Pushing Biomolecular Utility-Diversity Frontiers with Supergroup Relative Policy Optimization

Xinwu Ye [*1,2,3] **He Cao** [*2] **Hao Li** [4] **Bin Feng** [2] **Zijing Liu** [2] **Xiangru Tang** [5] **Yu Li** [2] **Shenghua Gao** [1]

Abstract

Biomolecular generators are often adapted with reward feedback to improve task-specific utility, but pushing utility alone can concentrate generation on a narrow family of candidates. Maintaining diversity is difficult because sample diversity is a set-level property. We introduce Supergroup Relative Policy Optimization (SGRPO), a flexible GRPO-style framework that directly constructs rewards from set-level diversity. For each condition, SGRPO samples a supergroup of candidate sets, compares their diversity under the same condition, and redistributes the group diversity reward to individual rollouts through leave-one-out diversity contributions before combining it with rollout-level utility. This design decouples SGRPO from a particular generator, utility reward, or diversity metric, and allows instantiation with different GRPO-style approaches. We evaluate SGRPO on *de novo* small-molecule design, pocket-based small-molecule design, and *de novo* protein design, instantiating it with both GRPO and Coupled-GRPO across autoregressive and discrete diffusion generators. Across decoding sweeps, SGRPO expands the utility-diversity Pareto frontier and achieves the best frontier-level metrics relative to pretrained generators, GRPO, and memory-assisted GRPO when applicable. Our analyses further show that direct set-level diversity rewards remain effective with small groups and help preserve broader generation-distribution coverage during post-training. The code is available at https://github.com/IDEA-XL/SGRPO.

[*]Equal contribution [1]The University of Hong Kong [2]International Digital Economy Academy [3]Beijing Institute of Collaborative Innovation [4]Peking University [5]Yale University. Correspondence to: Yu Li <liyu@idea.edu.cn>, Shenghua Gao <gaosh@hku.hk>.

Accepted at the 2026 Workshop on Generative and Agentic AI for Biology (ICML 2026)

1. Introduction

Biomolecular generation aims to produce candidates that satisfy chemical or biological design objectives, and reinforcement learning (RL) provides a natural framework for post-training pretrained generators from reward feedback toward desired properties, structures, or functions (Olivecrona et al., 2017). In practice, however, generation quality is not determined by utility alone. A model that maximizes a property score, docking proxy, or protein-level objective may concentrate probability mass on a narrow family of candidates, while a highly diverse generator may fail to deliver enough high-utility samples. This creates a utility-diversity trade-off: different downstream settings may prefer different operating points, often modulated by decoding choices such as temperature, so the relevant objective is not a single best reward value but an improved Pareto frontier of attainable utility-diversity pairs. While many successful RL approaches are closely tailored to specific model classes, molecular or protein representations, and design settings (Olivecrona et al., 2017; You et al., 2018; Ektefaie et al., 2024a; Wang et al., 2025), our goal is a broadly applicable post-training principle. We evaluate it across different generator families, conditioning settings, utility functions, and diversity metrics, while leaving broader task-specific instantiations to future work.

A more broadly applicable class of diversity-aware RL methods encourages diversity through memory- or history-dependent novelty penalties (Blaschke et al., 2020; Loeffler et al., 2024). Such methods down-weight candidates that are too similar to previously sampled molecules, scaffolds, clusters, or neighborhoods, and can be effective in practice. However, novelty relative to past samples is only an indirect surrogate for the diversity of the current candidate set produced under a given condition. As a result, these methods may over-penalize useful high-density modes or induce distributional drift during post-training. More fundamentally, the target quantity itself is set-level: diversity is defined over collections of samples, whereas policy optimization updates individual rollouts. This raises the central question of this paper: *can we optimize sample-set diversity directly, as a first-class objective, while still assigning useful credit to individual generated candidates?*

We address this with Supergroup Relative Policy Optimization (SGRPO), a simple framework for directly optimizing sample-set diversity together with rollout-level utility. For each condition, SGRPO samples multiple candidate sets from the current policy, scores each set using a user-specified diversity metric, and compares sets only against other sets generated under the same condition. To make this set-level signal actionable for policy learning, SGRPO redistributes each set's diversity reward to its members through leave-one-out diversity contributions, so candidates that genuinely support set diversity receive stronger credit. The resulting *supergroup*-relative advantage can be instantiated with different GRPO-style optimizers.

We instantiate SGRPO with two GRPO-style optimizers and evaluate it on three biomolecular generation settings: unconditional *de novo* small-molecule design with GenMol (Lee et al., 2025), pocket-based small-molecule design with GenMol-P, our pocket-conditioned variant of GenMol, and unconditional *de novo* protein design with ProGen2 (Nijkamp et al., 2023). Across decoding sweeps, SGRPO consistently improves the attainable utility-diversity Pareto frontier over pretrained generators, GRPO, and memory-assisted GRPO baselines when applicable. It remains effective even with small group sizes and better preserves generation-distribution coverage during post-training, showing that directly optimizing set-level diversity can yield robust gains across both molecule and protein generation.

2. Related Work

2.1. Objective Optimization in Biomolecular Generation

Objective optimization in biomolecular generation is commonly approached either by conditioning or guiding generators toward desired properties, structures, or functions (Lim et al., 2018; Kotsias et al., 2020; Bagal et al., 2021; Jolicoeur-Martineau et al., 2024; Dauparas et al., 2022; Runcie & Mey, 2023; Xiong et al., 2025), or by improving candidates from oracle feedback through methods such as latent-space Bayesian or evolutionary optimization, iterative retraining, preference optimization, and reinforcement learning (Gómez-Bombarelli et al., 2018; Griffiths & Hernández-Lobato, 2020; Castro et al., 2022; Brookes & Listgarten, 2018; Tripp et al., 2020; Cheng et al., 2024; Widatalla et al., 2024; Li et al., 2026). We focus on the RL branch, which provides a general feedback-driven post-training formulation in which generated candidates are scored by objectives such as molecular properties, stability, or multi-objective reward functions, and the generator is updated to increase the likelihood of high-reward samples. RL-based biomolecular optimization has been instantiated across diverse representations, including SMILES sequence models such as REINVENT, ReLeaSE, and ChemRLformer (Olivecrona et al., 2017; Popova et al., 2018; Ghugare et al., 2023), graph- or fragment-based molecular generators such as GCPN, MolDQN, RationaleRL, LibINVENT, and DrugEx v3 (You et al., 2018; Zhou et al., 2019; Jin et al., 2020; Fialková et al., 2021; Liu et al., 2023), and protein sequence or structure-conditioned generators such as model-based RL for biological sequence design, RL-DIF, and ProteinZero (Angermueller et al., 2019a; Ektefaie et al., 2024a; Wang et al., 2025). This broad applicability of RL motivates our focus on diversity-aware reward design at the post-training level.

2.2. Diversity-Promoting RL for Biomolecular Generation

Diversity-promoting RL has been explored in both molecular and protein generation to mitigate mode collapse and sample redundancy. One line of work builds diversity objectives around the structure of a specific generator or design task, for example by jointly generating multiple SMILES strings in a single sequence (Jang et al., 2024), exploiting augmented SMILES and score reuse (Bjerrum et al., 2023), incorporating diversity into fragment-based molecular construction (Yang et al., 2021), pairing exploitation and exploration policies during generation (Liu et al., 2019), or adding task-specific regularization in protein inverse folding and sequence design (Ektefaie et al., 2024b; Wang et al., 2025; Park et al., 2024). A more broadly applicable family instead promotes diversity through indirect reward shaping, such as diverse mini-batch selection (Svensson et al., 2025), memory- or scaffold-based penalties and filters (Blaschke et al., 2020; Pereira et al., 2021; Loeffler et al., 2024; Thomas et al., 2022; Gummesson Svensson et al., 2024; Zhu et al., 2025), distance-to-memory or novelty rewards (Hu et al., 2024; Svensson et al., 2024; Park et al., 2025; Chadi et al., 2023), entropy regularization (Stevens et al., 2025), or count-based visitation bonuses (Angermueller et al., 2019b). These approaches have shown empirical benefits, but they are either tightly coupled to particular generator interfaces or optimize indirect proxies such as novelty, entropy, or history-relative exploration rather than the diversity of the current generated sample set itself. Our work focuses on this latter gap.

3. Problem Setup: Utility–Diversity Frontier in Biomolecular Generation

3.1. Setup

We consider a conditional biomolecular generator $\pi_\theta(x \mid \mathcal{C})$, where x is a generated candidate and $\mathcal{C}$ is the conditioning input. Depending on the task, $\mathcal{C}$ may be empty, a task or property specification, or a target environment such as a protein binding pocket. The formulation is model-agnostic and applies to pretrained *de novo* molecular generators, pocket-conditioned molecular generators, and protein lan-

guage models. Each candidate receives an individual utility score $r(x, \mathcal{C})$. The exact form of r depends on the domain. For small molecules, it may combine drug-likeness and synthesizability, and in pocket-conditioned generation, it may additionally include target-specific terms such as docking. For proteins, utility may reflect sequence plausibility, stability, foldability, or developability.

We also care about diversity among the generated outputs. For a set of K candidates generated under the same condition, denoted by $G = \{x_1, \ldots, x_K\}$, let $D(G)$ be a set-level diversity score. This score may measure internal diversity, scaffold diversity, sequence diversity, or cluster coverage. The key point is that diversity is *not* a per-sample reward: in general, it depends on the relationships among samples in the set and cannot be reduced to independently scoring each candidate. Optimizing diversity, therefore, requires reasoning over groups of outputs rather than isolated generations.

3.2. Utility–diversity frontier

Let $p(\mathcal{C})$ denote the distribution over generation conditions. At inference time, the trained generator is paired with a decoding strategy $a \in \mathcal{A}$, such as a sampling temperature or related decoding hyperparameters. Together, (θ, a) determine two expected quantities: the expected individual utility, denoted by $U(\theta, a)$, and the expected set-level diversity, denoted by $V(\theta, a)$. Here $U(\theta, a)$ is computed from single generated samples, while $V(\theta, a)$ is computed from sets of K samples drawn under the same condition.

Varying the decoding strategy induces a set of attainable utility–diversity trade-offs for the generator, which we denote by $\mathcal{P}(\theta) = \{(U(\theta, a), V(\theta, a)) : a \in \mathcal{A}\}$. A point on this set is Pareto-optimal if no other decoding strategy achieves both higher utility and higher diversity at the same time.

Our goal is to improve this frontier itself. Rather than optimizing only utility or only diversity, we seek post-training methods that push $\mathcal{P}(\theta)$ outward, so that the same generator can achieve better utility at a fixed diversity level, better diversity at a fixed utility level, or both.

4. Supergroup Relative Policy Optimization

SGRPO is a post-training reinforcement learning method for improving the utility–diversity frontier of a pretrained biomolecular generator. Its central idea is simple: since diversity is a set-level property, training should compare *sets* of candidates generated under the same condition, rather than scoring each candidate in isolation. For each condition $\mathcal{C}$, SGRPO samples several candidate groups, scores each group by diversity, redistributes the group-level signal back to individual candidates according to their within-group contribution, and then applies a PPO-style update using a same-condition relative advantage. Figure 1 illustrates the overall pipeline, and detailed pseudocode is provided in Appendix A.

4.1. Same-condition supergroups

For a condition $\mathcal{C}$, we sample M groups from the old policy, each containing K independently generated rollouts. Denote the resulting collection by $\mathcal{S}(\mathcal{C}) = \{G_1, \ldots, G_M\}$, where $G_m = \{x_{m,1}, \ldots, x_{m,K}\}$ and each $x_{m,i} \sim \pi_{\theta_{\text{old}}}(\cdot \mid \mathcal{C})$. We refer to $\mathcal{S}(\mathcal{C})$ as a *supergroup*. It contains $N = MK$ candidates generated under the same condition, with M controlling how many alternative groups are compared and K controlling the size of each group.

Restricting comparisons to a single supergroup is important. In conditional biomolecular generation, different conditions can have very different intrinsic difficulty, so comparing samples across conditions would confound policy quality with condition difficulty. SGRPO instead performs only *local* comparisons: under the same $\mathcal{C}$, which groups are more diverse, and which rollouts are more useful?

4.2. Utility and group-level diversity

Each rollout $x_{m,i}$ receives an individual utility reward $r_{m,i} = r(x_{m,i}, \mathcal{C})$, while each group G_m receives a diversity score $R_m = D(G_m)$. Thus, utility is defined at the candidate level, whereas diversity is defined over the whole group.

To compare groups generated under the same condition, we center group diversity within the supergroup. Let $\bar{R} = \frac{1}{M}\sum_{h=1}^{M} R_h$. We define the group-relative diversity signal as

$$A_m^{\text{grp}} = R_m - \frac{1}{M-1}\sum_{h \neq m} R_h = \frac{M}{M-1}(R_m - \bar{R}). \quad (1)$$

This is simply a leave-one-out comparison among same-condition groups: $A_m^{\text{grp}} > 0$ means that G_m is more diverse than its alternatives, and $A_m^{\text{grp}} < 0$ means the opposite. For the normalized pairwise diversity used in our experiments, average diversity over groups of size K is an unbiased proxy for the diversity of a larger same-condition sample, so optimizing group diversity is aligned with improving diversity at the supergroup level. Formal statements are given in Appendix B.

4.3. Set-aware redistribution

The diversity score R_m evaluates an entire group, but policy optimization ultimately acts on individual rollouts. SGRPO bridges this gap by assigning more of the diversity signal to candidates that matter more for the diversity of their own group.

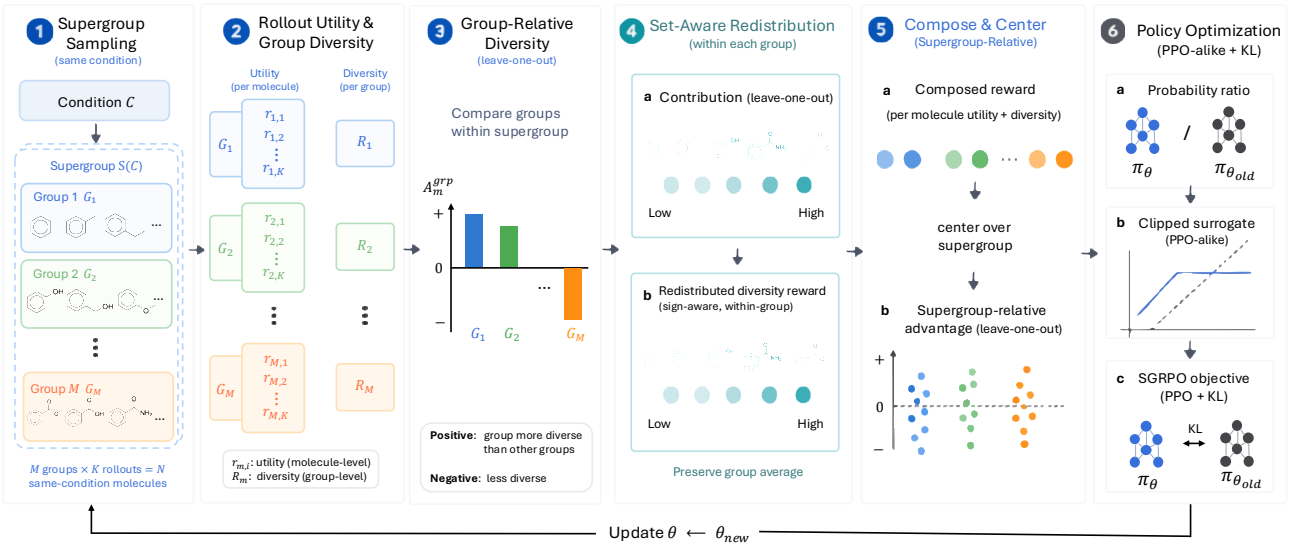

Figure 1. Overview of SGRPO. For each condition, SGRPO samples a same-condition supergroup, computes rollout-level utility and group-level diversity, compares groups by leave-one-out group-relative diversity, redistributes the diversity signal within each group according to leave-one-out set contributions, centers the composed rewards over the supergroup, and updates the policy with a PPO-style objective and KL regularization.

For each rollout $x_{m,i} \in G_m$, we first compute its leave-one-out contribution $c_{m,i} = D(G_m) - D(G_m \setminus \{x_{m,i}\})$, and standardize these contributions within the group as $z_{m,i} = (c_{m,i} - \bar{c}_m)/(\sigma(c_{m,\cdot}) + \zeta)$. We then form two sign-aware softmax weight vectors:

$$w^{\pm}_{m,i} = K \cdot \frac{\exp(\pm z_{m,i}/\tau_c)}{\sum_{j=1}^{K} \exp(\pm z_{m,j}/\tau_c)}. \tag{2}$$

By construction, $\sum_i w^{+}_{m,i} = \sum_i w^{-}_{m,i} = K$. Here $w^{+}_{m,i}$ emphasizes candidates with larger diversity contributions, while $w^{-}_{m,i}$ emphasizes candidates with smaller ones.

We then define the redistributed diversity reward

$$\widetilde{R}_{m,i} = R_m + [A^{\mathrm{grp}}_m]_+(w^{+}_{m,i} - 1) - [-A^{\mathrm{grp}}_m]_+(w^{-}_{m,i} - 1), \tag{3}$$

where $[a]_+ = \max(a, 0)$. The redistribution is sign-aware. If $A^{\mathrm{grp}}_m > 0$, then G_m is more diverse than its same-condition alternatives, and the extra positive signal is concentrated on candidates that contributed more to that diversity. If $A^{\mathrm{grp}}_m < 0$, then G_m is less diverse, and the negative signal is concentrated on candidates that contributed less. By construction, redistribution preserves the original group reward on average, i.e., $\frac{1}{K}\sum_{i=1}^{K} \widetilde{R}_{m,i} = R_m$.

4.4. Supergroup-relative policy update

We combine candidate-level utility and redistributed diversity into a single reward, $\widehat{r}_{m,i} = (1-\lambda) r_{m,i} + \lambda \widetilde{R}_{m,i}$, where $\lambda \in [0, 1]$ controls the utility–diversity trade-off. Let $\bar{r}_{\mathcal{S}} = \frac{1}{MK}\sum_{m=1}^{M}\sum_{i=1}^{K} \widehat{r}_{m,i}$ denote the average composed reward within the supergroup. The final supergroup-relative advantage is

$$A_{m,i} = \frac{MK}{MK-1}\left(\widehat{r}_{m,i} - \bar{r}_{\mathcal{S}}\right). \tag{4}$$

Equivalently, this is a leave-one-out baseline over all rollouts in the same supergroup. Since all rollouts in the supergroup share the same condition, $A_{m,i}$ measures whether a rollout is better or worse than its local same-condition alternatives after utility and diversity have been combined.

We then update the policy with a clipped PPO objective and a KL penalty to a reference policy π_{ref}. Let $\rho_{m,i} = \pi_\theta(x_{m,i} \mid \mathcal{C})/\pi_{\theta_{\mathrm{old}}}(x_{m,i} \mid \mathcal{C})$. The objective is

$$\begin{aligned}\mathcal{L}_{\mathrm{SGRPO}}(\theta) = -\,\mathbb{E}\Big[\min\Big(&\rho_{m,i}A_{m,i},\\ &\mathrm{clip}(\rho_{m,i}, 1-\epsilon, 1+\epsilon)A_{m,i}\Big)\Big]\\ &+ \beta\,\mathbb{E}\Big[\mathrm{KL}\Big(\pi_\theta(\cdot \mid \mathcal{C}) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid \mathcal{C})\Big)\Big]\end{aligned} \tag{5}$$

The expectation is over conditions, sampled supergroups, and rollouts. In practice, SGRPO alternates between sampling same-condition supergroups, computing group-level diversity and rollout-level redistributed rewards, and updating the policy with the objective above. Full pseudocode is provided in Appendix A.

5. Experiments

We evaluate whether SGRPO expands the utility-diversity Pareto frontier across three biomolecular generation settings: unconditional *de novo* small-molecule design, pocket-based small-molecule design, and *de novo* protein design.

In each setting, we decode each model under a sweep of operating points and summarize every operating point by its utility and set-level diversity. We compare the resulting frontiers against the pretrained generator, GRPO, and memory-assisted GRPO when applicable, using the same Pareto-level metrics across tasks. This evaluation tests whether supergroup-relative diversity pressure improves the trade-off frontier itself, rather than merely shifting generation toward higher utility or higher randomness.

5.1. Evaluation Protocol

Each experiment evaluates a generator under a range of task-specific decoding settings, treating each setting as one utility–diversity operating point. For a given model, this yields a set of points $A = \{(U_i, V_i)\}_{i=1}^{n}$, where U_i and V_i denote the utility and diversity of the i-th setting. Both metrics are scaled to $[0, 1]$, with higher values indicating better performance. We summarize performance by the non-dominated subset of A, denoted by $ND(A)$. A point belongs to $ND(A)$ if no other decoding setting achieves at least as much utility and at least as much diversity, with one of them being strictly better. In other words, $ND(A)$ is the Pareto frontier of the model under the evaluated decoding settings. [1]

Hypervolume. For each experiment, we use a common reference point $r_{\text{exp}} = (r_U, r_V)$, where r_U and r_V are the minimum utility and diversity observed across all operating points from all compared methods. In two dimensions, the hypervolume of a model is the area of the staircase-shaped region dominated by its non-dominated operating points and bounded below by r_{exp}. Equivalently, $HV(A; r_{\text{exp}}) = \text{Area}\big(\cup_{(U,V)\in ND(A)}[r_U, U] \times [r_V, V]\big)$. Thus, HV is the union area of axis-aligned rectangles induced by the non-dominated set, rather than the area of a single rectangle. Larger HV indicates that the frontier extends further toward high utility and high diversity and/or spans a broader utility–diversity range. Because the reference point is experiment-specific, HV is intended for within-experiment comparison rather than direct comparison across tasks.

Distance to Ideal Point. Let $z^\star = (U^\star, V^\star)$ denote the ideal point, whose coordinates are the best attainable values of the two objectives. For an operating-point set A, we report $\text{DIP}(A, z^\star) = \min_{(U,V)\in A} \sqrt{(U^\star - U)^2 + (V^\star - V)^2}$. Since utility and diversity are scaled to $[0, 1]$, we set $z^\star = (1, 1)$. Lower distance is better.

[1] Across all evaluated methods and decoding settings, output validity was 100% in our experiments, so the reported utility and diversity values are not confounded by differences in validity.

R2 Indicator. R2 evaluates an operating-point set under multiple utility-diversity preference weights. For a weight $\lambda = (\lambda_U, \lambda_V)$, we define the best weighted Tchebycheff shortfall as $g(A \mid \lambda, z^\star) = \min_{(U,V)\in A} \max\{\lambda_U(z_U^\star - U), \lambda_V(z_V^\star - V)\}$, and compute $R2(A, \Lambda, z^\star) = \frac{1}{|\Lambda|}\sum_{\lambda\in\Lambda} g(A \mid \lambda, z^\star)$. In our implementation, A is the full model-specific sweep set and $\Lambda = \{\lambda^{(\ell)} = (\ell/100, 1 - \ell/100)\}_{\ell=0}^{100}$. Lower R2 means a smaller average weighted worst-case shortfall to $z^\star$.

5.2. *De novo* Small-Molecule Design

Setup. We study unconditional *de novo* small-molecule design with GenMol (Lee et al., 2025), a discrete diffusion language model that generates molecules as SAFE strings (Noutahi et al., 2024). Unlike autoregressive LMs, GenMol generates samples through iterative denoising, so applying GRPO requires a diffusion-compatible training objective. We therefore instantiate SGRPO on top of coupled-GRPO (Gong et al., 2025), which adapts GRPO-style relative policy optimization to discrete diffusion models via coupled denoising samples. The molecule-level utility in this experiment is defined by drug-likeness and synthetic accessibility. We use QED (Bickerton et al., 2012) as a normalized drug-likeness score in $[0, 1]$, and denote the raw synthetic accessibility score (Ertl & Schuffenhauer, 2009) by $\text{SA}(x)$, where lower values indicate easier synthesis. We convert it into a high-is-better score $s_{\text{SA}}(x) = \text{clip}((6 - \text{SA}(x))/5, 0, 1)$, and define the rollout utility as $u(x) = \alpha_{\text{QED}}\text{QED}(x) + \alpha_{\text{SA}} s_{\text{SA}}(x)$. Unless otherwise noted, we use $\alpha_{\text{QED}} = 0.6$ and $\alpha_{\text{SA}} = 0.4$, slightly prioritizing drug-likeness while retaining synthetic accessibility as a feasibility-oriented component. Because our main comparisons are based on frontier-level metrics across decoding settings, rather than a single operating point, the conclusions do not hinge on a finely tuned choice of these scalarization weights. Sample diversity is measured by internal diversity over valid generated molecules using Morgan-fingerprint Tanimoto distances (Bajusz et al., 2015). Specifically, $V(S) = 1 - \frac{2}{|S|(|S|-1)}\sum_{i<j} s_{ij}$, where $s_{ij} = \text{Tan}(\phi(x_i), \phi(x_j))$, $\phi(x)$ is the Morgan fingerprint of molecule x, and Tan denotes Tanimoto similarity. We compare SGRPO (denoted as coupled-SGRPO) against the pretrained GenMol model, coupled-GRPO, and Memory-assisted RL-based coupled-GRPO (Blaschke et al., 2020). For Pareto evaluation, each model is decoded under the same sweep of GenMol randomness ρ and temperature τ, using the six settings $(0.1, 0.5)$, $(0.3, 0.8)$, $(0.5, 1.1)$, $(0.7, 1.4)$, $(0.9, 1.7)$, and $(1.0, 2.0)$. At each sweep point, we generate 1000 molecules per model and compute both utility metrics and internal diversity over the valid molecules generated at that point.

Table 1. Frontier-level metrics for the three tasks. Each cell reports the mean ± 95% confidence interval over five independent sweep runs. HV is higher-is-better, while distance to ideal point (DIP) and R2 are lower-is-better. For the two small-molecule tasks, GRPO, Mem-GRPO, and SGRPO denote coupled-GRPO, Memory-assisted coupled-GRPO, and coupled-SGRPO, respectively.

	De novo Small-Molecule Design				Pocket-Based Design			*De novo* Protein Design			
Metric	Original	GRPO	Mem-GRPO	SGRPO	Original	GRPO	SGRPO	Original	GRPO	Mem-GRPO	SGRPO
HV ↑	$0.0579_{\pm 0.0026}$	$0.0629_{\pm 0.0032}$	$0.0585_{\pm 0.0024}$	$\mathbf{0.0672_{\pm 0.0036}}$	$0.0293_{\pm 0.0011}$	$0.0090_{\pm 0.0000}$	$\mathbf{0.0654_{\pm 0.0002}}$	$0.2708_{\pm 0.0074}$	$0.2078_{\pm 0.0052}$	$0.0245_{\pm 0.0008}$	$\mathbf{0.3627_{\pm 0.0085}}$
DIP ↓	$0.2719_{\pm 0.0015}$	$0.2679_{\pm 0.0017}$	$0.2696_{\pm 0.0025}$	$\mathbf{0.2551_{\pm 0.0020}}$	$0.4643_{\pm 0.0015}$	$0.7527_{\pm 0.0001}$	$\mathbf{0.3818_{\pm 0.0003}}$	$0.4279_{\pm 0.0076}$	$0.6519_{\pm 0.0058}$	$1.0128_{\pm 0.0011}$	$\mathbf{0.3538_{\pm 0.0048}}$
R2 ↓	$0.1035_{\pm 0.0003}$	$0.1034_{\pm 0.0004}$	$0.1072_{\pm 0.0003}$	$\mathbf{0.0979_{\pm 0.0005}}$	$0.2382_{\pm 0.0011}$	$0.3874_{\pm 0.0000}$	$\mathbf{0.1809_{\pm 0.0002}}$	$0.2201_{\pm 0.0027}$	$0.3023_{\pm 0.0034}$	$0.4840_{\pm 0.0024}$	$\mathbf{0.1693_{\pm 0.0036}}$

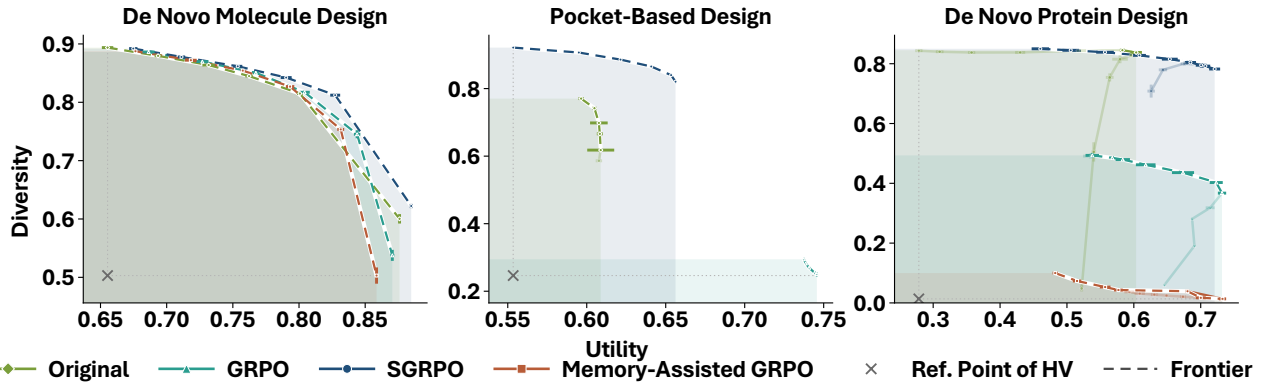

Figure 2. Utility–diversity operating points for de novo small-molecule design, pocket-based small-molecule design, and de novo protein design. Each marker corresponds to one decoding setting and reports the mean utility and diversity over five independent runs; error bars show 95% confidence intervals on both axes. Dashed lines trace the method-specific non-dominated subsets.

Result. SGRPO expands the utility–diversity frontier for *de novo* small-molecule design by improving the high-utility end of the trade-off. In Figure 2, all methods are similar under conservative decoding, but the baselines lose diversity more rapidly as decoding is pushed toward higher utility. Coupled-SGRPO shows a noticeably slower diversity drop, yielding a frontier that extends further right without an equally severe downward bend. This indicates that SGRPO mainly delays diversity collapse, rather than uniformly improving all operating points. Table 1 confirms the same trend quantitatively. Coupled-SGRPO achieves the best HV (0.0670) as well as the lowest DIP (0.2542) and R2 (0.0977), indicating a frontier that is both closer to the ideal point and more favorable overall. The gains are moderate in absolute size because the four methods already overlap substantially in the low- and mid-utility regime, but the ranking is consistent across all three frontier metrics.

5.3. Pocket-Based Small-Molecule Design

Setup. We train GenMol-P for pocket-based small-molecule design. GenMol-P initializes from the pretrained GenMol and adds pocket-prefix conditioning: a frozen ESM-IF1 (Hsu et al., 2022) pocket encoder embeds pocket, and a two-layer MLP projector maps these embeddings into the GenMol hidden space before molecular denoising. We supervised-tune GenMol-P on the CrossDocked2020 (Francoeur et al., 2020) training set. Following Section 5.2, we instantiate SGRPO as coupled-SGRPO for this discrete diffusion generator. The rollout utility augments the QED–SA utility with a target-dependent docking term: $u(x, \mathcal{C}) = \alpha_{\text{QED}}\text{QED}(x) + \alpha_{\text{SA}} s_{\text{SA}}(x) + \alpha_{\text{dock}} s_{\text{dock}}(x, \mathcal{C})$, where $s_{\text{dock}}(x, \mathcal{C}) = \text{clip}(-a_{\text{Vina}}(x, \mathcal{C})/10, 0, 1)$ maps the raw AutoDock Vina (Trott & Olson, 2010) score $a_{\text{Vina}}(x, \mathcal{C})$ to a high-is-better score in $[0, 1]$. Unless otherwise noted, we set $\alpha_{\text{QED}} = 0.3$, $\alpha_{\text{SA}} = 0.2$, and $\alpha_{\text{dock}} = 0.5$, giving docking the largest weight because pocket compatibility is the primary task objective, while QED and SA act as chemistry-oriented regularizers. We do not retune these coefficients per method, because evaluation is based on frontier-level metrics over a shared decoding sweep, and the results are not sensitive to the exact scalarization. We compare coupled-SGRPO against the original GenMol-P model and coupled-GRPO under the same six paired (ρ, τ) settings used in Section 5.2, which span conservative to exploratory decoding regimes, using pockets from the CrossDocked2020 test set. [2] At each sweep point, each model generates 16 ligands for each of 100 held-out pockets, for 1600 ligand samples in total. This protocol provides stable estimates at a manageable docking cost. Utility metrics are averaged over the valid generated ligands at that point, while diversity is computed within each pocket's 16 ligands and then averaged over pockets; this measures target-conditional diversity rather than conflating diversity with variation across different pockets.

[2] We exclude Memory-assisted coupled-GRPO here: global memory mixes unrelated pockets, while pocket-specific memories require separate optimization and extra compute.

Result. Pocket-based design is the setting where SGRPO helps most: because optimizing docking to a fixed pocket tends to collapse generation onto a few high-scoring chemotypes, coupled-GRPO improves utility only by sacrificing within-pocket diversity, whereas coupled-SGRPO shifts the GenMol-P operating-point trajectory outward relative to both the original model and coupled-GRPO and retains markedly higher diversity at comparable utility, especially in the high-utility regime where collapse is strongest (Figure 2). This matches the conditional nature of the task: multiple chemically distinct ligands can be similarly plausible for the same pocket, but utility-only relative updates overamplify small score differences and reinforce redundancy, while SGRPO explicitly rewards high-utility samples that also contribute marginal diversity. Accordingly, coupled-SGRPO achieves the best frontier-level performance in Table 1, with the largest HV and the smallest DIP and R2, and its advantage is more pronounced than in *de novo* molecule generation. Figure 2 shows all evaluated decoding settings for completeness, while HV, DIP, and R2 are computed only on the non-dominated subset, so some adjacent baseline points may improve in both utility and diversity before the true trade-off boundary is reached.

5.4. *De novo* Protein Design

Setup. We evaluate unconditional *de novo* protein design with ProGen2 (Nijkamp et al., 2023), an autoregressive amino-acid language model, and apply SGRPO via GRPO (Shao et al., 2024). The sequence-level utility combines naturalness, foldability, stability, and developability: $u(y) = \alpha_{\text{nat}} r_{\text{nat}}(y) + \alpha_{\text{fold}} r_{\text{fold}}(y) + \alpha_{\text{stab}} r_{\text{stab}}(y) + \alpha_{\text{dev}} r_{\text{dev}}(y)$. Here, the four terms are computed from ESM2, ESMFold (Lin et al., 2023), TemBERTure (Rodella et al., 2024), and ProteinSol-based scorers (Hebditch et al., 2017), respectively, with weights $\alpha_{\text{nat}} = 0.25$, $\alpha_{\text{fold}} = 0.30$, $\alpha_{\text{stab}} = 0.20$, and $\alpha_{\text{dev}} = 0.25$. Diversity is measured at the set level using normalized Levenshtein similarity over valid sequences: $V(S) = 1 - \frac{2}{|S|(|S|-1)} \sum_{i<j} s_{ij}^{\text{edit}}$. We compare SGRPO against the original ProGen2 model, GRPO, and Memory-assisted RL-based GRPO under the same temperature sweep $\tau \in \{0.1, 0.2, \ldots, 1.0, 1.1, 1.2\}$. At each temperature, we sample 512 sequences per model and evaluate both utility and diversity over valid outputs.

Result. SGRPO achieves the best utility–diversity trade-off in *de novo* protein design. In Figure 2, all post-training methods improve utility over the original ProGen2 model, but GRPO and Memory-assisted GRPO do so by collapsing diversity, with the latter showing the most severe mode concentration. By contrast, SGRPO reaches a similarly high-utility regime while preserving diversity much closer to the pretrained model, indicating that it improves sequence quality without sacrificing coverage of distinct sequence families. This pattern is reflected consistently in Table 1, where SGRPO attains the best HV, DIP, and R2.

6. Analysis

6.1. Ablation Study

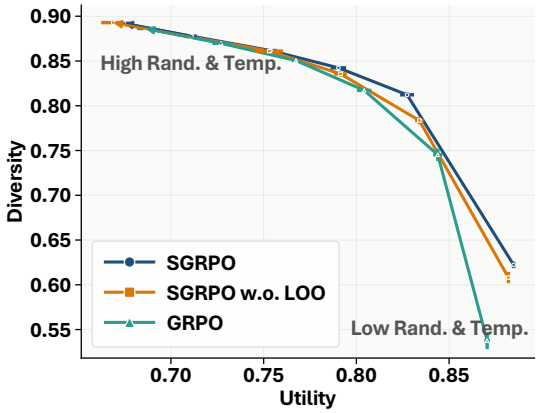

Figure 3. Ablation study on GenMol-based *de novo* small-molecule generation. Each point reports the mean over five independent sweeps, and error bars indicate 95% confidence intervals for utility and diversity. Removing the diversity term yields coupled-GRPO, while removing leave-one-out group credit weakens set-aware credit assignment.

We isolate two components of SGRPO on the GenMol-based *de novo* small-molecule design task: the supergroup diversity reward and the leave-one-out diversity contribution mechanism. As shown in Figure 3, removing the diversity reward gives the innermost utility-diversity curve. Adding the supergroup diversity reward without leave-one-out credit assignment moves the curve outward, indicating that group-level diversity pressure itself improves the trade-off. Full SGRPO further dominates this variant, showing that leave-one-out credit assignment is important for redistributing the group diversity reward to the rollouts that actually contribute to set-level diversity.

6.2. Training Dynamics of Generated Distributions

To understand why the three training algorithms lead to different final utility-diversity trade-offs, we use *de novo* protein design as a diagnostic setting and visualize how their generated sequence distributions move during training. For GRPO, Memory-assisted GRPO, and SGRPO, we sample 512 sequences from the shared original ProGen2 model, the checkpoint after 20 optimization steps, and the final checkpoint after 100 steps. We then pool all sequences and compute a shared two-dimensional UMAP embedding using distances derived from normalized Levenshtein similarity, so that movements are comparable across methods and checkpoints.

Figure 4 suggests two distinct training dynamics. After 20

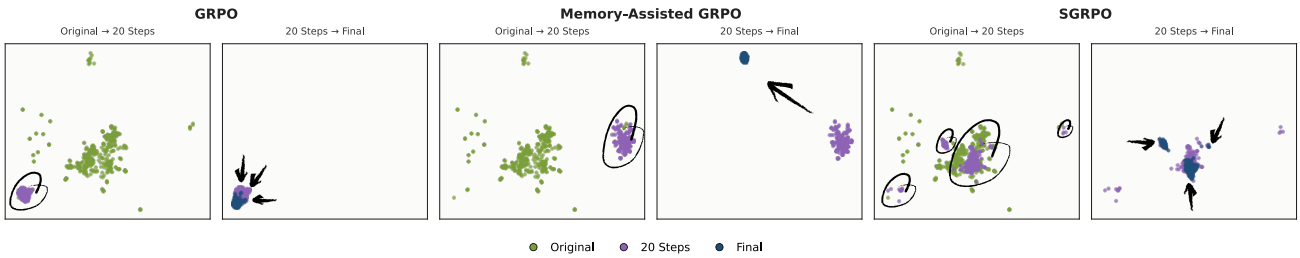

Figure 4. Distribution dynamics during ProGen2 post-training. SGRPO explores multiple clusters early and preserves them, whereas GRPO contracts and Memory-assisted GRPO drift toward a narrow distant region.

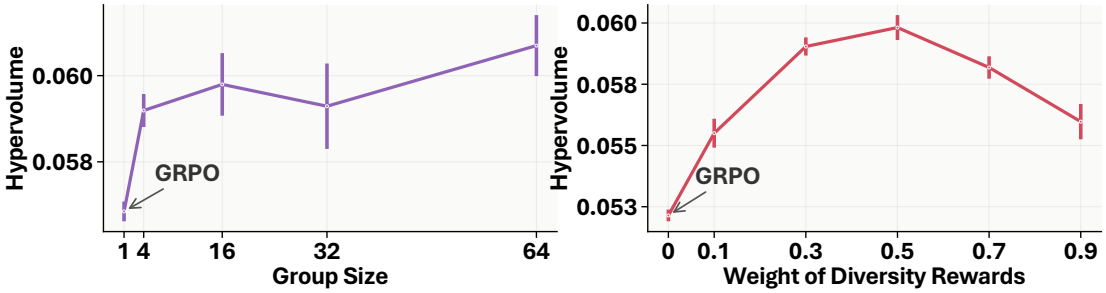

Figure 5. Analysis of diversity-estimator efficiency and group-reward weighting on the GenMol-based *de novo* small-molecule design task. Each point reports the mean HV over five independent sweeps, and error bars indicate 95% confidence intervals. Left: under a fixed total supergroup size, SGRPO already outperforms GRPO with small groups, indicating that useful group-diversity estimates do not require a large rollout multiplier. Right: SGRPO robustly improves HV over GRPO across all tested nonzero λ, with the strongest frontier expansion at $\lambda = 0.5$.

steps, GRPO and Memory-assisted GRPO each move into a relatively concentrated region of the embedding, whereas SGRPO spreads across multiple clusters, including the regions explored by the other two methods. This early behavior indicates that supergroup-relative diversity pressure promotes broader exploration. From 20 to 100 steps, GRPO contracts into an even smaller region. Memory-assisted GRPO instead drifts toward a distant, narrow region: its memory penalty discourages revisiting high-density regions, which is not equivalent to directly optimizing set diversity and can drive distributional drift. SGRPO refines within the explored regions while retaining multiple clusters, explaining why it reaches the high-utility regime while preserving substantially more sequence diversity.

6.3. Robustness to Diversity-Estimator Efficiency and Reward Weighting

We study two practical sensitivities of SGRPO on the GenMol-based *de novo* small-molecule design task: the efficiency of the group-diversity estimator and the choice of group-reward weight λ. In both cases, we train one model per setting, evaluate each model using the same decoding sweep as in Section 5.2, and measure the HV of the resulting utility-diversity frontier using a common lower reference point computed from all operating points in this experiment. To test tolerance to diversity-estimator inefficiency, we hold the total supergroup size fixed while varying the group size $K \in \{1, 4, 16, 32, 64\}$, where $K = 1$ recovers GRPO. As shown in the left panel of Figure 5, SGRPO already outperforms GRPO in the near-minimal setting $K = 4$, indicating that useful group-diversity estimates do not require a large multiplicative increase in rollouts. Larger groups generally lead to better frontier-level performance, suggesting that more efficient diversity estimates help but are not necessary for SGRPO to be effective.

We next vary the group-reward coefficient $\lambda \in \{0, 0.1, 0.3, 0.5, 0.7, 0.9\}$, which controls the trade-off between rollout-level utility and redistributed group-level diversity in the composed reward. The right panel of Figure 5 shows that every tested nonzero value of λ improves HV over the GRPO baseline $\lambda = 0$, demonstrating that the benefit of supergroup-relative diversity pressure is robust to reward weighting. Performance peaks at $\lambda = 0.5$, suggesting that the strongest frontier expansion is achieved by balancing molecule-level utility with set-level diversity. Overall, Figure 5 shows that SGRPO is robust to both imperfect diversity estimation and moderate variation in reward composition.

7. Discussion and Future Work

SGRPO trades an additional rollout structure for a direct set-level diversity signal. Unlike diversity-agnostic RL, SGRPO must sample groups of candidates under the same condi-

tions in order to estimate and compare group diversity. Our analysis shows that SGRPO remains effective with a near-minimal group size, but the method can still require more rollouts than objectives that score each candidate independently. Improving the efficiency of finite-group diversity estimators, or reusing rollouts across compatible diversity computations, is therefore an important direction for making SGRPO cheaper at larger scales.

SGRPO also adds the cost of computing the diversity reward and the leave-one-out diversity contribution. For the pairwise-similarity diversity objectives used in our experiments, this cost is dominated by constructing the within-group similarity matrix. Let a supergroup contain M same-condition groups, each with K rollouts, and let C_{sim} denote the cost of one similarity evaluation. Computing all within-group pairs requires $O(MK^2C_{\mathrm{sim}})$ work, or $MK(K-1)/2$ pair evaluations up to constants. The same similarity matrix is then reused to compute both the group diversity reward and all leave-one-out contributions, so leave-one-out credit assignment does not require a second pass over the pairwise similarities. Our implementation follows this reuse pattern to avoid redundant similarity computation.

This overhead should be interpreted relative to other diversity-aware training mechanisms. Under the matched-rollout protocol used in our experiments, Memory-Assisted GRPO evaluates the same MK generated candidates but compares high-scoring candidates against a memory with B index–bucket pairs. Its memory lookup therefore scales as $O(MKBC_{\mathrm{sim}})$ in the number of stored similarity neighborhoods. Once B is group-scale or larger, this lookup can exceed the pairwise group-diversity computation. In the *de novo* small-molecule run, for example, SGRPO uses $K = 64$ and $M = 8$, while the memory already contains more than 300 index–bucket pairs after the first training step.

The diversity-computation overhead is nevertheless real. For pairwise-similarity diversity, beyond the similarity-matrix reuse used here, future implementations could cache pairwise similarities across repeated candidates or cache reusable metric-specific computations such as fingerprints, embeddings, or nearest-neighbor structures. For diversity notions that are not naturally pairwise-similarity based, the appropriate efficiency strategy may be different: GPU-batched diversity computation, differentiable or learned proxy metrics, or approximate set summaries may be needed to keep set-level rewards practical at larger rollout scales.

Our experiments evaluate the utility-diversity trade-off with respect to the scalar utility specified for each task. These utilities aggregate several design-relevant components, such as drug-likeness, synthesizability, docking affinity, foldability, stability, and developability, using task-specific weights. This scalarization is the shared interface through which a design preference is provided to reward-based post-training: the method is given a utility function and a diversity metric, and the evaluation asks whether the attainable trade-off between them improves. Questions about how to choose, calibrate, or audit the internal utility components are important reward-design questions shared by all reward-based post-training methods. They are complementary to the contribution studied here: given a user-specified utility and diversity metric, SGRPO directly improves the attainable trade-off between them. Although the scope of this work treats utility and diversity as user-specified evaluation axes, a natural extension is to expose the utility components themselves as additional controllable axes, allowing future SGRPO variants to study diversity jointly with more fine-grained within-utility trade-offs.

SGRPO is intentionally decoupled from a specific generator, task, utility objective, or diversity metric. This makes it easy to instantiate through different GRPO-style optimizers, but it also leaves room for more specialized designs. Future work could use the same supergroup-relative framework with task-specific diversity notions, condition-aware chemical series constraints, structure-aware protein diversity metrics, or adaptive group construction rules that better match the scientific objective of a particular design campaign.

8. Conclusion

We presented SGRPO, a GRPO-style framework for directly combining rollout-level utility with set-level sample diversity in biomolecular post-training. The central idea is simple: sample multiple candidate sets under the same condition, score their diversity, compare them within the supergroup, and redistribute the resulting set reward to individual rollouts through leave-one-out diversity contributions. Across *de novo* small-molecule design, pocket-based small-molecule design, and *de novo* protein design, SGRPO improves the utility-diversity Pareto frontier over pretrained generators and RL baselines. These results suggest that directly rewarding diverse generated sets is a practical way to expand the operating points available to biomolecular generation models.

References

Angermueller, C., Dohan, D., Belanger, D., Deshpande, R., Murphy, K., and Colwell, L. Model-based reinforcement learning for biological sequence design. In *International conference on learning representations*, 2019a.

Angermueller, C., Dohan, D., Belanger, D., Deshpande, R., Murphy, K., and Colwell, L. Model-based reinforcement learning for biological sequence design. In *International conference on learning representations*, 2019b.

Bagal, V., Aggarwal, R., Vinod, P., and Priyakumar, U. D. Molgpt: molecular generation using a transformer-decoder model. *Journal of chemical information and modeling*, 62(9):2064–2076, 2021.

Bajusz, D., Rácz, A., and Héberger, K. Why is tanimoto index an appropriate choice for fingerprint-based similarity calculations? *Journal of cheminformatics*, 7(1):20, 2015.

Bickerton, G. R., Paolini, G. V., Besnard, J., Muresan, S., and Hopkins, A. L. Quantifying the chemical beauty of drugs. *Nature chemistry*, 4(2):90–98, 2012.

Bjerrum, E. J., Margreitter, C., Blaschke, T., Kolarova, S., and de Castro, R. L.-R. Faster and more diverse de novo molecular optimization with double-loop reinforcement learning using augmented smiles. *Journal of Computer-Aided Molecular Design*, 37(8):373–394, 2023.

Blaschke, T., Engkvist, O., Bajorath, J., and Chen, H. Memory-assisted reinforcement learning for diverse molecular de novo design. *Journal of cheminformatics*, 12(1):68, 2020.

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

Castro, E., Godavarthi, A., Rubinfien, J., Givechian, K., Bhaskar, D., and Krishnaswamy, S. Transformer-based protein generation with regularized latent space optimization. *Nature Machine Intelligence*, 4(10):840–851, 2022.

Chadi, M.-A., Mousannif, H., and Aamouche, A. Curiosity as a self-supervised method to improve exploration in de novo drug design. In *2023 International Conference on Information Technology Research and Innovation (ICITRI)*, pp. 151–156. IEEE, 2023.

Cheng, X., Zhou, X., Yang, Y., Bao, Y., and Gu, Q. Decomposed direct preference optimization for structure-based drug design. *arXiv preprint arXiv:2407.13981*, 2024.

Dauparas, J., Anishchenko, I., Bennett, N., Bai, H., Ragotte, R. J., Milles, L. F., Wicky, B. I., Courbet, A., de Haas, R. J., Bethel, N., et al. Robust deep learning–based protein sequence design using proteinmpnn. *Science*, 378 (6615):49–56, 2022.

Ektefaie, Y., Viessmann, O., Narayanan, S., Dresser, D., Kim, J. M., and Mkrtchyan, A. Reinforcement learning on structure-conditioned categorical diffusion for protein inverse folding. *arXiv preprint arXiv:2410.17173*, 2024a.

Ektefaie, Y., Viessmann, O., Narayanan, S., Dresser, D., Kim, J. M., and Mkrtchyan, A. Reinforcement learning on structure-conditioned categorical diffusion for protein inverse folding. *arXiv preprint arXiv:2410.17173*, 2024b.

Ertl, P. and Schuffenhauer, A. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of cheminformatics*, 1(1):8, 2009.

Fialková, V., Zhao, J., Papadopoulos, K., Engkvist, O., Bjerrum, E. J., Kogej, T., and Patronov, A. Libinvent: reaction-based generative scaffold decoration for in silico library design. *Journal of Chemical Information and Modeling*, 62(9):2046–2063, 2021.

Francoeur, P. G., Masuda, T., Sunseri, J., Jia, A., Iovanisci, R. B., Snyder, I., and Koes, D. R. Three-dimensional convolutional neural networks and a cross-docked data set for structure-based drug design. *Journal of chemical information and modeling*, 60(9):4200–4215, 2020.

Ghugare, R., Miret, S., Hugessen, A., Phielipp, M., and Berseth, G. Searching for high-value molecules using reinforcement learning and transformers. *arXiv preprint arXiv:2310.02902*, 2023.

Gómez-Bombarelli, R., Wei, J. N., Duvenaud, D., Hernández-Lobato, J. M., Sánchez-Lengeling, B., Sheberla, D., Aguilera-Iparraguirre, J., Hirzel, T. D., Adams, R. P., and Aspuru-Guzik, A. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

Gong, S., Zhang, R., Zheng, H., Gu, J., Jaitly, N., Kong, L., and Zhang, Y. Diffucoder: Understanding and improving masked diffusion models for code generation. *arXiv preprint arXiv:2506.20639*, 2025.

Griffiths, R.-R. and Hernández-Lobato, J. M. Constrained bayesian optimization for automatic chemical design using variational autoencoders. *Chemical science*, 11(2): 577–586, 2020.

Gummesson Svensson, H., Tyrchan, C., Engkvist, O., and Haghir Chehreghani, M. Utilizing reinforcement learning for de novo drug design. *Machine Learning*, 113(7): 4811–4843, 2024.

Hebditch, M., Carballo-Amador, M. A., Charonis, S., Curtis, R., and Warwicker, J. Protein–sol: a web tool for predicting protein solubility from sequence. *Bioinformatics*, 33 (19):3098–3100, 2017.

Hsu, C., Verkuil, R., Liu, J., Lin, Z., Hie, B., Sercu, T., Lerer, A., and Rives, A. Learning inverse folding from millions of predicted structures. In *International conference on machine learning*, pp. 8946–8970. PMLR, 2022.

Hu, X., Liu, G., Yao, Q., Zhao, Y., and Zhang, H. Hamiltonian diversity: effectively measuring molecular diversity by shortest hamiltonian circuits. *Journal of Cheminformatics*, 16(1):94, 2024.

Jang, H., Jang, Y., Kim, J., and Ahn, S. Can llms generate diverse molecules? towards alignment with structural diversity. *arXiv preprint arXiv:2410.03138*, 2024.

Jin, W., Barzilay, R., and Jaakkola, T. Multi-objective molecule generation using interpretable substructures. In *International conference on machine learning*, pp. 4849–4859. PMLR, 2020.

Jolicoeur-Martineau, A., Baratin, A., Kwon, K., Knyazev, B., and Zhang, Y. Any-property-conditional molecule generation with self-criticism using spanning trees. *arXiv preprint arXiv:2407.09357*, 2024.

Kotsias, P.-C., Arús-Pous, J., Chen, H., Engkvist, O., Tyrchan, C., and Bjerrum, E. J. Direct steering of de novo molecular generation with descriptor conditional recurrent neural networks. *Nature Machine Intelligence*, 2(5): 254–265, 2020.

Lee, S., Kreis, K., Veccham, S. P., Liu, M., Reidenbach, D., Peng, Y., Paliwal, S., Nie, W., and Vahdat, A. Genmol: A drug discovery generalist with discrete diffusion. *arXiv preprint arXiv:2501.06158*, 2025.

Li, Y., Jiang, Z., Dai, E., Wang, L., Ye, W.-C., and Liu, L. Cagenmol: Condition-aware diffusion language model for goal-directed molecular generation. *arXiv preprint arXiv:2604.11483*, 2026.

Lim, J., Ryu, S., Kim, J. W., and Kim, W. Y. Molecular generative model based on conditional variational autoencoder for de novo molecular design. *Journal of cheminformatics*, 10(1):31, 2018.

Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637): 1123–1130, 2023.

Liu, X., Ye, K., Van Vlijmen, H. W., IJzerman, A. P., and Van Westen, G. J. An exploration strategy improves the diversity of de novo ligands using deep reinforcement learning: a case for the adenosine a2a receptor. *Journal of cheminformatics*, 11(1):35, 2019.

Liu, X., Ye, K., van Vlijmen, H. W., IJzerman, A. P., and van Westen, G. J. Drugex v3: scaffold-constrained drug design with graph transformer-based reinforcement learning. *Journal of Cheminformatics*, 15(1):24, 2023.

Loeffler, H. H., He, J., Tibo, A., Janet, J. P., Voronov, A., Mervin, L. H., and Engkvist, O. Reinvent 4: Modern ai–driven generative molecule design. *Journal of Cheminformatics*, 16(1):20, 2024.

Nijkamp, E., Ruffolo, J. A., Weinstein, E. N., Naik, N., and Madani, A. Progen2: exploring the boundaries of protein language models. *Cell systems*, 14(11):968–978, 2023.

Noutahi, E., Gabellini, C., Craig, M., Lim, J. S., and Tossou, P. Gotta be safe: a new framework for molecular design. *Digital Discovery*, 3(4):796–804, 2024.

Olivecrona, M., Blaschke, T., Engkvist, O., and Chen, H. Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics*, 9(1):48, 2017.

Park, J., Ahn, J., Choi, J., and Kim, J. Mol-air: Molecular reinforcement learning with adaptive intrinsic rewards for goal-directed molecular generation. *Journal of Chemical Information and Modeling*, 65(5):2283–2296, 2025.

Park, R., Hsu, D. J., Roland, C. B., Korshunova, M., Tessler, C., Mannor, S., Viessmann, O., and Trentini, B. Improving inverse folding for peptide design with diversity-regularized direct preference optimization. *arXiv preprint arXiv:2410.19471*, 2024.

Pereira, T., Abbasi, M., Ribeiro, B., and Arrais, J. P. Diversity oriented deep reinforcement learning for targeted molecule generation. *Journal of cheminformatics*, 13(1): 21, 2021.

Popova, M., Isayev, O., and Tropsha, A. Deep reinforcement learning for de novo drug design. *Science advances*, 4(7): eaap7885, 2018.

Rodella, C., Lazaridi, S., and Lemmin, T. Temberture: advancing protein thermostability prediction with deep learning and attention mechanisms. *Bioinformatics Advances*, 4(1):vbae103, 2024.

Runcie, N. T. and Mey, A. S. Silvr: guided diffusion for molecule generation. *Journal of chemical information and modeling*, 63(19):5996–6005, 2023.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y., Wu, Y., et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Stevens, T. S., Nolan, O., Robert, J.-L., and Van Sloun, R. J. Sequential posterior sampling with diffusion models. In

ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1–5. IEEE, 2025.

Svensson, H. G., Tyrchan, C., Engkvist, O., and Chehreghani, M. H. Diversity-aware reinforcement learning for de novo drug design. *arXiv preprint arXiv:2410.10431*, 2024.

Svensson, H. G., Engkvist, O., Janet, J. P., Tyrchan, C., and Chehreghani, M. H. Diverse mini-batch selection in reinforcement learning for efficient chemical exploration in de novo drug design. *arXiv preprint arXiv:2506.21158*, 2025.

Thomas, M., O'Boyle, N. M., Bender, A., and De Graaf, C. Augmented hill-climb increases reinforcement learning efficiency for language-based de novo molecule generation. *Journal of cheminformatics*, 14(1):68, 2022.

Tripp, A., Daxberger, E., and Hernández-Lobato, J. M. Sample-efficient optimization in the latent space of deep generative models via weighted retraining. *Advances in Neural Information Processing Systems*, 33:11259–11272, 2020.

Trott, O. and Olson, A. J. Autodock vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of computational chemistry*, 31(2):455–461, 2010.

Wang, Z., Fan, J., Guo, R., Nguyen, T., Ji, H., and Liu, G. Proteinzero: Self-improving protein generation via online reinforcement learning. *arXiv preprint arXiv:2506.07459*, 2025.

Widatalla, T., Rafailov, R., and Hie, B. Aligning protein generative models with experimental fitness via direct preference optimization. *bioRxiv*, pp. 2024–05, 2024.

Xiong, J., Gaur, I., Lukarska, M., Nisonoff, H., Oltrogge, L. M., Savage, D. F., and Listgarten, J. Proteinguide: On-the-fly property guidance for protein sequence generative models. *arXiv preprint arXiv:2505.04823*, 2025.

Yang, S., Hwang, D., Lee, S., Ryu, S., and Hwang, S. J. Hit and lead discovery with explorative rl and fragment-based molecule generation. *Advances in Neural Information Processing Systems*, 34:7924–7936, 2021.

You, J., Liu, B., Ying, Z., Pande, V., and Leskovec, J. Graph convolutional policy network for goal-directed molecular graph generation. *Advances in neural information processing systems*, 31, 2018.

Zhou, Z., Kearnes, S., Li, L., Zare, R. N., and Riley, P. Optimization of molecules via deep reinforcement learning. *Scientific reports*, 9(1):10752, 2019.

Zhu, X., Zhao, Z., and Zhu, F. Scaffold-driven molecular generation via reinforced rnn with centroid distance evaluation. *Expert Systems with Applications*, 292:128606, 2025.

A. Full Training Procedure of SGRPO

This appendix provides the full training procedure of Supergroup Relative Policy Optimization (SGRPO), corresponding to Section 4 in the main text. For each condition, SGRPO samples a same-condition supergroup, computes rollout-level utility and group-level diversity, redistributes the diversity signal to individual rollouts in a set-aware manner, forms supergroup-relative advantages, and updates the policy with a PPO-style objective regularized toward a reference policy.

Algorithm 1: Supergroup Relative Policy Optimization (SGRPO)

Input: initial policy π_θ, reference policy π_{ref}, condition batch $\{\mathcal{C}_b\}_{b=1}^{B}$, groups per condition M, rollouts per group K, diversity weight λ, PPO clip parameter ϵ, KL coefficient β, contribution temperature τ_c
for $t = 1$ to T **do**
  Set $\theta_{\mathrm{old}} \leftarrow \theta$
  for $b = 1$ to B **do**
    Sample a same-condition supergroup $\mathcal{S}_b = \{G_{b,1}, \ldots, G_{b,M}\}$, where each group is $G_{b,m} = \{x_{b,m,1}, \ldots, x_{b,m,K}\}$ with $x_{b,m,i} \sim \pi_{\theta_{\mathrm{old}}}(\cdot \mid \mathcal{C}_b)$
    Compute rollout utilities $r_{b,m,i} = r(x_{b,m,i}, \mathcal{C}_b)$ for all m, i
    Compute group diversities $R_{b,m} = D(G_{b,m})$ for all m
    Compute the supergroup mean diversity $\bar{R}_b = \frac{1}{M}\sum_{h=1}^{M} R_{b,h}$
    Compute group-relative diversity signals $A^{\mathrm{grp}}_{b,m} = \frac{M}{M-1}(R_{b,m} - \bar{R}_b)$ for all m
    for $m = 1$ to M **do**
      For each rollout $x_{b,m,i}$, compute leave-one-out contribution $c_{b,m,i} = D(G_{b,m}) - D(G_{b,m} \setminus \{x_{b,m,i}\})$
      Compute within-group standardized contributions $z_{b,m,i} = \dfrac{c_{b,m,i} - \bar{c}_{b,m}}{\sigma(c_{b,m,\cdot}) + \zeta}$ for all i
      Form sign-aware redistribution weights $w^{\pm}_{b,m,i} = K\,\dfrac{\exp(\pm z_{b,m,i}/\tau_c)}{\sum_{j=1}^{K}\exp(\pm z_{b,m,j}/\tau_c)}$ for all i
      Assign redistributed diversity rewards $\widetilde{R}_{b,m,i} = R_{b,m} + [A^{\mathrm{grp}}_{b,m}]_+(w^{+}_{b,m,i} - 1) - [-A^{\mathrm{grp}}_{b,m}]_+(w^{-}_{b,m,i} - 1)$ for all i
    end for
    Form combined rollout rewards $\widehat{r}_{b,m,i} = (1-\lambda) r_{b,m,i} + \lambda \widetilde{R}_{b,m,i}$ for all m, i
    Compute the supergroup mean reward $\bar{r}_{\mathcal{S}_b} = \frac{1}{MK}\sum_{m=1}^{M}\sum_{i=1}^{K}\widehat{r}_{b,m,i}$
    Compute supergroup-relative advantages $A_{b,m,i} = \frac{MK}{MK-1}\big(\widehat{r}_{b,m,i} - \bar{r}_{\mathcal{S}_b}\big)$ for all m, i
  end for
  Update θ by minimizing the PPO-style objective with KL regularization to π_{ref} using all collected tuples $(\mathcal{C}_b, x_{b,m,i}, A_{b,m,i})$
end for
Return: trained policy π_θ

Implementation notes. All relative comparisons in SGRPO are performed within the same-condition supergroup. In particular, group-relative diversity signals are centered only across the M groups sampled for the same condition, and supergroup-relative advantages are centered only across the corresponding MK rollouts. This avoids confounding policy quality with variation in condition difficulty. In practice, the diversity metric $D(\cdot)$ can be instantiated according to the domain, and the utility reward $r(x, \mathcal{C})$ can be any task-specific scalar oracle.

B. Properties of Small-Group Pairwise Diversity Rewards

In this appendix, we justify the use of small-group diversity as a training signal in SGRPO. We show two properties for the normalized pairwise diversity used in our experiments: (i) *partition consistency*, namely that the average diversity of randomly partitioned groups is an unbiased proxy for the diversity of the full same-condition sample; and (ii) *concentration*, namely that this proxy becomes more stable as the group size increases.

Setup. Let $\mathcal{X} = \{x_1, \ldots, x_N\}$ be a same-condition sample of size $N = MK$, where M is the number of groups and K is the group size. A *balanced partition* Π divides $\mathcal{X}$ into M disjoint groups of size K:

$$\Pi = \{G_1, \ldots, G_M\}, \qquad |G_m| = K, \qquad \bigsqcup_{m=1}^{M} G_m = \mathcal{X}.$$

For a set $A = \{a_1, \ldots, a_L\}$, define the normalized pairwise diversity

$$D_L(A) = \frac{2}{L(L-1)} \sum_{1 \le i < j \le L} d(a_i, a_j), \qquad d(x, x') := 1 - s(x, x'), \tag{6}$$

where $s(x, x') \in [0, 1]$ is a biomolecular similarity function. Equivalently, $D_L(A)$ is the average pairwise dissimilarity under $d \in [0, 1]$.

Given a balanced partition $\Pi = \{G_1, \ldots, G_M\}$, define the average small-group diversity

$$\overline{D}_{M,K}(\mathcal{X}, \Pi) = \frac{1}{M} \sum_{m=1}^{M} D_K(G_m). \tag{7}$$

B.1. Partition Consistency

Proposition B.1 (Partition consistency of normalized pairwise diversity)**.** *Let $\mathcal{X} = \{x_1, \ldots, x_N\}$ be any fixed set of size $N = MK$. If Π is drawn uniformly from all balanced partitions of $\mathcal{X}$ into M groups of size K, then*

$$\mathbb{E}_\Pi\big[\overline{D}_{M,K}(\mathcal{X}, \Pi)\big] = D_N(\mathcal{X}). \tag{8}$$

Proof. For brevity, let $d_{ij} := d(x_i, x_j)$. For a random balanced partition Π, define the indicator

$$I_{ij}(\Pi) = \mathbf{1}\{x_i \text{ and } x_j \text{ belong to the same group under } \Pi\}.$$

Then

$$\begin{aligned} \overline{D}_{M,K}(\mathcal{X}, \Pi) &= \frac{1}{M} \sum_{m=1}^{M} \frac{2}{K(K-1)} \sum_{\substack{i<j \\ x_i, x_j \in G_m}} d_{ij} \\ &= \frac{1}{M} \cdot \frac{2}{K(K-1)} \sum_{1 \le i < j \le N} I_{ij}(\Pi)\, d_{ij}. \end{aligned} \tag{9}$$

Taking expectation over Π gives

$$\mathbb{E}_\Pi\big[\overline{D}_{M,K}(\mathcal{X}, \Pi)\big] = \frac{1}{M} \cdot \frac{2}{K(K-1)} \sum_{1 \le i < j \le N} \mathbb{E}_\Pi[I_{ij}(\Pi)]\, d_{ij}. \tag{10}$$

By symmetry of the uniform balanced partition, for any fixed pair (i, j),

$$\mathbb{P}_\Pi(I_{ij}(\Pi) = 1) = \frac{K-1}{N-1}, \tag{11}$$

because once x_i is assigned to a group, exactly $K - 1$ of the remaining $N - 1$ positions lie in that same group. Substituting Eq. (11) into Eq. (10),

$$\begin{aligned} \mathbb{E}_\Pi\big[\overline{D}_{M,K}(\mathcal{X}, \Pi)\big] &= \frac{1}{M} \cdot \frac{2}{K(K-1)} \cdot \frac{K-1}{N-1} \sum_{1 \le i < j \le N} d_{ij} \\ &= \frac{2}{MK(N-1)} \sum_{1 \le i < j \le N} d_{ij} \\ &= \frac{2}{N(N-1)} \sum_{1 \le i < j \le N} d_{ij} = D_N(\mathcal{X}), \end{aligned} \tag{12}$$

where we used $N = MK$ in the last step. □

Proposition B.1 shows that SGRPO does not optimize an unrelated small-set objective: for normalized pairwise diversity, the average diversity of random groups is exactly aligned, in expectation, with the diversity of the full same-condition sample.

B.2. Concentration Around Full-Sample Diversity

We now show that the average small-group diversity not only matches the full-sample diversity in expectation, but also concentrates around it when the groups are sufficiently large.

Proposition B.2 (Concentration of average small-group diversity)**.** *Fix a condition c, and let $X_1, \dots, X_N \overset{\text{i.i.d.}}{\sim} \pi_\theta(\cdot \mid c)$ with $N = MK$. Let $\mathcal{X} = \{X_1, \dots, X_N\}$, and let Π be a uniformly random balanced partition of $\mathcal{X}$ into M groups of size K, independent of the samples. Define*

$$\mu(c) := \mathbb{E}_{X,X' \sim \pi_\theta(\cdot|c)}\big[d(X, X')\big]. \tag{13}$$

Then, for any $\varepsilon > 0$,

$$\mathbb{P}\big(\big|\overline{D}_{M,K}(\mathcal{X}, \Pi) - D_N(\mathcal{X})\big| \geq \varepsilon\big) \leq 4 \exp\left(-\frac{1}{2} M \left\lfloor \frac{K}{2} \right\rfloor \varepsilon^2\right). \tag{14}$$

Consequently, for any $\delta \in (0, 1)$,

$$\big|\overline{D}_{M,K}(\mathcal{X}, \Pi) - D_N(\mathcal{X})\big| \leq \varepsilon \tag{15}$$

with probability at least $1 - \delta$ whenever

$$M \left\lfloor \frac{K}{2} \right\rfloor \geq \frac{2 \log(4/\delta)}{\varepsilon^2}. \tag{16}$$

Equivalently, a sufficient lower bound on the group size is

$$K \geq 2 \left\lceil \frac{2 \log(4/\delta)}{M \varepsilon^2} \right\rceil. \tag{17}$$

Proof. Both $D_N(\mathcal{X})$ and $\overline{D}_{M,K}(\mathcal{X}, \Pi)$ estimate the same population quantity $\mu(c)$ defined in Eq. (13).

First, $D_N(\mathcal{X})$ is a bounded U-statistic of order two with kernel $d(\cdot, \cdot) \in [0, 1]$. By Hoeffding's concentration inequality for bounded U-statistics,

$$\mathbb{P}(|D_N(\mathcal{X}) - \mu(c)| \geq t) \leq 2 \exp\left(-2 \left\lfloor \frac{N}{2} \right\rfloor t^2\right). \tag{18}$$

Next, consider $\overline{D}_{M,K}(\mathcal{X}, \Pi)$. Because the samples are i.i.d. and the partition is independent of the samples, its distribution is the same as that obtained by first drawing MK i.i.d. samples and then forming M disjoint blocks of size K. Hence

$$\overline{D}_{M,K}(\mathcal{X}, \Pi) = \frac{1}{M} \sum_{m=1}^{M} U_m,$$

where $U_1, \dots, U_M$ are independent copies of a bounded order-two U-statistic based on K i.i.d. samples with mean $\mu(c)$. For each U_m, Hoeffding's inequality yields

$$\mathbb{P}(|U_m - \mu(c)| \geq t) \leq 2 \exp\left(-2 \left\lfloor \frac{K}{2} \right\rfloor t^2\right).$$

Equivalently, each $U_m - \mu(c)$ is sub-Gaussian with variance proxy proportional to $1/\lfloor K/2 \rfloor$. Averaging the M independent terms, therefore, gives

$$\mathbb{P}\big(\big|\overline{D}_{M,K}(\mathcal{X}, \Pi) - \mu(c)\big| \geq t\big) \leq 2 \exp\left(-2M \left\lfloor \frac{K}{2} \right\rfloor t^2\right). \tag{19}$$

Applying the triangle inequality,

$$\big|\overline{D}_{M,K}(\mathcal{X}, \Pi) - D_N(\mathcal{X})\big| \leq \big|\overline{D}_{M,K}(\mathcal{X}, \Pi) - \mu(c)\big| + |D_N(\mathcal{X}) - \mu(c)|. \tag{20}$$

Setting $t = \varepsilon/2$ in Eqs. (18) and (19), and then using a union bound, we obtain

$$\begin{aligned} &\mathbb{P}\big(\big|\overline{D}_{M,K}(\mathcal{X}, \Pi) - D_N(\mathcal{X})\big| \geq \varepsilon\big) \\ &\leq 2 \exp\left(-\frac{1}{2} M \left\lfloor \frac{K}{2} \right\rfloor \varepsilon^2\right) + 2 \exp\left(-\frac{1}{2} \left\lfloor \frac{N}{2} \right\rfloor \varepsilon^2\right). \end{aligned} \tag{21}$$

Since $N = MK$ and $\lfloor N/2 \rfloor \geq M\lfloor K/2 \rfloor$, the second term is no larger than the first, which gives Eq. (14). Finally, solving

$$4\exp\left(-\frac{1}{2}M\left\lfloor\frac{K}{2}\right\rfloor\varepsilon^2\right) \leq \delta$$

yields the sufficient condition in Eq. (16), and Eq. (17) follows immediately. □

Proposition B.2 clarifies the role of the group size K. The partition-consistency result removes bias at the objective level, while the concentration result shows that increasing K improves the stability of the small-group diversity signal as a proxy for full-sample diversity.

C. GenMol-P Implementation

C.1. Method

GenMol-P extends GenMol (Lee et al., 2025) from unconditional molecular generation to pocket-conditioned molecular generation by adding a continuous pocket prefix to the discrete diffusion language model. Let $x = (x_1, \ldots, x_T)$ denote the SAFE-token sequence of a ligand and let $\mathcal{C}$ denote a protein pocket. In GenMol-P, $\mathcal{C}$ is represented by the residue sequence together with backbone coordinates (N, C_α, C) for each pocket residue. A frozen ESM-IF1 encoder (Hsu et al., 2022) maps these pocket backbone coordinates to residue-level embeddings $h_1, \ldots, h_L$, where L is the number of pocket residues. A trainable two-layer projector P_ψ then maps each residue embedding into the GenMol hidden space, producing prefix vectors $p_\ell = P_\psi(h_\ell) \in \mathbb{R}^d$, with $d = 768$. The projector consists of a linear layer, GELU nonlinearity, a second linear layer, and layer normalization.

The projected pocket vectors are inserted as a prefix before the molecular token embeddings. Given corrupted molecule tokens $\tilde{x}_t$ at diffusion time t, GenMol-P forms the transformer input

$$[p_1, \ldots, p_L, e(\tilde{x}_{t,1}), \ldots, e(\tilde{x}_{t,T})],$$

where $e(\cdot)$ is the molecular token embedding. The transformer attends jointly over the pocket prefix and molecular positions, but logits are read only from the molecular positions. Thus, the pocket prefix conditions molecular denoising without being treated as tokens to be generated. The total prefix-plus-molecule length is capped at 256 positions, matching the GenMol positional budget; training examples that exceed this budget are excluded during preprocessing. This construction keeps the molecular generation mechanism unchanged while allowing the denoising network to condition on the target pocket through continuous structural context.

C.2. Training

GenMol-P is supervised-tuned on CrossDocked2020 (Francoeur et al., 2020) pocket-ligand pairs. For each complex, the ligand SMILES string is converted to a SAFE string and tokenized with the GenMol tokenizer. The pocket is converted into a residue-level structural prefix by extracting, or deterministically reconstructing, the N, C_α, and C backbone coordinates for each pocket residue. We discard examples without an assigned train/validation/test split, examples whose ligand cannot be converted to a nonempty SAFE string, malformed pockets with missing backbone information, and examples whose pocket-prefix plus ligand-token length exceeds 256.

The supervised objective is the same masked discrete diffusion objective as GenMol, with the pocket prefix supplied as additional context. For a training pair $(x, \mathcal{C})$, we sample a diffusion time $t \in [\varepsilon_{\text{time}}, 1]$ with $\varepsilon_{\text{time}} = 10^{-3}$, corrupt the ligand token sequence through the masked discrete diffusion forward process, and train the model to recover the original SAFE tokens from $(\tilde{x}_t, \mathcal{C})$. We use antithetic time sampling within minibatches and optimize the globally averaged molecular-token loss. The ESM-IF1 pocket encoder is frozen throughout training; the GenMol backbone and the pocket projector are trainable. The GenMol backbone is initialized from the pretrained GenMol checkpoint, so supervised tuning learns pocket conditioning while retaining the molecular prior learned by the unconditional generator.

The reported GenMol-P checkpoint is trained on 8 H200 GPUs with bf16 precision, per-device batch size 384, and no gradient accumulation, giving a global batch size of 3072. We use AdamW with learning rate 3×10^{-4}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, optimizer $\epsilon = 10^{-8}$, and zero weight decay. The learning-rate schedule is constant after 2500 warmup steps, gradients are clipped at norm 1.0, and an exponential moving average with decay 0.9999 is maintained over the

trainable parameters. The original GenMol-P model used in the pocket-conditioned experiments is the verified checkpoint at 5,500 supervised optimization steps, and the subsequent coupled-GRPO and coupled-SGRPO runs are initialized from this checkpoint.

D. Experimental Implementation Details

D.1. *De novo* Small-Molecule Design

Model. The *de novo* small-molecule design experiment uses GenMol (Lee et al., 2025) as the base generator. GenMol is a masked discrete diffusion language model over SAFE molecular strings (Noutahi et al., 2024). SAFE represents a molecule as an unordered sequence of fragment blocks: a molecule is decomposed into BRICS fragments, each fragment is written as a contiguous string with its attachment points preserved, and fragments are concatenated with separator tokens. This representation is well matched to non-autoregressive denoising because the molecular identity is insensitive to the order in which fragment blocks are listed. As a result, GenMol can model the whole molecular string bidirectionally and fill multiple masked positions in parallel, instead of committing to a left-to-right token order.

GenMol follows the masked discrete diffusion formulation of masked diffusion language models. For a clean SAFE sequence $x = (x_1, \ldots, x_L)$, the forward process independently corrupts each token by interpolating between the clean token and a mask token m:

$$q(z_t^l \mid x_l) = \mathrm{Cat}\left(z_t^l;\, \alpha_t x_l + (1-\alpha_t)m\right),$$

where z_t^l is the noisy token at diffusion time t, and α_t decreases from $\alpha_0 = 1$ to $\alpha_1 = 0$. A BERT-style denoiser $x_\theta(z_t, t)$ predicts the clean token distribution at each position from the partially masked sequence. It is trained with the masked-diffusion negative-ELBO objective, which can be viewed as a time-weighted masked language modeling loss:

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{x,t,z_t}\left[-\sum_{l:z_t^l=m} \log x_\theta^l(x_l \mid z_t, t)\right].$$

During generation, GenMol starts from a masked SAFE sequence and simulates the reverse process. Unmasked tokens are kept fixed, while each masked position is predicted from $x_\theta(z_t, t)$ and sampled with temperature τ. GenMol then confirms only the most confident predictions and leaves the remaining positions masked for later denoising steps. With randomness r, the confidence score for a sampled token $\hat{x}_l$ is

$$c_t^l = \log p_\theta^l(\hat{x}_l \mid z_t, t) + rt\epsilon_l, \qquad \epsilon_l \sim \mathrm{Gumbel}(0,1).$$

This confidence-based parallel unmasking gives GenMol a native diffusion sampler rather than an autoregressive action sequence. All post-training methods in this experiment, therefore, use the coupled-GRPO instantiation described in the main text, which evaluates completed molecules under paired diffusion masks while preserving GenMol's native denoising sampler. The original-model baseline is the pretrained GenMol checkpoint without any RL post-training.

Reward definition. The rollout-level utility is the same QED–SA reward used in Section 5.2. For a generated molecule x, we compute QED (Bickerton et al., 2012) and the raw synthetic accessibility score $\mathrm{SA}(x)$ (Ertl & Schuffenhauer, 2009). QED is a normalized drug-likeness score that combines several medicinal-chemistry descriptors, including molecular weight, lipophilicity, hydrogen-bond donors and acceptors, polar surface area, rotatable bonds, aromatic rings, and structural alerts, into a single high-is-better value in $[0, 1]$. The raw SA score is a heuristic estimate of synthetic difficulty: it rewards molecules composed of common chemical fragments and penalizes structural complexity, such as large rings, stereochemical complexity, and unusual molecular topology. Since lower raw SA indicates easier synthesis, we use the high-is-better transformation $s_{\mathrm{SA}}(x) = \mathrm{clip}((6 - \mathrm{SA}(x))/5, 0, 1)$. The rollout utility is $u(x) = 0.6\,\mathrm{QED}(x) + 0.4\,s_{\mathrm{SA}}(x)$. For SGRPO, the group-level diversity reward is internal diversity over valid molecules, computed as one minus the mean pairwise Tanimoto similarity between Morgan fingerprints. Morgan fingerprints encode circular atom neighborhoods as binary substructure features, and Tanimoto similarity measures the overlap between two such feature sets. Thus, SGRPO optimizes the same molecule-level quality reward as GRPO, but adds an explicit finite-group estimate of sample diversity.

Baselines. We compare SGRPO against three baselines. The first is the pretrained GenMol model, denoted Original, which measures the utility-diversity frontier before RL post-training. The second is coupled-GRPO (Gong et al., 2025),

denoted GRPO in the small-molecule figures and tables, which uses the rollout utility reward but has no explicit set-level diversity reward. Directly applying GRPO to a masked diffusion generator is nontrivial because a completed sample does not come with the autoregressive factorization normally used to compute token-level policy ratios. Treating the whole completion as fully masked gives a weak likelihood proxy, while independently resampling masks gives high-variance estimates. Coupled-GRPO keeps the GRPO reward and advantage structure but changes how the diffusion policy ratio is estimated.

For a condition c, coupled-GRPO samples a group of completed molecules $\{x_i\}_{i=1}^{G}$ and computes a group-relative utility advantage

$$A_i = u(x_i) - \frac{1}{G}\sum_{j=1}^{G} u(x_j).$$

The update then uses a PPO-style clipped objective, but the log-probability proxy is computed through coupled diffusion masks. For each completed molecule x_i, coupled-GRPO samples timestep pairs $(t, \hat{t})$ with $t + \hat{t} = T$, and constructs complementary completion masks M_t and $\widehat{M}_{\hat{t}}$ such that $M_t^l + \widehat{M}_{\hat{t}}^l = 1$ for every completion token l. A compact way to write the corresponding log-probability proxy is

$$\ell_\theta(x_i; c) = \sum_{(t,\hat{t})} \left[\sum_{l: M_t^l = 1} \log p_\theta(x_i^l \mid c, x_i^{\neg M_t}, t) + \sum_{l: \widehat{M}_{\hat{t}}^l = 1} \log p_\theta(x_i^l \mid c, x_i^{\neg \widehat{M}_{\hat{t}}}, \hat{t}) \right],$$

where $x_i^{\neg M}$ denotes the visible, unmasked part of the completion under mask M. Thus, each token contributes exactly once across the coupled pair, but is evaluated under a realistic, partially observed context rather than an all-mask context. The policy ratio in the clipped GRPO objective is then formed from this coupled log-probability proxy. Coupled-GRPO therefore preserves the native diffusion denoising interface and reduces the variance of the policy-gradient estimate relative to independently sampled masks; in our baseline, it optimizes only rollout-level utility, without any set-level diversity reward.

The third baseline is memory-assisted coupled-GRPO, denoted Memory-Assisted GRPO, which augments coupled-GRPO with the memory unit of memory-assisted RL (Blaschke et al., 2020). We use the compound-similarity version of this memory: it is organized as index–bucket pairs, where each index is a molecule that represents a similarity neighborhood, and each bucket stores the high-scoring generated molecules assigned to that neighborhood. For a newly generated molecule x, the memory is queried only if its utility score exceeds a threshold η. The method then compares x with all indexed molecules using Morgan-fingerprint Tanimoto similarity. If no index has similarity at least γ, a new index–bucket pair is created with x, and x is stored in the new bucket. If x matches an existing index and the corresponding bucket has not reached its capacity C, x is added to that bucket and the reward is left unchanged. If the matched bucket is already full, the memory returns zero, and the reward is suppressed before GRPO advantage computation. Thus, Memory-Assisted GRPO encourages diversity indirectly by suppressing reward in historically saturated similarity neighborhoods, rather than by optimizing the internal diversity of the current generated set.

Training details. All RL-trained *de novo* small-molecule models start from the same GenMol checkpoint and are trained on 8 H200 GPUs with bf16 precision and seed 42. Training rollouts are sampled with random masking enabled, GenMol randomness 0.3, sampling temperature 1.0, minimum added length 60, generation batch size 1024, and one training iteration per rollout batch. GRPO uses 512 rollouts per prompt, a per-device training batch size of 1024, learning rate 5×10^{-5}, Adam $(\beta_1, \beta_2) = (0.9, 0.99)$, optimizer $\epsilon = 10^{-8}$, weight decay 0.1, maximum gradient norm 0.2, KL coefficient $\beta = 0.005$, clipping parameter $\epsilon_{\text{clip}} = 0.5$, cosine learning-rate decay with minimum learning-rate ratio 0.1, and warmup ratio 10^{-4}. The reference model is synchronized every 64 steps with a mixup coefficient 0.6. The reported GRPO model is the 2,000-step checkpoint. Memory-Assisted GRPO uses the same GRPO settings and additionally enables the history-based memory with bucket size 25, score threshold 0.9, and similarity cutoff 0.4; the reported model is also the 2,000-step checkpoint. SGRPO uses the same optimizer, KL, clipping, scheduler, and training-sampling settings, but keeps the total rollout budget matched by forming 8 same-condition groups with 64 rollouts per group. We set the group-reward weight to $\lambda = 0.5$ and use leave-one-out group credit at temperature 1.0. The reported SGRPO model is the 2,000-step checkpoint.

Evaluation protocol. For the main Pareto curve, we evaluate the four models under the paired GenMol decoding sweep $(\rho, \tau) \in \{(0.1, 0.5), (0.3, 0.8), (0.5, 1.1), (0.7, 1.4), (0.9, 1.7), (1.0, 2.0)\}$, where ρ is GenMol decoding randomness and

τ is sampling temperature. Each model generates 1,000 molecules at each sweep point. QED, transformed SA, and utility are averaged over valid generated molecules at that sweep point, and diversity is computed over the same valid set using Morgan-fingerprint Tanimoto distance. In practice, all generated molecules were valid under our decoding and postprocessing pipeline for every method and sweep point, so the valid set coincides with the full generated set in this experiment. The frontier metrics in Table 1 are computed from these six operating points for each method.

D.2. Pocket-Based Small-Molecule Design

Model. The pocket-based small-molecule experiment uses GenMol-P, the pocket-conditioned extension of GenMol described in Appendix C. GenMol-P conditions molecular denoising on a protein pocket by prepending a continuous pocket prefix, obtained from frozen ESM-IF1 pocket embeddings and a trainable projector, before the SAFE-token embeddings. The original GenMol-P model used in this experiment is the supervised CrossDocked2020 checkpoint at 5,500 optimization steps. All RL post-training methods start from this same checkpoint and use the coupled-GRPO family of objectives because the generator remains a discrete diffusion model.

Reward definition. The pocket-conditioned utility extends the *de novo* QED–SA reward with a target-dependent docking term. The QED and transformed SA components retain the same interpretation as in the unconditional small-molecule task: QED favors drug-like physicochemical profiles, while s_{SA} favors molecules estimated to be easier to synthesize. For ligand x and pocket $\mathcal{C}$, we define $u(x,\mathcal{C}) = 0.3\,\mathrm{QED}(x) + 0.2\,s_{\mathrm{SA}}(x) + 0.5\,s_{\mathrm{dock}}(x,\mathcal{C})$, where $s_{\mathrm{SA}}(x) = \mathrm{clip}((6-\mathrm{SA}(x))/5, 0, 1)$. The docking term is derived from AutoDock Vina (Trott & Olson, 2010), which searches ligand poses in the pocket and scores them with an empirical approximation to binding affinity. If $a_{\mathrm{Vina}}(x,\mathcal{C})$ is the raw Vina affinity, lower and more negative values indicate stronger predicted binding, so we use the high-is-better transformation $s_{\mathrm{dock}}(x,\mathcal{C}) = \mathrm{clip}(-a_{\mathrm{Vina}}(x,\mathcal{C})/10, 0, 1)$. Diversity is condition-specific: for each pocket, we compute internal diversity among the ligands generated for that same pocket using the Morgan-fingerprint Tanimoto distance, and then average this value over pockets. This avoids rewarding diversity across unrelated targets, where chemical dissimilarity may simply reflect different pocket requirements rather than useful within-pocket diversity.

Baselines. We compare SGRPO against the original supervised GenMol-P model and a coupled-GRPO model trained with the same QED–SA–Vina rollout utility. We do not include a memory-assisted GRPO baseline in the main pocket-based comparison. The memory mechanism stores historically generated candidates across optimization steps, but the scientific diversity objective in this task is pocket-conditional: diversity should be encouraged among ligands for the same pocket, not across ligands generated for unrelated pockets. Making a memory-assisted baseline operate at the pocket level would require repeated optimization steps on the same pocket so that the memory becomes meaningful, which would change the data exposure and make the comparison no longer matched.

Training details. Both pocket-conditioned RL methods are trained on 8 H200 GPUs with bf16 precision from the same 5,500-step GenMol-P checkpoint and seed 42. Training rollouts are sampled with random masking enabled, GenMol randomness 0.3, sampling temperature 1.0, minimum added length 60, generation batch size 384, and one training iteration per rollout batch. Coupled-GRPO uses 192 rollouts per pocket condition, per-device training batch size 384, learning rate 5×10^{-5}, Adam $(\beta_1, \beta_2) = (0.9, 0.99)$, optimizer $\epsilon = 10^{-8}$, weight decay 0.1, maximum gradient norm 0.2, KL coefficient $\beta = 0.005$, clipping parameter $\epsilon_{\mathrm{clip}} = 0.5$, cosine learning-rate decay with minimum learning-rate ratio 0.1, and warmup ratio 10^{-4}. The reference model is synchronized every 64 steps with a mixup coefficient 0.6. AutoDock Vina rewards are computed with fast search mode, one output pose, docking batch size 384, and a timeout of 1800 seconds. The reported GRPO checkpoint is trained for 1,000 steps. Coupled-SGRPO keeps the total rollout budget matched by forming 8 same-pocket groups with 24 rollouts per group. It uses the same optimizer, KL, clipping, scheduler, docking, and training-sampling settings as GRPO, sets the group-reward weight to $\lambda = 0.9$, and uses leave-one-out group credit at temperature 1.0. The reported SGRPO checkpoint is also trained for 1,000 steps.

Evaluation protocol. The main pocket-based Pareto curve uses pockets from the CrossDocked2020 test set. We evaluate each model under the same paired (ρ, τ) sweep as the *de novo* small-molecule experiment: $(0.1, 0.5)$, $(0.3, 0.8)$, $(0.5, 1.1)$, $(0.7, 1.4)$, $(0.9, 1.7)$, and $(1.0, 2.0)$. At each sweep point, each model generates 16 ligands for each of 100 test pockets, giving 1,600 ligand samples per method and sweep point. We scored generated ligands with QED, transformed SA, and AutoDock Vina. Utility metrics are averaged over valid generated ligands, while diversity is computed within each pocket's 16 ligands and then averaged over the 100 pockets. In practice, all generated ligands were valid under our decoding and

postprocessing pipeline for every method and sweep point, so validity does not affect the frontier comparison in this experiment. The frontier metrics in Table 1 are computed from the resulting six utility-diversity operating points.

D.3. *De novo* Protein Design

Model. The *de novo* protein design experiment uses ProGen2-small (Nijkamp et al., 2023), an autoregressive protein language model, as the base generator. ProGen2 models raw protein sequences with a decoder-only Transformer trained for next-token prediction. For an amino-acid sequence $y = (a_1, \ldots, a_L)$, the model defines a left-to-right sequence distribution

$$p_\theta(y) = \prod_{t=1}^{L+1} p_\theta(a_t \mid a_{<t}),$$

where a_{L+1} denotes the terminal token. Pretraining minimizes the corresponding negative log-likelihood over large collections of unaligned protein sequences, so generation is performed by repeatedly sampling the next amino-acid token from $p_\theta(\cdot \mid a_{<t})$ until the terminal token or a maximum length is reached.

Architecturally, ProGen2 is a causal Transformer decoder with rotary positional encodings and a parallelized residual block in which self-attention and the feed-forward network are applied to the same normalized hidden state.

$$h^{(m+1)} = h^{(m)} + \mathrm{Attn}(\mathrm{LN}(h^{(m)})) + \mathrm{MLP}(\mathrm{LN}(h^{(m)})).$$

The ProGen2 family scales this architecture across model sizes; we use the official ProGen2-small checkpoint, which has 151M parameters, 12 layers, 16 attention heads, head dimension 64, and context length 1024. This checkpoint is pretrained on the standard ProGen2 mixture of UniRef90 and BFD30 sequences, giving it broad coverage of natural protein sequence statistics. The original baseline is this checkpoint evaluated without RL post-training. Since ProGen2 provides an explicit token-level likelihood for every generated sequence, SGRPO can be instantiated with standard GRPO rather than the coupled-GRPO estimator required for diffusion generators. All post-training methods generate unconditional amino-acid sequences from the same prompt set and are evaluated at the 100-step checkpoint used in the main comparison.

Reward definition. The sequence-level utility combines four normalized protein-design scores, each targeting a different failure mode of unconstrained protein generation. Naturalness is computed from the average per-token log-likelihood under ESM2, so sequences that look more plausible under a large protein language model receive higher scores. Foldability is the ESMFold mean pLDDT score divided by 100; pLDDT is a per-residue confidence estimate, so this term favors sequences whose predicted structures are internally confident. Stability is based on TemBERTure-predicted melting temperature, which serves as a sequence-based proxy for thermal robustness. Developability combines Protein-Sol solubility with simple liability filters, so it penalizes sequences that may be hard to express or handle experimentally. Naturalness and stability are quantile-normalized using calibration sequences, while foldability and developability are already on a $[0, 1]$-compatible scale. The rollout utility is $u(y) = 0.25\, r_{\mathrm{nat}}(y) + 0.30\, r_{\mathrm{fold}}(y) + 0.20\, r_{\mathrm{stab}}(y) + 0.25\, r_{\mathrm{dev}}(y)$. For developability, the underlying score is $r_{\mathrm{dev}}(y) = 0.8\, r_{\mathrm{sol}}(y) + 0.2\, r_{\mathrm{liability}}(y)$, where r_{sol} is the Protein-Sol score clipped to $[0, 1]$ and $r_{\mathrm{liability}}$ penalizes transmembrane-like hydrophobicity, low sequence complexity, long hydrophobic runs, and cysteine outlier frequency. Sequence diversity is measured as one minus the average normalized Levenshtein similarity over valid generated sequences; the normalized Levenshtein similarity compares two sequences by their edit distance after accounting for sequence length, so lower similarity corresponds to larger sequence-level variation.

Baselines. We compare SGRPO against the original ProGen2-small generator, GRPO, and Memory-Assisted GRPO. GRPO optimizes only the rollout-level protein utility. Memory-Assisted GRPO uses the same rollout utility but adds the history-based diversity memory over generated amino-acid sequences; sequence similarity in this memory is normalized Levenshtein similarity. The low diversity of Memory-Assisted GRPO in the protein experiment should be interpreted as a consequence of its history-relative novelty mechanism rather than a difference in training budget: all RL methods start from the same ProGen2 checkpoint, use the same rollout utility and matched GRPO training settings, and are evaluated under the same temperature sweep.

Training details. All ProGen2 post-training runs are trained on 8 H200 GPUs and use the official ProGen2-small checkpoint as both the initial policy and the initial reference model. GRPO uses group size 96, per-device training batch size 192, gradient accumulation 1, maximum generation length 256, top-$p = 0.95$, training sampling temperature 0.8, learning

rate 5×10^{-5}, Adam $(\beta_1, \beta_2) = (0.9, 0.999)$, optimizer $\epsilon = 10^{-8}$, zero weight decay, maximum gradient norm 1.0, KL coefficient $\beta = 0.01$, and clipping parameter $\epsilon_{\text{clip}} = 0.2$. Memory-Assisted GRPO uses the same training settings and enables the history-based diversity memory with bucket size 25, score threshold 0.6, and similarity cutoff 0.6. In each supergroup, SGRPO keeps the rollout budget matched by forming 8 groups with 12 sequences per group. It uses the same optimizer, generation, KL, and clipping settings as GRPO, sets the group-reward weight to $\lambda = 0.8$, and uses leave-one-out group credit at temperature 1.0. For all three post-training methods, reward calibration uses 1,024 generated sequences. Naturalness, stability, and developability rewards are computed at every step, while the more expensive foldability reward is computed every 4 steps. Each model is trained 100 steps.

Evaluation protocol. The protein Pareto curve is evaluated with a temperature sweep $\tau \in \{0.1, 0.2, \ldots, 1.0, 1.1, 1.2\}$. At each temperature, each model generates 512 sequences. All generated sequences consisted of valid amino-acid tokens and passed our evaluation preprocessing checks, so validity was 100% for all methods across the full temperature sweep. Naturalness and stability are calibrated once using a fixed set of 256 calibration sequences. The resulting normalization parameters are then held fixed for all models and all temperatures in the sweep. The plotted utility is the weighted utility defined above, and the frontier metrics in Table 1 are computed from the 12 temperature-specific operating points for each method.