Guiding Generative Protein Language Models with Reinforcement Learning

Filippo Stocco[1,2], Maria Artigues-Lleixà[1,2], Andrea Hunklinger[2,3], Talal Widatalla[4,5], Marc Güell[1,6], Noelia Ferruz[2,1,*]

[1]Department of Medicine and Life Sciences, Universitat Pompeu Fabra, Barcelona, Spain
[2]Centre for Genomic Regulation, the Barcelona Institute of Science and Technology, Dr Aiguader 88, Barcelona 08003, Spain
[3]Universitat de Barcelona, Facultat de Farmàcia i Ciències de l'Alimentació, Avda. Diagonal 643, Barcelona 08028, Spain
[4]Stanford University,
[5]Arc Institute
[6]ICREA, Institució Catalana de Recerca i Estudis Avançats, Barcelona, Spain
*E-mail: noelia.ferruz@crg.eu

Autoregressive protein language models (pLMs) have emerged as powerful tools to efficiently design functional proteins with extraordinary diversity, as evidenced by the successful generation of diverse enzyme families, including lysozymes or carbonic anhydrases. However, a fundamental limitation of pLMs is their propensity to sample from dense regions within the training distribution, which constrains their ability to sample from rare, high-value regions of the sequence space. This limitation becomes particularly critical in applications targeting underrepresented distribution tails, such as engineering for enzymatic activity or binding affinity. To address this challenge, we implement DPO_pLM, a reinforcement learning (RL) framework for protein sequence optimization with pLMs. Drawing inspiration from the success of RL in aligning language models to human preferences, we approach protein optimization as an iterative process that fine-tunes pLM weights to maximize a reward provided by an external oracle. Our strategy demonstrates that RL can efficiently optimize for a variety of properties, even when sparsely represented in the training data, achieving significant improvements within hours while preserving sequence diversity. We applied DPO_pLM to the design of EGFR binders, successfully identifying nanomolar binders within hours. Our code is publicly available at https://github.com/AI4PDLab/DPO_pLM.

Introduction

Protein engineering aims to create new sequences that optimize specific properties, such as activity, stability, or regioselectivity. This task presents a complex optimization problem due to the vast space of potential sequences and the slow, costly nature of wet lab validation. The gold standard in protein optimization is directed evolution, which operates similarly to a greedy hill-climbing algorithm within the local region of sequence space. While the methodology has proven remarkably successful[1,2], including being recognized with a Nobel Prize (Arnold, 2018), it remains inefficient in sampling vast phenotype landscapes, often limited to local exploration. We currently lack tools to orient long evolutionary trajectories towards specific biological functions.

In the past five years, protein language models (pLMs) have emerged as transformative tools for various applications, from structure prediction[3] to protein design[4]. Specifically, autoregressive pLMs, trained for next-token (amino acid) prediction, can efficiently generate diverse protein sequences by sampling from learned probability distributions[5]. These models have achieved notable success in enzyme design, including the design of lysozymes[6], superoxide dismutases[7], carbonic anhydrases[8], lactate dehydrogenases[8], or triosephosphate isomerases[9] with activity comparable to their natural counterparts[9], even with sequence identities below 40% to any natural protein. pLMs learn intrinsic patterns in the underlying protein training sets, enabling the rapid generation of diverse and potentially functional sequences. While pLMs allow exploration of a high-quality subspace of the possible genotypes, it is a challenge to converge to more specific intended phenotypes. Achieving such control would require generating sequences that efficiently sample from rare events, effectively balancing exploration-exploitation tradeoffs.

Reinforcement learning (RL) offers a paradigm for directing deep neural models to optimize specific properties defined by an oracle and could efficiently guide pLMs toward optimizing a certain property. In RL, a model -termed agent in this context- learns to maximize positive feedback, or reward, from its environment, a concept grounded in psychological[10] and neuroscientific studies of animal behavior, where negative stimuli (e.g., pain or hunger) and positive reinforcements (e.g., pleasure or food intake) shape actions. The influence of RL has met a wide range of success, with applications ranging from robotics[11] to gaming[12–14] passing by autonomous driving[15].

With large language models (LLMs) becoming commonplace in our daily lives, recent RL research has now shifted toward aligning model outputs with human preferences (RLHF)[16]. Implementations like Proximal Policy Optimization (PPO)[17] and Direct Preference Optimization (DPO)[18] are standard methods for refining online models like ChatGPT[19] and Gemini[20]. Since autoregressive pLMs share the architecture and training objectives of LLMs, we propose that they can leverage the advances in RLHF to improve the process of protein engineering. In particular, rather than aligning pLMs to human feedback, we can tune these models to maximize feedback from an external oracle, such as fold topologies, predicted stability, or experimental enzymatic activity.

Preliminary efforts applying reinforcement learning to protein engineering are showcasing the potential of the methodology. Angermueller et al. introduced DyNA PPO, a model-based extension of PPO to the design of DNA and protein sequence[21]. Wang et al introduced EvoPlay[22], a framework based on the single-player version of AlphaZero, that produces iterative mutations. Recently, Widatalla et al applied DPO to align the structure-conditioned model ESM-IF[23], achieving competitive thermostability prediction[24]. Furthermore, Mistani et al. applied ranked preference optimization to ProtGPT2[25] for the design of binders[26]. While these works highlight the potential of RL in protein design, there is to date no publicly available, broadly applicable, experimentally-validated implementation of RL to generative pLMs.

In this study, we introduce DPO_pLM, an adaptation of DPO to the context of generative pLMs and evaluate its effectiveness in optimizing several properties. Specifically, we aim to efficiently sample rare events, i.e., achieve the controllable generation of sequences that would otherwise occur infrequently -if ever- in unsupervised settings, significantly delaying experimental success. We successfully control the generation of specific topologies and enzymatic functions, optimize numerical, unbound metrics like ESM-1v[27], and ProteinMPNN[28]

log-likelihoods, and reward models trained from activity data. Additionally, we demonstrate its suitability in experimental settings, with the design of epidermal growth factor receptor (EGFR) protein binders that show nanomolar affinity. We demonstrate that DPO_pLM can effectively optimize user-defined properties in just a few iterations while maintaining sequence diversity. Compared to finetuning of pLMs, the current gold standard in protein engineering with pLMs[6,9], DPO_pLM shows superior performance, reduced computational resource requirements, and less susceptibility to catastrophic forgetting[29]. Our code is publicly available and applicable to any generative pLM at https://github.com/AI4PDLab/DPO_pLM.

Results

pLMs do not transcend the limits of their underlying training set

LLMs generate text that resembles human-created pieces, across a broad range of topics, that often surpass average human performance[30,31]. Similarly, we here observe that pLMs generate sequences with properties close to those found in the training dataset, still in regions significantly distant to natural data points (**Fig. 1a**). ZymCTRL is a conditional model trained on the known enzyme space, where every sequence has been passed with its corresponding Enzyme Commission number (EC number or label), learning a joint sequence-function distribution. Leveraging ZymCTRL we recently generated seven carbonic anhydrases (CA) that showed almost wild-type activity levels, yet having sequence identities as low as 39%[8]. In a subsequent experiment, 200 generated α-amylases were tested, showing in many cases higher activities than the wild-type cutoff[32]. When taking a closer look, we nevertheless noticed that their activities followed the same distribution as the training set (**Fig 1b**). We wondered whether this behavior is empirically reproduced in other properties. We thus generated 40.000 CA sequences (EC label 4.2.1.1) and compared their properties against the original training set, containing originally the same number of sequences. We observe that the produced sequences always recapitulate the training set distribution, replicating the training set normal modes. In particular, sequence lengths (**Fig 1c**), ESMFold-predicted pLDDT (**Fig. 1d**), the proportion of globular domains predicted per sequence (**Fig 1e**), and the proportion of generated domains predicted to fold into the α or beta-conserved CA topologies (**Fig 1f**). We observe this trend in other works in protein design[6], in the context of LLMs[33], and in other neural networks, such as for image generation[34]. Reproducing the training set distributions is nonetheless remarkable: with up to 70% of single mutations in a sequence being deleterious for function[35], the chances of designing a protein with over 100 mutations while preserving wild-type activity levels would be virtually null, yet pLMs achieve this quite efficiently[6,8,9]. Nevertheless, unsupervised pLMs, in their current form, cannot be guided towards specific directions i.e., to controllably tune a protein's activity, thermostability, or any other property at a user's will.

There has been intense research to tackle complex optimization problems. In this particular field, traditional computational protein design algorithms were until recently approached as a landscape defined by a physicochemical energy function, in which heuristic algorithms such as Monte Carlo Metropolis would step-wise search to find a satisfying local minimum[36]. Other methods are inspired by biological phenomena, where, for example, genetic algorithms proceed by combining the best performers in a given generation, to produce offspring that eventually resemble and surpass the properties of the parents[37], and reinforcement learning. Given the recent success of DPO aligning LLMs to human preferences, we ought to provide a framework and evaluate the performance of DPO for pLM-guided protein design.

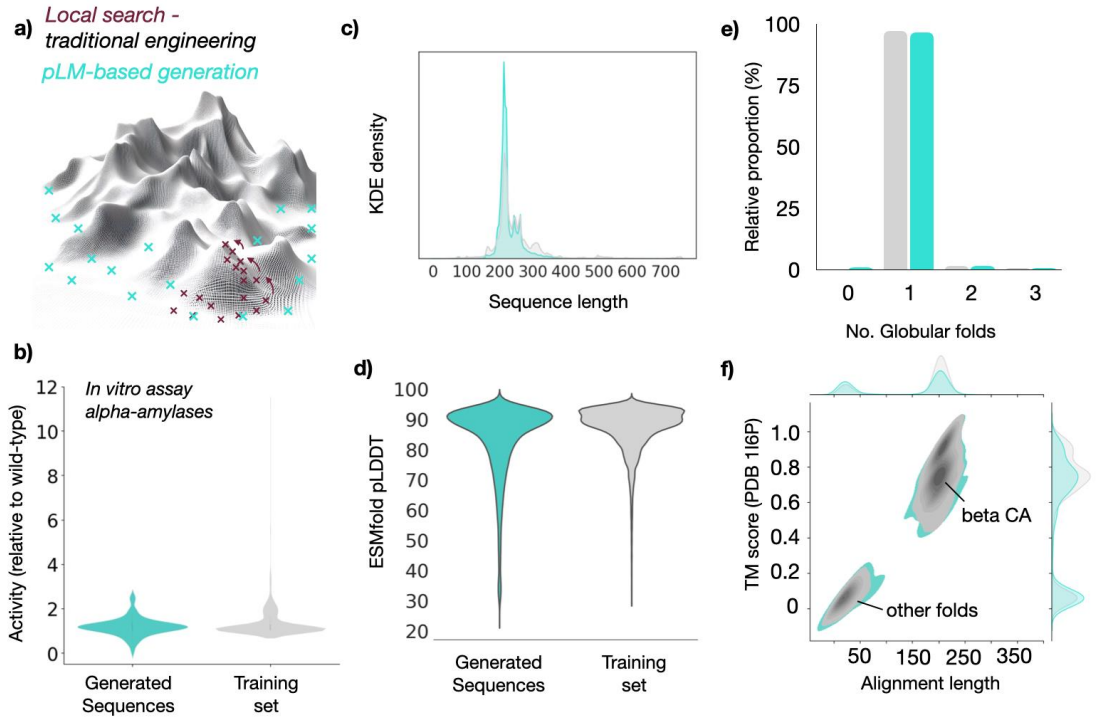


Figure 1: **Current pLMs replicate the distributions of their underlying training sets. (a)** Generation sampling paradigms, LLMs tend to explore vast regions, yet in the most probable areas and replicate training distributions for **(a)** experimental α-amylase activities, **(b)** sequence length, **(c)** predicted pLDDT, **(e)** predicted number of globular folds and **(f)** generated topologies**.** In panels b)-f), gray represents natural sequences and cyan generated ones.

Reinforcement Learning guides pLMs towards specific properties

RL has played a central role in optimizing online LLMs, such as ChatGPT[19] and Gemini[20]. In this scenario, RL frameworks use human feedback to align the model generation process to human preferences, decreasing the probability of producing harmful or unhelpful answers. In RLFH, users provide feedback to various model outputs, which serve as input to train a reward model (RM). The RM is used to evaluate future generations and iteratively fine-tune the LLM to align the generated text to human preferences[17]. This method nevertheless requires extensive human annotation and is strongly influenced by the quality of the RM. Rafailov et al. (2023)[18] introduced DPO, significantly mitigating these limitations. DPO finetunes the LLM by aligning the distance between ranked points and LLMs' log-likelihoods, eliminating the need for laborious RM training, and yielding to more stable fitting[38]. As opposed to assigning numerical feedback values and training RMs, the LLM in this case learns from pairs of ranked feedback points (e.g. response *i* is preferred to response *j*), a loss objective that mathematically is defined as:

$$L_{DPO}(\pi_\theta;\pi_{ref}) = -E_{(x,y_w,y_l)\sim D}\left[log\,\sigma\left(\beta log\,\frac{\pi_\theta(y_i|x)}{\pi_{ref}(y_i|x)} - \beta log\,\frac{\pi_\theta(y_j|x)}{\pi_{ref}(y_j|x)}\right)\right] \quad (1)$$

Where $\pi_\theta(y_i|x)$ and $\pi_{ref}(y_i|x)$ represent the current and original (reference) models for the *i* sequence-reward point, and beta is a tunable hyperparameter. We have adapted DPO to the general case of pLM protein design, where multiple, diverse sequences are generated in batch, an oracle assigns them a certain fitness value, and value-sequence pairs are reverted

to the model as feedback, improving the pLM in an iterative fashion (**Fig. 2a**). We observe that computing the negative log likelihood per sequence yielded better performance (**Fig. S1**), and use this adaptation in DPO_pLM. Beyond the original DPO application using ranked feedback pairs, we apply an alternative considering all ranking points and another that explicitly introduces the reward feedback as recently introduced by Widatalla et al.[24], allowing models to incorporate scalar weights derived from in-silico predictions or wet-lab experimental data. We evaluate and implement our framework to the enzyme model ZymCTRL[8], but note that our implementation is amenable to any autoregressive pLM, and the code is readily amenable for any generative pLM available in HuggingFace[39].

DPO drives generation toward specific folds

Carbonic anhydrases (CA) (EC 4.2.1.1) are remarkably diverse, with eight distinct subtypes (α, β, γ, δ, ζ, η, θ, and ι), exhibiting no similarities in sequence or structure, yet arriving at the same catalytic reaction due to complex convergent evolutionary events[40]. Pre-trained ZymCTRL predominantly generates sequences of the α and β CA classes, with a majority of β-CA, reproducing the proportions in the original training set (**Fig. 1f**). To bias the model towards generating a specific fold, we used TM-score as the oracle[41], employing to this end Foldseek[41] on the ESMFold[3] predicted structures. TM-score is a metric with possible values ranging from 0 to 1, used to assess the topological similarity between two protein structures. Values of 1 indicate an identical topology, and values over 0.5 would indicate structures with approximately the same fold as found in databases like CATH[42]. We sought to control the proportion of generated sequences toward generating a majority of α CA. To this end, we produced 200 sequences per iteration, with label 4.2.1.1 as prompt, and computed TM-scores against the representative α CA (PDB 2VVB) for each sequence. This score was introduced as input in our reward function (**Fig. 2a**, **Methods**). Interestingly, initial experiments using only the TM-score, led to increasing sequence length over the course of the iterations, possibly as a form of reward hacking[43] (**Fig. S2**). Complementing the reward function with the alignment length (**Methods, Table S2**), enhanced performance (**Fig. 2b**). After six iterations, the majority of the sequences reached a TM score of 0.8, with over 95% of the sequences adopting the intended fold by iteration 10 (**Fig. 2b,** right). The model demonstrated stability for over 20 iterations, with no severe decline in sequence quality or diversity (**Fig. 2b,c**), as evidenced by a stable number of MMseqs clusters at 80% and sequence identities to the training set averaging 50%. These first analyses suggested that DPO_pLM can optimize toward the intended property, allowing for several reward terms, while preserving sequence diversity.

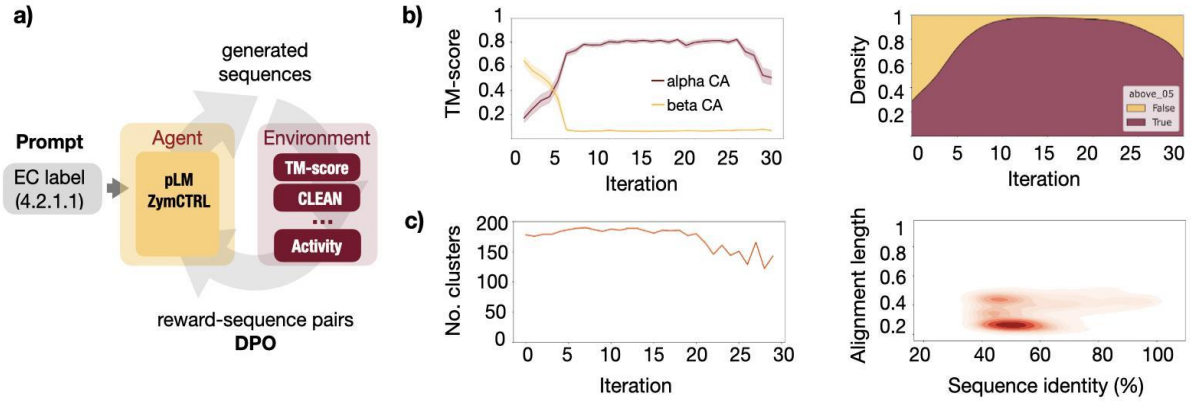


Figure 2: **DPO guiding the generation of intended carbonic anhydrase topologies. (a)** Schematic representation of DPO_pLM, where an agent, the pLM ZymCTRL in this study, generates sequences that are evaluated by an environment, which can contain the combination of several oracles. The scores take part in a reward function, either alone or in more complex combination with other terms, which assigns a reward to each sequence and revert this information to the model. The model improves over time with feedback. **(b)** Evolution of TM-score for α (red) and β (yellow) α and beta CA over the iterations, as an average (left) or counting instances with TM-score values over 0.5 to the representative α CA. **(c)** Distribution of sequences over the iterations, with the number of clusters at 80% (left) and sequence identity to the training set, darker hues corresponding to later iterations (right).

Complex multi-objective reward functions enable the generation of underrepresented enzyme classes

Public enzyme databases contain a highly unbalanced representation of enzyme classes, due to challenges sequencing remote organisms and the infrequency of certain enzyme families. For example, while we find EC labels with more than one million members in the Uniprot database[44], 9% of EC labels in BRENDA have only one known enzyme representative[8]. This unequal sampling is the current object of intense research, both in elegant engineering[45], transfer knowledge paradigms[8], prompting techniques[46] and automated data generation efforts[32,47]. Despite advances, unbalanced training still poses significant challenges when generating enzymes for underrepresented labels, especially in the most data-hungry regimes. For instance, when prompted with the EC label for pancreatic ribonuclease (EC 4.6.1.18), ZymCTRL generates sequences where CLEAN[48] would only assign 10% of the generated sequences as ribonucleases (**Fig. 3a**).

To address this limitation, we used DPO_pLM to guide the ratio of 4.6.1.18 CLEAN-labelled sequences per generation. CLEAN is a state-of-the-art model that leverages contrastive learning to predict EC labels given a protein sequence. This tool encodes the sequence in CLEAN's latent space and infers an EC label, based on the distance of the sequence compared with the labeled clusters emerged during training. In our case, the cosine distance between candidate sequences and the cluster center of the target EC label was used as a scoring function, serving as a multidimensional direction vector for model alignment. To prevent deviations in sequence length from the wild type, we incorporated a length-based discount factor into the weight (**Methods, Table S1**). The model aligned to the objective function, in particular, 60% of the sequences were labeled as 4.6.1.18 after only two iterations (**Fig. 3a**). However, runs that did not include pLDDT optimization in the scoring function led to suboptimal structures (**Fig. S1**). To prevent this, we enhanced the scoring function's

complexity and applied RL to simultaneously reward the proportion of correctly labeled sequences, a length discount, and their pLDDT values (**Methods, Table S1**). The model successfully aligned with both objectives, achieving a pLDDT score of 80 (**Fig. 3b**) compared to the previous value of 40 in earlier experiments (**Fig. S1**). As an indirect consequence, the TM-score against a reference pancreatic ribonuclease (PDB 11BA), also improved over the iterations (**Fig. 3c**).

Upon closer analysis of the model, we noticed an interesting performance boost in other EC labels (**Fig. 3d**). Specifically, EC labels closely related to those targeted during training appeared to benefit indirectly from the reinforcement campaign, displaying an improved sequence quality after 30 iterations (Fig 3d, EC 4.6.1.13), while labels farther away did not seem to decrease performance (EC 4.2.1.1). The same effect was observed in pLDDT (**Fig. 3e**). This has important implications, because by optimizing a single EC label, the model seems to jointly improve other labels, opening a new highly efficient approach for model optimization that does not require additional training data.

DPO optimizes unbounded functions

Previous experiments build on properties with upper bound limits, i.e., the proportion of α CA, or CLEAN-aligned sequences are limited to a value of 100%. We ought to understand how DPO_pLM would behave in protein engineering campaigns, where the variables to optimize are continuous and can extend well-beyond training set instances. To this end, we implemented reward functions maximize ESM1v[27] and ProteinMPNN[28] log likelihoods (**Methods**), which have recently been shown to correlate with experimental outcomes in protein engineering campaigns[7]. ESM1v is a sequence-based model capable of predicting the effects of single-point mutations, and its negative log-likelihood can be used as a proxy of how the amino acid composition of a given protein aligns with the model's learned statistical distribution. Additionally, we also use ProteinMPNN as a structure-based model that takes the predicted folded structure to compute an average negative log-likelihood score for the query amino acid at each residue site. Using carbonic anhydrase (EC 4.2.1.1) and chorismate mutase (EC 5.4.99.5) as prompts for ZymCTRL, we applied iterative steps of reinforcement learning (RL) with the objective of increasing these scores. In both cases, we observed a consistent, maintained improvement, reflected by higher negative log-likelihood values (**Fig. 3f, g**).

Additionally, we applied DPO_pLM using experimental data as a reward, with the goal of implementing an *in silico* directed evolution approach to identify potentially more active candidates. Starting with the dataset of α-amylases sourced from the Protein Engineering Tournament[32], we trained a reward model capable of predicting the Specific Activity Performance Index (SAPI). This value is calculated as the ratio of the specific activities (variant over reference) normalized by concentration. Supervised fine-tuning of ESM1v and ESM2 was performed using a LoRA adapter as previously described[49] (**Methods**). The predictions from both models were then averaged and employed as weights for DPO optimization. Consistent with other methodologies, DPO_pLM successfully enhanced the predicted specific activity of the enzymes (**Fig. 3h)**.

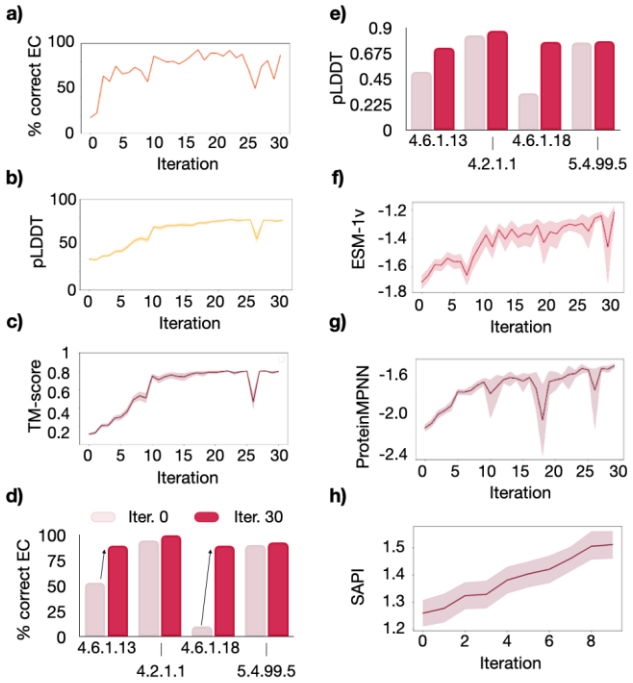


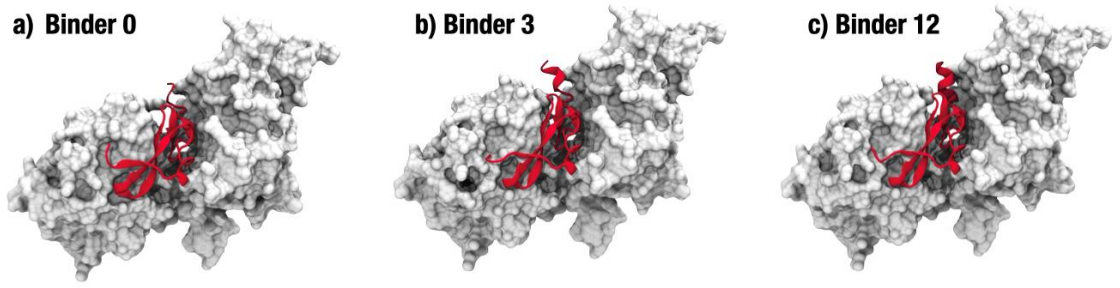
Figure 3: DPO_pLM optimizes CLEAN-labelled sequences and numerical metrics. a) Percentage of 4.6.1.18-labelled proteins by CLEAN over 30 iterations. **b)** Mean pLDDT value of generated sequences over 30 iterations. **c)** TM-score values of generated sequences over 30 iterations. **(d)** Percentage of sequences whose CLEAN annotation corresponds to EC label prompted to ZymCTRL, at iteration 0 (pre-trained model) and 30 (after CLEAN optimization). **e)** ESMfold pLDDT values for the same models and EC labels as in **d)**. ESM-1v progression of values over 30 iterations. **f)** ESM-1v and **g)** ProteinMPNN values of generated sequences across 30 iterations. **h)** Predicted Specific Activity Performance Index (SAPI) of amylase over 30 iterations.

DPO_pLM applied to the designs of nanomolar EGFR binders

As part of the Adaptyvbio round 2 challenge, we delved into the potential of using DPO_pLM to guide ZymCTRL toward the design of binders for the Epidermal Growth Factor Receptor (EGFR), known to play a role with pathogenesis and progression of different carcinoma types[50,51]. While ZymCTRL is an enzyme-specific pLM, we hypothesized that DPO_pLM could guide its generation toward the design of successful (non-catalytic) binders as well if guided with the right oracles. Given the challenging nature of EGFR as a target and the constrained time frame, we fine-tuned ZymCTRL by curating a dataset of Epidermal Growth Factor (EGF) sequences homologous, pre-pending them with the artificial EC label 1.3.3.18 (**Methods**). After this process, ZymCTRL generated sequences resembling EGF. Building on this fine-tuned model, we applied DPO_pLM, using a reward function incorporating the metrics used for scoring submissions (PEA, ESM2 log likelihood, and pLDDT). Due to the slower running times with AlphaFold2, we limited each iteration to 20 generated sequences. These metrics were monitored over 12 iterations (**Figure S6**). At the conclusion of the campaign, 2000 sequences were generated from models at iterations 0, 3, and 12. These sequences were ranked based on perplexity, and the top-scoring ones were selected for submission, where six ranked within the top 100 among a total of over 1000 submissions[52] and were selected for experimental validation.

Experimental assays revealed nanomolar binding affinities for three of the designs (**Fig 5**), and all six of them expressed solubly. These results hold great potential for further exploration,

considering that the model does not have any epitope information or explicit knowledge of the metrics used for optimization, and the short optimization protocol we used in this case. Binders 0 and 3, outperformed the wild-type EGF with average K_D values of 328nM and 346nM, having sequence identities of 71% and 54% to EGF, respectively (**Fig. 5a, 5b**). For reference, EGF yielded an average K_D of 759nM under the same experimental conditions. Additionally, binder 12, with a sequence identity of 62% to EGF, demonstrated a performance comparable to the wild type (average K_D of 820nM, **Fig. 4c**). To further enhance performance, we conducted a second, post-deadline DPO_pLM campaign changing the objective function (**Fig. S6**). This modification led to better overall performance and improved metrics, which could lead to better performance in future binder design campaigns.



	Rep. 1	Rep. 2
K_D (M)	2.61E-07	3.95E-07
k_{off} (1/s)	8.89E-03	8.73E-03
k_{on} (1/Ms)	2.25E+04	3.35E+04

	Rep. 1	Rep. 2
K_D (M)	4.81E-04	2.11E-07
k_{off} (1/s)	1.71E-02	1.84E-02
k_{on} (1/Ms)	3.55E+04	8.74E+04

	Rep. 1	Rep. 2
K_D (M)	9.10E-07	7.3E-07
k_{off} (1/s)	1.99E-02	--
k_{on} (1/Ms)	2.19E+04	--

Figure 4: Design of EGFR nanomolar binders with DPO_pLM. (a) Binder 0, **(b)** Binder 3, **(c)** Binder 12.

DPO outperforms recursive finetuning on synthetic data

Finetuning refers to the process of retraining LLMs on smaller, tailored, specific training sets with the goal that it specializes in a single task while keeping its previously acquired general knowledge[53]. Here, we wondered whether recursive finetuning with synthetic data (self-fine-tuning, s-FT) could achieve similar performance as DPO_pLM to guide the generation process towards certain properties. In particular, we iteratively re-trained ZymCTRL using the top-scoring sequences generated at each iteration, as evaluated by a user-predefined scoring function. This process comprises three sequential steps: (1) generating 2,000 new sequences from ZymCTRL, (2) ranking these sequences based on the defined scoring function and (3) finetuning the model using the top 10% candidates. This finetuned model is then used to initiate another cycle, repeating the process for a total of 30 iterations, to best compare to the RL approach.

We applied s-FT to maximize the generation of α CA and augment the proportion of 4.6.1.18-labeled sequences as previously done with DPO_pLM. While a strategy combining sequences following a length factor promoted the generation of sequences with the canonical α CA length (260 aa, **Table S1**), the enrichment of the α CA fold with this method was modest, reaching only 20% after 30 iterations (**Fig 5a**). In the last iteration, the maximum TM-score reached was 0.71, considerably lower than the maximum reached at iteration 0 (0.97, **Fig 5b**). When pushed further than 30 iterations, the model underwent collapse and started producing redundant sequences (**Fig S4d**), a phenomenon that has recently been observed for LLMs[54].

Following the same methodology, we attempted to enrich the proportion of sequences labeled as 4.6.1.18 with CLEAN. Through this process, s-FT can only reach a maximum of 30% generated sequences with the correct EC label predicted by CLEAN in around 25 iterations, before decreasing performance. Interestingly, the sequences seem to be very diverse, with the number of clusters at both 50% and 90% amounting to the number of generated sequences (**Fig 5c**).

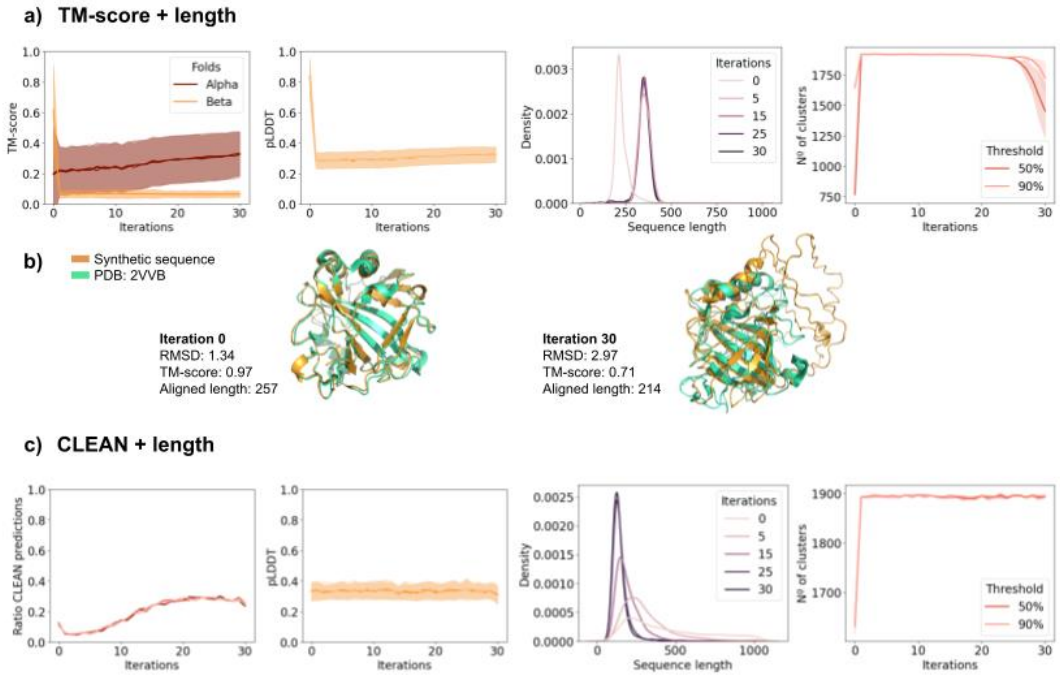


Figure 5. s-FT alignment towards specific properties is modest and leads to disordered proteins. (a) Metrics from the s-FT run with TM-score and sequence length as reward function. From left to right: TM-scores against PDB 2VVB, pLDDT values progression, distribution of sequence lengths and nº of clusters at 50% and 90% sequence identity over iterations. All plots reflect data points from 30 iterations for 3 independent runs of s-FT with the same reward. **(b)** Superimposed structures of the PDB 2VVB and synthetic sequences produced at iteration 0 and 30 for s-FT TM-score. **(c)** Metrics from the s-FT run with CLEAN distance and sequence length as reward function. From left to right: ratio of 4.6.1.18-labeled proteins by CLEAN, pLDDT values progression, distribution of sequence lengths and nº of clusters at 50% and 90% sequence identity over iterations. All plots reflect data points from 30 iterations for 3 independent runs of s-FT with the same reward.

DPO can discriminate at the amino acid level relevant features

Rafailov et al. (2024) have recently shown that DPO implicitly learns a token-level reward function, even if trained to reward entire sequences at once[55], and that by doing so is capable of learning credit assignment, i.e., spot erroneous tokens. Mathematically, this is captured as:

$$r(s,a) = \beta \log \pi_\theta(s|a) - \beta \log \pi_{ref}(s|a) \quad (2)$$

In our case, the reward function for a given state-action pair $r(s,a)$ is the result of the subtraction of the negative log likelihood of the current policy (π) and the negative log likelihood reference policy (π_{ref})(pre-trained ZymCTRL), which reverses the original equation (2) and it is coherent without loss definition of DPO_pLM (6). In practical terms, this means we can compute a reward per token, and assign different ‘credits’, i.e., amino acids that maximize the reward.

Starting from our experiment with pancreatic ribonuclease (EC number 4.6.1.18) and pLDDT reinforcement, we computed the reward by retrieving the per-token negative log-likelihood level during a single forward pass of the output model at each iteration. The results were then visualized as a heatmap (**Fig. 5b**). Interestingly, the iterative process revealed the emergence of putative relevant amino acids. Mutations in individual amino acids induced localized changes in the reward function *r* (**Fig. 5d**). Altering the EC label tokens caused the reward to drop to zero, a condition possible only when the log likelihoods of the sequence are not changed before and after the reinforcement campaign, in other words, where the reference policy and the updated policy are equal (**Fig. S6**).

These experiments highlight that DPO_pLM successfully aligns to the oracle, but may not align with actual biological significance, underscoring the importance of selecting an appropriate scoring function to guide the model toward biologically-relevant feature alignment. This approach has promising applications in areas such as explainability and protein design. Specifically, the token-level reward can be utilized to understand which amino acids are considered important by the environment, but also generate sequences with higher rewards by applying beam search over the tokens, to pick the token between the ones with higher reward in an iterative fashion.

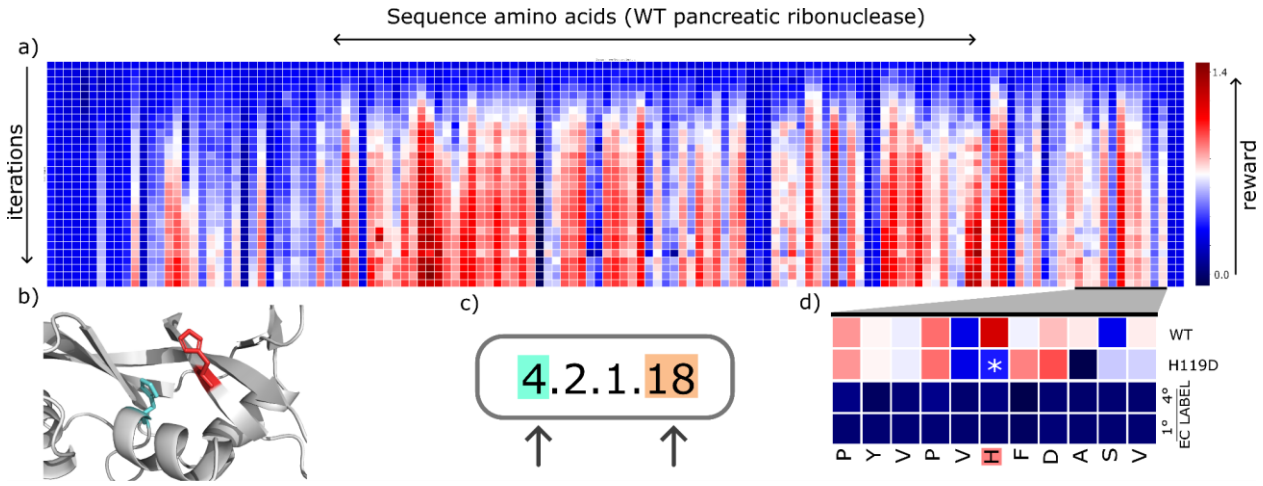


Figure 5. DPO aligns over iterations to the objective function (a) Token-level reward (*r*) computed for a WT pancreatic Ribonuclease sequence (UniProt ID: P00694) across the 30 optimization iterations. **b)** Catalytic residues. In red the histidine targeted by the mutation in position 119. **c)** EC label for pancreatic ribonuclease showing the two numbers that were changed were changed to 4.6.1.13 in the first case and to 1.2.1.18 in the second. **(d)** Detailed view of 10 amino acids at the C-terminal region, including one of the catalytic centers (histidine highlighted in red). DPO effectively identifies relevant tokens with higher rewards (red). Mutations in specific amino acids reduce the reward without significantly affecting the remaining sequence tokens (full sequence provided in **Figure S4**). Mutations at EC-label tokens result in a global reward drop to zero (blue), both at the first (1º) or last (4º) EC number level.

Discussion

pLMs are powerful tools to sample from a given distribution, however, they face limitations when features of interest are rare or not explicitly present in the training data. While RL has become the *de facto* standard for aligning LLMs, in the context of pLMs, human feedback is insufficient for ranking outputs due to a limited understanding of the nuanced "grammar" of biological sequences.

Here, we expand the potential of RL for protein design by implementing, evaluating, and releasing DPO_pLM. To evaluate the method, we explored steering the sampling space of the conditional enzyme model ZymCTRL through interaction with external oracles. DPO_pLM demonstrated robustness across diverse tasks, enriching desired folds and enzyme classes in a multi-objective manner. Additionally, DPO_pLM iteratively maximized ProteinMPNN, ESM-1v and predicted activity scores, highlighting potential for the optimization of virtually unbounded functions, like thermostability or enzymatic kinetics. Additionally, we test DPO_pLM for the design of EGFR binders, producing three nanomolar binders (50% success rate) after training for 12 iterations of 20 feedback points each.

DPO_pLM offers significant advantages over fine-tuning, the current gold standard for pLMs. Unlike fine-tuning, which minimizes negative log-likelihoods, DPO maximizes preference rewards, enabling it to also learn from negative data—often abundant in protein engineering campaigns. Additionally, it requires no extra data, as the model learns directly from its own generations. Notably, fine-tuning one enzyme classification (EC) label improved predictions for others, suggesting a new route to improve existing pLMs without additional data and in minimal training times.

Token-level reward application offers a compelling opportunity to mutate biologically significant amino acids identified as critical by the policy. A similar strategy, involving masking relevant tokens, was recently shown to have good outcomes[45]. As suggested by Rafailov et al. (2024), token-level rewards could also be integrated into beam search to explore sequence space using diverse generative approaches after model alignment to desired objectives. However, it is essential to acknowledge that features identified by the model, as what the oracle defines as "relevant" may not align with biological relevance. Incorporating more sophisticated oracles could improve the fidelity and biological applicability of identified amino acids.

DPO_pLM marks a significant advancement in pLM-driven protein design, addressing the limitations of traditional pLMs in optimizing specific properties. By leveraging RL, it enables efficient optimization for complex properties, even when sparsely represented in training data, while preserving sequence diversity. The successful application of DPO_pLM to design nanomolar EGFR binders highlights its practical utility in accelerating protein engineering tasks. Future efforts will focus on the integration of DPO_pLM into automated experimental laboratories.

Code Availability Statement

The code is available under the MIT license at: https://github.com/AI4PDLab/DPO_pLM

Acknowledgments

We are most grateful to AdaptyvBio, for testing our EGFR binders in their protein characterization platform as part of the Adaptyvbio Protein Design Competition Round 2, and Julian Englert for assistance. We acknowledge Dana Cortade and Dave Estell for help interpreting the experimental α-amylase data, and the Protein Engineering Tournament for kindly testing our sequences. We are grateful to Michael Heinzinger, Maurice Brenner, and Alex Vicente for their insightful discussions. We thank Emyr James and the CRG IT team for support in using the CRG GPU cluster. We acknowledge support of the Spanish Ministry of Science and Innovation through the Centro de Excelencia Severo Ochoa (CEX2020-001049-S, MCIN/AEI /10.13039/501100011033), and the Generalitat de Catalunya through the CERCA programme. N.F. acknowledges support from a Ramón y Cajal contract RYC2021-034367-I funded by MCIN/AEI/10.13039/501100011033 and by the European Union NextGenerationEU/PRTR. M.A.L acknowledges support from an FI Fellowship (AGAUR-Catalan Government) co-funded by the European Social Fund (Award 2024 FI-3 00065). This project has received funding from the European Union's Horizon Europe under the grant agreement No 101120466 (MSCA-DN supporting A.H.). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

Conflict of interest

The authors declare no conflict of interest.

Methods

Direct Preference Optimization

From the original formulation of DPO, the loss (L_{DPO}) to be minimized is formulated as follows:

$$L_{DPO}(\pi_\theta;\pi_{ref}) = -E_{(x,y_w,y_l)\sim D}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(x)}{\pi_{ref}(x)} - \beta \log \frac{\pi_\theta(x)}{\pi_{ref}(x)}\right)\right] \quad (3)$$

where $\pi_\theta$is the policy to be updated, π_{ref} the original frozen model, β rules how much the model will drift from the reference, and y_w and y_l the selected and the rejected options for each pair comparison. In a recent contribution by (Widatalla et al., 2024), two additional loss functions have been introduced: the ranked and weighted form.

Ranked Loss:

$$L_{DPO_{ranked}}(\pi_\theta;\pi_{ref}) = -E_D \sum_{k=1}^{K}\left[\beta \log \frac{\pi_\theta(x)}{\pi_{ref}(x)} - \log \sum_{j=k}^{K} exp\left(\beta \log \frac{\pi_\theta(x)}{\pi_{ref}(x)}\right)\right] \quad (4)$$

Weighted loss:

$$L_{DPO_{weighted}}(\pi_\theta;\pi_{ref}) = -E_D \sum_{k=1}^{K} w^k\left[\beta \log \frac{\pi_\theta(x)}{\pi_{ref}(x)} - \log \sum_{j=k}^{K} exp\left(\beta \log \frac{\pi_\theta(x)}{\pi_{ref}(x)}\right)\right] \quad (5)$$

Which, in other words, is the cross entropy of the ratio $r = \beta log\ \frac{\pi_\theta(x)}{\pi_{ref}(x)}$ and the weight value w. In the our implementation, the ratio r is the difference of the log likelihood of the sequence from the updated model minus the log likelihood of the reference model. The hyperparameter β determines how far the model should drift from the original probability distribution π. SoftMax has been applied to the weights w, on the batch dimension. The pytorch function F.cross_entropy has been applied to the ratio and the weights. We tested both the log likelihood, but also the negative likelihood for each sequence. The latter yielded better results, and if not differently specified, all the results were obtained with the negative log likelihood. A comparison of the two can be found at the supplementary **Figure S1.**

The use of the negative log likelihood results in the inversion of the numerator and the denominator, resulting into this formula:

$$L_{DPO_{pLM}}(\pi_\theta; \pi_{ref}) = -E_D \sum_{k=1}^{K} w^k \left[\beta\ log\ \frac{\pi_{ref}(x)}{\pi_\theta(x)} - log\ \sum_{j=k}^{K} exp\left(\beta\ log\ \frac{\pi_{ref}(x)}{\pi_\theta(x)}\right)\right] \quad (6)$$

The inverted ratio allows the model to have a more conservative update of the model, resulting in a stable version of the original implementation.

Self-fine-tuning (s-FT)

s-FT involves iterative refinement of ZymCTRL by leveraging the top-scoring sequences generated at each iteration, as evaluated by a predefined scoring function $f(seq)$. The pipeline follows three sequential steps: 1) generate 2,000 synthetic sequences from ZymCTRL, 2) rank these sequences based on $f(seq)$ and 3) fine-tune the model using the top 200 candidates. This process iterates for 30 iterations. Triplicates of the training were carried out to increase the statistical significance of the results obtained.

Model training and generation

During s-FT, we retrained ZymCTRL with the Huggingface transformer's Trainer class, loaded from Hugging Face's transformers API (version 4.45.2). The finetuning was performed with a training and evaluation batch size of 4 and 1 with a learning rate of $8x10^{-6}$. Every 10 steps the model was evaluated and saved. The process was carried out for 25 epochs each iteration. The model with the lowest evaluation loss was chosen as the input for the next iteration. In all cases (DPO_pLM and s-FT) we generated using as inference parameters top_k=9, top_p=1, repetition_penalty=1.2, and do_sampling=True.

Structural Similarity: TM Score and Sequence Identity

Generated sequences were folded using ESMFold[3] loaded from Hugging Face's transformers API (version 4.45.2), and the resulting PDB structures were superimposed onto the wild-type structures from the Protein Data Bank using Foldseek[41] (release 9-427df8a). Sequence identity was computed by aligning generated sequences to the training set of each updated ZymCTRL's model using MMseqs2 (Release 15-6f452).

Functional Annotation: CLEAN Class Enzyme Prediction and Cosine Similarity

Protein embeddings were derived using ESM1b[57], yielding matrices of size $L \times dim$ (where L is the protein length and dim is the embedding dimensionality). These embeddings were inferred into CLEAN's optimized embedding space (v1.0.1) to compute the cosine similarity

between the studied sequence and the center of the latent space cluster corresponding to the target EC number.

Statistical Metrics: ESM1v and ProteinMPNN

ProteinMPNN score was retrieved using the "score_only" flag of ProteinMPNN GitHub repository (v1.0.1) and multiplied by -1 to be maximized with DPO as other features. ESM1v score is the average of the log probabilities of amino acids at each sequence position and were computed without masking relying on the previously established methods of sampling[7].

Activity prediction: training of regression models

Two classification models were trained following the notebook in Schmirler et. al 2024[49]. The models utilized were ESM2 and ESM1v, in both cases with LoRA adapters. The two predicted activity by the two models has been averaged and used as reward ZymCTRL. We used the publicly available at the Protein Engineering Tournament GitHub repository[32]. The dataset comprises sequences paired with their respective Specific Activity Performance Index (SAPI), which represents the ratio of the specific activity of a variant to that of the reference. Specific activity is defined as the ratio of activity to the measured protein concentration. To enhance data quality, entries with no activity or expression values below 0.5 were excluded. The models were trained for 57 epochs using an 80/20 split for training and validation. A batch size of 4 and a learning rate of 3×10^{-4} were applied during training. Learning curves and Spearman correlations are illustrated in **Figure S5**.

Fine Tuning of ZymCTRL and Reinforcement Campaign: EGFR binders design

ZymCTRL was fine-tuned using 600 sequences retrieved via BLAST, with the wild-type epidermal growth factor as the query. The label 4.6.1.18 was employed for annotation. Fine-tuning was conducted with a learning rate of 8×10^{-6} over 100 epochs. Detailed information about the training process is available in the DPO_pLM repository. From the fine-tuned model, 20 sequences were generated and subsequently folded using AlphaFold Colab[58,59]. Folding was performed with a single model, three recycles, and in single-sequence mode, utilizing the alphafold2_multimer_v3 mode. The computed metrics from this process were used as weights to optimize the model for designing improved binders.

Training and evaluation:

Training was conducted on a GPU H100 for each task, ranging from a few GPU hours to one day of training to reach the 30 iterations for DPO and several days per experiment for s-FT. In most cases, unless explicitly reported in the main text, the hyperparameters for DPO are stated in **Table S2**.

Code Availability Statement

The code is available under the MIT license at: https://github.com/AI4PDLab/DPO_pLM, including all the scripts for the experiments that are presented in this paper.

Acknowledgments

We are most grateful to AdaptyvBio, for testing our EGFR binders in their protein characterization platform as part of the Adaptyvbio Protein Design Competition Round 2, and Julian Englert for assistance. We acknowledge Dana Cortade and Dave Estell for help interpreting the experimental α-amylase data, and the Protein Engineering Tournament for kindly testing our sequences. We are grateful to Michael Heinzinger, Maurice Brenner, and Alex Vicente for their insightful discussions. We thank Emyr James and the CRG IT team for support in using the CRG GPU cluster. We acknowledge support of the Spanish Ministry of Science and Innovation through the Centro de Excelencia Severo Ochoa (CEX2020-001049-S, MCIN/AEI /10.13039/501100011033), and the Generalitat de Catalunya through the CERCA programme. N.F. acknowledges support from a Ramón y Cajal contract RYC2021-034367-I funded by MCIN/AEI/10.13039/501100011033 and by the European Union NextGenerationEU/PRTR. M.A.L acknowledges support from an FI Fellowship (AGAUR-Catalan Government) co-funded by the European Social Fund (Award 2024 FI-3 00065). This project has received funding from the European Union's Horizon Europe under the grant agreement No 101120466 (MSCA-DN supporting A.H.). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

Conflict of interest

The authors declare no conflict of interest.

References

1. Arnold, F. H. Directed evolution: Bringing new chemistry to life. *Angew. Chem. Int. Ed Engl.* **57,** 4143–4148 (2018).
2. McCafferty, J., Griffiths, A. D., Winter, G. & Chiswell, D. J. Phage antibodies: filamentous phage displaying antibody variable domains. *Nature* **348,** 552–554 (1990).
3. Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., dos Santos Costa, A., Fazel-Zarandi, M., Sercu, T., Candido, S. & Rives, A. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *bioRxiv* 2022.07.20.500902 (2022). doi:10.1101/2022.07.20.500902
4. Ruffolo, J. A. & Madani, A. Designing proteins with language models. *Nat. Biotechnol.* **42,** 200–202 (2024).
5. Ferruz, N. & Höcker, B. Controllable protein design with language models. *Nat. Mach. Intell.* **4,** 521–532 (2022).
6. Madani, A., Krause, B., Greene, E. R., Subramanian, S., Mohr, B. P., Holton, J. M., Olmos, J. L., Jr, Xiong, C., Sun, Z. Z., Socher, R., Fraser, J. S. & Naik, N. Large language models generate functional protein sequences across diverse families. *Nat. Biotechnol.* **41,** 1099–1106 (2023).
7. Johnson, S. R., Fu, X., Viknander, S., Goldin, C., Monaco, S., Zelezniak, A. & Yang, K. K. Computational Scoring and Experimental Evaluation of Enzymes Generated by Neural Networks. *bioRxiv* 2023.03.04.531015 (2023). doi:10.1101/2023.03.04.531015
8. Munsamy, G., Illanes-Vicioso, R., Funcillo, S., Nakou, I. T., Lindner, S., Ayres, G., Sheehan, L. S., Moss, S., Eckhard, U., Lorenz, P. & Ferruz, N. Conditional language models enable the efficient design of proficient enzymes. *bioRxiv* (2024). doi:10.1101/2024.05.03.592223
9. Romero-Romero, S., Braun, A. E., Kossendey, T., Ferruz, N., Schmidt, S. & Höcker, B. De novo design of triosephosphate isomerases using generative language models. *bioRxiv* (2024). doi:10.1101/2024.11.10.622869
10. Dayan, P. & Berridge, K. C. Model-based and model-free Pavlovian reward learning:

revaluation, revision, and revelation. *Cogn. Affect. Behav. Neurosci.* **14,** 473–492 (2014).

11. Dittert, S., Moens, V. & De Fabritiis, G. BricksRL: A platform for democratizing robotics and reinforcement learning research and education with LEGO. *arXiv [cs.RO]* (2024). at <http://arxiv.org/abs/2406.17490>
12. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S. & Hassabis, D. Human-level control through deep reinforcement learning. *Nature* **518,** 529–533 (2015).
13. Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T. & Hassabis, D. Mastering the game of Go with deep neural networks and tree search. *Nature* **529,** 484–489 (2016).
14. Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K. & Hassabis, D. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science* **362,** 1140–1144 (2018).
15. Kiran, B. R., Sobh, I., Talpaert, V., Mannion, P., Sallab, A. A. A., Yogamani, S. & Pérez, P. Deep reinforcement learning for autonomous driving: A survey. *arXiv [cs.LG]* (2020). at <http://arxiv.org/abs/2002.00444>
16. Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J. & Lowe, R. Training language models to follow instructions with human feedback. *arXiv [cs.CL]* (2022). at <http://arxiv.org/abs/2203.02155>
17. Schulman, J., Wolski, F., Dhariwal, P., Radford, A. & Klimov, O. Proximal Policy Optimization Algorithms. *arXiv [cs.LG]* (2017). at <http://arxiv.org/abs/1707.06347>

18. Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C. D. & Finn, C. Direct Preference Optimization: Your language model is secretly a reward model. *arXiv [cs.LG]* (2023). at <http://arxiv.org/abs/2305.18290>
19. ChatGPT. at <https://chat.openai.com/chat>
20. Gemini Team. Gemini: A Family of Highly Capable Multimodal Models. *arXiv [cs.CL]* (2023). at <http://arxiv.org/abs/2312.11805>
21. Angermueller, C. & Dohan, D. Model-based reinforcement learning for bio- logical sequence design. at <https://storage.googleapis.com/gweb-research2023-media/pubtools/5917.pdf>
22. Wang, Y., Tang, H., Huang, L., Pan, L., Yang, L., Yang, H., Mu, F. & Yang, M. Self-play reinforcement learning guides protein engineering. *Nat. Mach. Intell.* **5,** 845–860 (2023).
23. Hsu, C., Verkuil, R., Liu, J., Lin, Z., Hie, B., Sercu, T., Lerer, A. & Rives, A. Learning inverse folding from millions of predicted structures. *Systems Biology* (2022). at <https://www.biorxiv.org/content/10.1101/2022.04.10.487779v1>
24. Widatalla, T., Rafailov, R. & Hie, B. Aligning protein generative models with experimental fitness via Direct Preference Optimization. *bioRxiv* (2024). doi:10.1101/2024.05.20.595026
25. Ferruz, N., Schmidt, S. & Höcker, B. ProtGPT2 is a deep unsupervised language model for protein design. *Nat. Commun.* **13,** 4348 (2022).
26. Mistani, P. & Mysore, V. Preference optimization of protein language models as a multi-objective binder design paradigm. *arXiv [physics.bio-ph]* (2024). at <http://arxiv.org/abs/2403.04187>
27. Meier, J., Rao, R., Verkuil, R., Liu, J., Sercu, T. & Rives, A. Language models enable zero-shot prediction of the effects of mutations on protein function. *Synthetic Biology* (2021). at <https://www.biorxiv.org/content/10.1101/2021.07.09.450648v1.full>
28. Dauparas, J., Anishchenko, I., Bennett, N., Bai, H., Ragotte, R. J., Milles, L. F., Wicky, B. I. M., Courbet, A., de Haas, R. J., Bethel, N., Leung, P. J. Y., Huddy, T. F., Pellock, S., Tischer, D., Chan, F., Koepnick, B., Nguyen, H., Kang, A., Sankaran, B., Bera, A. K.,

King, N. P. & Baker, D. Robust deep learning-based protein sequence design using ProteinMPNN. *Science* **378,** 49–56 (2022).

29. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D. & Hadsell, R. Overcoming catastrophic forgetting in neural networks. *arXiv [cs.LG]* (2016). at <http://arxiv.org/abs/1612.00796>
30. Luo, X., Rechardt, A., Sun, G., Nejad, K. K., Yáñez, F., Yilmaz, B., Lee, K., Cohen, A. O., Borghesani, V., Pashkov, A., Marinazzo, D., Nicholas, J., Salatiello, A., Sucholutsky, I., Minervini, P., Razavi, S., Rocca, R., Yusifov, E., Okalova, T., Gu, N., Ferianc, M., Khona, M., Patil, K. R., Lee, P.-S., Mata, R., Myers, N. E., Bizley, J. K., Musslick, S., Bilgin, I. P., Niso, G., Ales, J. M., Gaebler, M., Ratan Murty, N. A., Loued-Khenissi, L., Behler, A., Hall, C. M., Dafflon, J., Bao, S. D. & Love, B. C. Large language models surpass human experts in predicting neuroscience results. *Nat. Hum. Behav.* 1–11 (2024).
31. Street, W., Siy, J. O., Keeling, G., Baranes, A., Barnett, B., McKibben, M., Kanyere, T., Lentz, A., Arcas, B. A. y. & Dunbar, R. I. M. LLMs achieve adult human performance on higher-order theory of mind tasks. *arXiv [cs.AI]* (2024). at <http://arxiv.org/abs/2405.18870>
32. Armer, C., Kane, H., Cortade, D. L., Redestig, H., Estell, D. A., Yusuf, A., Rollins, N., Spinner, H., Marks, D., Brunette, T. J., Kelly, P. J. & DeBenedictis, E. Results of the Protein Engineering Tournament: An open science benchmark for protein modeling and design. *bioRxiv* (2024). doi:10.1101/2024.08.12.606135
33. Kang, K., Setlur, A., Ghosh, D., Steinhardt, J., Tomlin, C., Levine, S. & Kumar, A. What do learning dynamics reveal about generalization in LLM reasoning? *arXiv [cs.LG]* (2024). at <http://arxiv.org/abs/2411.07681>
34. Wenger, E. AI produces gibberish when trained on too much AI-generated data. *Nature* **631,** 742–743 (2024).
35. Guo, H. H., Choe, J. & Loeb, L. A. Protein tolerance to random amino acid change.

Proc. Natl. Acad. Sci. U. S. A. **101,** 9205–9210 (2004).

36. Gainza, P., Nisonoff, H. M. & Donald, B. R. Algorithms for protein design. *Curr. Opin. Struct. Biol.* **39,** 16–26 (2016).
37. Jones, D. T. De novo protein design using pairwise potentials and a genetic algorithm: De novo protein design. *Protein Sci.* **3,** 567–574 (1994).
38. Grattafiori, A *et al*. The Llama 3 herd of models. *arXiv [cs.AI]* (2024). at <http://arxiv.org/abs/2407.21783>
39. Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q. & Rush, A. M. HuggingFace's transformers: State-of-the-art natural language processing. *arXiv [cs.CL]* (2019). at <http://arxiv.org/abs/1910.03771>
40. Banerjee, S. & Deshpande, P. A. On origin and evolution of carbonic anhydrase isozymes: A phylogenetic analysis from whole-enzyme to active site. *Comput. Biol. Chem.* **61,** 121–129 (2016).
41. van Kempen, M., Kim, S. S., Tumescheit, C., Mirdita, M., Lee, J., Gilchrist, C. L. M., Söding, J. & Steinegger, M. Fast and accurate protein structure search with Foldseek. *Nat. Biotechnol.* **42,** 243–246 (2024).
42. Chandonia, J.-M., Guan, L., Lin, S., Yu, C., Fox, N. K. & Brenner, S. E. SCOPe: improvements to the structural classification of proteins - extended database to facilitate variant interpretation and machine learning. *Nucleic Acids Res.* **50,** D553–D559 (2022).
43. Specification gaming: the flip side of AI ingenuity. *Google DeepMind* at <https://deepmind.google/discover/blog/specification-gaming-the-flip-side-of-ai-ingenuity/>
44. UniProt Consortium. UniProt: the Universal Protein Knowledgebase in 2023. *Nucleic Acids Res.* **51,** D523–D531 (2023).
45. Ismail, A. A., Oikarinen, T., Wang, A., Adebayo, J., Stanton, S., Joren, T., Kleinhenz, J., Goodman, A., Bravo, H. C., Cho, K. & Frey, N. C. Concept Bottleneck Language Models

For protein design. *arXiv [cs.LG]* (2024). at <http://arxiv.org/abs/2411.06090>

46. Hesslow, D., Zanichelli, N., Notin, P., Poli, I. & Marks, D. RITA: a Study on Scaling Up Generative Protein Sequence Models. *arXiv [q-bio.QM]* (2022). at <http://arxiv.org/abs/2205.05789>
47. Adaptyv Bio. at <https://www.adaptyvbio.com/>
48. Yu, T., Cui, H., Li, J. C., Luo, Y., Jiang, G. & Zhao, H. Enzyme function prediction using contrastive learning. *Science* **379,** 1358–1363 (2023).
49. Schmirler, R., Heinzinger, M. & Rost, B. Fine-tuning protein language models boosts predictions across diverse tasks. *Nat. Commun.* **15,** 7407 (2024).
50. Normanno, N., De Luca, A., Bianco, C., Strizzi, L., Mancino, M., Maiello, M. R., Carotenuto, A., De Feo, G., Caponigro, F. & Salomon, D. S. Epidermal growth factor receptor (EGFR) signaling in cancer. *Gene* **366,** 2–16 (2006).
51. Buch, I., Ferruz, N. & De Fabritiis, G. Computational Modeling of an Epidermal Growth Factor Receptor Single-Mutation Resistance to Cetuximab in Colorectal Cancer Treatment. *J. Chem. Inf. Model.* **53,** 3123–3126 (2013).
52. Protein Optimization 103: Racing to the Top 100. at <https://www.adaptyvbio.com/blog/po103>
53. Radford, A. & Narasimhan, K. Improving language understanding by generative pre-training. (2018). at <https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf>
54. Shumailov, I., Shumaylov, Z., Zhao, Y., Papernot, N., Anderson, R. & Gal, Y. AI models collapse when trained on recursively generated data. *Nature* **631,** 755–759 (2024).
55. Rafailov, R., Hejna, J., Park, R. & Finn, C. From r to Q^*: Your Language Model is Secretly a Q-Function. *arXiv [cs.LG]* (2024). at <http://arxiv.org/abs/2404.12358>
56. Zhang, Z., Wayment-Steele, H. K., Brixi, G., Wang, H., Kern, D. & Ovchinnikov, S. Protein language models learn evolutionary statistics of interacting sequence motifs. *Proc. Natl. Acad. Sci. U. S. A.* **121,** e2406285121 (2024).
57. Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., Guo, D., Ott, M., Zitnick, C. L.,

Ma, J. & Fergus, R. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proc. Natl. Acad. Sci. U. S. A.* **118,** e2016239118 (2021).

58. Mirdita M, Schütze K, Moriwaki Y, Heo L, Ovchinnikov S and Steinegger M. ColabFold: Making protein folding accessible to all. Nature Methods (2022) doi: 10.1038/s41592-022-01488-1

59. Evans et al. "Protein complex prediction with AlphaFold-Multimer." biorxiv (2021) doi: 10.1101/2021.10.04.463034v1