g-DPO: Scalable Preference Optimization for Protein Language Models

Constance Ferragu **Jonathan D. Ziegler** **Nicolas Deutschmann**
Arthur Lindoulsi **Eli Bixby** **Cradle ML Team**
Cradle
Zürich, Switzerland
constance@cradle.bio

Abstract

Direct Preference Optimization (DPO) is an effective approach for aligning protein language models with experimental design goals. However, DPO faces a scalability bottleneck: the number of possible training pairs grows quadratically with the number of labeled sequences, leading to prohibitive training times even for modestly sized datasets. We introduce g-DPO, a framework that (i) uses sequence space clustering to prune redundant pairs while preserving training signal, and (ii) amortizes likelihood computations with group-based approximations. Across three protein engineering tasks, g-DPO maintains *in-silico* and *in-vitro* performance that is statistically indistinguishable from standard DPO, while converging 1.8 to 3.7 times faster, with greater gains expected as the size of the dataset increases.

1 Introduction

Protein language models (PLMs), trained with self-supervised learning on large sequence datasets, capture relevant structure and function signals [14, 17, 25, 33]. To use these models for protein engineering tasks, such as optimizing binding affinity, thermostability, or catalytic activity, they are typically fine-tuned on labeled experimental datasets, many of which naturally define preferences (e.g., higher fluorescence is better, lower toxicity is better).

Direct Preference Optimization (DPO) is a natural fit for this setting. DPO fine-tunes a language model using pairwise preferences, maximizing the likelihood of preferred sequences relative to dispreferred ones [24], without requiring a separate reward model as in RLHF [6]. Recent studies have applied DPO to protein sequence design [2, 30, 32, 37], however, unlike in NLP, where explicit human preferences are available [3, 22], experimental protein datasets often provide scalar labels. Constructing preferences by exhaustively comparing all samples is infeasible because the number of pairs grows quadratically with dataset size. Moreover, not all comparisons are equally informative. Distant comparisons in sequence space tend to collapse into coarse binary signals, while local comparisons capture subtle and often non-additive effects of a few mutations [10, 12], providing a more valuable learning signal, which is crucial in late-stage optimization.

We introduce group-DPO (g-DPO), a DPO framework for experimentally labeled protein data that addresses these challenges. This study contributes: **(1) Scalable preference sampling:** Sequence-space clustering prunes redundant comparisons and focuses training on informative local neighborhoods in input space. **(2) Efficient training:** Building on this clustering, we exploit union masking to amortize log-likelihood computations across groups of sequences, further improving convergence time. **(3) Empirical validation:** Across three protein mutational landscapes, g-DPO maintains the *in-silico* and *in-vitro* performance of standard DPO, while converging 1.8 to 3.7 times faster, with larger convergence time gains expected as the size of the dataset increases.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: 2nd Workshop on Multi-modal Foundation Models and Large Language Models for Life Sciences.

2 Related work: Constructing pairs for DPO with PLMs

Direct Preference Optimization (DPO). DPO fine-tunes language models directly from pairwise comparisons, avoiding the need for a separate reward model [24]. Given a distribution $\mathcal{D}$ of pairs of sequences (y_w, y_l), where y_w (winner) is preferred over y_l (loser), a policy π_θ is optimized to increase the likelihood of sampling y_w over y_l under a regularization constraint formulated as a bound on the KL divergence between π_θ and a reference policy π_{ref}. This objective is expressed as a loss function:

$$\mathcal{L}_{\text{DPO}}(\pi_\theta, \pi_{\text{ref}}) = -\mathbb{E}_{(y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\Big(\beta\log\frac{\pi_\theta(y_w)}{\pi_{\text{ref}}(y_w)} - \beta\log\frac{\pi_\theta(y_l)}{\pi_{\text{ref}}(y_l)}\Big)\right], \tag{1}$$

where $\beta > 0$ is the regularization hyperparameter and σ is the sigmoid function. A complete derivation is provided in A.1. A primary challenge in applying DPO to PLMs is constructing a preference distribution $\mathcal{D}$ from scalar labels, as exhaustively pairing labeled sequences does not scale.

2.1 Preference sampling for DPO with PLMs

Output space partitioning. A common strategy is to partition labeled sequences by thresholds defined by design objectives (e.g., "get sequences above x stability") or quantiles, and assign sequences above as "preferred" and sequences below as "dispreferred" [20, 26, 34–36]. In terms of complexity, subsampling schemes, such as fixing k losers per winner, can be used and scale effectively with dataset size. Although this aligns well for coarse optimization goals, treating all pairs within the same partition as equivalent reduces resolution among high-performing sequences, which is limiting in late-stage optimization.

Rank-space sampling. Unlike thresholding, which collapses labels into binary partitions, rankings preserve relative relationships across all sequences. Methods such as RRHF [38] and PRO [28] show that training on rankings (or their induced pairs) leads to a more stable convergence in aligning LMs with human preferences. Applied to proteins, CtrlProt [15] shows that a rank-wise preference objective improves controllability over pairwise methods for multi-objective optimization. Widatalla et al. [32] evaluate a 'gap level' heuristic that down-samples pairs of sequences at fixed distances in rank space, and find that most gap levels have comparable performance to randomly sampling pairs. This suggests that not all pairs are equally informative and that randomly pruning pairs in the output space risks diluting the training signal.

Informative pairs. Other methods have been proposed to prioritize preference pairs that provide a stronger training signal. Maru et al. [18] propose a hybrid sampling strategy that mixes global and local perturbations, selecting pairs with large predicted stability differences for the former and pairs ten mutations apart for the latter. They find no improvement over randomly selecting pairs. Beyond proteins, other strategies have been proposed to improve informativeness, such as stratified sampling to ensure coverage across score bins, oversampling rare but informative regions, or curriculum learning to gradually introduce harder comparisons as the model improves [4, 7, 11, 13, 23].

3 Methods: Group-DPO (g-DPO)

In NLP, preferences are inherently structured: multiple outputs are generated from the same prompt, and annotators decide which is better [3, 22]. This provides a shared context that makes likelihood ratios between outputs meaningful. On the other hand, comparing outputs from different prompts yields likelihoods ratios that are dominated by prompt differences rather than output preference. Proteins have a related dynamic: even when variants come from the same wild type, comparing distant variants conflates the effects of many mutations, collapsing into coarse signals, whereas local comparisons better capture subtle, non-additive effects [10, 12]. Our work, therefore, leverages proximity in sequence space to construct preference pairs, enabling the model to learn fine-grained mutational effects while reducing the number of training pairs.

Our g-DPO training framework is conducted in three stages. We begin with unsupervised fine-tuning of a PLM on evolutionarily related sequences of a wild type protein (evo-tuning) [1, 8]. Next, given an experimental mutant dataset of the relevant wild type, we apply union mask clustering, a greedy agglomerative procedure that groups sequences based on shared mutational positions.

Finally, we sample groups of sequences from each cluster and evaluate the DPO loss over all preferences within each group to further fine-tune our evo-tuned model. Although some assay variants could appear within the evo-tuning training data, this does not constitute data leakage as evo-tuning uses no assay labels. This unsupervised design choice is intended to provide general evolutionary context to the model, opposed to labeled samples for model alignment.

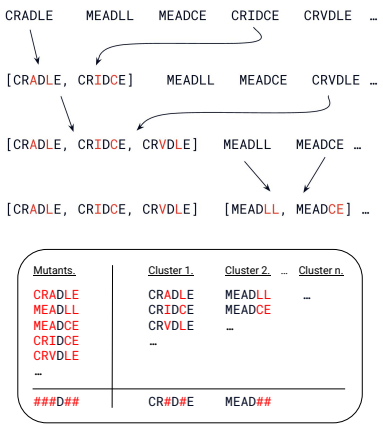


Figure 1: Clustering of mutants with overlapping mutations.

3.1 Union mask clustering

Given a set of aligned sequences $S = (s^1, \ldots, s^n),\ s^i \in \mathcal{A}^L$, over the amino acid alphabet $\mathcal{A}$, define the *union mask* as $M(S) = \{i \in [\![L]\!] : \exists j, k \text{ such that } s_i^{(j)} \neq s_i^{(k)}\}$, and let $m(S) := |M(S)|$ denote the size of the union mask. We cluster sequences with a greedy agglomerative procedure with a linkage that prefers merges that keep the union masks small:

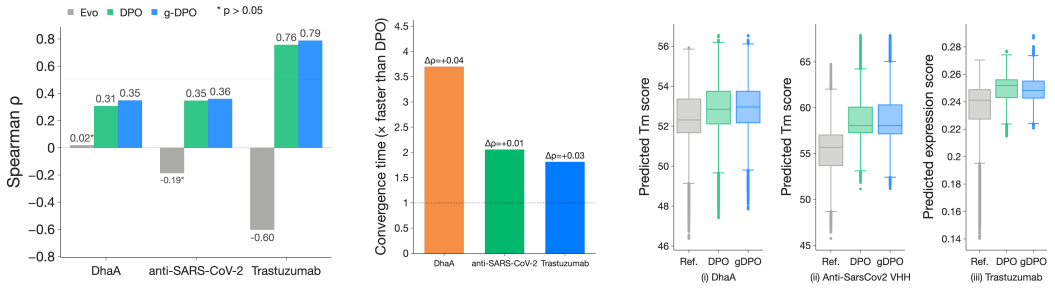
1. **Initialization.** Let $\mathcal{C} \leftarrow \{\{s^1\}, \ldots, \{s^n\}\}$. For each cluster $C \in \mathcal{C}$, store its union mask $M(C)$.
2. **Linkage.** For clusters C_i, C_j, define the cost of merging C_j into C_i as
$$\phi(C_i, C_j) = m(C_i \cup C_j) - m(C_i) = \big|M(C_i) \cup M(C_j) \cup M(\{s, s'\})\big| - m(C_i),$$
where $s \in C_i$ and $s' \in C_j$ are arbitrarily chosen.
3. **Greedy merge.** Repeatedly select
$$(C_p, C_q) = \arg\min_{i \neq j} \phi(C_i, C_j),$$
breaking ties by minimum $m(C_i)$. Replace clusters C_p and C_q by $C_p \cup C_q$, and store the new union mask $M(C_p \cup C_q) = M(C_p) \cup M(C_q) \cup M(\{s, s'\})$.
4. **Stopping.** Stop when the next best merge would violate the union mask ratio threshold $\tau \in [0, 1]$,
$$\min_{i \neq j} \phi(C_i, C_j) > \tau L.$$

Time complexity. Calculating $\phi(C_i, C_j)$ is $O(L)$, since $M(C_i)$ and $M(C_j)$ are stored, and only $M(s, s')$ must be computed. In each merging step, computing all costs ϕ is $O(n^2 L)$. Using a heap over cluster pairs, the final time complexity is $O(n^2 \log n + n^2 L)$. When n is large, we first coarse-cluster with MMseqs2 [29], which provides approximate linear-time clustering for large datasets, and then run union mask clustering independently within those buckets.

3.2 Group sampling

Log likelihood approximation. Evaluating the DPO loss on a pair (y_w, y_l) requires likelihoods for both sequences. For masked language models (MLMs) like ESM-2 [14], computing pseudo log-likelihood (PLL) [27] scores has become the standard way to approximate sequence likelihoods. This requires one forward pass per position and is therefore expensive. To reduce this cost, we exploit the union mask $D = M(\{y_w, y_l\})$, and create a jointly masked input $y_{\setminus D}$, where all differing positions between y_w and y_l are masked. One forward pass on $y_{\setminus D}$ provides logits for all positions. We then approximate $\log p(y_w) \approx \sum_{i \in D} \log p(y_{wi} | y_{\setminus D})$ (same for y_l). Positions outside D do not need to be evaluated since their logits agree. This is equivalent to a mean-field approximation, where we assume that, conditional on the observed tokens, the masked positions are independent. Although approximate, it is reliable when the differing positions are few compared to the sequence length, since the error scales linearly with the size of the difference mask. Similar multi-mask approximations have already been applied successfully in prior work [5, 19, 21, 39].

Since union-mask clustering controls the union-mask size of clusters, we can extend this approximation beyond pairs. We uniformly sample without replacement groups of g sequences from each cluster. We can then approximate the likelihood of each sequence in a group with a single forward pass. We then evaluate the DPO loss over all pairwise comparisons inside the group. Sampling continues until every sequence has appeared in at least one group, which defines one training epoch.



(a) (b) (c)

Figure 2: *In-silico* evaluation of g-DPO. (a) Spearman correlation (ρ) of evo-tuned, DPO, and g-DPO models across the three datasets. (b) Relative convergence speed of g-DPO compared to DPO, measured on a single NVIDIA A100 GPU, with $\Delta\rho$ indicating performance differences. (c) Predicted property distributions of sequences generated from the reference, DPO, and g-DPO models using beam search. Kolmogorov-Smirnov tests confirm that g-DPO and DPO yield nearly identical, and improved distributions over the reference.

4 Experiments

Evaluating generative models for protein optimization is challenging and ultimately requires experimental evidence. Our goal is to test whether g-DPO (i) preserves or improves model quality relative to DPO while (ii) reducing training cost, and (iii) whether *in-silico* improvements translate to experimental outcomes. We evaluate g-DPO on experimentally measured variants derived from three wild type proteins:

- **Anti-SARS-CoV-2 VHH** [9]: thermostability ($N = 462$, mutation breadth 47.1%).
- **Trastuzumab scFv**: expression ($N = 76$, mutation breadth 13.1%).
- **Haloalkane dehalogenase (DhaA)**: thermostability ($N = 474$, mutation breadth 40.3%).

We use ESM-2-650 [14] as our pre-trained model. Additional dataset and training details are provided in Appendix C.

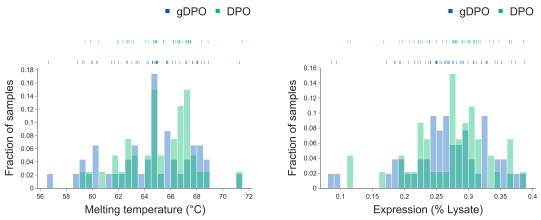
4.1 *In-silico* validation

For each dataset, we report (i) rank correlation between PLL scores and experimental measurements on holdout test sets, which tests whether model likelihoods align with ground truth preferences; (ii) convergence time to early stopping, to measure training efficiency; and (iii) generative quality, measured by scoring sequences generated with beam search [31]. We score the generated sequences with independent predictors trained on the same experimental datasets.

Both DPO and g-DPO improve rank correlation relative to the evo-tuned reference model (Figure 2a) and shift the distribution of generated sequences toward better predicted function (Figure 2c). The distributions produced by DPO and g-DPO are statistically indistinguishable, indicating that g-DPO preserves generative quality while reducing training cost (Figure 2b). Statistical significance was assessed with two-sample KS tests, with full results reported in Table 2.

4.2 *In-vitro* validation

To test whether *in-silico* improvements translate to experimental outcomes, we validated a subset of generated sequences in the lab for two of the three optimization tasks: thermostability of DhaA and expression of Trastuzumab. Candidates were



(a) (b)

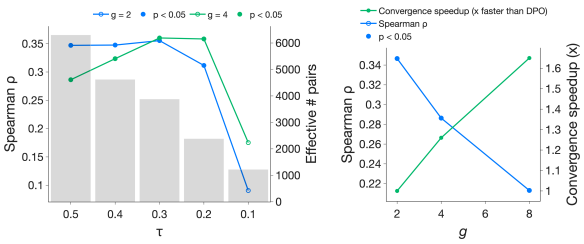
Figure 3: *In-vitro* validation. Distribution of assay measurements for sequences designed with g-DPO and DPO on (a) thermostability of DhaA and (b) expression of Trastuzumab. Distributions show that both models yield comparable outcomes.

down-selected from the *in-silico* generation pool using a criterion that averages performance across the top-k predicted sequences ($k = 3$), following the Monte Carlo sampling strategy of DiscoBAX [16]. Assay results confirm that DPO and g-DPO yield comparable experimental outcomes, with no significant differences observed between their distributions.

4.3 Ablations

We ablate clustering (τ) and grouping (g) to understand their individual effects and how they interact. All ablations are reported on the Anti-SARS-CoV-2 VHH dataset.

Clustering, no grouping. We sweep the clustering threshold τ from $0.5 \rightarrow 0.1$, where a smaller τ yields more clusters, reduces the number of training pairs and speeds up convergence. In Figure 4a with $g = 2$, we see that as τ decreases to ≈ 0.3, model performance remains unchanged. Beyond this threshold, performance declines as clusters become too tight and useful signal is lost. This indicates that many pairs are redundant and can be safely pruned in input space up to a moderate threshold ($\tau \approx 0.3$).



(a) (b)

Figure 4: (a) Effect of clustering threshold τ on performance and training pairs. (b) Effect of grouping size g on performance and convergence speed without clustering.

Grouping, no clustering. Grouping amortizes likelihood computations by sharing a union mask and a single forward pass across g sequences, but comes at a cost. When mutations span a large fraction of the sequence, union masks become too large, likelihoods are overly approximated, and model performance drops (Figure 4b). Hence, grouping alone is insufficient when the mutation span is high.

Clustering with grouping. Combining clustering with grouping ($g = 4$ in Figure 4a) yields the expected trade-off. With little to no clustering, grouping degrades model performance, as previously mentioned. As the clustering threshold increases, performance improves and eventually matches that of $g = 2$ at $\tau \approx 0.3$, while convergence is significantly faster for $g = 4$. Beyond $\tau \approx 0.3$, performance drops as clustering becomes too strict and valuable training signal is lost. This drop is less severe for $g = 4$, probably because within-group preferences are processed in one batch, with coupled likelihoods and update steps, while with $g = 2$, they can be split across batches. Practically, clustering controls the training signal (removing redundant pairs), and grouping controls efficiency; however, accurate grouping is dependent on a reasonable clustering level for the likelihood approximation to hold.

5 Conclusion, limitations and future work

We present g-DPO, a scalable variant of DPO for PLMs, that combines sequence-space clustering, to prune redundant preference pairs, with grouped likelihood amortization, to reduce computation. Across three protein engineering tasks, g-DPO matches DPO performance on both *in-silico* metrics and *in-vitro* results, while converging up to 3.7× faster. In practice, moderate clustering with grouping yields the best trade-off; excessive clustering removes training signal, and grouping alone can degrade performance when mutation span is large and likelihood approximations break down.

Our experiments span three mutational landscapes with 76-474 variants, representative of typical assay-driven protein engineering datasets. However, extending evaluation to larger and more diverse benchmarks (multiple wild types, broader mutation spans) would allow to further test the scalability and robustness of g-DPO. We expect the scalability advantage of g-DPO to become even more important in such settings. A natural extension is to move beyond a pairwise objective and explore rank-based objectives which could leverage the full ranking information of variants within clusters. Moreover, while this study focused on unimodal protein sequence PLMs, the clustering and grouping strategies are modality-agnostic. Extending g-DPO to multi-modal foundation models (e.g., sequence-structure or sequence-function) is a promising direction for future work.

References

[1] Ethan C Alley, Grigory Khimulya, Surojit Biswas, Mohammed AlQuraishi, and George M Church. Unified rational protein engineering with sequence-based deep representation learning. *Nature methods*, 16(12):1315–1322, 2019.

[2] Mohammad Gheshlaghi Azar, Mark Rowland, Bilal Piot, Daniel Guo, Daniele Calandriello, Michal Valko, and Rémi Munos. A General Theoretical Paradigm to Understand Learning from Human Preferences, November 2023.

[3] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[4] Shihao Cai, Chongming Gao, Yang Zhang, Wentao Shi, Jizhi Zhang, Keqin Bao, Qifan Wang, and Fuli Feng. K-order ranking preference optimization for large language models. *arXiv preprint arXiv:2506.00441*, 2025.

[5] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11315–11325, 2022.

[6] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

[7] Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, Nicu Sebe, and Mubarak Shah. Curriculum direct preference optimization for diffusion and consistency models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 2824–2834, 2025.

[8] Cade Gordon, Amy X Lu, and Pieter Abbeel. Protein language model fitness is a matter of preference. *bioRxiv*, pages 2024–10, 2024.

[9] Leo Hanke, Laura Vidakovics Perez, Daniel J Sheward, Hrishikesh Das, Tim Schulte, Ainhoa Moliner-Morro, Martin Corcoran, Adnane Achour, Gunilla B Karlsson Hedestam, B Martin Hällberg, et al. An alpaca nanobody neutralizes sars-cov-2 by blocking receptor interaction. *Nature communications*, 11(1):4420, 2020.

[10] Yuuki Hayashi, Takuyo Aita, Hitoshi Toyota, Yuzuru Husimi, Itaru Urabe, and Tetsuya Yomo. Experimental rugged fitness landscape in protein sequence space. *PLoS One*, 1(1):e96, 2006.

[11] Junyu Hou. De novo molecular design enabled by direct preference optimization and curriculum learning. *arXiv preprint arXiv:2504.01389*, 2025.

[12] Milo S Johnson, Gautam Reddy, and Michael M Desai. Epistasis and evolution: recent advances and an outlook for prediction. *BMC biology*, 21(1):120, 2023.

[13] Xiaoqiang Lin, Arun Verma, Zhongxiang Dai, Daniela Rus, See-Kiong Ng, and Bryan Kian Hsiang Low. Activedpo: Active direct preference optimization for sample-efficient alignment. *arXiv preprint arXiv:2505.19241*, 2025.

[14] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.

[15] Xiangyu Liu, Yi Liu, Silei Chen, and Wei Hu. Controllable protein sequence generation with llm preference optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 505–513, 2025.

[16] Clare Lyle, Arash Mehrjou, Pascal Notin, Andrew Jesson, Stefan Bauer, Yarin Gal, and Patrick Schwab. Discobax: Discovery of optimal intervention sets in genomic experiment design. In *International Conference on Machine Learning*, pages 23170–23189. PMLR, 2023.

[17] Ali Madani, Ben Krause, Eric R Greene, Subu Subramanian, Benjamin P Mohr, James M Holton, Jose Luis Olmos Jr, Caiming Xiong, Zachary Z Sun, Richard Socher, et al. Large language models generate functional protein sequences across diverse families. *Nature biotechnology*, 41(8):1099–1106, 2023.

[18] Nahum Maru. Learning new biophysical controls in protein language models via supervised and preference-based fine-tuning. 2025.

[19] Joshua Meier, Roshan Rao, Robert Verkuil, Jason Liu, Tom Sercu, and Alex Rives. Language models enable zero-shot prediction of the effects of mutations on protein function. *Advances in neural information processing systems*, 34:29287–29303, 2021.

[20] Pouria Mistani and Venkatesh Mysore. Preference optimization of protein language models as a multi-objective binder design paradigm. *arXiv preprint arXiv:2403.04187*, 2024.

[21] Hunter Nisonoff, Junhao Xiong, Stephan Allenspach, and Jennifer Listgarten. Unlocking guidance for discrete state-space diffusion and flow models. *arXiv preprint arXiv:2406.01572*, 2024.

[22] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[23] Pulkit Pattnaik, Rishabh Maheshwary, Kelechi Ogueji, Vikas Yadav, and Sathwik Tejaswi Madhusudhan. Curry-dpo: Enhancing alignment using curriculum learning & ranked preferences. *arXiv preprint arXiv:2403.07230*, 2024.

[24] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Thirty-Seventh Conference on Neural Information Processing Systems*, November 2023.

[25] Roshan M Rao, Jason Liu, Robert Verkuil, Joshua Meier, John Canny, Pieter Abbeel, Tom Sercu, and Alexander Rives. Msa transformer. In *International conference on machine learning*, pages 8844–8856. PMLR, 2021.

[26] Dingyi Rong, Haotian Lu, Wenzhuo Zheng, Fan Zhang, Shuangjia Zheng, and Ning Liu. Enerbridge-dpo: Energy-guided protein inverse folding with markov bridges and direct preference optimization. *arXiv preprint arXiv:2506.09496*, 2025.

[27] Julian Salazar, Davis Liang, Toan Q Nguyen, and Katrin Kirchhoff. Masked language model scoring. *arXiv preprint arXiv:1910.14659*, 2019.

[28] Feifan Song, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang. Preference ranking optimization for human alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18990–18998, 2024.

[29] Martin Steinegger and Johannes Söding. Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature biotechnology*, 35(11):1026–1028, 2017.

[30] Filippo Stocco, Maria Artigues-Lleixa, Andrea Hunklinger, Talal Widatalla, Marc Guell, and Noelia Ferruz. Guiding generative protein language models with reinforcement learning, 2025.

[31] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

[32] Talal Widatalla, Rafael Rafailov, and Brian Hie. Aligning protein generative models with experimental fitness via Direct Preference Optimization, May 2024.

[33] Yijia Xiao, Wanjia Zhao, Junkai Zhang, Yiqiao Jin, Han Zhang, Zhicheng Ren, Renliang Sun, Haixin Wang, Guancheng Wan, Pan Lu, et al. Protein large language models: A comprehensive survey. *arXiv preprint arXiv:2502.17504*, 2025.

[34] Junde Xu, Zijun Gao, Xinyi Zhou, Jie Hu, Xingyi Cheng, Le Song, Guangyong Chen, Pheng-Ann Heng, and Jiezhong Qiu. Protein inverse folding from structure feedback. *arXiv preprint arXiv:2506.03028*, 2025.

[35] Fanglei Xue, Andrew Kubaney, Zhichun Guo, Joseph K Min, Ge Liu, Yi Yang, and David Baker. Improving protein sequence design through designability preference optimization. *arXiv preprint arXiv:2506.00297*, 2025.

[36] Fanglei Xue, Andrew Kubaney, Zhichun Guo, Joseph K. Min, Ge Liu, Yi Yang, and David Baker. Improving protein sequence design through designability preference optimization, 2025.

[37] Jason Yang, Wenda Chu, Daniel Khalil, Raul Astudillo, Bruce J Wittmann, Frances H Arnold, and Yisong Yue. Steering generative models with experimental data for protein fitness optimization. *arXiv preprint arXiv:2505.15093*, 2025.

[38] Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. Rrhf: Rank responses to align language models with human feedback without tears. *arXiv preprint arXiv:2304.05302*, 2023.

[39] Kaiwen Zheng, Yongxin Chen, Hanzi Mao, Ming-Yu Liu, Jun Zhu, and Qinsheng Zhang. Masked diffusion models are secretly time-agnostic masked models and exploit inaccurate categorical sampling. *arXiv preprint arXiv:2409.02908*, 2024.

A Preliminaries

A.1 Direct Preference Optimization (DPO) Overview

Given a pre-trained reference model π_{ref}, the goal of DPO is to learn a policy π_θ that maximizes the probability of generating preferred sequences over dispreferred ones.

Consider a dataset of N pairwise rankings, $D = \{(x^{(i)}, y_w^{(i)}, y_l^{(i)})\}_{i=1}^N$, where $x^{(i)}$ is the masked input and $y_w^{(i)}$ and $y_l^{(i)}$ are the preferred and dispreferred outputs. The preference distribution p^* can be expressed given a Bradley-Terry reward model r^* and a parameterization of the score $s(x, y) = \exp(r^*(x, y))$,

$$p^*(y_w \succ y_l \mid x) = \frac{\exp(r^*(x, y_w))}{\exp(r^*(x, y_w)) + \exp(r^*(x, y_l))} = \sigma(r^*(x, y_w) - r^*(x, y_l))$$

Under the RLHF framework, a parameterized reward model $r_\Phi(x, y)$ can be estimated to match the optimal reward function π_*, using the negative log-likelihood loss,

$$L_R(r_\Phi, D) = -\mathbb{E}_{(x,y_w,y_l)\sim D}\left[\log \sigma(r_\Phi(x, y_w) - r_\Phi(x, y_l))\right]$$

This reward model can then be used to optimize the optimal policy,

$$\pi^* = \arg\max \mathbb{E}_{x\in D, y\sim\pi_\theta}[r_\Phi(x, y)] - \beta D_{\text{KL}}[\pi_\theta(y \mid x) \| \pi_{\text{ref}}(y \mid x)]$$

The KL term is there to ensure that the outputs from the learned policy do not diverge too much from the original policy, as we assume that the fine-tuned base model already produces reliable outputs. Using Gibbs' inequality, we can derive the optimal solution to this optimization problem,

$$\pi^*(y \mid x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y \mid x) e^{\frac{1}{\beta} r_\Phi(x,y)}$$

where $Z(x) = \sum_y \pi_{\text{ref}}(y \mid x) e^{\frac{1}{\beta} r_\Phi(x,y)}$, an untractable term. We can reorganize this term to solve for r_Φ or r^* with its corresponding policy π^*, which we can plug back into (2) to obtain,

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(y_w, y_l)\sim D}\left[\log \sigma\big(\beta \log \tfrac{\pi_\theta(y_w)}{\pi_{\text{ref}}(y_w)} - \beta \log \tfrac{\pi_\theta(y_l)}{\pi_{\text{ref}}(y_l)}\big)\right]$$

where σ is the sigmoid function and β is a scaling parameter.

This objective encourages the policy to increase the likelihood of preferred sequences relative to dispreferred ones, while the reference model serves as a regularizer to prevent the policy from deviating too far from the original distribution.

B g-DPO Details

B.1 Union mask clustering algorithm

Algorithm 1 Greedy Union-Mask Clustering

Require: Sequences $\{s^1, \ldots, s^n\}, s^i \in \mathcal{A}^L$; length L; union-mask ratio threshold $\tau \in [0, 1]$.
Ensure: A set of clusters $\mathcal{C}$.

Initialization: $\mathcal{C} \leftarrow \big\{\{s^1\}, \ldots, \{s^n\}\big\}$.
for all $C \in \mathcal{C}$ **do**
    Store its union mask $M(C)$ and its size $m(C) = |M(C)|$.
end for
function $\phi(C_i, C_j)$ ▷ Merge cost of C_j into C_i
    Choose arbitrary $s \in C_i$, $s' \in C_j$.
    return $|\, M(C_i) \cup M(C_j) \cup M(\{s, s'\}) \,| - m(C_i)$.
end function
while true **do**
    $(p, q) \leftarrow \arg\min_{i \neq j} \big(\phi(C_i, C_j)\big)$ ▷ Tie-break: smaller $m(C_i)$
    $c^\star \leftarrow \phi(C_p, C_q)$
    if $c^\star > \tau L$ **then** ▷ Stopping rule
        break
    end if
    Choose arbitrary $s \in C_p$, $s' \in C_q$.
    $C_{\text{new}} \leftarrow C_p \cup C_q$
    $M(C_{\text{new}}) \leftarrow M(C_p) \cup M(C_q) \cup M(\{s, s'\})$
    $m(C_{\text{new}}) \leftarrow |M(C_{\text{new}})|$
    $\mathcal{C} \leftarrow \big(\mathcal{C} \setminus \{C_p, C_q\}\big) \cup \{C_{\text{new}}\}$
end while
return $\mathcal{C}$

B.2 Single forward pass log-likelihood approximation algorithm

The DPO loss requires log-likelihood differences between sequences. By constructing a union mask for a group G, all sequences within G are evaluated under the same masked context, making their likelihoods directly comparable. This ensures that the log-likelihood ratios used in the loss are valid. Importantly, the approximation only holds for sequences scored under the same union mask.

Algorithm 2 Union-mask LL approximation

Require: Cluster $C \subset \mathcal{C}$; group $G \subset C$ of size g; MLM model π; mask token `[MASK]`

Ensure: Per-sequence approximate likelihoods $\{\widetilde{\log p_\pi}(y)\}_{y \in G}$

function UNIONMASK(G)
  return $D \leftarrow \{\, p \in [\![L]\!] \mid \exists\, y, y' \in G :\ y_p \neq y'_p \,\}$
end function
function MASK(y, D)
  $x \leftarrow y;\quad x_p \leftarrow$ `[MASK]` for all $p \in D$
  return x
end function
function LIKELIHOODS(G, π)
  $D \leftarrow$ UNIONMASK(G)
  Choose arbitrary $y \in G$
  $x \leftarrow$ MASK(y, D)
  $\text{logits} \leftarrow \pi(x)$ ▷ single forward pass; returns token logits for all positions
  for all $y \in G$ **do**
    $\widetilde{\log p_\pi}(y) \leftarrow \sum_{p \in D} \log \mathrm{softmax}(\text{logits}_p)[\, y_p \,]$
  end for
  return $D,\ \{\widetilde{\log p_\pi}(y)\}_{y \in G}$
end function

B.3 Training speedup

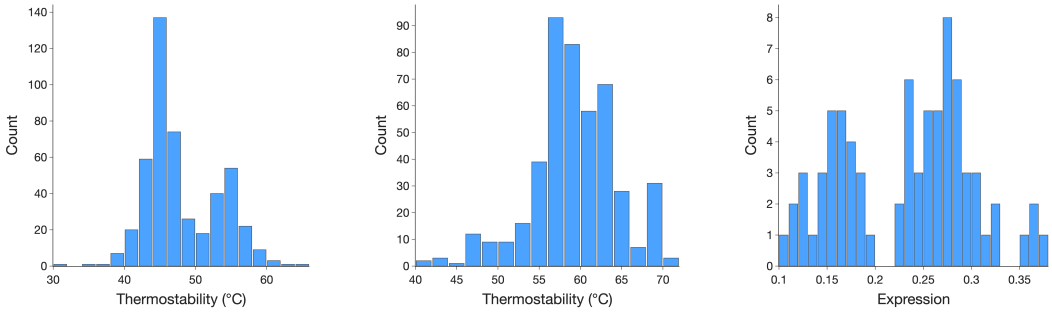
g-DPO improves training efficiency in two ways. (1) Clustering prunes pairs that are redundant or less informative, reducing the quadratic growth of pairs and focusing updates on local neighborhoods that provide stronger training signal. (2) Grouping amortizes likelihood evaluations: for a group of size g, all $\binom{g}{2}$ pairwise preferences are computed from a single forward pass under the shared union mask, rather than one forward pass per pair. For example, with $g = 4$, 6 pairs are obtained from one forward pass instead of 6 forward passes. These two mechanisms together reduce both the number of pairs processed and the cost per pair, resulting in 1.8–$3.7\times$ faster convergence in our experiments. As datasets grow, the number of redundant pairs increases, but clustering should discard them. Consequently, the relative efficiency gain of g-DPO compounds with dataset size.

C Experiment details

C.1 Training details

We start with pre-trained ESM-2-650 (weights are open-sourced) [14], which we fine-tune on evolutionarily related sequences of our wild type sequence for each corresponding dataset [1, 8]. We follow the evo-tuning framework defined by [1]. Then, this model is further fine-tuned using the g-DPO framework with SGD using a learning rate of 7×10^{-4} and no weight decay, with a 300-step linear warmup. The DPO loss is computed with $\beta = 0.04$ with a batch size of 64. Sequences were clustered with a maximum union mask size of $0.3L$, and groups of $g = 4$ were sampled from each cluster. The loss was applied over all within-group pairs. This was the final configuration, though we ran hyperparameter sweeps and expect the optimal group size to depend on dataset scale and clustering. Validation loss was monitored every 250 steps, with checkpointing on validation loss and relative early stopping on validation loss that checks there has not been more than a 1% improvement with a patience of 3 validation intervals. Training was performed on a single NVIDIA A100 GPU.

C.2 Datasets



(a) DhaA (b) Anti-SARS-CoV-2 VHH (c) Trastuzumab scFv

Figure 5: Dataset distributions.

anti-SARS-CoV-2 VHH: The wild type[9] already expresses well and has a binding affinity in the single-digit nanomolar range.

DhaA: Haloalkane dehalogenase catalyzes the hydrolysis of halogenated compounds by cleavage of the carbon-halogen bond.

Trastuzumab scFv: An scFv version of the cancer drug Trastuzumab, which expresses poorly and has a binding affinity of 3–5 nanomolar.

Protein	Function	N	Mutation breadth	WT measurement
anti-SARS-CoV-2 VHH	Thermostability	462	47.1%	58.67 C
Trastuzumab scFv	Expression	76	13.1%	0.28
Haloalkane dehalogenase	Thermostability	474	40.3%	45.09 C

Table 1: Optimized function, dataset size (N), mutation breadth (% of positions mutated), and wild type measurement for datasets used in experiments.

D Additional results

D.1 *In-silico* results

We report Kendall's τ as a complementary rank correlation metric to Spearman ρ. While Spearman captures monotonic correlation, Kendall's τ measures the fraction of concordant versus discordant pairs and is therefore more sensitive to local ordering differences. Formally, it is defined as

$$\tau = \frac{(\#\text{concordant pairs}) - (\#\text{discordant pairs})}{\#\text{total pairs}}.$$

We report additional results with Kendall's τ in Figure 6 and Figure 7. The consistent trends across Spearman ρ and Kendall's τ confirm that our conclusions are robust to the choice of ranking measure.

D.2 Statistical significance tests

We include two-sample Kolmogorov-Smirnov (KS) tests on the predicted property distributions of generated sequences (see Table 2). While the KS tests indicate statistically significant differences between DPO and g-DPO, the effect sizes are very small, showing that the distributions are nearly identical in practice. This supports our conclusion that the computational gains of g-DPO do not come at the expense of model quality.

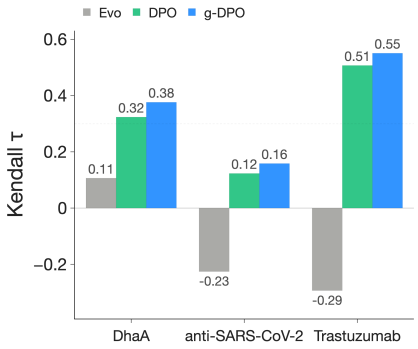


Figure 6: Kendall tau (τ) correlation between PLL scores and experimental measurements on holdout test sets for the evo-tune, DPO, and g-DPO models.

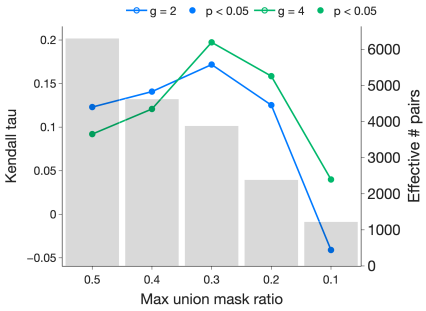


Figure 7: Effect of clustering threshold τ on model performance and number of effective training pairs. Moderate clustering prunes redundant pairs without loss of performance.

Dataset	**Pair**	D	**p-value**	**Statistic location**	**Sign**
DhaA	g-DPO - DPO	0.0290	< 0.01	51.90	-1
DhaA	g-DPO - Ref	0.1736	< 0.01	52.49	-1
DhaA	DPO - Ref	0.1851	< 0.01	52.07	-1
Ty1	g-DPO - DPO	0.0385	< 0.01	56.99	+1
Ty1	g-DPO - Ref	0.5325	< 0.01	56.89	-1
Ty1	DPO - Ref	0.5695	< 0.01	56.90	-1
Her2	g-DPO - DPO	0.0829	< 0.01	0.2506	+1
Her2	g-DPO - Ref	0.3477	< 0.01	0.2408	-1
Her2	DPO - Ref	0.3587	< 0.01	0.2411	-1

Table 2: Two-sample Kolmogorov-Smirnov tests comparing predicted property distributions between models. The KS statistic D reports the maximum distance between empirical CDFs; smaller values indicate closer distributions. p-values below 0.05 indicate statistically significant differences.