

---

# Ambient Proteins: Training Diffusion Models on Low Quality Structures

---

**Giannis Daras** \*  
CSAIL, MIT  
gdaras@mit.edu

**Jeffrey Ouyang-Zhang** \*  
Computer Science, UT Austin  
jozhang@utexas.edu

**Krithika Ravishankar**  
Computer Science, UT Austin  
krithravi@utexas.edu

**Costis Daskalakis**  
CSAIL, MIT  
costis@csail.mit.edu

**Adam Klivans**  
Computer Science, UT Austin  
klivans@cs.utexas.edu

**Daniel J. Diaz**  
Computer Science, UT Austin  
danny.diaz@utexas.edu

## Abstract

We present *Ambient Protein Diffusion*, a framework for training protein diffusion models that generates structures with unprecedented diversity and quality. State-of-the-art generative models are trained on computationally derived structures from AlphaFold2 (AF), as experimentally determined structures are relatively scarce. The resulting models are therefore limited by the quality of synthetic datasets. Since the accuracy of AF predictions degrades with increasing protein length and complexity, *de novo* generation of long, complex proteins remains challenging. Ambient Protein Diffusion overcomes this problem by treating low-confidence AF structures as corrupted data. Rather than simply filtering out low-quality AF structures, our method adjusts the diffusion objective for each structure based on its corruption level, allowing the model to learn from both high and low quality structures. Empirically, Ambient Protein Diffusion yields major improvements: on proteins with 700 residues, diversity increases from 45% to 86% from the previous state-of-the-art, and designability improves from 68% to 86%. All of our code, models and datasets are available under the following repository: <https://github.com/jozhang97/ambient-proteins>.

## 1 Introduction

Proteins are the fundamental building blocks of life. They accelerate chemical reactions by many orders of magnitude, convert sunlight into food, and underpin the myriads of processes within cells and organisms with the level of accuracy and precision required to sustain life [6, 32]. Unlike computational protein engineering—which focuses on improving the developability or function of existing proteins through computationally guided mutations for practical biotechnological applications [21, 40, 24, 41, 8, 36, 20, 52]—*de novo* protein design aims to create entirely new proteins with specified structures and functions, ultimately seeking to discover folds and activities not found in nature [12]. Since protein function is largely determined by tertiary and quaternary structure, generative machine learning frameworks for protein design focus on learning the sparse, evolutionarily sampled landscape of protein structures, with the goal of generating novel, functional backbone scaffolds beyond those observed in nature [50, 33, 34, 28, 23, 53, 22, 7, 55, 47].

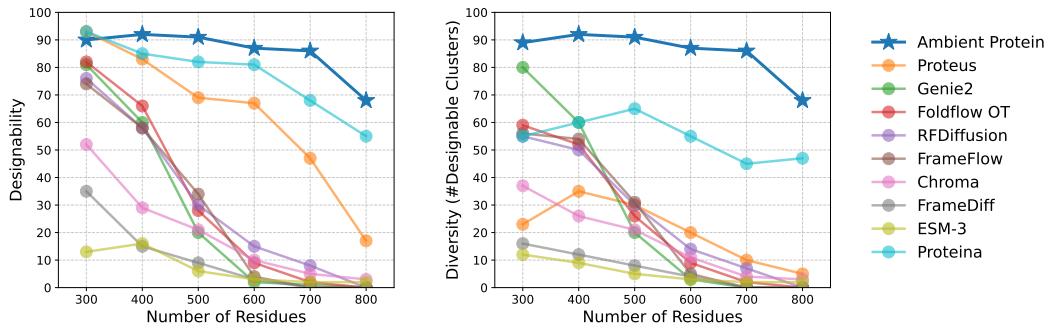
Recent breakthroughs in machine learning-based structure prediction—most notably AlphaFold2 [29]—have made it possible to infer accurate protein structures directly from se-

---

\*Equal contribution.

quence [29, 11, 35]. This progress has enabled the creation of large-scale structural resources such as the AlphaFold Database (AFDB), which contains over 214M predicted structures from UniProtKB sequences [13, 46]. In parallel, high-throughput tools for sequence and structure comparison, such as MMSeqs2 and FoldSeek, have facilitated the curation of large, diverse training datasets from AFDB [5]. Among them, the 2.3M AFDB cluster dataset, has already been shown to improve the capabilities of generative models for protein structure design [34, 23].

The quality of a generative model depends on the size and fidelity of its training data. While AlphaFold2 (AF) has enabled large-scale protein structure prediction, its outputs often contain biological or computational inaccuracies [51]. To estimate the reliability of a predicted structure, AlphaFold provides a per-residue confidence score, the predicted Local Distance Difference Test (pLDDT), which is a proxy of local structural accuracy. In practice, researchers frequently filter predicted structures based on average pLDDT scores, training only on high-confidence subsets (typically using a cutoff of  $p\text{LDDT} > 80$ ). However, lower pLDDT scores are disproportionately associated with longer and more structurally complex proteins. As a result, filtering based on pLDDT introduces a bias toward smaller, simpler folds, reducing structural diversity in the training set and impairing the model’s ability to generalize to more complex regions of structure space—including longer proteins. Notably, many low-pLDDT structures still contain well-folded domains that are misoriented with respect to each other, as reflected by low predicted alignment error (pAE). These structures can still offer valuable domain-level and coarse-grained information about the structure distribution, which is discarded by overly aggressive filtering.



**Figure 1: Long protein generation performance.** We train Ambient Protein Diffusion on proteins up to 768 residues and sample sequences ranging from 300 to 800 residues. Our 17M parameters model significantly outperforms the previous state-of-the-art Proteina [23], which is a 200M parameters model. Ambient Protein Diffusion generates both diverse and designable structures across all lengths.

To mitigate these issues, we depart from the standard paradigm of aggressive filtering of low-confidence structures. Instead, we introduce *Ambient Protein Diffusion* —a framework for training diffusion models that incorporates proteins with noisy or incomplete structures directly into the training process. Ambient Protein Diffusion builds on recent advances in learning generative models from corrupted data [14, 16, 1, 15, 2, 42, 4, 49, 17, 37, 43, 27], which have explored controlled corruption settings such as additive Gaussian noise [14, 16, 1, 17] and masking [15, 2]. Our framework generalizes these techniques to arbitrary, unknown corruption processes, enabling the training of generative models in scientific domains where the corruption mechanism is complex and non-parametric. In our setting, the AlphaFold prediction errors represent such a corruption: they are structured, not explicitly modeled, and vary across protein size and topology. Yet, our method effectively leverages these imperfect samples to significantly advance the capabilities of generative protein models. For example, on proteins with 700 residues, our 16.7M parameter model improves diversity from 45% to 86% and increases designability from 68% to 86% compared to the previous state-of-the-art, Proteina [23], a 200M parameter model. Below, we summarize our key contributions:

- We generalize recent approaches for training generative models on corrupted data to handle arbitrary, non-parametric, and unknown corruption processes, enabling their application to scientific domains.

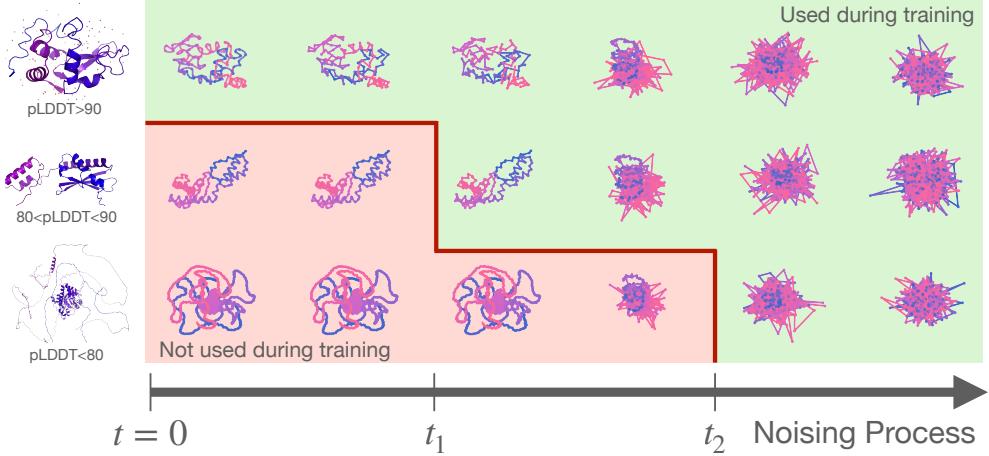


Figure 2: **Overview of Ambient Protein Diffusion on AlphaFold structures.** Rows 1-3 show the noising process (from left to the right) of three different AlphaFold proteins based on their average pLDCT (top: high, middle: medium, and bottom: low). These proteins are only used during training at the green diffusion times. At these noise levels, any initial AlphaFold prediction errors in low-pLDCT proteins have effectively been “erased” by the added noise, and the distributions of low- and high-pLDCT proteins have merged.

- We demonstrate that our framework, Ambient Protein Diffusion, effectively leverages low-pLDCT AlphaFold predictions, allowing the model to learn from all available samples without distorting the underlying structure distribution.
- We further construct a new training set from the AFDB cluster dataset optimized for geometric diversity irrespective of their evolutionary relationship, yielding a broader and more representative sampling of structural space for generative modeling.
- We achieve state-of-the-art results in both diversity and designability for protein generation, improve diversity by 45% and designability by 24% on long proteins (800 residues), and establish the Pareto frontier between these objectives on short proteins (< 256 residues). Our models also achieve state-of-the-art novelty scores in both short and long protein generation, indicating much lower memorization of the training set.

## 2 Background and Related Work

**De novo Protein Generation.** Most *de novo* protein generation frameworks that operate in structure space follow a three-step pipeline: (1) a generative model samples a three-dimensional backbone structure; (2) an inverse folding model (e.g., ProteinMPNN [19]) proposes amino acid sequences likely to fold into the generated backbone; and (3) these sequences are evaluated by a structure prediction model (e.g., ESMFold [35]) to identify the ones that best recapitulate the target fold.

Pioneering methods such as RFDiffusion [50] and Chroma [28] have established strong baselines for backbone generation. More recent advances include Genie [33], which introduces a denoising diffusion model with an SE(3)-equivariant network that generates proteins as point clouds of reference frames; Genie2 [34], which scales Genie using synthetic AlphaFold structures to improve training data diversity; and Proteína [23], which replaces diffusion with flow matching and scales both model size and dataset scale by orders of magnitude to improve performance on longer and more complex monomeric proteins.

Ambient Protein Diffusion is built using the Genie architecture and makes use of ambient protein diffusion to achieve state-of-the-art results with substantially shorter training times, much fewer parameters (16.7M vs 200M), and significantly reduced computational requirements.

**Training Datasets.** Recent advances in structure prediction—most notably AlphaFold2 [29] and ESMFold [35]—have dramatically expanded the available structural data, enabling the prediction

of  $\sim 214M$  and  $\sim 617M$  monomeric protein structures from UniProtKB (via the AlphaFold Protein Structure Database) [46] and metagenomic libraries (via the ESM Atlas) [35], respectively. While this explosion of computational structures presents unprecedented opportunities, it also poses significant challenges for downstream bioinformatic analysis and model training, particularly due to the scale, redundancy, and uneven quality of the predicted structures. To address this, prior work applied MMSeqs2 [26] and FoldSeek [45] to cluster the AlphaFold Database (AFDB), yielding  $\sim 2.3M$  clusters shown to capture evolutionary relationships between predicted structures [5]. This AFDB cluster dataset has since served as the foundational dataset to train several generative protein structure models [23, 34].

In this work, we apply a reparameterized FoldSeek to the AFDB cluster dataset to maximize geometric diversity rather than evolutionary insights. Our goal is to construct a dataset better suited for learning a generative model of protein structure space—one that emphasizes structural rather than evolutionary variation. Starting from the 2.3 million AFDB clusters, we use the cluster representatives with average pLDDT  $> 70$  ( $\sim 1.29M$  structures) and apply our geometric clustering procedure. The resulting dataset comprises roughly  $\sim 292K$  structurally diverse clusters.

**Diffusion Models.** The goal in diffusion modeling is to sample from an unknown density  $p_0$  that we have samples available. Formally, let  $\mathcal{D} = \{x_0^i\}_{i=1}^N$  a dataset of  $N$  independent samples, where  $X_0^i \sim p_0$ . The unknown distribution  $p_0$  is potentially complex, high-dimensional and multimodal. To make the sampling problem more tractable, in diffusion modeling we target smoothed densities  $p_t$  defined as the convolution with a Gaussian:  $p_t = p_0 * \mathcal{N}(0, \sigma^2(t)I_d)$ <sup>2</sup>, where  $\sigma(t)$  is an increasing function of  $t$ , with  $\sigma(0) = 0$ . In particular, the object of interest in diffusion modeling is the score-function of the smoothed densities, defined as  $\nabla_{x_t} \log p_t(x_t)$ . The latter is connected to the optimal denoiser (in the  $l_2$  sense) through Tweedie's Formula:  $\nabla \log p_t(x_t) = \frac{\mathbb{E}[X_0|X_t=x_t] - x_t}{\sigma^2(t)}$ .

Given access to  $\mathbb{E}[X_0|X_t = x_t]$  one can sample from the distribution  $p_0$  of interest by running a discretized version of a reverse diffusion process [3, 44, 9, 10]. Hence, the sampling problem becomes equivalent to the problem of approximating the set of functions  $\{\mathbb{E}[X_0|X_t = \cdot]\}_{t=0}^T$ . Given a sufficiently rich family of functions  $\{h_\theta : \theta \in \Theta\}$ , the conditional expectation at a particular time  $t$  can be learned by minimizing the objective:

$$J(\theta) = \mathbb{E}_{t \in \mathcal{U}[0, T]} \mathbb{E}_{x_0, x_t|t} \left[ \|h_\theta(x_t) - x_0\|^2 \right]. \quad (1)$$

In the context of protein diffusion models for backbones,  $X_0$  captures the 3-D co-ordinates for each residues of the protein.

**Learning from noisy data.** Recent work has explored the problem of learning diffusion models from corrupted data. Typically, the corruption process is simple, e.g. it can be additive Gaussian noise as in [14, 16, 1], or masking as in [1, 2]. Even in works where the corruption process is more general, the degradation needs to be known and multiple diffusion trainings are required until an Expectation-Maximization algorithm converges [42, 4]. In this work, we deviate from this setting as the corruption process is unknown and complex, which may include AlphaFold learning and hallucination errors, and noise inherent to the structural biology technique used to solve the structure, etc. We also target a single diffusion training instead of performing multiple EM iterations. The method is detailed in Section 3.

Our work generalizes the techniques developed in [14, 16] for the additive Gaussian noise case. Particularly, in [16], the authors consider learning from a dataset  $\mathcal{D} = \{(x_{t_i}^i, t_i)\}_{i=1}^N$  of samples noised with additive Gaussian noise of different variances  $\{\sigma^2(t_i)\}_{i=1}^N$ . Formally, let  $X_{t_i} = X_0 + \sigma(t_i)Z$ , where  $X_0 \sim p_0, Z \sim \mathcal{N}(0, I_d)$ . Each point  $X_{t_i}$  contributes to the learning only for  $t \geq t_i$ , using the objective:

$$\hat{J}(\theta) = \mathbb{E}_{t \in \mathcal{U}[0, T]} \sum_{\substack{x_{t_i} \in \mathcal{D}: t > t_i}} \mathbb{E}_{x_t|x_{t_i}, t_i} \left[ \|\alpha(t, t_i)h_\theta(x_t, t) + (1 - \alpha(t, t_i))x_t - x_{t_i}\|^2 \right], \quad (2)$$

$\alpha(t, t_i) = \frac{\sigma^2(t) - \sigma^2(t_i)}{\sigma^2(t)}$ . As the number of samples grows to infinity, Equation 2 also recovers the conditional expectation  $\mathbb{E}[X_0|X_t = x_t]$ , but it does so while being able to utilize noisy samples.

<sup>2</sup>Alternative formulations of diffusion modeling, such as the Variance Preserving case, are equivalent to this case up to a simple reparametrization. For the ease of analysis, we focus our presentation on corruptions of the form  $X_t = X_0 + \sigma_t Z, \quad Z \sim \mathcal{N}(0, I_d)$ .

This objective recovers the true minimizer because one can prove that the conditional expectation  $\mathbb{E}[X_{t_i}|X_t = x_t]$ , lies in the line that connects the current noisy point  $x_t$  and the prediction of the clean image,  $\mathbb{E}[X_0|X_t = x_t]$ .

**Distribution merging under noise.** A key idea in our method will be that distribution distances contract as we increase the noise level added. In the context of diffusion models, this property has been leveraged in the SDEdit [39] paper to allow diffusion models to perform stroke-based editing at inference time without any finetuning. Most relevant to our work, Ambient Omni [18] uses the distribution contraction property (together with other innovations) to use blurry and out-of-distribution data during training. In this work, we focus on *synthetic data* from AlphaFold that arise from various corruption sources: biological ones (errors in the crystallography process), computational ones (errors in the solution of the inverse problem from the crystallography data to the modeled structure), and learning ones (AlphaFold mistakes due to the limited size of the training dataset and hallucinations).

### 3 Method

#### 3.1 Building Intuition

We are given access to samples from the AlphaFold distribution  $\tilde{p}_0$  and aim to learn how to sample from the true distribution of experimentally solved structures,  $p_0$ , without an explicit degradation model mapping  $p_0 \rightarrow \tilde{p}_0$ . Our key insight is that, regardless of how  $\tilde{p}_0$  deviates from  $p_0$ , adding noise to both distributions causes them to contract toward one another. As the noise level increases, the distributions  $\tilde{p}_t$  and  $p_t$  become progressively more aligned. This is because it is known that Gaussian noise contracts distribution distances (KL divergence) in the following sense:

$$D_{\text{KL}}(p_t \parallel \tilde{p}_t) \leq D_{\text{KL}}(p_{t'} \parallel \tilde{p}_{t'}), \quad \forall t \geq t'. \quad (3)$$

In fact, as  $t \rightarrow \infty$ , we have that:  $D_{\text{KL}}(p_t \parallel \tilde{p}_t) \rightarrow 0$ , as both distributions converge to the same Gaussian. We now define the concept of merging of two distributions towards the same measure.

**Definition 3.1 ( $\epsilon$ -merged)** *We say that two distributions,  $p$  and  $\tilde{p}$  are  $\epsilon$ -merged, if the KL distance between the two is upper-bounded, by  $\epsilon$ , i.e., if  $D_{\text{KL}}(p \parallel \tilde{p}) \leq \epsilon$ .*

Similarly, we define the merging time of two distributions as the minimal amount of noise we need to add such that the two distributions become  $\epsilon$ -merged. Formally,

**Definition 3.2 ( $\epsilon$ -merging time)** *Let two distributions  $p, \tilde{p}$ . We define their  $\epsilon$ -merging time as follows:  $t_n(p, \tilde{p}, \epsilon) = \inf\{t : D_{\text{KL}}(p * \mathcal{N}(0, \sigma(t)^2 I) \parallel \tilde{p} * \mathcal{N}(0, \sigma(t)^2 I)) \leq \epsilon\}$ .*

Assuming we can estimate the  $\epsilon$ -merging time between two distributions  $p$  and  $\tilde{p}$ , our key idea is to treat samples from  $\tilde{p}_t$  as approximate samples from  $p_t$  for all timesteps  $t > t_n(p, \tilde{p}, \epsilon)$ . This idea is illustrated in Figure 2. The intuition is that once the distributions have sufficiently merged under noise, the residual shift becomes negligible and samples from  $\tilde{p}_t$  can be used for learning  $p_t$ . This holds because: (i) the learning algorithm may not be sensitive to small distributional discrepancies at high noise levels, and (ii) even if some bias is introduced, the remaining diffusion trajectory for times  $t \leq t_n(p, \tilde{p}, \epsilon)$  is robust to small initial distributional mismatch due to its inherent stochasticity. For a more in depth analysis of the mathematics behind this intuitive idea we refer the reader to the work of Daras et al. [18].

**Sample dependent noise levels.** At a high level, our objective is to determine the  $\epsilon$ -merging time between the distribution of AlphaFold-predicted structures and that of experimentally resolved proteins. A key challenge arises from the fact that the AlphaFold distribution is highly heterogeneous in structural fidelity—that is, the accuracy with which AlphaFold predicts the true protein structure varies widely across samples. It is well established that short, structurally simple proteins are predicted with higher confidence, while longer and more complex proteins tend to yield lower-confidence predictions. This trend is illustrated in Figure 3B (Left). If we were to assign a single noise level across the entire AlphaFold dataset, we would need to select a relatively high noise level to accommodate the lowest-confidence predictions, particularly from long proteins. This would unnecessarily degrade the training signal for high-confidence structures—regardless of protein length—and limit the model’s

ability to learn from clean supervision. To address this, we treat the AlphaFold dataset as a mixture of  $K$  sub-distributions,  $q_1, q_2, \dots, q_K$ , each representing a distinct confidence regime. We then assign each sub-distribution an appropriate noise level, sufficient to bring it  $\epsilon$ -close to the distribution of high-confidence structures under the same noise schedule. This formulation allows the model to effectively learn from high-confidence AlphaFold predictions and incorporate low-confidence structures in a controlled manner, mitigating the degradation typically caused by noisy training data.

A natural way to decompose the AlphaFold distribution into a mixture of quality-specific sub-distributions is to leverage AlphaFold’s self-reported confidence metric—the average predicted Local Distance Difference Test (pLDDT) score—as a proxy for predicted structural fidelity. In particular, given a dataset  $\mathcal{D} = \{(x_0^{(i)}, \text{pLDDT}^{(i)})\}_{i=1}^N$ , we consider  $K$  distributions (where  $K$  is a hyperparameter to be chosen) with empirical observations for the  $j$ -th distribution being all the samples  $\{(x_0^{(i)}, \text{pLDDT}^{(i)}) : c_{\min}^{(j)} \leq \text{pLDDT}^{(i)} \leq c_{\max}^{(j)}\}$ , for some hyperparameters  $c_{\min}^{(j)}, c_{\max}^{(j)}$ .

**Choice of sub-distribution boundaries.** In this work, we adopt a deliberately simple and conservative strategy by partitioning the AlphaFold dataset into three discrete quality regimes based on the average pLDDT score: high-quality proteins ( $\text{pLDDT} > 90$ ), medium-quality proteins ( $\text{pLDDT}$  in  $[80, 90]$ ) and low-quality proteins ( $\text{pLDDT}$  in  $[70, 80]$ ). We acknowledge that this discretization is coarse and that more principled alternatives may yield further improvements—for instance, by optimizing the bin boundaries or learning a continuous mapping from pLDDT to diffusion time. Despite the simplicity of our choices, our experimental results demonstrate that even a naive quality-aware decomposition can lead to important gains in performance across both short and long proteins. It is important to emphasize that there are two sources of benefit over filtering methods: 1) low-quality data (previously discarded) increases diversity, and 2) the distinction we do between medium-quality and high-quality data boosts designability. The population of proteins that have pLDDT lower than 70 has a very high merging time, and we did not see significant benefits from including them. To improve training efficiency and save computational resources, we decided to discard this subpopulation. We underline that training algorithms that are adaptive to local corruption (rather than global, i.e. average pLDDT) might be able to benefit from such proteins—we leave this direction for future work.

### 3.2 Ambient Protein Diffusion Algorithm

Our algorithm takes as input a dataset of protein structures together with their average pLDDT score,  $\mathcal{D} = \{(x_0^{(i)}, \text{pLDDT}^{(i)})\}_{i=1}^N$ , a diffusion schedule,  $\sigma(t)$ , and a mapping function  $f : [0, 100] \mapsto \mathbb{R}^+$  that translates the average pLDDT value of a protein to its estimated  $\epsilon$ -merging time.

**Annotation stage.** The first step of the algorithm replaces each protein in the dataset with a noisy version of itself, where the noise level is determined by the mapping function  $f$ . This function is a hyperparameter for our algorithm—in our experiments, we opt for a rather simple choice (see Appendix Table 16 for a full description of our training configuration), but in principle this can be an arbitrary function defined by the user based on their specific domain knowledge or experimental findings. After this transformation, each protein can be treated as a sample from the target distribution convolved with a Gaussian at its assigned noise level. This transformation step is only performed once during dataset preprocessing, i.e., we replace the low-quality protein with a noisy version of itself *before* we start the training. This is important because adding different noise realizations across epochs can lead to recovery of the original low-quality protein if the noise is averaged out.

**Loss function.** After the annotation stage, we need to solve a training problem where we have data corrupted at different noise levels with additive Gaussian noise, as in [14, 16]. Hence, we can use the objective of Equation 2. Instead of directly applying the loss, we first need to rescale each time  $t$  to account for the vanishing gradient effect that is due to the multiplicative factor  $a(t)$ . Specifically, we need to rescale the loss at time  $t$  with:  $w(t) = \frac{1}{a^2(t)} = \frac{\sigma^4(t)}{(\sigma^2(t) - \sigma^2(t_i))^2}$  such that we balance the different timesteps. This weighting is derived following the methodology of the EDM [30] paper (Appendix, p. 26, Section B6). We underline that this rescaling was not mentioned in the original paper of Daras et al. [16, 14], for training with noisy data. Yet, we find this rescaling critical for the success of our method. We hypothesize that the authors of [14, 16] did not encounter this issue because there were at most two noise levels considered, while in AF predicted protein structures there is a whole spectrum of assigned noise levels based on the predicted quality (measured by average

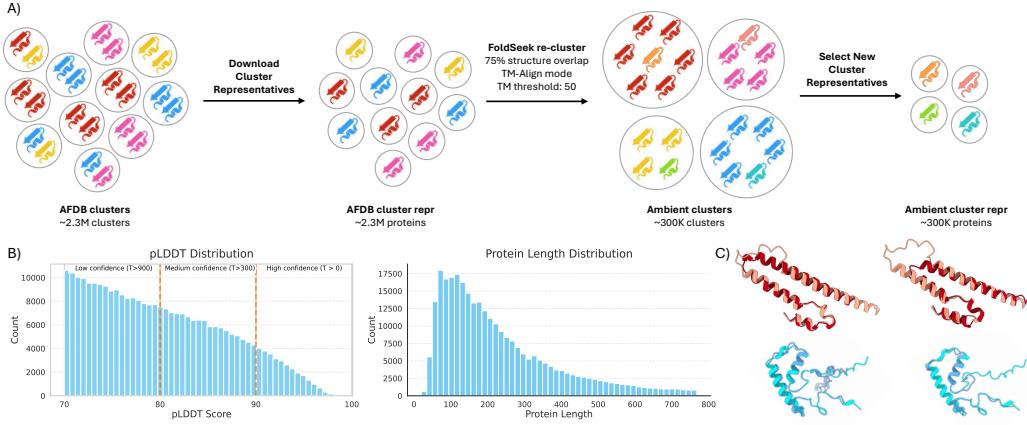


Figure 3: **Reclustering the AFDB cluster dataset to improve generative protein modeling.** (A) Starting from the 2.3M AFDB clustered dataset, we cluster the representatives with FoldSeek optimized for geometric similarity: alignment-type set to TM-Align, TM threshold set to 0.5, and coverage set to 0.75. This results in a 300K clusters ( $p\text{LDDT} > 70$ ) from which we keep the representatives for training. (B)  $p\text{LDDT}$  and protein length statistics for our new training set. (C) Overlay of two Ambient clusters: Top row: Representative (beige; UniProt A0A2W1EPG1) overlaid with members A0A820X4G2 (left, red) and A0A446YZW1 (right, red). Bottom row: Representative (cyan; UniProt A0A395XDB6) overlaid with members A0A3B0YRI6 (left, blue) and L0S475 (right, blue). In the original AFDB cluster dataset, each of these six proteins was designated as the representative of its own cluster despite their similarities in structural features.

$p\text{LDDT}$ ) of a protein structure. We provide further details about the loss implementation in the Appendix (Section E) and pseudocode in Algorithm 1.

**Uniform Protein Sampling in terms of diffusion times.** To perform a training update for a diffusion model, we typically sample a point from the training distribution and then we uniformly sample the noise level  $t$ . However, since in our case we are dealing with noisy data, not all times  $t$  are allowed for a given protein, i.e. a protein with  $p\text{LDDT}^{(i)}$  is only used for times  $t \geq f(p\text{LDDT}^{(i)})$ . To avoid spending most of the training updates on very noisy proteins, we opt for sampling first the diffusion time and then select from the eligible proteins that can be used in that diffusion time. This strategy ensures balanced coverage across the diffusion trajectory—from low to high noise—while still leveraging the diversity of low-confidence structures ( $p\text{LDDT} < 80$ ) in our training dataset.

**Summarizing:** Our algorithm requires three simple changes to the regular diffusion training: 1) an annotation stage (before training) where each low-quality protein is replaced with a noisy version of itself, 2) a change in the way we fetch samples from the dataset so that we do not overallocate training updates to highly noisy proteins and 3) a change in the loss function to account for the fact that for some proteins we do not have access to an uncorrupted structures.

### 3.3 Reclustering AFDB clusters for generative modeling applications

On top of our algorithmic contributions, we also reconsider the choices made for the training dataset. The AFDB clustered dataset [5] has been used to train several generative protein models [34, 23]. However, the original intent behind the clustering was to study structure evolution. Thus, the hyperparameters were chosen to obtain clusters of homologous structures, and the authors report that 97.4% of pairwise comparisons within clusters are conserved at the H-group (Homology) level of the ECOD hierarchical domain classification (median TM-score 0.71). While these FoldSeek hyperparameters are well-suited for evolutionary analysis, we found that the AFDB cluster dataset has a significant degree of structural duplication and near-duplication between clusters that are more distantly evolutionarily related (see Figure 3C). This structural redundancy leads to an imbalanced training set, where structural motifs from the larger protein superfamilies are overrepresented.

Given this finding, we hypothesize that the datasets for generative modeling of protein structures—particularly for backbone-based models—benefit more from clusters defined purely by geometric similarity. To that end, we construct a new clustering dataset derived from the AFDB cluster representatives, with an exclusive focus on structural topology. Specifically, these are the changes we made to the FoldSeek hyperparameters: we switch the alignment-type from 3Di+AA to TM-Align to improve fidelity, we use a TM-score threshold of 0.5, and we relax the alignment coverage from 0.9 to 0.75. We did the latter to improve clustering of AlphaFolded proteins with extended, unfolded N- or C-terminal regions (i.e., noodle tails) (Figure 3C). This approach produced a more balanced dataset that samples structural folds more uniformly, independent of their evolutionary relationships. Ablations that disentangle the contribution of this reclustering from our ambient training approach are given in Figure 4.

## 4 Experimental Results

We build on the Genie2 codebase [34]. Our model architecture follows the Genie2 architecture except that it is scaled larger, using 8 triangle layers as opposed to 5. We train Ambient Protein Diffusion in 3 stages with increasingly longer proteins, eventually reaching proteins up to length 768. For details on the training process, see Appendix Section E.3, and for details on metrics, see Appendix Section D. We underline that the computational cost of training our model is relatively low compared to the prior state of the art Proteína model. This is due to the decreased size of our model (< 17M vs 200M) and training set ( $\sim 290K$  vs  $\sim 780K$ ). We further note that our goal is to develop models that perform well across a range of tasks, including long-protein generation, motif scaffolding, and more. To this end, we train only two models for the purposes of this paper: one model optimized for long-protein generation (Figure 1) and another optimized for short-protein generation (Figure 5).

### 4.1 Comparisons on unconditional generation of longer proteins

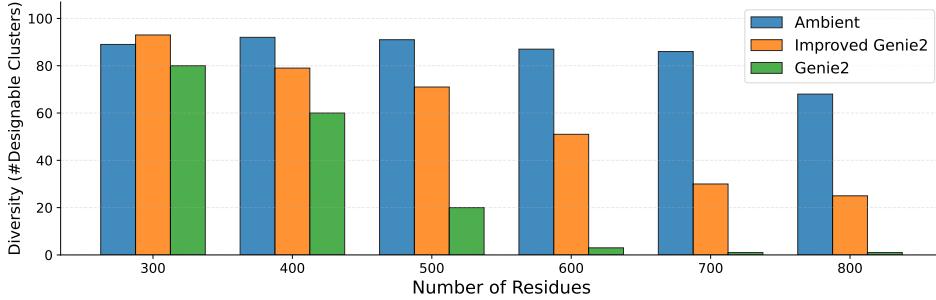
In Figure 1, we compare Ambient Protein Diffusion performance on generating backbone for proteins with length ranging from 300 to 800 residues. To directly compare with Proteína on long-protein generation, we adopt its three-stage training and evaluation protocol. During training, the maximum sequence length is capped at 768 residues. For evaluation, we sample 100 protein backbones at each target length and evaluate them using the designability and diversity metrics. Since Ambient Protein Diffusion builds on Genie2, we use the same sampling procedure—running 1000 diffusion steps with a noise scale of  $\gamma = 0.6$ . This noise scale parameter controls the trade-off between the designability and diversity by reducing the amount of stochasticity added in the reverse process, as it is typically done in the protein generative modeling literature (see Appendix E.4.1 for details).

Ambient Protein Diffusion achieves designability and diversity scores exceeding 90% for proteins between 300 and 500 residues, and maintains scores above 85% for lengths up to 700 residues. For 800-residue proteins, both metrics decline to 68%. Compared to Proteína, Ambient Protein Diffusion outperforms by 26% in designability and 91% in diversity at length 700, and by 24% and 45%, respectively, at length 800. At every protein length, Ambient Protein Diffusion’s diversity is equal to its designability, indicating that every designable protein is unique. This is not the case for Proteína, where diversity scores consistently fall below designability, regardless of protein length.

Taken together, these results demonstrate the impact of ambient diffusion on backbone-based generative models and highlight the strength of Genie2’s equivariant architecture. Our 17M parameter model trained on approximately 290K AlphaFold structures significantly outperforms a 200M-parameter transformer model trained on roughly 780K proteins. Our results show that smaller, more efficient models can surpass larger transformer baselines in both structural diversity and designability.

### 4.2 Ablating the significance of Ambient Diffusion

In comparison to Genie2, the starting point of our implementation, we made the following changes: 1) made the model bigger, 2) trained on longer proteins, 3) reclustered the dataset to optimize for geometric similarity rather than evolutionary similarity, and 4) used low pLDDT AlphaFold proteins as noisy data using our Ambient Protein Diffusion framework. To quantify how much of the improvement comes from the latter step, we train an Improved Genie2 without the Ambient Framework for training with corrupted data and we report results in Figure 4. We find that while



**Figure 4: Ablation to quantify the effect of the Ambient Protein Diffusion framework.** We improve upon Genie2 by making the architecture bigger, reclustering the dataset as in Section 3.3 and by finetuning on longer proteins (up to 768 aminoacids). The resulting improved Genie2, shown in Orange, outperforms Genie2 but still lags behind our Ambient Protein Diffusion Model, shown in blue. The only difference between the two models is that Ambient Protein Diffusion uses low pLDDT AlphaFold structures as noisy data, as explained in Section 3.2. Ambient Protein Diffusion consistently outperforms both the improved Genie2 baseline and Genie2, with increasingly significant improvements as sequence length grows.

our two models perform similarly on proteins of 300 residues, the designability and diversity of the baseline diminish on longer proteins. For proteins with 800 residues, the number of designable clusters drops from 68% to 25%. Ambient Protein Diffusion shows a marked improvement, maintaining a stable number of designable clusters. We underline that the difference comes solely from the training algorithm since the architecture, model size, hyperparameters, and inference algorithm remain the same. We note that this experiment only ablates the impact of treating AlphaFold data as noisy, and fixes the reclustered dataset (Section 3.3) for both models. We examine the impact of reclustering separately in Section B.3 and find that reclustering increases diversity; however, without Ambient training, the designability remains low. Both innovations are needed to achieve optimal results.

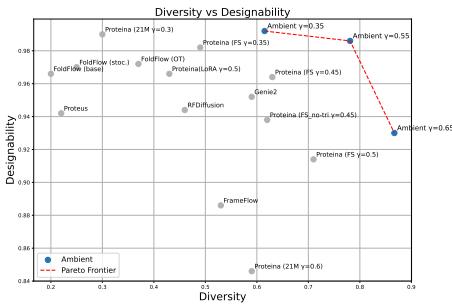
### 4.3 Comparisons on unconditional generation of shorter proteins

In this experiment, we evaluate the model on the unconditional generation of shorter proteins in Figure 5. We provide training details for the model optimized for short protein generation in the Appendix Section E.3.2.

Following the Genie2 protocol, we generate 5 structures for each sequence length from 50 to 256 residues, yielding a total of 1,035 structures. The generated structures are evaluated for both designability and diversity. In line with prior work, we sweep the noise scale  $\gamma$  to explore the trade-off between designability and diversity. Ambient Protein Diffusion outperforms previous methods on both metrics, establishing a new Pareto frontier that achieves superior performance compared to all existing models, including Proteina.

### 4.4 Novelty scores

The next step is to show that our method does not just memorize proteins, but instead it can generate novel designable structures that are distinct from the training set. Here, we compute the TM-novelty metric following the protocol of Geffner et al. [23]. However, we found two key issues when reproducing the literature TM-novelty scores, which we discuss in detail in Appendix Section D.3. First, since Spring 2025 FoldSeek resolved a bug in computing the alntmscore (Github issue 312), which makes all previous reported values incorrect. Second, we found that Geffner et al. [23]



**Figure 5: Designability - diversity trade-off for short protein generation** (up to 256 residues). Ambient dominates completely the Pareto frontier between designability and diversity, while using a  $12.88 \times$  smaller model. We further do so without using any higher-order sampler or (auto-) guidance method.

Table 1: **PDB and AFDB TM-Noveltiy for short protein generation.** For each model, we sample 5 structures for each sequence length from 50 to 256 residues, yielding a total of 1,035 structures. A lower TM-Noveltiy score is better.

Model	PDB Novelty (↓)	AFDB Novelty (↓)
<i>Ambient Proteins</i>		
$\gamma = 0.35$	0.774	<b>0.848</b>
$\gamma = 0.55$	<b>0.773</b>	0.851
$\gamma = 0.65$	0.774	0.858
<i>Baselines</i>		
Genie2	0.789	0.862
RFDiffusion	0.853	0.923
Proteus	0.842	0.884
Chroma	0.824	0.885
Proteina (FS $\gamma = 0.35$ )	0.853	0.910
Proteina (FS $\gamma = 0.45$ )	0.837	0.898
Proteina (FS $\gamma = 0.5$ )	0.831	0.893
Proteina (FS_no-tri $\gamma = 0.45$ )	0.832	0.891
Proteina (21M $\gamma = 0.3$ )	0.889	0.932
Proteina (21M $\gamma = 0.6$ )	0.854	0.905

inadvertently used the alnTM-Score from the row with the highest qTM-Score, rather than from the row with the highest alnTM-Score. To ensure accurate benchmarking with the literature, we recalculated TM-Noveltiy with the patched FoldSeek v10 for several baselines, explicitly selecting the max alnTM-Score for each query. We report results in Table 1. For backward compatibility, we also reproduced literature results using the unpatched FoldSeek v9 and the default qTM-Score sorting (Table 15). Moving forward, we strongly recommend that adopting FoldSeek v10 and always using the max alnTM-Score value to determine the novelty per query when computing TM-Noveltiy.

Using both versions of FoldSeek, Ambient Protein Diffusion sets new state-of-the-art TM-novelty scores on both the PDB and AFDB (588K) benchmarks. In the short-evaluation regime ( $\leq 256$  AAs) using FoldSeek v10, we exceed the next-best model (Genie2) by 2.0% on PDB and 1.6% on AFDB—despite Genie2’s restriction to proteins no longer than 256 AAs. Against Proteina, we further boost TM-novelty by 7.0% on PDB and 4.8% on AFDB. In the long-evaluation regime (300–800 AAs), we focus on comparison with Proteina. Here, we achieve TM-novelty scores of 0.682 on PDB and 0.740 on AFDB, representing improvements of 18.4% and 16.2%, respectively. Together, these results demonstrate that Ambient Protein Diffusion, driven by the Ambient loss and cluster dataset, produces the most novel proteins across both short and long-sequence settings.

#### 4.5 Motif Scaffolding

As a final evaluation, we compare our method to prior work in motif scaffolding. The full results are shown in Appendix Figure 7 and in Appendix Tables 11 and 12. With  $\gamma = 0.45$ , Ambient Protein Diffusion generates 1,923 unique successful scaffolds for single-motif tasks, a significant improvement over Genie2’s 1,445 [34] and performs comparably to a Proteína model (2,094 [23]), which is much larger (200M parameters vs 17M parameters) and is optimized specifically for motif scaffolding. For multi-motif scaffolding, Ambient Protein Diffusion generates 89 unique successful structures across 5 of the 6 problems, outperforming Genie2, which produces 40 and solves 4.

### 5 Conclusion

We introduced *Ambient Protein Diffusion*, a framework for protein structure generation that leverages low-confidence AlphaFold structures as a source of noisy training data. Ambient Protein Diffusion enables the generation of long protein structures with unprecedented levels of designability, diversity and novelty. Diversity increases as it can use low-confidence AlphaFold structures that are typically discarded and designability increases as we separate the pristine quality proteins structures from the medium quality AlphaFold predictions. Ambient Protein Diffusion represents a foundational step toward robust de novo protein design at more natural, biologically relevant lengths.

## 6 Acknowledgements

This research has been supported by NSF Awards CCF-1901292, ONR grants N00014-25-1-2116, N00014-25-1-2296, a Simons Investigator Award, and the Simons Collaboration on the Theory of Algorithmic Fairness. The experiments were run on the Vista GPU Cluster through the Center for Generative AI (CGAI) and the Texas Advanced Computing Center (TACC) at UT Austin.

**Note on authorship.** William Daspit and Qiang Liu joined the team after an initial version of this project was submitted to NeurIPS. Both authors made major contributions to the final version. Due to the NeurIPS policy preventing authorship changes after submission, their names could not appear on the paper or the conference website. However, all the authors of this manuscript agree that they made very significant contributions and that this project would not be the same without their work. A version of this paper with the updated authorlist is available on bioarXiv: <https://www.biorxiv.org/content/10.1101/2025.07.03.663105>.

## References

- [1] Asad Aali, Marius Arvinte, Sidharth Kumar, and Jonathan I Tamir. Solving inverse problems with score-based generative priors learned from noisy data. *arXiv preprint arXiv:2305.01166*, 2023.
- [2] Asad Aali, Giannis Daras, Brett Levac, Sidharth Kumar, Alex Dimakis, and Jon Tamir. Ambient diffusion posterior sampling: Solving inverse problems with diffusion models trained on corrupted data. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=qeXcMutEZY>.
- [3] Brian D.O. Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.
- [4] Weimin Bai, Yifei Wang, Wenzheng Chen, and He Sun. An expectation-maximization algorithm for training clean diffusion models from corrupted observations. *arXiv preprint arXiv:2407.01014*, 2024.
- [5] Inigo Barrio-Hernandez, Jingi Yeo, Jürgen Jänes, Milot Mirdita, Cameron LM Gilchrist, Tanita Wein, Mihaly Varadi, Sameer Velankar, Pedro Beltrao, and Martin Steinegger. Clustering predicted structures at the scale of the known protein universe. *Nature*, 622(7983):637–645, 2023.
- [6] Jeremy M Berg, John L Tymoczko, and Lubert Stryer. *Biochemistry (loose-leaf)*. Macmillan, 2007.
- [7] Avishek Joey Bose, Tara Akhound-Sadegh, Guillaume Huguet, Kilian Fatras, Jarrid Rector-Brooks, Cheng-Hao Liu, Andrei Cristian Nica, Maksym Korablyov, Michael Bronstein, and Alexander Tong. Se (3)-stochastic flow matching for protein backbone generation. *arXiv preprint arXiv:2310.02391*, 2023.
- [8] Jose M Carceller, Bhumika Jayee, Claire G Page, Daniel G Oblinsky, Gustavo Mondragón-Solórzano, Nithin Chintala, Jingzhe Cao, Zayed Alassad, Zheyu Zhang, Nathaniel White, et al. Engineering a photoenzyme to use red light. *Chem*, 11(2), 2025.
- [9] Sitan Chen, Sinho Chewi, Jerry Li, Yuanzhi Li, Adil Salim, and Anru R Zhang. Sampling is as easy as learning the score: theory for diffusion models with minimal data assumptions. *arXiv preprint arXiv:2209.11215*, 2022.
- [10] Sitan Chen, Giannis Daras, and Alex Dimakis. Restoration-degradation beyond linear diffusions: A non-asymptotic analysis for DDIM-type samplers. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 4462–4484. PMLR, 23–29 Jul 2023. URL <https://proceedings.mlr.press/v202/chen23e.html>.

[11] Xinshi Chen, Yuxuan Zhang, Chan Lu, Wenzhi Ma, Jiaqi Guan, Chengyue Gong, Jincai Yang, Hanyu Zhang, Ke Zhang, Shenghao Wu, Kuangqi Zhou, Yanping Yang, Zhenyu Liu, Lan Wang, Bo Shi, Shaochen Shi, and Wenzhi Xiao. Proteinix - advancing structure prediction through a comprehensive alphafold3 reproduction. *bioRxiv*, 2025. doi: 10.1101/2025.01.08.631967.

[12] Alexander E Chu, Tianyu Lu, and Po-Ssu Huang. Sparks of function by de novo protein design. *Nature biotechnology*, 42(2):203–215, 2024.

[13] UniProt Consortium. Uniprot: a hub for protein information. *Nucleic acids research*, 43(D1): D204–D212, 2015.

[14] Giannis Daras, Yuval Dagan, Alexandros G Dimakis, and Constantinos Daskalakis. Consistent diffusion models: Mitigating sampling drift by learning to be consistent. *arXiv preprint arXiv:2302.09057*, 2023.

[15] Giannis Daras, Kulin Shah, Yuval Dagan, Aravind Gollakota, Alex Dimakis, and Adam Klivans. Ambient diffusion: Learning clean distributions from corrupted data. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=wBJBLy9kBY>.

[16] Giannis Daras, Alexandros G Dimakis, and Constantinos Daskalakis. Consistent diffusion meets tweedie: Training exact ambient diffusion models with noisy data. *arXiv preprint arXiv:2404.10177*, 2024.

[17] Giannis Daras, Yeshwanth Cherapanamjeri, and Constantinos Costis Daskalakis. How much is a noisy image worth? data scaling laws for ambient diffusion. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=qZwtPEw2qN>.

[18] Giannis Daras, Adrian Rodriguez-Munoz, Adam Klivans, Antonio Torralba, and Constantinos Daskalakis. Ambient diffusion omni: Training good models with bad data, 2025.

[19] Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J Ragotte, Lukas F Milles, Basile IM Wicky, Alexis Courbet, Rob J de Haas, Neville Bethel, et al. Robust deep learning-based protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022.

[20] Daniel J Diaz, Anastasiya V Kulikova, Andrew D Ellington, and Claus O Wilke. Using machine learning to predict the effects and consequences of mutations in proteins. *Current opinion in structural biology*, 78:102518, 2023.

[21] Daniel J Diaz, Chengyue Gong, Jeffrey Ouyang-Zhang, James M Loy, Jordan Wells, David Yang, Andrew D Ellington, Alexandros G Dimakis, and Adam R Klivans. Stability oracle: a structure-based graph-transformer framework for identifying stabilizing mutations. *Nature Communications*, 15(1):6170, 2024.

[22] Cong Fu, Keqiang Yan, Limei Wang, Wing Yee Au, Michael Curtis McThrow, Tao Komikado, Koji Maruhashi, Kanji Uchino, Xiaoning Qian, and Shuiwang Ji. A latent diffusion model for protein structure generation. In *Learning on Graphs Conference*, pages 29–1. PMLR, 2024.

[23] Tomas Geffner, Kieran Didi, Zuobai Zhang, Danny Reidenbach, Zhonglin Cao, Jason Yim, Mario Geiger, Christian Dallago, Emine Kucukbenli, Arash Vahdat, et al. Proteina: Scaling flow-based protein structure generative models. *arXiv preprint arXiv:2503.00710*, 2025.

[24] Chengyue Gong, Adam Klivans, James Madigan Loy, Tianlong Chen, Daniel Jesus Diaz, et al. Evolution-inspired loss functions for protein representation learning. In *Forty-first International Conference on Machine Learning*, 2024.

[25] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[26] Maria Hauser, Martin Steinegger, and Johannes Söding. Mmseqs software suite for fast and deep clustering and searching of large protein sequence sets. *Bioinformatics*, 32(9):1323–1330, 2016.

[27] Danial Hosseintabar, Fan Chen, Giannis Daras, Antonio Torralba, and Constantinos Daskalakis. Diffem: Learning from corrupted data with diffusion models via expectation maximization. *arXiv preprint arXiv:2510.12691*, 2025.

[28] John B Ingraham, Max Baranov, Zak Costello, Karl W Barber, Wujie Wang, Ahmed Ismail, Vincent Frappier, Dana M Lord, Christopher Ng-Thow-Hing, Erik R Van Vlack, et al. Illuminating protein space with a programmable generative model. *Nature*, 623(7989):1070–1078, 2023.

[29] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.

[30] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *arXiv preprint arXiv:2206.00364*, 2022.

[31] Tero Karras, Miika Aittala, Tuomas Kynkänniemi, Jaakko Lehtinen, Timo Aila, and Samuli Laine. Guiding a diffusion model with a bad version of itself. *Advances in Neural Information Processing Systems*, 37:52996–53021, 2024.

[32] Albert L Lehninger, David L Nelson, and Michael M Cox. *Lehninger principles of biochemistry*. Macmillan, 2005.

[33] Yeqing Lin and Mohammed AlQuraishi. Generating novel, designable, and diverse protein structures by equivariantly diffusing oriented residue clouds. *arXiv preprint arXiv:2301.12485*, 2023.

[34] Yeqing Lin, Minji Lee, Zhao Zhang, and Mohammed AlQuraishi. Out of many, one: Designing and scaffolding proteins at the scale of the structural universe with genie 2. *arXiv preprint arXiv:2405.15489*, 2024.

[35] 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.

[36] Yi Liu, Sophie G Bender, Damien Sorigue, Daniel J Diaz, Andrew D Ellington, Greg Mann, Simon Allmendinger, and Todd K Hyster. Asymmetric synthesis of  $\alpha$ -chloroamides via photoenzymatic hydroalkylation of olefins. *Journal of the American Chemical Society*, 146(11):7191–7197, 2024.

[37] Haoye Lu, Qifan Wu, and Yaoliang Yu. SFBD: A method for training diffusion models with noisy data. In *Frontiers in Probabilistic Inference: Learning meets Sampling*, 2025. URL <https://openreview.net/forum?id=6HN14zuHRb>.

[38] Nanye Ma, Shangyuan Tong, Haolin Jia, Hexiang Hu, Yu-Chuan Su, Mingda Zhang, Xuan Yang, Yandong Li, Tommi Jaakkola, Xuhui Jia, et al. Inference-time scaling for diffusion models beyond scaling denoising steps. *arXiv preprint arXiv:2501.09732*, 2025.

[39] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021.

[40] Jeffrey Ouyang-Zhang, Daniel Diaz, Adam Klivans, and Philipp Krähenbühl. Predicting a protein’s stability under a million mutations. *Advances in Neural Information Processing Systems*, 36:76229–76247, 2023.

[41] Jeffrey Ouyang-Zhang, Chengyue Gong, Yue Zhao, Philipp Krähenbühl, Adam R Klivans, and Daniel J Diaz. Distilling structural representations into protein sequence models. *bioRxiv*, pages 2024–11, 2024.

[42] François Rozet, Gérôme Andry, François Lanusse, and Gilles Louppe. Learning diffusion priors from observations by expectation maximization. *arXiv preprint arXiv:2405.13712*, 2024.

[43] Kulin Shah, Alkis Kalavasis, Adam R. Klivans, and Giannis Daras. Does generation require memorization? creative diffusion models using ambient diffusion, 2025.

[44] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[45] Michel van Kempen, Stephanie S Kim, Charlotte Tumescheit, Milot Mirdita, Cameron LM Gilchrist, Johannes Söding, and Martin Steinegger. Foldseek: fast and accurate protein structure search. *Biorxiv*, pages 2022–02, 2022.

[46] Mihaly Varadi, Stephen Anyango, Mandar Deshpande, Sreenath Nair, Cindy Natassia, Galabina Yordanova, David Yuan, Oana Stroe, Gemma Wood, Agata Laydon, et al. AlphaFold protein structure database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic acids research*, 50(D1):D439–D444, 2022.

[47] Chentong Wang, Yannan Qu, Zhangzhi Peng, Yukai Wang, Hongli Zhu, Dachuan Chen, and Longxing Cao. Proteus: exploring protein structure generation for enhanced designability and efficiency. *bioRxiv*, pages 2024–02, 2024.

[48] Jue Wang, Sidney Lisanza, David Juergens, Doug Tischer, Joseph L Watson, Karla M Castro, Robert Ragotte, Amijai Saragovi, Lukas F Milles, Minkyung Baek, et al. Scaffolding protein functional sites using deep learning. *Science*, 377(6604):387–394, 2022.

[49] Yifei Wang, Weimin Bai, Weijian Luo, Wenzheng Chen, and He Sun. Integrating amortized inference with diffusion models for learning clean distribution from corrupted images. *arXiv preprint arXiv:2407.11162*, 2024.

[50] Joseph L Watson, David Juergens, Nathaniel R Bennett, Brian L Trippe, Jason Yim, Helen E Eisenach, Woody Ahern, Andrew J Borst, Robert J Ragotte, Lukas F Milles, et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976):1089–1100, 2023.

[51] Carter J Wilson, Wing-Yiu Choy, and Mikko Karttunen. AlphaFold2: a role for disordered protein/region prediction? *International Journal of Molecular Sciences*, 23(9):4591, 2022.

[52] Seung-Gyun Woo, Danny J Diaz, Wantae Kim, Mason Galliver, and Andrew D Ellington. Machine learning-guided engineering of t7 rna polymerase and mrna capping enzymes for enhanced gene expression in eukaryotic systems. *Chemical Engineering Journal*, page 165191, 2025.

[53] Kevin E Wu, Kevin K Yang, Rianne van den Berg, Sarah Alamdari, James Y Zou, Alex X Lu, and Ava P Amini. Protein structure generation via folding diffusion. *Nature communications*, 15(1):1059, 2024.

[54] Jinrui Xu and Yang Zhang. How significant is a protein structure similarity with tm-score= 0.5? *Bioinformatics*, 26(7):889–895, 2010.

[55] Jason Yim, Andrew Campbell, Andrew YK Foong, Michael Gastegger, José Jiménez-Luna, Sarah Lewis, Victor Garcia Satorras, Bastiaan S Veeling, Regina Barzilay, Tommi Jaakkola, et al. Fast protein backbone generation with se (3) flow matching. *arXiv preprint arXiv:2310.05297*, 2023.

## NeurIPS Paper Checklist

### 1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: **[Yes]**

Justification: The claims made in the abstract and introduction are validated in the experimental results.

## 2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: See conclusion section.

## 3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper does not introduce any new theoretical results.

## 4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide thorough implementation details in the paper and supplement.

## 5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Code and data are publicly available.

## 6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We provide thorough implementation details in the paper and supplement.

## 7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Our evaluation follows the standard deterministic benchmarking setup commonly used in the protein design literature.

## 8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We detail the compute requirements in the implementation details.

## 9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: We have reviewed the ethics guidelines and find our paper conforms.

## 10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [\[Yes\]](#)

Justification: See conclusion section.

#### 11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [\[NA\]](#)

Justification: Paper poses no such risks.

#### 12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [\[Yes\]](#)

Justification: All existing models and datasets are properly credited.

#### 13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [\[Yes\]](#)

Justification: New datasets, code, and models are released, with instructions on how to use them.

#### 14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [\[NA\]](#)

Justification: No human subjects.

#### 15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [\[NA\]](#)

Justification: No human subjects.

#### 16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [\[NA\]](#)

Justification: LLM are not part of the core method.

## A Limitations and Future Work

This work represents a first step toward protein generative models that make better use of the synthetic structures from the AlphaFold Database. Nevertheless, there are several clear avenues for improvement. (i) Structure-quality metric. We rely on AlphaFold’s self-reported pLDDT score—a coarse, residue-averaged confidence measure that can itself be noisy or misleading, (ii) Data coverage. We only use one representative per AFDB cluster rather than incorporating all available cluster members, and we build on the existing AFDB clustering rather than reclustering the full 214M-structure dataset, (iii) pLDDT-merging-time mapping. Our choice of how to translate pLDDT values into merging thresholds was driven by empirical tuning rather than by a systematic ablation study or principled selection criterion, (iv) Experimental validation. Ultimately, the real test of any generative model is whether its predictions hold up in the laboratory. We have yet to confirm our structures experimentally.

## B Ablations

### B.1 Mapping function ablations

The algorithm described in Section 3.2 assumes a mapping function that maps a protein’s pLDDT to a diffusion time after which the protein can be used for learning. For all the experiments of the paper, we used a discrete map (detailed in Table 16); in particular proteins that have  $\text{pLDDT} \geq 90$  are used everywhere, proteins with  $\text{pLDDT} \in [80, 90)$  are used for only the last 400 diffusion steps (out of the 1000 total) and proteins that have  $\text{pLDDT} \in [70, 80)$  are only used for the last 100 diffusion steps. This mapping was chosen to represent three coarse categories; clean samples, semi-clean proteins and very noisy AlphaFold predictions. In this section, we ablate the choice of the mapping function to 1) show the robustness of our method to reasonable choices, and 2) show that it is possible to achieve improved results by optimizing this mapping.

We show results in Table 2 for small changes to the default choice used in the paper. Specifically, we change what happens when we inflate or deflate all the pLDDT boundaries in the discrete chosen mapping by 5 pLDDT points and we observe the differences in designability and diversity. For all these results, we use noise scale  $\gamma = 0.6$  during sampling. As shown, our method is relatively robust, as small perturbations to the mapping don’t lead to a very substantial change. In fact, the inflated thresholds lead to a model that belongs to the Pareto frontier of Figure 4 in the paper. Running the same model with  $\gamma = 0.65$  leads to another Pareto point, achieving an (86.6, 0.882) designability-diversity pair.

plddt_to_timestep	Designability	Diversity	Notes
(80,0)	95.2	0.590	Genie2 baseline
(90,0),(80,300),(70,900)	98.6	0.781	Paper choice
(85,0),(75,300),(70,900)	96.4	0.780	Deflated thresholds by 5
(95,0),(85,300),(75,900)	96.8	0.783	Inflated thresholds by 5

Table 2: **Effect of changing the pLDDT-to-timestep function.** Slight adjustments to the thresholds lead to reasonable results, indicating that the method is robust.

Now that we established the robustness of our approach, we present further results that can be obtained by optimizing the pLDDT to timestep mapping. Towards that goal, we try using more discrete bins and we also explore two continuous generalizations to interpolate between the pLDDT boundaries used in the paper. In particular, we try a piecewise continuous function and a sigmoid function. To explain with an example, before all proteins with  $\text{pLDDT} \in (80, 90]$  were used for times 300-1000. Under the piecewise continuous generalization, a protein with  $\text{pLDDT} x$  in  $(80, 90]$  will be used for times  $t \geq 300 \cdot (90 - x)/(90 - 80)$ , e.g. a protein with  $\text{pLDDT} 82$  is used for times  $t \geq 240$ . The results are given in Table 3.

As shown, by optimizing the mapping function, we can even outperform the results we obtained in the paper. That said, even a simplistic choice, as the ones we opted for in the paper, can already yield significant boosts over the Genie2 baseline.

pLDDT to timestep mapping	Designability	Diversity	Pareto	Notes
(90,0),(80,300),(70,900)	98.6	0.781	✓ Pareto optimal	Paper choice
Piecewise linear	96.2	0.827	✓ Pareto optimal	
Sigmoid	98.2	0.752		
(90,0),(85,300),(80,600),(75,900),(70,950)	95.9	0.742		Extra bins 1
(90,0),(85,200),(80,300),(75,850),(70,900)	94.2	0.858	✓ Pareto optimal	Extra bins 2

Table 3: **Effect of different pLDDT-to-timestep mappings.** Several mappings achieve Pareto optimality, balancing designability and diversity.

## B.2 Weighting function ablation

In the paper, we used the weighting term  $w(t) = \frac{1}{\sigma^2(t)} = \frac{\sigma^4(t)}{(\sigma^2(t) - \sigma^2(t_i))^2}$  in the Ambient loss to balance the contribution of different diffusion times and avoid vanishing gradients. We ablate this choice in Table 4. As shown, removing that weighting leads to significant deterioration of the obtained performance, both in designability and diversity. The calculations for this weighting follow the EDM [30] methodology, Appendix p. 26, Section B6.

Weighting	Designability	Diversity	Pareto
Paper choice	98.6	0.781	✓ Pareto optimal
Constant (w=1)	97.4	0.733	

Table 4: **Effect of weighting term**  $w(t) = \frac{\sigma^4(t)}{(\sigma^2(t) - \sigma^2(t_i))^2}$  **in the loss.**

## B.3 Dataset reclustering ablation

In Figure 4 of the main paper, we fix the dataset to the geometric reclustered version we created in this work, and we ablate the effect of Ambient training. In this section, we perform additional experiments to understand the standalone value of geometric reclustering. We report results with all combinations of Ambient vs. no Ambient and reclustering vs. no reclustering in Table 5. As shown in the Table, reclustering leads to improved diversity, but without Ambient designability is low. Both innovations are needed to achieve the optimal performance.

Setting	Dataset	Designability $\uparrow$	Diversity $\uparrow$
No Ambient	Genie2	95.2	0.590
No Ambient	Re-clustered	81.4	0.902
Ambient	Genie2	96.4	0.501
Ambient	Re-clustered	98.6	0.781

Table 5: **Effect of dataset reclustering.** Re-clustering increases diversity, but without Ambient training, the designability is low. The combination of the two innovations yields the optimal results.

## B.4 Loss ablation

For low pLDDT structures,  $x_0$  cannot be trusted. Instead, our loss involves predicting a noisy version of the low pLDDT protein, which can be trusted more than the original protein because, intuitively, the noise has erased some of the AlphaFold prediction mistakes. For completeness and to support our argument, we provide one further ablation where we train a model with the same setup as the short-protein generation in the paper, but using  $x_0$  prediction loss instead of the proposed loss. Results are shown in Table 6.

For a more detailed discussion on the benefits of the Ambient loss over the original diffusion loss, we refer the reader to the work of Shah et al. [43].

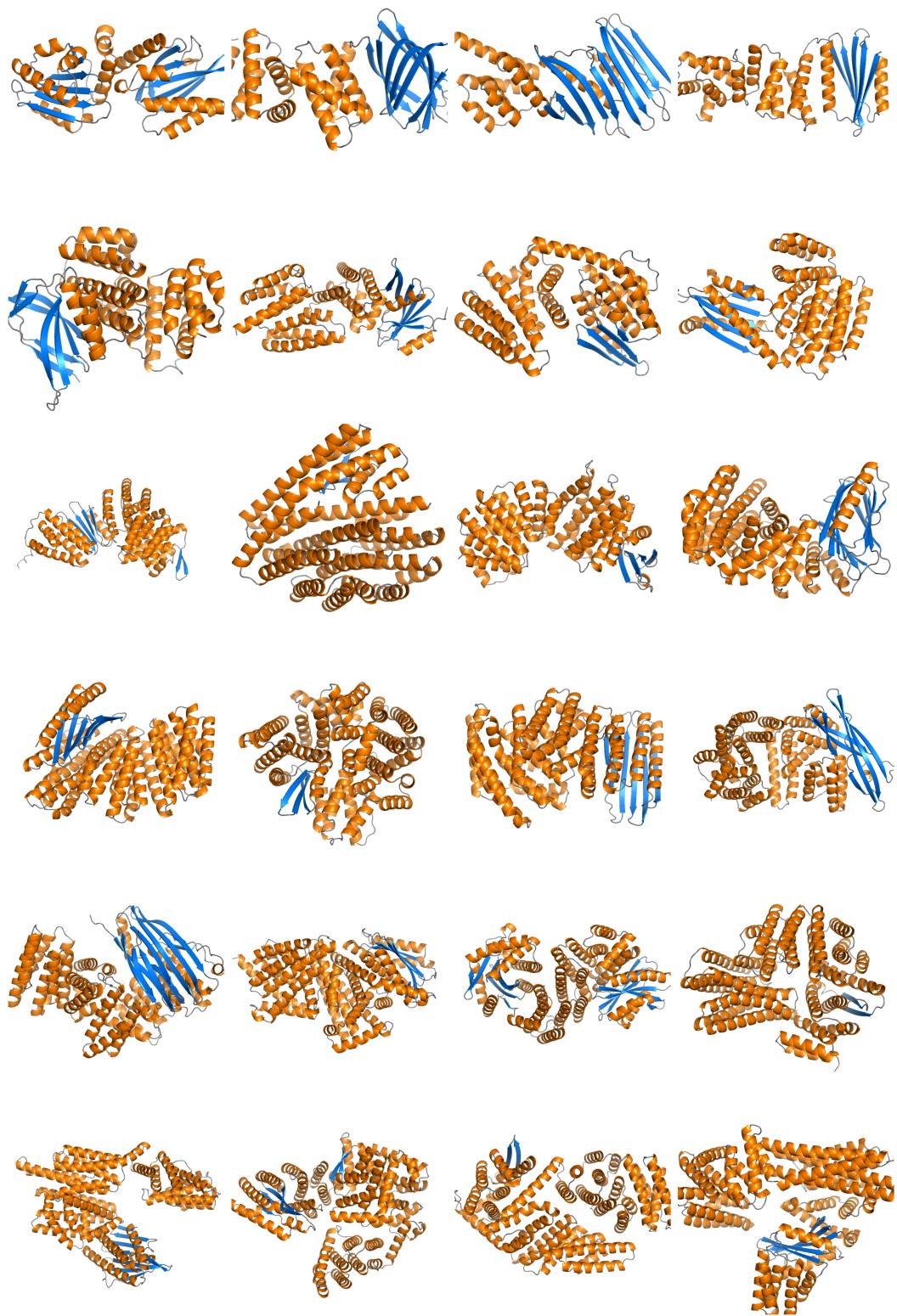


Figure 6: Additional qualitative visualizations of unconditional generations.

Loss Type	Designability	Diversity
Ambient loss	98.6	0.781
$x_0$ prediction loss	98.5	0.753

Table 6: **Effect of Ambient loss.** Ambient loss achieves slightly higher designability and diversity compared to  $x_0$  prediction loss.

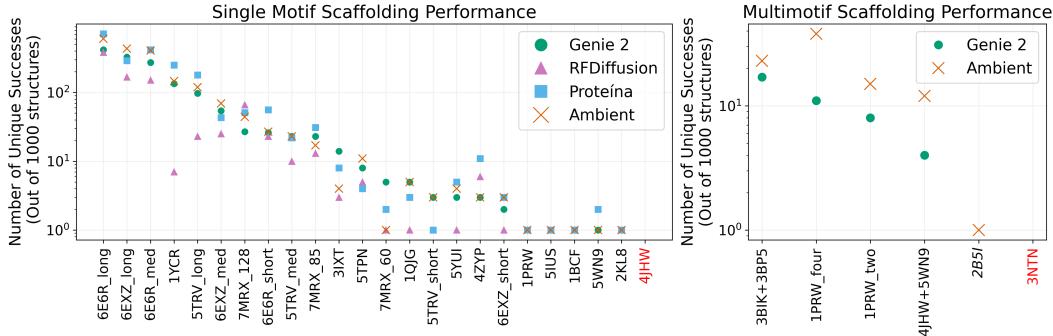


Figure 7: **Performance on Motif Scaffolding Tasks.** We compare Ambient Protein Diffusion to state-of-the-art models for motif scaffolding. The graphs show the number of unique successful scaffolds generated for each single- and multi-motif task. No model produced successful scaffolds for **4JHW** and **3NTN**. Only Ambient Protein Diffusion produced a valid solution for multi-motif scaffolding of **2B5I**.

## C Additional Results

### C.1 Motif Scaffolding Results

We additionally compare our method to prior work on motif scaffolding in Figure 7, with full results provided in the supplement. Our evaluation follows the Genie2 benchmark, which comprises 24 single-motif and 6 multi-motif design tasks [34, 50]. For each task, we generate 1,000 scaffold samples using a noise scale of  $\gamma = 0.45$ . A design is considered successful if it (1) satisfies Genie2’s motif designability criteria and (2) preserves the motif with an RMSD below 1Å. Among successful designs, a scaffold is counts as unique if its TM-score is at most 0.6 when compared to any other successful scaffold. A task is considered solved if at least one successful scaffold is generated.

With  $\gamma = 0.45$ , Ambient Protein Diffusion generates 1,923 unique successful scaffolds for single-motif tasks, a significant improvement over Genie2’s 1,445 [34] and performs comparably to a Proteína model (2,094 [23]) that is much larger (200M parameters vs 17M parameters) and is optimized specifically for motif scaffolding. Notably, all methods solve a similar number of motifs – RFDiffusion solves 22 of the 24 tasks, while Ambient Protein Diffusion, Genie2, and Proteína each solve the same 23 tasks. For multi-motif scaffolding, Ambient Protein Diffusion generates 89 unique successful structures across 5 of the 6 benchmark problems, outperforming Genie2, which produces 40 and solves 4. Ambient Protein Diffusion performs particularly well on the 1PRW\_four motif (38 vs. 11 successful structures) in which a scaffold is generated surrounding a calcium binding motif [48]. Overall, Ambient Protein Diffusion outperforms existing methods such as Genie2 and RFDiffusion on single-motif tasks and matches the performance of a Proteína model optimized specifically for motif-scaffolding.

We want to underline that when counting successes in motif scaffolding it is important to fix the scaffold length for a given motif for all the baselines. In particular, we noticed that increasing the protein length leads to consistent improvements. For fair comparison between models, we provide results for Ambient Protein Diffusion, Genie, and RFDiffusion, where increasing the scaffold lengths by 1.5 increased the total number of successes. Assuming we use the midpoint of the minimum and maximum scaffold lengths as provided in the Genie 2 benchmark (structures are provided by the RFDiffusion benchmark, but the lengths are from Genie), Genie generates 1327 successes, Ambient

Protein Diffusion generates 1849, and RFDiffusion generates 694. If we increase the length by 1.5, Genie, Ambient Protein Diffusion and RFDiffusion generate 2486, 3776 and 1033 successful scaffolds respectively.

## C.2 Secondary Structure Evaluations

In this section, we systematically evaluate the Secondary Structure Proportions of our generated proteins. Specifically, we first use biotite to assign secondary structure labels to all the generated proteins. Then, we look at the designable PDB files, we calculate the percentages of alpha/beta/loops, and then we average those over all designable PDBs. The results are shown in Table 7.

Method	$\alpha$ -Helix (%) ↓	$\beta$ -Sheet (%) ↑
Genie2	72.7	4.8
Proteina MFS 0.35	71.6	5.8
Proteina MFS 0.45	68.1	6.9
Proteina MFS 0.50	67.0	7.2
Ambient Protein Diffusion (scale 0.5)	69.9	7.9
Ambient Protein Diffusion (scale 0.4)	66.6	9.5

Table 7: **Secondary structure composition across methods.** Lower  $\alpha$ -Helix and higher  $\beta$ -Sheet percentages indicate more natural outputs.

As shown, Ambient Protein Diffusion generates proteins with more beta sheets and loops compared to Proteina and Genie2, illustrating progress towards designing more natural proteins. That said, all the models, including ours, show a bias towards alpha helices. One way to mitigate this bias is through secondary structure conditioning. In our code, we further release a model that can be conditioned on a per-residue SS label. As we increase the Classifier Free Guidance strength towards beta sheets, this model generates proteins with more interesting SS structure, as summarized in Table 8. Preliminary designability/diversity evaluations show weaker performance of this model compared to the unconditional models of our main paper. Optimally balancing structural diversity and strong designability/diversity metrics remains a challenging open problem.

Guidance strength	$\alpha$ -Helix (%) ↓	$\beta$ -Sheet (%) ↑
0.2	82.1	1.5
0.4	61.8	11.4
0.6	30.3	33.6
0.7	21.7	36.1

Table 8: **Effect of guidance strength on secondary structure composition.** Increasing the strength reduces  $\alpha$ -Helix content and increases  $\beta$ -Sheet content, indicating a shift towards more natural outputs.

## C.3 Complete Tabular Results

This section presents the full numerical tables corresponding to result figures shown in the text. Specifically,

- Table 9 enumerates the results in Figure 1 and Figure 4.
- Table 10 enumerate the results in Figure 5.
- Table 11 enumerates partial results in Figure 7.
- Table 12 enumerates partial results in Figure 7.

Table 9: **Long protein generation performance.** Best values per residue length are highlighted in bold. Ambient Proteins results are shown for  $\gamma = 0.6$ . Found in main text in Figure 1.

Residue Length	Model	Designability (%)	Diversity (Clusters)
300	Ambient	90	89
	Proteina	<b>93</b>	55
	Proteus	<b>93</b>	23
	Improved Genie2	<b>93</b>	<b>93</b>
	Genie2	81	80
400	Ambient	<b>92</b>	<b>92</b>
	Proteina	85	60
	Proteus	83	35
	Improved Genie2	79	79
	Genie2	60	60
500	Ambient	<b>91</b>	<b>91</b>
	Proteina	82	65
	Proteus	69	30
	Improved Genie2	71	71
	Genie2	20	20
600	Ambient	<b>87</b>	<b>87</b>
	Proteina	81	55
	Proteus	67	20
	Improved Genie2	51	51
	Genie2	2	3
700	Ambient	<b>86</b>	<b>86</b>
	Proteina	68	45
	Proteus	47	10
	Improved Genie2	30	30
	Genie2	1	0
800	Ambient	<b>68</b>	<b>68</b>
	Proteina	55	47
	Proteus	17	5
	Improved Genie2	25	25
	Genie2	0	0

Table 10: **Designability-diversity trade-off for short protein generation.** Designability and diversity for short protein generation. Found in main text in Figure 5.

Model	Designability (%↑)	Diversity (↑)
<i>Ambient Proteins</i>		
$\gamma = 0.35$	<b>99.2</b>	0.615
$\gamma = 0.55$	98.6	0.781
$\gamma = 0.65$	93.0	<b>0.867</b>
<i>Baselines</i>		
Genie2	95.2	0.59
FoldFlow (base)	96.6	0.20
FoldFlow (stoc.)	97.0	0.25
FoldFlow (OT)	97.2	0.37
FrameFlow	88.6	0.53
RFDiffusion	94.4	0.46
Proteus	94.2	0.22
Proteina (FS $\gamma = 0.35$ )	98.2	0.49
Proteina (FS $\gamma = 0.45$ )	96.4	0.63
Proteina (FS $\gamma = 0.5$ )	91.4	0.71
Proteina (FS_no-tri $\gamma = 0.45$ )	93.8	0.62
Proteina (21M $\gamma = 0.3$ )	99.0	0.30
Proteina (21M $\gamma = 0.6$ )	84.6	0.59
Proteina (LoRA $\gamma = 0.5$ )	96.6	0.43

Table 11: **Performance on Single Motif Scaffolding Tasks** Ambient Protein Diffusion achieves superior results to Genie 2 and RFDiffusion and performs on par with Proteína. Crucially, our model achieves these results zero-shot, i.e., unlike Proteína, it is not optimized for motif scaffolding and still achieves comparable performance while being an order of magnitude smaller. Found in text in Figure 7.

Motif Name	Genie 2	RFDiffusion	Proteína	Ambient Protein Diffusion
<b>6E6R_long</b>	415	381	<b>713</b>	601
<b>6EXZ_long</b>	326	167	290	<b>432</b>
<b>6E6R_med</b>	272	151	<b>417</b>	406
<b>1YCR</b>	134	7	<b>249</b>	146
<b>5TRV_long</b>	97	23	<b>179</b>	119
<b>6EXZ_med</b>	54	25	43	<b>69</b>
<b>7MRX_128</b>	27	66	<b>51</b>	44
<b>6E6R_short</b>	26	23	<b>56</b>	27
<b>5TRV_med</b>	<b>23</b>	10	22	<b>23</b>
<b>7MRX_85</b>	23	13	<b>31</b>	17
<b>3IXT</b>	<b>14</b>	3	8	4
<b>5TPN</b>	8	5	4	<b>11</b>
<b>7MRX_60</b>	<b>5</b>	1	2	1
<b>1QJG</b>	<b>5</b>	1	3	<b>5</b>
<b>5TRV_short</b>	<b>3</b>	1	1	<b>3</b>
<b>5YUI</b>	3	1	<b>5</b>	4
<b>4ZYP</b>	3	6	<b>11</b>	3
<b>6EXZ_short</b>	2	1	<b>3</b>	<b>3</b>
<b>1PRW</b>	<b>1</b>	<b>1</b>	<b>1</b>	<b>1</b>
<b>5IUS</b>	<b>1</b>	<b>1</b>	<b>1</b>	<b>1</b>
<b>1BCF</b>	<b>1</b>	<b>1</b>	1	<b>1</b>
<b>5WN9</b>	1	0	<b>2</b>	1
<b>2KL8</b>	<b>1</b>	<b>1</b>	1	1
<b>4JHW</b>	0	0	0	0
<b>Total</b>	1445	889	<b>2094</b>	1923

Table 12: **Performance on Multi Motif Scaffolding Tasks** Ambient Protein Diffusion achieves consistently superior results to the predecessor Genie-2 model, despite using the same architecture, i.e. the benefit comes from better use of the data. The motif 2B5I is only solved by Ambient Protein Diffusion. Found in text in Figure 7.

Motif Name	Genie 2	Ambient Protein Diffusion
<b>3BIK+3BP5</b>	17	<b>23</b>
<b>1PRW_four</b>	11	<b>38</b>
<b>1PRW_two</b>	8	<b>15</b>
<b>4JHW+5WN9</b>	4	<b>12</b>
<b>2B5I</b>	0	<b>1</b>
<b>3NTN</b>	0	0
<b>Total</b>	40	<b>89</b>

Table 13: **Performance Comparison on Single Motif Scaffolding Tasks with Constant Midpoint Length**

Motif Name	Genie2	Ambient	RFDiffusion	Lengths
<b>6E6R_long</b>	406	<b>601</b>	208	108
<b>6EXZ_long</b>	296	<b>432</b>	158	110
<b>6E6R_med</b>	286	<b>406</b>	70	78
<b>1YCR</b>	78	<b>95</b>	3	70
<b>5TRV_long</b>	85	<b>119</b>	40	116
<b>6EXZ_med</b>	46	<b>69</b>	17	80
<b>7MRX_128</b>	28	<b>44</b>	52	128
<b>6E6R_short</b>	25	<b>27</b>	5	48
<b>5TRV_med</b>	<b>33</b>	23	9	86
<b>7MRX_85</b>	<b>24</b>	17	12	85
<b>3IXT</b>	<b>5</b>	1	1	62
<b>5TPN</b>	<b>2</b>	1	1	62
<b>7MRX_60</b>	<b>1</b>	<b>1</b>	<b>1</b>	60
<b>1QJG</b>	2	2	<b>103</b>	78
<b>5TRV_short</b>	2	1	<b>4</b>	56
<b>5YUI</b>	1	<b>2</b>	1	75
<b>4ZYP</b>	1	<b>2</b>	<b>2</b>	40
<b>6EXZ_short</b>	1	1	<b>2</b>	50
<b>1PRW</b>	<b>1</b>	<b>1</b>	1	82
<b>5IUS</b>	<b>1</b>	<b>1</b>	1	99
<b>1BCF</b>	<b>1</b>	<b>1</b>	1	124
<b>5WN9</b>	<b>1</b>	<b>1</b>	1	42
<b>2KL8</b>	<b>1</b>	<b>1</b>	1	79
<b>4JHW</b>	<b>0</b>	<b>0</b>	<b>0</b>	75
<b>Total</b>	1327	<b>1849</b>	694	

## D Evaluation Metrics

Evaluation of a protein generative model is challenging and there have been a few metrics that have been proposed. In what follows, we explain standard metrics in the protein-generative modeling literature that we will use in our Experimental Results section. Our experiments report using Proteína’s definitions of the metrics when possible.

### D.1 Designability

Designability (also referred to as refoldability) assesses the structural plausibility of generated proteins. Given a generated backbone, ProteinMPNN [19] generates eight plausible amino acid sequences for that backbone. ESMFold then folds each sequence and the resulting eight structures are compared to the original backbone. The self-consistency RMSD (scRMSD) is defined as the smallest root mean squared deviation between the generated backbone and each of the eight refolded structures. A backbone is considered *designable* if  $\text{scRMSD} < 2 \text{ \AA}$  and designability is defined as the percentage of generated backbones that meet this criterion.

### D.2 Diversity

Diversity quantifies the structural variability among the generated proteins. Designable backbones are clustered using Foldseek with a TM-score threshold of 0.5. Diversity is then defined as:

$$\text{Diversity} = \frac{\text{Number of Designable Clusters}}{\text{Number of Designable Samples}}.$$

This metric reflects the proportion of structurally distinct (i.e., non-redundant) designable backbones among all designable samples.

Table 14: Performance Comparison on Single Motif Scaffolding Tasks with 1.5x Length

Motif Name	Genie2	Ambient	RFDiffusion	Lengths
<b>6E6R_long</b>	439	<b>720</b>	243	162
<b>6EXZ_long</b>	526	<b>848</b>	193	165
<b>6E6R_med</b>	393	<b>633</b>	74	117
<b>1YCR</b>	321	<b>459</b>	25	105
<b>5TRV_long</b>	64	<b>102</b>	14	174
<b>6EXZ_med</b>	349	<b>490</b>	19	120
<b>7MRX_128</b>	14	<b>23</b>	<b>84</b>	192
<b>6E6R_short</b>	145	<b>215</b>	6	72
<b>5TRV_med</b>	82	<b>109</b>	5	129
<b>7MRX_85</b>	23	<b>42</b>	15	127
<b>3IXT</b>	<b>48</b>	27	1	93
<b>5TPN</b>	16	<b>40</b>	13	93
<b>7MRX_60</b>	9	<b>10</b>	2	90
<b>1QJG</b>	0	2	<b>323</b>	117
<b>5TRV_short</b>	<b>11</b>	10	2	84
<b>5YUI</b>	4	<b>9</b>	1	112
<b>4ZYP</b>	6	<b>13</b>	6	60
<b>6EXZ_short</b>	<b>31</b>	17	2	75
<b>1PRW</b>	<b>1</b>	<b>1</b>	<b>1</b>	123
<b>5IUS</b>	<b>1</b>	<b>1</b>	<b>1</b>	149
<b>1BCF</b>	1	<b>2</b>	1	186
<b>5WN9</b>	1	<b>2</b>	0	63
<b>2KL8</b>	<b>1</b>	<b>1</b>	<b>1</b>	118
<b>4JHW</b>	0	0	<b>1</b>	112
<b>Total</b>	2486	<b>3776</b>	1033	

Table 15: **PDB and AFDB TM-Noveltiy for short protein generation using FoldSeek-v9 and max qTM-score row.** We recompute all values using Geffner et al. [23] method. Numbers reported by Geffner et al. [23] are shown in parenthesis. These numbers are reported for backwards comparisons only and we strongly encourage the community to use the corrected TM-Noveltiy scores reported in the main text (see Appendix D.3).

Model	PDB Novelty (↓)	AFDB Novelty (↓)
<i>Ambient Proteins</i>		
$\gamma = 0.35$	<b>0.604</b>	<b>0.663</b>
$\gamma = 0.55$	0.606	0.671
$\gamma = 0.65$	0.608	0.673
<i>Baselines</i>		
Genie2	0.621 (0.63)	0.685 (0.69)
RFDiffusion	0.711 (0.71)	0.779 (0.77)
Proteus	0.741 (0.74)	0.766 (0.76)
Chroma	0.686 (0.69)	0.732 (0.74)
Proteina (FS $\gamma = 0.35$ )	0.727 (0.71)	0.783 (0.77)
Proteina (FS $\gamma = 0.45$ )	0.709 (0.69)	0.769 (0.75)
Proteina (FS $\gamma = 0.5$ )	0.698 (0.69)	0.760 (0.75)
Proteina (FS_no-tri $\gamma = 0.45$ )	0.702 (0.69)	0.759 (0.76)
Proteina (21M $\gamma = 0.3$ )	0.811 (0.81)	0.841 (0.84)
Proteina (21M $\gamma = 0.6$ )	0.761 (0.72)	0.797 (0.77)

### D.3 Novelty

**Metric definition.** Novelty is a metric that assesses the uniqueness of the generated backbones in comparison to existing structures in a database. We compute the novelty score with respect to both AFDB and PDB datasets following Geffner et al. [23]. To compute novelty, we measure the structural similarity of each designable protein to those in the dataset using FoldSeek’s easy-search command used by Proteina:

```
foldseek easy-search <path_sample> <database_path> <out_file> <tmp_path>
-alignment-type 1 -exhaustive-search -tmscore-threshold 0.0 -max-seqs
10000000000 -format-output query,target,alntmscore,lddt
```

For each designable backbone, we keep the max alntmscore value rather than the alntmscore value of the first row, which is the max qtm score value. The novelty of the dataset is the average of these maximum alntmscore values, representing how distinct our generated structures are from the proteins in the reference database (i.e., we do `df.groupby("query")["alntmscore"].max().mean()`). Perhaps counterintuitively, high novelty is not desired since it implies high similarity to the existing database.

**Bug in novelty computation.** Several of our evaluation metrics —TM-Diversity and TM-Novelty— depend on FoldSeek’s TM-score implementation. In Fall 2024, however, FoldSeek developers identified a bug in the alntmscore output (see Github issue 312 titled "alntmscore output is wrong" for details), which means that all previously reported TM-based metrics in the literature that did not use FoldSeek v10 (release 10-941cd33) are incorrect. Additionally, we found that Geffner et al. [23] mistakenly computed TM-novelty by taking the alnTM-Score from the row with the highest qTM-Score, rather than from the row with the highest alnTM-Score. This oversight arises because Foldseek’s easy-search command, by default, sorts its output in descending order by qTM-Score—irrespective of the requested output format.

To ensure accurate and comparable benchmarking with the literature, we recalculated TM-Novelty with the patched FoldSeek v10 (release 10-941cd33), explicitly selecting the maximum alnTM-Score for each query. For backward compatibility, we also reproduced literature results using the unpatched FoldSeek v9 (release 9-427df8a) and the default max qTM-Score row. Moving forward, we strongly recommend that the community adopt FoldSeek v10 and always sort using the alnTM-Score output to determine the maximum TM-Score per query and correctly compute TM-Novelty. Using both versions of FoldSeek, Ambient Protein Diffusion sets new state-of-the-art TM-novelty scores on both the PDB and AFDB (588K) benchmarks.

## E Full Training Algorithm and Implementation Details

### E.1 Additional Implementation Details

**Loss buffer.** The loss rescaling introduced in the main paper ensures balanced weighting across noise levels. At the same time, it also introduces a potential instability: the loss explodes as  $\sigma(t)$  approaches  $\sigma(t_i)$ . To mitigate this instability, we define a buffer zone around each protein’s assigned noise level. Specifically, given a protein’s assigned noise level  $t_i$ , it is only used during training at timesteps  $t + \tau$ , where  $\tau$  is a buffer hyperparameter that controls the exclusion margin. This constraint prevents the model from encountering degenerate gradient behavior near the rescaling boundaries and is only applied to medium and low confidence structures ( $p\text{LDDT} < 90$ ). We underline that is similar to how in normal diffusion there is a buffer time zone around  $t = 0$  that is never sampled.

**Ambient in high-noise regime.** As explained in the main paper, each protein is only used for a subset of diffusion times according to its average pLDDT value. The proteins that have super high PLDDT ( $> 90$ ) are considered clean data and can be used with the normal training objective. However, as found in [43], using the Ambient training objective for high-noise might be useful even if clean data is available. Intuitively, this objective prevents memorization and promotes diversity in the outputs. We ablated this design choice, and we found a slight increase in diversity for the same designability by using this. Hence, we used this tool from [43] for all our Ambient Protein Diffusion trainings.

## E.2 Algorithm

We provide the full algorithm in Algorithm 1. We commit to open-sourcing our code and models to facilitate the broader adoption of our method from the community.

---

### Algorithm 1 Ambient Protein Diffusion: Training Algorithm.

---

**Require:** untrained network  $h_\theta$ , dataset  $\mathcal{D} = \{(x_0^{(i)}, \text{pLDDT}^{(i)})\}_{i=1}^N$ , pLDDT to diffusion time mapping function  $f : [0, 100] \mapsto \mathbb{R}^+$ , noise scheduling  $\sigma(t)$ , batch size  $B$ , diffusion time  $T$ , buffer  $\tau$ .

1:  $\tilde{\mathcal{D}} \leftarrow \left\{ \left( x_0^{(i)} + f(\text{pLDDT}^{(i)})\epsilon^{(i)}, f(\text{pLDDT}^{(i)}) \right) \mid (x_0^{(i)}, \text{pLDDT}^{(i)}) \in \mathcal{D}, \epsilon^{(i)} \sim \mathcal{N}(0, I_d) \right\}$   $\triangleright$  Noise each point in the training set according to its pLDDT and get (noisy, noise level) pairs.

2: **while** not converged **do**

3:      $t_s^{(1)}, \dots, t_s^{(B)} \leftarrow$  Sample uniformly  $B$  times in  $[0, T]$   $\triangleright$  Sample diffusion times for this batch.

4:      $\tilde{\mathcal{D}}_p \leftarrow \text{shuffle}(\tilde{\mathcal{D}})$   $\triangleright$  Shuffle dataset.

5:     loss  $\leftarrow 0$   $\triangleright$  Initialize loss.

6:     pos  $\leftarrow 0$   $\triangleright$  Initialize index at shuffled dataset.

7:     **for**  $i \in [1, B]$  **do**

8:         **while** True **do**  $\triangleright$  find the first eligible point

9:              $y, t_y \leftarrow \tilde{\mathcal{D}}_p[\text{pos}]$

10:             **if**  $t_y \geq t_s^i + \tau$  **then**

11:                 break

12:             **else**

13:                 pos  $\leftarrow$  pos + 1  $\triangleright$  Move to the next point in the dataset.

14:             **end if**

15:     **end while**

16:      $\epsilon \sim \mathcal{N}(0, I)$   $\triangleright$  Sample noise.

17:      $t \leftarrow t_s^{(i)}$   $\triangleright$  Time to be used in this training update.

18:      $t_i \leftarrow t_y$   $\triangleright$  Assigned time based on the PLDDT value

19:      $x_{t_i} \leftarrow y$   $\triangleright$  Noised point to the assigned time.

20:      $x_t \leftarrow x_{t_i} + \sqrt{\sigma^2(t) - \sigma^2(t_i)}\epsilon$   $\triangleright$  Add additional noise.

21:      $\alpha(t, t_i) \leftarrow \frac{\sigma^2(t) - \sigma^2(t_i)}{\sigma^2(t)}$ .

22:      $w(t, t_i) \leftarrow \frac{\sigma^4(t)}{(\sigma^2(t) - \sigma^2(t_i))^2}$   $\triangleright$  Loss reweighting.

23:     loss  $\leftarrow$  loss +  $w(t, t_i) \|\alpha(t, t_i)h_\theta(x_t, t) + (1 - \alpha(t, t_i))x_t - x_{t_i}\|^2$   $\triangleright$  Ambient loss

24:     **end for**

25:     loss  $\leftarrow \frac{\text{loss}}{B}$   $\triangleright$  Compute average loss.

26:      $\theta \leftarrow \theta - \eta \nabla_\theta \text{loss}$   $\triangleright$  Update network parameters via backpropagation.

27: **end while**

---

## E.3 Model and Training Hyperparameters

### E.3.1 Hyperparameters for model optimized for long generation.

We train Ambient Protein Diffusion in 3 stages with increasingly longer proteins. In the first stage, we train on proteins from 50 to 256 residues for 200 epochs on our ambient clusters dataset using the representatives ( $\sim 196,000$  proteins). Since we increased the batch size to 384 items, we adopted a learning rate schedule to improve convergence [25]. We train with the AdamW optimizer with a maximal learning rate of  $1.0 \times 10^{-4}$ . During the second and third stage, we include additional cluster representatives of at most 512 and 712 residues, which scales our dataset to  $\sim 269,000$  and  $\sim 291,000$  proteins respectively. Training is performed on 48 GH200 GPUs and runs in 18, 48, and 48 hours for each stage respectively. We underline that the computational cost of training our model, while significant, is still relatively low compared to the Proteína’s estimated 14 days training on 128 A100 GPUs. This is due to the decreased size of our model (< 17M vs 200M) and training set ( $\sim 290K$  vs  $\sim 780K$ ).

Table 16 includes a more thorough list of the hyperparameters used for our experiments.

Table 16: **Hyperparameters of the diffusion protein model.** Dashes (-) indicate that the value is the same as the previous column. The Ambient walls correspond to the assigned diffusion times based on the protein’s pLDDT (times are from 1 to 1000). Proteins with  $p\text{LDDT} > 90$  are used everywhere. Proteins with  $p\text{LDDT} > 80$  are used for times in  $[600, 1000]$  and proteins with  $p\text{LDDT} > 70$  are used for times in  $[900, 1000]$ . We underline that these hyperparameters were not particularly optimized, and even more benefits might be observed by properly tuning these values.

Hyperparameter	Genie2	Ambient (Stage 1)	Stage 2	Stage 3
<b><i>Diffusion</i></b>				
Number of timesteps	1,000	-	-	-
Noise schedule	Cosine	-	-	-
Allowed times:	-	$\begin{cases} [1, 1000], p\text{LDDT} \geq 90 \\ [600, 1000], 90 > p\text{LDDT} \geq 80 \\ [900, 1000], 80 > p\text{LDDT} \geq 70 \end{cases}$	-	-
<b><i>Model Architecture</i></b>				
Single feature dimension	384	-	-	-
Pair feature dimension	128	-	-	-
Pair transform layers	5	8	8	8
Triangle dropout	0.25	-	-	-
Structure layers	8	-	-	-
<b><i>Training</i></b>				
Optimizer	AdamW	-	-	-
Number of training proteins	586k	196k	269k	291k
Number epochs	40	200	50	20
Warmup iterations	10,000	1,000	500	100
Total batch size	384	384	96	48
Learning rate	$1.0 \times 10^{-4}$	$1.0 \times 10^{-4}$	$1.0 \times 10^{-5}$	$1.0 \times 10^{-5}$
Weight decay	0.05	-	-	-
Minimum protein length	20	20	50	50
Maximum protein length	256	256	512	768
Minimum mean pLDDT	80	70	70	70
<b><i>Compute Resources</i></b>				
Number of GPUs	48	48	48	48
Training time	18 hr	18hr	48hr	48hr

### E.3.2 Hyperparameters for model optimized for short generation.

The Ambient Protein Diffusion model used in this experiment was trained on a dataset filtered with a TM-Align threshold of 0.4 (as opposed to 0.5), resulting in a training set of approximately 90K cluster representative proteins. While it is well known that protein pairs with TM-scores above 0.5 typically share the same fold, and those below 0.5 generally do not, we find that the trade-off between designability and diversity is sensitive to the underlying structural heterogeneity of the dataset. Notably, clustering with a TM-align threshold of 0.4, which corresponds to less than a 1% chance of shared global topology, slightly outperforms the 0.5 threshold, which reflects a  $\sim 38\%$  probability of topological similarity [54].

## E.4 Sampling

### E.4.1 Noise scale

In diffusion modeling, one designs a forward Ito corruption process:

$$dX_t = f(X_t, t)dt + g(t)dB_t, \quad (4)$$

defined by the drift function  $f(\cdot, \cdot)$  and the noise coefficient  $g(\cdot)$ . This process gets initialized at a distribution  $p_0$  and diffuses over time, defining smoother densities  $p_t$ . Due to a remarkable result by Anderson [3], sampling from  $p_0$  is achieved by running the reverse process:

$$dX_{T-t} = (-f(X_{T-t}, T-t) + g^2(T-t)\nabla \log p_{T-t}(X_{T-t}))dt + g(T-t)dB_{T-t}, \quad (5)$$

initialized at  $p_T$ .

However, in the context of protein generative models, it has been observed that sampling from a discretized version of the reverse process of Equation 6 does not lead to good performance as measured by the available metrics. Hence, it is common practice in the protein generative modeling literature to sample from a tilted measure using the process:

$$dX_{T-t} = (-f(X_{T-t}, T-t) + g^2(T-t)\nabla \log p_{T-t}(X_{T-t}))dt + g(T-t)\sqrt{\gamma}dB_{T-t}, \quad (6)$$

where the parameter  $\gamma$  controls the stochasticity added to the generation. Typically, this parameter is set to values  $\gamma < 1$  leading to more designable proteins at the expense of reduced diversity in the generated samples. The goal is often to optimally control the trade-off between designability and diversity, i.e. to be able to produce a wide range of structurally and functionally diverse proteins. Unless stated otherwise, for the experiments in this paper, we use  $\gamma = 0.6$  (as done in Genie2 [34]).

### E.4.2 Hyperparameters and sampling methods

For sampling, we follow the exact same parameters as Genie2. In particular, we run 1000 sampling steps using a simple first-order discretization of Equation (6). We underline that results could be further enhanced by using more advanced sampling techniques such as autoguidance [31] (used in Proteina [23]), higher-order samplers [30] and test-time scaling [38] methods.