenenholtz, Robert Strome, Alan M. Moses, Alex Lu, Nicolo Fusi, Ava P. Amini, and Kevin Kaichuang Yang. Protein generation with evolutionary diffusion: sequence is all you need. *bioRxiv*, 2024. doi: 10.1101/2023.09.11.556673. URL https://www.biorxiv.org/content/10.1101/2023.09.11.556673v2.

Yashas Annadani, Syrine Belakaria, Stefano Ermon, Stefan Bauer, and Barbara Engelhardt. Preference-guided diffusion for multi-objective offline optimization. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 15738–15761. Curran Associates, Inc., 2025. doi: 10.52202/085713-0533. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/175fa4fc8f275f877ec85340131c5d7a-Paper-Conference.pdf.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 17981–17993. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/958c530554f78bcd8e97125b70e6973d-Paper.pdf.

Antoni Beltran, Mohsin M. Naqvi, Andre J. Faure, and Ben Lehner. The allosteric landscape of the src kinase. *Science Advances*, 12(7):eaea2726, 2026. doi: 10.1126/sciadv.aea2726. URL https://www.science.org/doi/abs/10.1126/sciadv.aea2726.

Samuel Daulton, Maximilian Balandat, and Eytan Bakshy. Differentiable expected hypervolume improvement for parallel multi-objective bayesian optimization. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 9851–9864. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/6fec24eac8f18ed793f5eaad3dd7977c-Paper.pdf.

Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. *CoRR*, abs/2105.05233, 2021. URL https://arxiv.org/abs/2105.05233.

Chase R. Freschlin, Sarah A. Fahlberg, Pete Heinzelman, and Philip A. Romero. Neural network extrapolation to distant regions of the protein fitness landscape. *Nature Communications*, 15:6405,

2024. doi: 10.1038/s41467-024-50712-3. URL https://www.nature.com/articles/s41467-024-50712-3.

Nate Gruver, Samuel Stanton, Nathan Frey, Tim G. J. Rudner, Isidro Hotzel, Julien Lafrance-Vanasse, Arvind Rajpal, Kyunghyun Cho, and Andrew G Wilson. Protein design with guided discrete diffusion. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 12489–12517. Curran Associates, Inc., 2023. doi: 10.52202/075280-0547. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/29591f355702c3f4436991335784b503-Paper-Conference.pdf.

Moksh Jain, Sharath Chandra Raparthy, Alex Hernández-García, Jarrid Rector-Brooks, Yoshua Bengio, Santiago Miret, and Emmanuel Bengio. Multi-objective GFlowNets. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 14631–14653. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/jain23a.html.

L. Steven Johnson, Sean R. Eddy, and Elon Portugaly. Hidden Markov model speed heuristic and iterative HMM search procedure. *BMC Bioinformatics*, 11(1):431, aug 2010. doi: 10.1186/1471-2105-11-431. URL https://doi.org/10.1186/1471-2105-11-431.

J. Knowles. Parego: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1):50–66, 2006. doi: 10.1109/TEVC.2005.851274.

Xiner Li, Yulai Zhao, Chenyu Wang, Gabriele Scalia, Gokcen Eraslan, Surag Nair, Tommaso Biancalani, Shuiwang Ji, Aviv Regev, Sergey Levine, and Masatoshi Uehara. Derivative-free guidance in continuous and discrete diffusion models with soft value-based decoding. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 95507–95545. Curran Associates, Inc., 2025. doi: 10.52202/085713-3194. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/899af0d66d8850318a20781484416152-Paper-Conference.pdf.

Hocheol Lim, Geon-Ho Lee, Hyein Cho, and Kyoung Tai No. SAGE-Prot: scoring-assisted generative exploration for multi-objective protein design. *Briefings in Bioinformatics*, 26(6):bbaf585, 2025. doi: 10.1093/bib/bbaf585. URL https://doi.org/10.1093/bib/bbaf585.

Xi Lin, Xiaoyuan Zhang, Zhiyuan Yang, Fei Liu, Zhenkun Wang, and Qingfu Zhang. Smooth tchebycheff scalarization for multi-objective optimization. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 30479–30509. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/lin24y.html.

Jiaqi Luo, Kerr Ding, and Yunan Luo. Pareto-optimal sampling for multi-objective protein sequence design. *iScience*, 28(3):112119, 2025. ISSN 2589-0042. doi: https://doi.org/10.1016/j.isci.2025.112119. URL https://www.sciencedirect.com/science/article/pii/S2589004225003797.

Aina Martí-Aranda and Ben Lehner. Seven complete comparative maps of allosteric mutations in a protein family. *Nature Communications*, 17(1):5487, 2026. doi: 10.1038/s41467-026-71005-x. URL https://www.nature.com/articles/s41467-026-71005-x.

Kaisa Miettinen. *Nonlinear Multiobjective Optimization*, volume 12 of *International Series in Operations Research & Management Science*. Springer, New York, NY, 1998. doi: 10.1007/978-1-4615-5563-6. URL https://link.springer.com/book/10.1007/978-1-4615-5563-6.

Erik Nijkamp, Jeffrey A. Ruffolo, Eli N. Weinstein, Nikhil Naik, and Ali Madani. Progen2: Exploring the boundaries of protein language models. *Cell Systems*, 14(11):968–978.e3, 2023.

ISSN 2405-4712. doi: https://doi.org/10.1016/j.cels.2023.10.002. URL https://www.sciencedirect.com/science/article/pii/S2405471223002727.

C. Anders Olson, Nicholas C. Wu, and Ren Sun. A comprehensive biophysical description of pairwise epistasis throughout an entire protein domain. *Current Biology*, 24(22):2643–2651, 2014. ISSN 0960-9822. doi: https://doi.org/10.1016/j.cub.2014.09.072. URL https://www.sciencedirect.com/science/article/pii/S0960982214012688.

Biswajit Paria, Kirthevasan Kandasamy, and Barnabás Póczos. A flexible framework for multi-objective bayesian optimization using random scalarizations. In Ryan P. Adams and Vibhav Gogate (eds.), *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*, volume 115 of *Proceedings of Machine Learning Research*, pp. 766–776. PMLR, 22–25 Jul 2020. URL https://proceedings.mlr.press/v115/paria20a.html.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 53728–53741. Curran Associates, Inc., 2023. doi: 10.52202/075280-2338. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/a85b405ed65c6477a4fe8302b5e06ce7-Paper-Conference.pdf.

Zhizhou Ren, Jiahan Li, Fan Ding, Yuan Zhou, Jianzhu Ma, and Jian Peng. Proximal exploration for model-guided protein sequence design. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 18520–18536. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/ren22a.html.

Adam J. Riesselman, John B. Ingraham, and Debora S. Marks. Deep generative models of genetic variation capture the effects of mutations. *Nature Methods*, 15:816–822, 2018. doi: 10.1038/s41592-018-0138-4. URL https://www.nature.com/articles/s41592-018-0138-4.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 130136–130184. Curran Associates, Inc., 2024. doi: 10.52202/079017-4135. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/eb0b13cc515724ab8015bc978fdde0ad-Paper-Conference.pdf.

Mahakaran Sandhu, Adam C. Mater, Dana S. Matthews, Matthew A. Spence, Artem A. Lenskiy, and Colin Jackson. Investigating the determinants of performance in machine learning for protein fitness prediction. *Protein Science*, 34(8):e70235, 2025. doi: 10.1002/pro.70235. URL https://onlinelibrary.wiley.com/doi/10.1002/pro.70235.

Jatan Shrestha, Santeri Heiskanen, Kari Hepola, Severi Rissanen, Pekka Jääskeläinen, and Joni Pajarinen. Pareto-conditioned diffusion models for offline multi-objective optimization. 2026. URL https://arxiv.org/abs/2602.00737.

Samuel Stanton, Wesley Maddox, Nate Gruver, Phillip Maffettone, Emily Delaney, Peyton Greenside, and Andrew Gordon Wilson. Accelerating Bayesian optimization for biological sequence design with denoising autoencoders. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 20459–20478. PMLR, 17–23 Jul 2022.

Daniel Steinberg, Asiri Wijesinghe, Rafael Oliveira, Piotr Koniusz, Cheng Soon Ong, and Edwin Bonilla. Amortized active generation of pareto sets. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp.

85808–85833. Curran Associates, Inc., 2025. doi: 10.52202/085713-2874. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/7c180af017258d239bac6248d1eb26ac-Paper-Conference.pdf.

Martin Steinegger and Johannes Söding. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature Biotechnology*, 35(11):1026–1028, 2017. doi: 10.1038/nbt.3988. URL https://doi.org/10.1038/nbt.3988.

Baris E. Suzek, Hongzhan Huang, Peter McGarvey, Raja Mazumder, and Cathy H. Wu. Uniref: comprehensive and non-redundant uniprot reference clusters. *Bioinformatics*, 23(10):1282–1288, 05 2007. ISSN 1367-4803. doi: 10.1093/bioinformatics/btm098. URL https://doi.org/10.1093/bioinformatics/btm098.

Sophia Tang, Yinuo Zhang, and Pranam Chatterjee. PepTune: De novo generation of therapeutic peptides with multi-objective-guided discrete diffusion. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 59017–59065. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/tang25n.html.

Masatoshi Uehara, Xingyu Su, Yulai Zhao, Xiner Li, Aviv Regev, Shuiwang Ji, Sergey Levine, and Tommaso Biancalani. Reward-guided iterative refinement in diffusion models at test-time with applications to protein and DNA design. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 60515–60529. PMLR, 13–19 Jul 2025a. URL https://proceedings.mlr.press/v267/uehara25a.html.

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, 2025b. 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, 2025. URL https://arxiv.org/abs/2410.13643.

Chenchun Weng, Andre J. Faure, Albert Escobedo, and Ben Lehner. The energetic and allosteric landscape for KRAS inhibition. *Nature*, 626(7999):643–652, 2024. doi: 10.1038/s41586-023-06954-0. URL https://www.nature.com/articles/s41586-023-06954-0.

Nicholas C. Wu, C. Anders Olson, and Ren Sun. High-throughput identification of protein mutant stability computed from a double mutant fitness landscape. *Protein Science*, 25(2):530–539, feb 2016. doi: 10.1002/pro.2840.

Junhao Xiong, Ishan Gaur, Maria Lukarska, Hunter Nisonoff, Luke M. Oltrogge, David F. Savage, and Jennifer Listgarten. Property guidance for protein sequence generative models with ProteinGuide. *Nature Biotechnology*, 2026. doi: 10.1038/s41587-026-03207-z. URL https://www.nature.com/articles/s41587-026-03207-z.

Jason Yang, Wenda Chu, Daniel Khalil, Raul Astudillo, Bruce J. Wittmann, Frances H. Arnold, and Yisong Yue. Steering generative models with experimental data for protein fitness optimization, 2025a. URL https://arxiv.org/abs/2505.15093.

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, 2025b. doi: 10.1038/s41467-025-55987-8. URL https://doi.org/10.1038/s41467-025-55987-8.

Kevin K. Yang, Zachary Wu, and Frances H. Arnold. Machine-learning-guided directed evolution for protein engineering. *Nature Methods*, 16:687–694, 2019. doi: 10.1038/s41592-019-0496-6. URL https://www.nature.com/articles/s41592-019-0496-6.

Ye Yuan, Can Chen, Christopher Pal, and Xue Liu. Paretoflow: Guided flows in multi-objective optimization, 2025. URL https://arxiv.org/abs/2412.03718.

Bingliang Zhang, Wenda Chu, Julius Berner, Chenlin Meng, Anima Anandkumar, and Yang Song. Improving diffusion inverse problem solving with decoupled noise annealing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20895–20905, 2025. doi: 10.1109/CVPR52734.2025.01946. URL https://openaccess.thecvf.com/content/CVPR2025/html/Zhang_Improving_Diffusion_Inverse_Problem_Solving_with_Decoupled_Noise_Annealing_CVPR_2025_paper.html.

Table A.1: **Summary of used datasets.** Length L of the sequences, multiple sequence alignment (MSA) counts in the training datasets for the generative model, number of sequences in the train and test datasets per objective for the oracle models.

Dataset	Length	Objectives	MSA size	Train fitness	Test fitness	Reference
GB1	56	binding stability†	128,881	385,860 382,946	96,465 95,693	Olson et al. (2014), Wu et al. (2016)
KRAS	188	abundance DARPin K55 binding	105,807	17,035	4,025	Weng et al. (2024)
ERBIN-PDZ1	90	abundance CRIPT binding	71,641	26,996	3,375	Martí-Aranda & Lehner (2026)
NHERF3-PDZ2	111	abundance CEP164 binding	39,809	30,761	3,847	Martí-Aranda & Lehner (2026)
Src‡	269	abundance kinase activity	753,770	41,599	5,201	Beltran et al. (2026)
NHERF3-PDZ1	106	abundance DAP1 binding UHMK binding	63,450	24,971	3,121	Martí-Aranda & Lehner (2026)

† GB1 stability is inferred from single-mutant estimates, summed for double mutants.
‡ Src models residues 268–536; assays used the full-length protein.

A Datasets and preprocessing

For each protein task, we use two complementary datasets that differ in purpose. The first is an unlabeled homolog dataset gathered from UniRef90 (Suzek et al., 2007), providing a natural-sequence distribution to train discrete diffusion models. The second is a labeled sequence–fitness dataset, obtained from mutational studies, which is used to train the prediction oracles. These two data sources are kept separate throughout: no fitness labels are used during diffusion training, and no homolog data are treated as fitness labels. An overview of the used data is represented in Table A.1.

For each task, the datasets for the diffusion models are obtained by searching UniRef90 (Suzek et al., 2007) for homologs of a parent sequence using two iterations of JackHMMER (Johnson et al., 2010). The resulting multiple sequence alignments (MSAs) are processed by removing insertions relative to the parent and replacing alignment gaps with the corresponding parent residues. A homolog is retained only if it covers at least 75% of the parent sequence without gaps prior to gap filling. The processed sequences form fixed-length pseudo-natural sequences that are used to train the generative model. To reduce redundancy among homologous sequences, we cluster them using MMseqs2 (Steinegger & Söding, 2017) at 80% sequence identity, i.e., sequences sharing at least 80% identical residues are grouped into the same cluster. We hold out 5% of clusters and their associated sequences as a validation set. The remaining sequences are resampled with replacement using weights proportional to $(1 + \log n_c)^{-1}$, where n_c is the cluster size. This reduces the contribution of large, closely related clusters. Because the MSA for GB1 contains relatively few homologs, we augment each homolog with all single amino acid substitutions.

The fitness datasets are obtained from published mutational studies listed in Table A.1. They contain single and double amino acid mutations with associated fitness values. For GB1, the stability objective is constructed from published single-mutant stability estimates $\Delta\Delta G_U$ (Wu et al., 2016), by augmenting the training data with double-mutant stability values obtained by summing the corresponding single-mutant effects. For other proteins, the fitness values are taken directly from the corresponding mutational studies. These sequence-fitness pairs are used as labeled data to train the oracle models to serve as a proxy for wet-lab assay measurements, allowing generated variants to be evaluated in silico.

Table B.1: **Generative models.**

Model	Family	Corruption	Trainable parameters	Architecture and initialization
D3PM	Discrete diffusion	Uniform	37.9 M	ByteNet denoiser finetuned from EvoDiff 38M-Uniform (Alamdari et al., 2024).
MDLM	Discrete diffusion	Absorbing state	28.6 M	Transformer denoiser (Sahoo et al., 2024); trained from scratch.

Table C.1: **Fitness oracle accuracy.** Pearson correlations r on the test set for each oracle.

Protein	Objective 1	r	Objective 2	r	Objective 3	r
GB1	Binding	0.89	Inferred stability	1.00	–	–
ERBIN-PDZ1	Abundance	0.94	CRIPT binding	0.95	–	–
NHERF3-PDZ2	Abundance	0.94	CEP164 binding	0.94	–	–
KRAS	Abundance	0.83	DARPin K55 binding	0.89	–	–
Src	Abundance	0.87	Kinase activity	0.76	–	–
NHERF3-PDZ1	Abundance	0.91	DAP1 binding	0.93	UHMK1 binding	0.94

B Diffusion models and sample quality

Table B.1 lists the model sizes, architectures and initialization for the generative models, used in their training.

Figure B.1 summarizes properties of sequences drawn from each diffusion model for each protein task. The violin plots compare the perplexity of the training data with the sequences generated by the models, uniformly random sequences, and the training sequences. Perplexities are calculated with respect to the $764\,\mathrm{M}$-parameter ProGen2-base model (Nijkamp et al., 2023). A sequence with low perplexity corresponds to a high likelihood in this model. This allows comparing the distributions of the learned models with the target training data. The scatter plots show the mean oracle fitness against the sequence diversity of the generated samples. Sequence diversity is the average Shannon entropy of the amino acid distribution generated at each sequence position.

Across all datasets, the random samples consistently yield the highest perplexity. In contrast, the natural MSA sequences that were used to train the generative models ("Target") have much lower perplexity. The learned generative models's perplexities are clearly different from those of random sequences and better approximate the training data's distribution.

The right panels show the trade-off between the diversity of the generated sequences and their mean fitness in each objective. Random samples achieve high diversity but have the lowest mean predicted fitness, whereas the target lies in the high-fitness, low-diversity corner. This pattern is preserved across all six protein groups.

C Fitness oracle evaluation and validation

To train the oracle models, we use an ensemble of ten one-dimensional convolutional neural networks (CNNs) to predict the objectives. CNNs have been shown to have better extrapolation behavior than fully connected networks (Freschlin et al., 2024; Sandhu et al., 2025) Each network takes an one-hot encoded amino acid sequence as input and comprises two convolutional layers, each followed by batch normalization and GELU activation. A final linear regression head is applied to the flattened features of the previous layers. All oracles use 64 and 32 channels, respectively, with kernel size 7. The ensemble members are trained with different random seeds using Adam and a mean squared error loss. The oracle's prediction $\bar{f}_m(x)$ is the average of all ensemble members' predictions. Table C.1 reports the Pearson correlations on test data splits for the resulting CNN ensembles. Figures C.1 and C.2 show the correlation between the data and oracles and the oracle predictions on the MSA sequences that were used to train the generative models.

Since the training data for the oracle only come from a limited neighborhood around a reference sequence, it has to extrapolate for more distant sequences, and predictions become increasingly unreliable. To discourage optimizing in biologically uninformed regions, we add a penalty to the raw oracle predictions:

$$f_m(x) = \bar{f}_m(x) - \lambda_m \max(0, d_{\mathrm{H}}(x, x_{\mathrm{ref}}) - d_0).$$

Here, d_{H} is the Hamming distance from a reference sequence x_{ref}, and d_0 defines a reference distance in which predictions remain unchanged before the penalty is applied. The parameter $\lambda_m \geq 0$ controls the strength of the penalty for the objective f_m. In our experiments, we set $d_0 = 4$ for GB1, ERBIN-PDZ1, NHERF3-PDZ2 and NHERF3-PDZ1, which have shorter sequences, and $d_0 = 8$ for KRAS and Src, which have longer sequences. We set λ_m to the standard deviation of the true fitness values for objective f_m in the training data. This makes the penalty magnitude comparable to the spread of the training labels, so the penalization strength adjusts to objectives with different fitness scales.

D Surrogate architecture and training

The surrogates are trained on the evaluations collected during the optimization campaign. The labeled data used to train the oracles is not used to train surrogates.

Each surrogate is an MLP predicting each objective. The MLP takes one-hot-encoded amino acid sequences as input, embeds them into a 64-dimensional space, and flattens the features. For CG-steering, these features are concatenated with a 64-dimensional time embedding constructed from fixed Gaussian Fourier features, with frequency standard deviation 30, followed by a linear layer of the same output size. The features are then passed through three fully connected layers of width 64, with ReLU activations between them. A regression head then applies a further width-64 linear layer, a ReLU, and a final linear layer to produce the objective predictions.

For CG, the surrogate training samples a diffusion timestep uniformly for each training data, and the diffusion model's forward corruption process is applied to the sequence. The surrogate receives the corrupted sequence and its timestep, while the regression target remains the oracle objective vector of the original clean sequence. This provides training examples at different noise levels without additional oracle queries. For DAPS, the surrogates are trained on clean sequences.

In each optimization round, we initialize an ensemble of ten surrogates with different random seeds and train each member from scratch on all observations accumulated within the campaign. Training minimizes the mean squared error jointly over all objectives, using Adam with a learning rate of 10^{-3} and a shuffled batch size of 20. Each member is trained for 1000 epochs for CG and 200 epochs for DAPS.

E Steering and steering sensitivity

All experiments use $\rho = 0.05$ for the augmented Chebyshev scalarization.

Figures E.1 and E.2 show the effect of different steering strengths on the hypervolume.

F Extended optimization results

Figure F.1 shows how the Hamming distance of sampled sequences from the parent evolves across the 8 rounds for the KRAS task. The dotted vertical line marks the threshold $d_0 = 8$ above which the distance penalty from Appendix C is applied. The distance distributions show that different methods, generative models, and steering mechanisms yield different evaluations in the sequence space.

For all methods, a substantial part of the evaluated sequences are beyond the d_0 distance for the penalty. The penalty, therefore, acts as a soft penalty to sequences far away from the base sequence, allowing evaluations outside the mutational neighborhood of the oracle training data.

For MODS with MDML as the generative model yields a distribution of Hamming distances that is closer to the base sequence but generally wider. For MODS with D3PM the distances are generally larger but more concentrated.

Table F.1: **Holm-adjusted p-values.** Entries compare the MODS configurations with the highest observed mean for each problem against each baseline in Table 1. Values are computed using Welch's t-tests, with Holm's adjustment applied jointly to all $2 \times 5 = 10$ MODS–baseline comparisons within each problem. All adjusted p-values are below 0.05.

Baseline	GB1	ERBIN PDZ1	NHERF3 PDZ2	KRAS	Src	NHERF3 PDZ1
MDLM (unsteered)	$6.51{\cdot}10^{-8}$	$1.71{\cdot}10^{-7}$	$6.23{\cdot}10^{-7}$	$6.99{\cdot}10^{-4}$	$6.03{\cdot}10^{-3}$	$1.62{\cdot}10^{-6}$
D3PM (unsteered)	$3.33{\cdot}10^{-11}$	$6.29{\cdot}10^{-8}$	$1.33{\cdot}10^{-8}$	$2.86{\cdot}10^{-6}$	$4.34{\cdot}10^{-6}$	$3.67{\cdot}10^{-6}$
ARLM + DPO	$3.55{\cdot}10^{-10}$	$1.20{\cdot}10^{-7}$	$2.02{\cdot}10^{-6}$	$1.19{\cdot}10^{-3}$	$8.22{\cdot}10^{-3}$	$4.51{\cdot}10^{-6}$
SAGE-Prot	$7.58{\cdot}10^{-9}$	$5.00{\cdot}10^{-8}$	$3.96{\cdot}10^{-5}$	$1.43{\cdot}10^{-3}$	$1.55{\cdot}10^{-2}$	$3.09{\cdot}10^{-6}$
LaMBO-2	$1.73{\cdot}10^{-7}$	$6.65{\cdot}10^{-5}$	$2.02{\cdot}10^{-6}$	$2.86{\cdot}10^{-6}$	$1.60{\cdot}10^{-5}$	$1.62{\cdot}10^{-6}$

Comparing these results with the achieved hypervolumes in Figure 3, shows no direct relationship between the distance of the parent sequence and the hypervolume.

Table F.2: **Final hypervolume across all evaluated methods.** Entries report mean ± one standard error of the mean over five repetitions. Bold indicates the highest mean among different parameters of a method. Green highlights indicates the highest mean among all methods.

Setting	GB1	ERBIN PDZ1	NHERF3 PDZ2	KRAS	Src	NHERF3 PDZ1
MDLM (unsteered)						
	231.6 ± 11.4	231.4 ± 2.0	145.4 ± 5.0	70.0 ± 6.1	64.0 ± 2.4	2244.8 ± 78.8
D3PM (unsteered)						
	225.0 ± 4.8	221.0 ± 5.5	95.7 ± 3.4	40.7 ± 2.1	44.7 ± 1.0	1004.0 ± 28.4
MODS: MDLM + CG						
$\tau = 0.4^7$	**518.0** ± 13.0	**318.4** ± 14.6	184.2 ± 11.8	99.0 ± 2.5	78.1 ± 4.7	**3873.3** ± 92.0
$\tau = 0.4^6$	458.9 ± 7.7	316.5 ± 8.8	**223.0** ± 2.8	**110.4** ± 1.6	**81.7** ± 1.5	3718.9 ± 30.8
$\tau = 0.4^5$	405.5 ± 9.5	295.8 ± 7.0	210.2 ± 2.3	108.8 ± 1.3	76.8 ± 5.0	3396.0 ± 72.7
$\tau = 0.4^4$	332.9 ± 2.6	258.7 ± 4.7	190.6 ± 4.0	100.8 ± 0.7	75.8 ± 1.2	2865.6 ± 32.9
MODS: MDLM + DAPS						
$\alpha = 2^8$	421.0 ± 8.8	262.1 ± 2.7	173.7 ± 4.3	82.1 ± 3.2	58.4 ± 2.9	2380.5 ± 41.8
$\alpha = 2^{10}$	590.4 ± 12.4	330.0 ± 2.2	228.9 ± 3.4	111.0 ± 0.8	64.3 ± 2.3	3566.7 ± 65.2
$\alpha = 2^{12}$ *	685.3 ± 10.4	391.4 ± 3.7	254.9 ± 5.3	**128.2** ± 3.6	**66.4** ± 2.8	4403.5 ± 97.0
$\alpha = 2^{14}$	**801.6** ± 15.3	**401.1** ± 6.6	**262.1** ± 6.1	110.7 ± 3.0	61.4 ± 0.6	**4846.1** ± 109.9
MODS: D3PM + CG						
$\tau = 0.4^7$	**490.7** ± 7.5	241.3 ± 4.4	**135.2** ± 4.8	**66.1** ± 1.7	**60.2** ± 0.6	**1441.1** ± 127.0
$\tau = 0.4^6$	483.8 ± 9.7	233.2 ± 4.1	129.3 ± 2.7	56.4 ± 0.8	56.2 ± 1.1	1187.5 ± 75.9
$\tau = 0.4^5$	434.3 ± 8.0	**241.8** ± 4.3	121.7 ± 7.2	51.2 ± 0.5	55.1 ± 0.7	1177.6 ± 28.6
$\tau = 0.4^4$	370.5 ± 5.4	241.5 ± 4.5	114.2 ± 2.2	47.7 ± 0.7	53.5 ± 1.2	1149.8 ± 37.8
MODS: D3PM + DAPS						
$\alpha = 2^8$	564.6 ± 3.5	296.0 ± 7.5	160.5 ± 3.7	62.6 ± 1.2	64.7 ± 1.6	1989.8 ± 45.8
$\alpha = 2^{10}$	647.3 ± 5.9	381.0 ± 3.3	218.0 ± 2.3	92.8 ± 1.9	79.9 ± 0.7	3298.6 ± 143.0
$\alpha = 2^{12}$	743.3 ± 5.7	406.5 ± 5.4	259.4 ± 4.8	100.7 ± 1.3	**85.8** ± 0.9	4159.2 ± 60.1
$\alpha = 2^{14}$ *	**781.4** ± 5.9	**413.8** ± 4.2	**271.5** ± 4.1	**106.5** ± 2.3	85.2 ± 0.3	**4278.9** ± 72.5
ARLM + DPO						
$\beta = 0.02$	285.9 ± 22.5	247.9 ± 4.4	165.2 ± 5.1	96.0 ± 1.4	65.8 ± 1.9	2205.4 ± 143.4
$\beta = 0.1$	**328.4** ± 6.8	**252.4** ± 5.3	**165.7** ± 2.1	97.5 ± 1.6	**67.5** ± 2.4	**2272.7** ± 39.7
$\beta = 0.5$	281.5 ± 7.5	239.9 ± 2.4	165.3 ± 3.9	**98.5** ± 2.5	65.1 ± 2.1	2214.5 ± 121.6
$\beta = 2$	245.6 ± 6.2	241.2 ± 2.1	160.4 ± 3.8	77.2 ± 6.0	62.9 ± 1.5	1859.1 ± 70.3
SAGE-Prot						
	210.8 ± 2.5	230.5 ± 2.6	130.2 ± 9.1	95.6 ± 1.3	66.2 ± 3.2	1960.6 ± 126.3
LaMBO-2						
	234.2 ± 13.5	216.2 ± 12.8	121.4 ± 6.4	44.1 ± 2.2	45.7 ± 1.2	1296.8 ± 136.3

τ: CG temperature; α: DAPS steering strength; β: DPO coefficient.
*: variants shown in Figure 3, Figure 4 and Table 1.

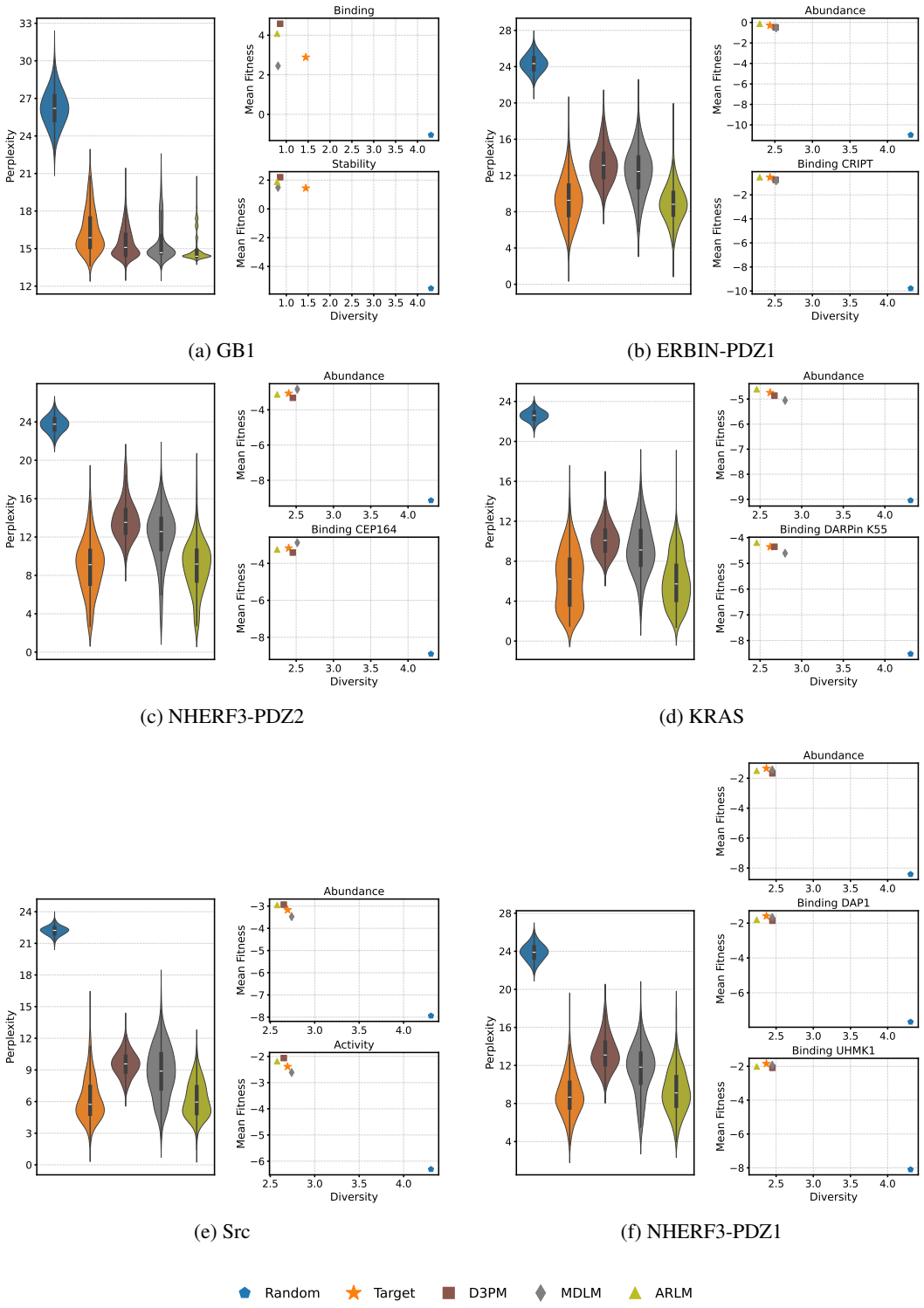


Figure B.1: **Generative model evaluation.** Perplexity distributions are shown on the left. The right panels report the mean oracle fitness versus sequence diversity for all objectives.

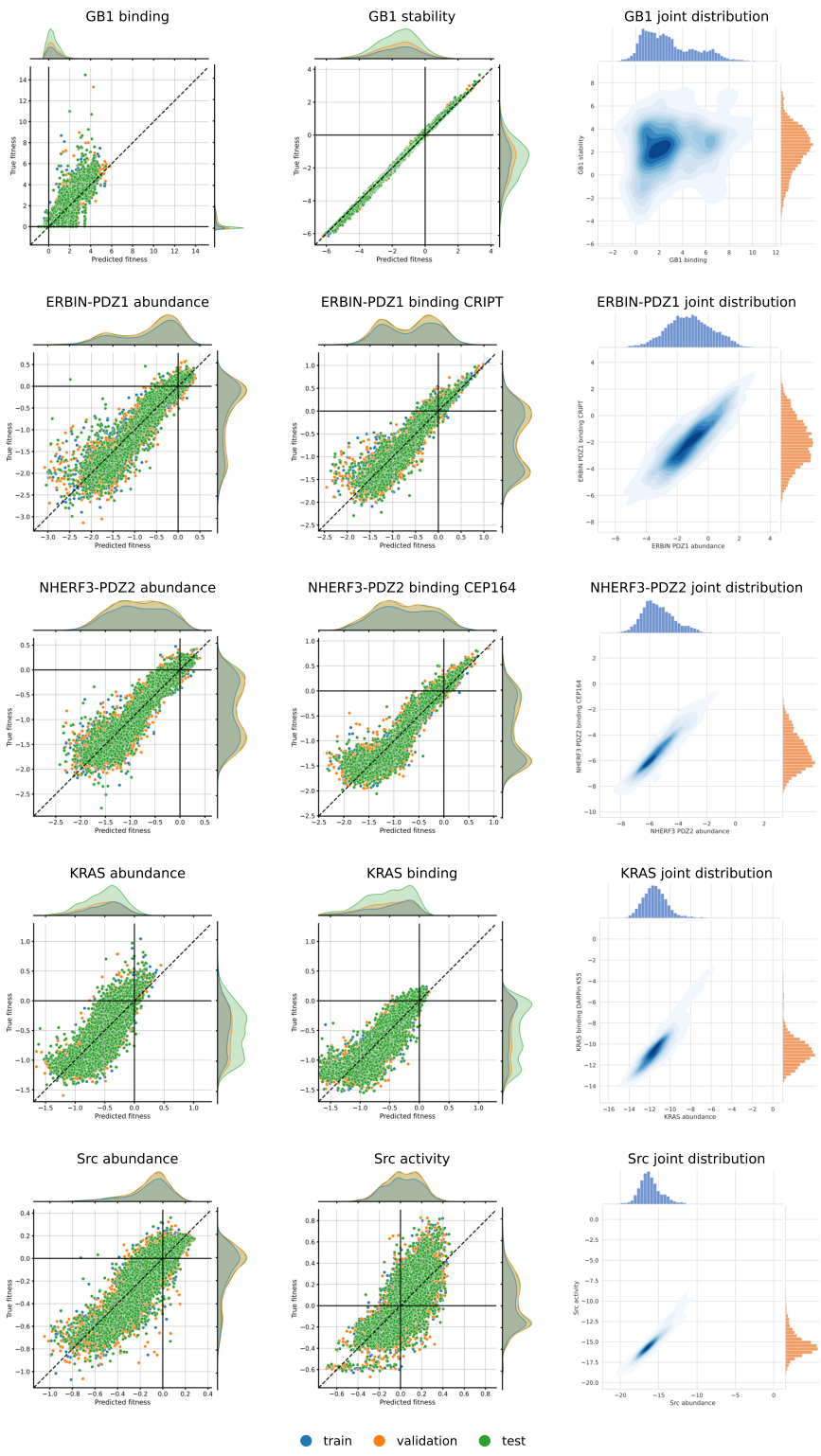


Figure C.1: **Oracle accuracy and joint oracle distributions for two-objective tasks.** The first two columns compare predicted and true fitness for each objective. The third column shows kernel density estimates of joint oracle values for MSA sequences, with marginal histograms. Darker blue indicates higher density.

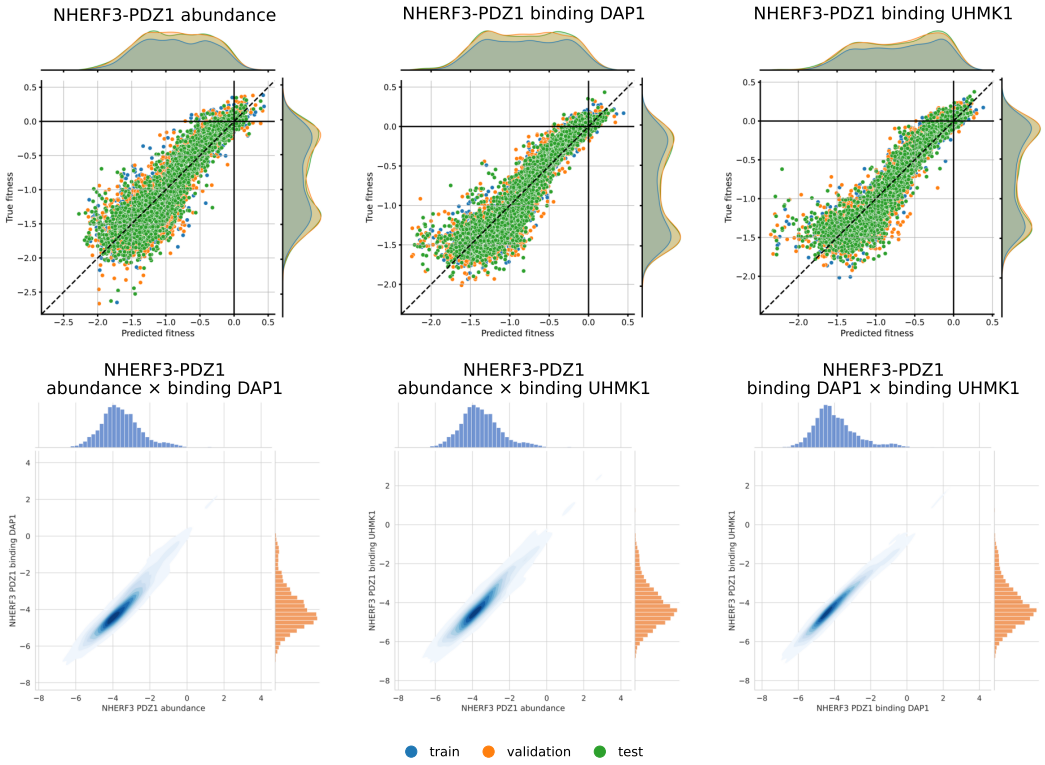
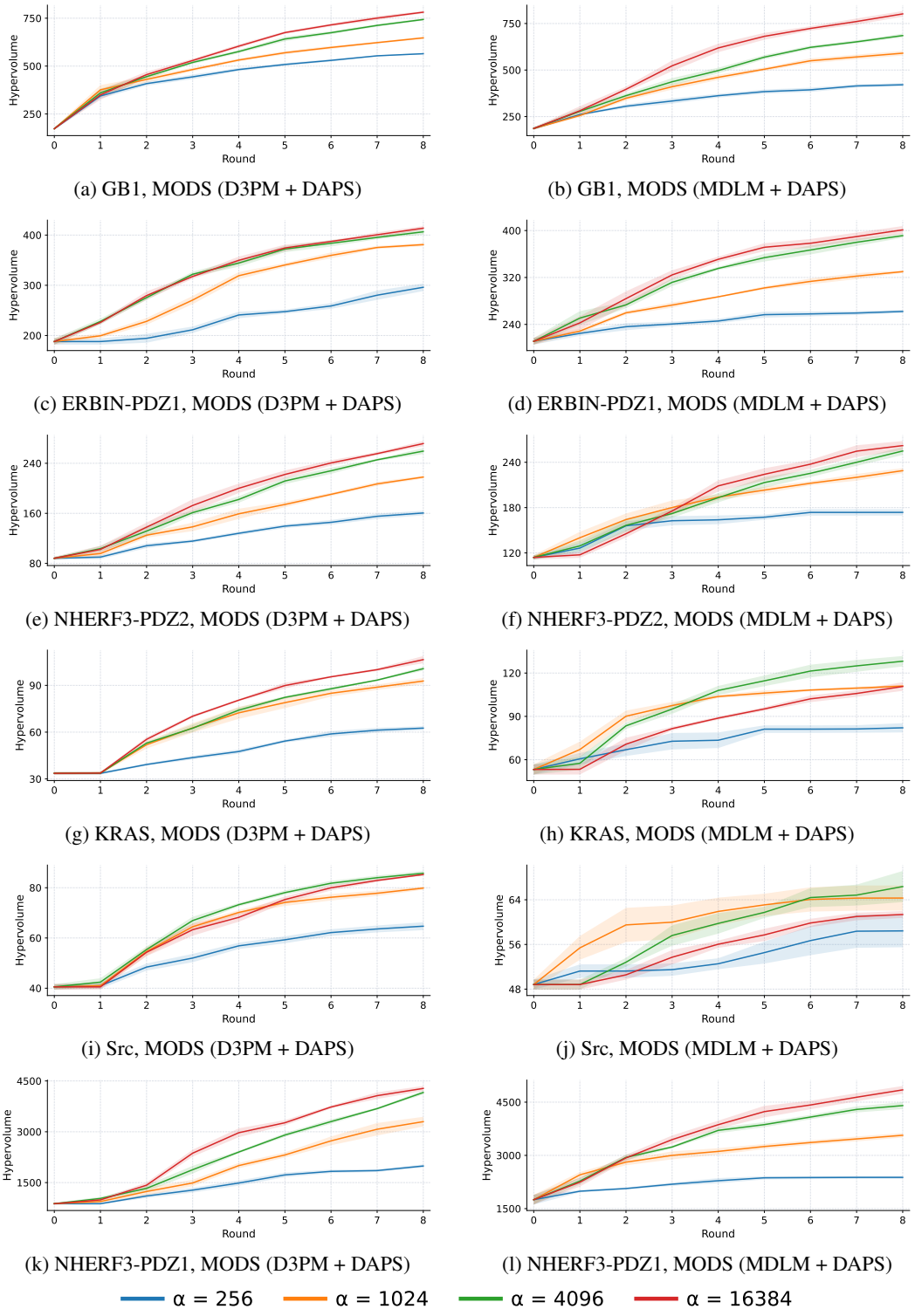


Figure C.2: **Oracle accuracy and joint oracle distributions for three-objective task.** The first two columns compare predicted and true fitness for each objective. The third column shows kernel density estimates of joint oracle values for MSA sequences, with marginal histograms. Darker blue indicates higher density.



(a) GB1, MODS (D3PM + DAPS)

(b) GB1, MODS (MDLM + DAPS)

(c) ERBIN-PDZ1, MODS (D3PM + DAPS)

(d) ERBIN-PDZ1, MODS (MDLM + DAPS)

(e) NHERF3-PDZ2, MODS (D3PM + DAPS)

(f) NHERF3-PDZ2, MODS (MDLM + DAPS)

(g) KRAS, MODS (D3PM + DAPS)

(h) KRAS, MODS (MDLM + DAPS)

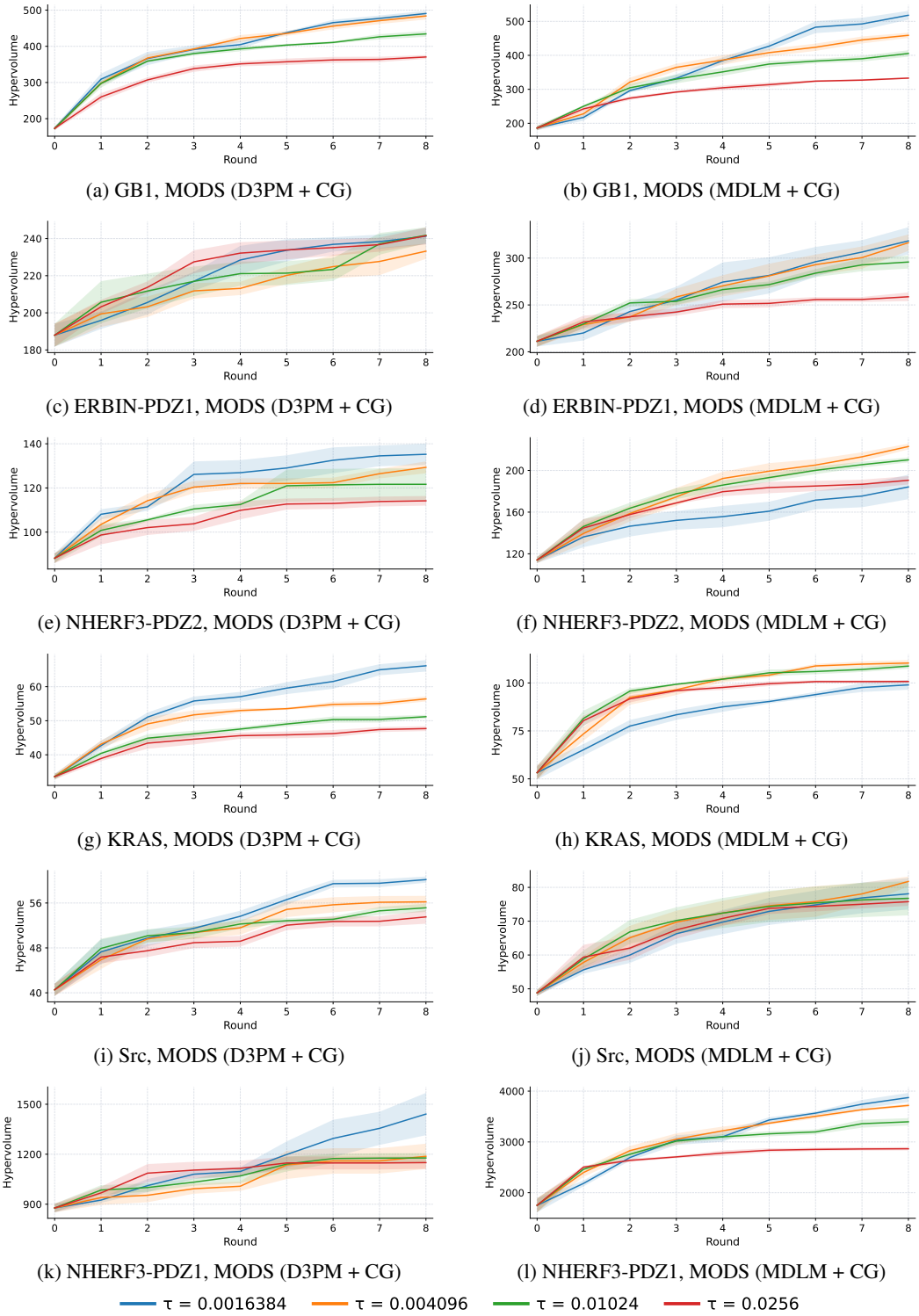
(i) Src, MODS (D3PM + DAPS)

(j) Src, MODS (MDLM + DAPS)

(k) NHERF3-PDZ1, MODS (D3PM + DAPS)

(l) NHERF3-PDZ1, MODS (MDLM + DAPS)

Figure E.1: **Sensitivity of DAPS to the steering strength.** Large α values correspond to weak regularization and strong steering. Lines indicate the mean hypervolume over five repeats. Shaded regions denote the mean $\pm$ one standard error of the mean over five repetitions.



(a) GB1, MODS (D3PM + CG)

(b) GB1, MODS (MDLM + CG)

(c) ERBIN-PDZ1, MODS (D3PM + CG)

(d) ERBIN-PDZ1, MODS (MDLM + CG)

(e) NHERF3-PDZ2, MODS (D3PM + CG)

(f) NHERF3-PDZ2, MODS (MDLM + CG)

(g) KRAS, MODS (D3PM + CG)

(h) KRAS, MODS (MDLM + CG)

(i) Src, MODS (D3PM + CG)

(j) Src, MODS (MDLM + CG)

(k) NHERF3-PDZ1, MODS (D3PM + CG)

(l) NHERF3-PDZ1, MODS (MDLM + CG)

Figure E.2: **Sensitivity of classifier guidance to the steering strength.** Small $\tau := \alpha^{-1}$ values correspond to weak regularization and strong steering. Lines indicate the mean hypervolume over five repeats. Shaded regions denote the mean $\pm$ one standard error of the mean over five repetitions.

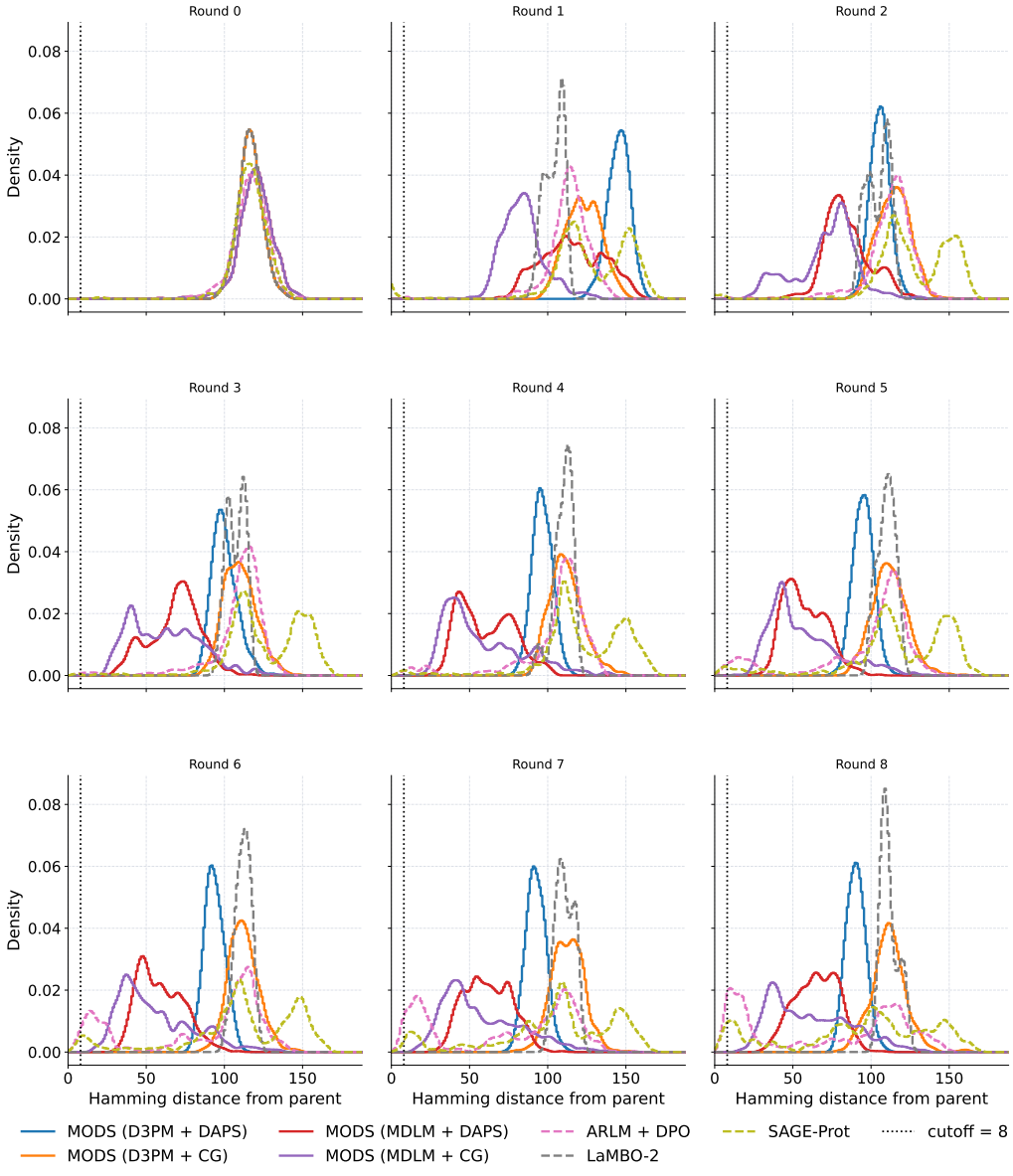


Figure F.1: **Hamming distances evolution of generated proteins throughout campaign.** Each panel shows the distribution of distances from the parent sequence for the evaluated sequences in each round. The dotted vertical line marks the distance threshold $d_0 = 8$.