GENERATING ALL-ATOM PROTEIN STRUCTURE FROM SEQUENCE-ONLY TRAINING DATA

Amy X. Lu[1,2]* **Wilson Yan**[1] **Sarah A. Robinson**[2] **Kevin K. Yang**[3] **Vladimir Gligorijevic**[2]
Kyunghyun Cho[2,4] **Richard Bonneau**[2] **Pieter Abbeel**[1] **Nathan Frey**[2]

[1]UC Berkeley [2]Prescient Design, Genentech [3]Microsoft Research [4]New York University

ABSTRACT

Generative models for protein design are gaining interest for their potential scientific impact. However, protein function is mediated by many modalities, and simultaneously generating multiple modalities remains a challenge. We propose **PLAID (Protein Latent Induced Diffusion)**, a method for multimodal protein generation that learns and samples from the *latent space of a predictor*, mapping from a more abundant data modality (e.g., sequence) to a less abundant one (e.g., crystallography structure). Specifically, we address the *all-atom* structure generation setting, which requires producing both the 3D structure and 1D sequence to define side-chain atom placements. Importantly, PLAID **only requires sequence inputs to obtain latent representations during training**, enabling the use of sequence databases for generative model training and augmenting the data distribution by 2 to 4 orders of magnitude compared to experimental structure databases. Sequence-only training also allows access to more annotations for conditioning generation. As a demonstration, we use compositional conditioning on 2,219 functions from Gene Ontology and 3,617 organisms across the tree of life. Despite not using structure inputs during training, generated samples exhibit strong structural quality and consistency. Function-conditioned generations learn side-chain residue identities and atomic positions at active sites, as well as hydrophobicity patterns of transmembrane proteins, while maintaining overall sequence diversity. Model weights and code are publicly available at `github.com/amyxlu/plaid`.

1 INTRODUCTION

Generative models of proteins promise to accelerate innovation in bioengineering by proposing designs that achieve novel functions. Many protein functions are mediated by their structure. This includes the identity, placement, and biophysical properties of both side-chain and backbone atoms, collectively referred to as the *all-atom structure*. However, to know which side-chain atoms to place, one must first know the *sequence*; all-atom structure generation thus can be seen as a multimodal problem requiring simultaneous generation of sequence and structure.

While generative modeling for protein structure has seen rapid recent progress, several important challenges still remain: **(1)** Existing protein structure and sequence generation methods often treat sequence and structure as *separate modalities*; structure-generation methods often produce only backbone atoms [1, 2, 3, 4]. **(2)** Methods that do address all-atom

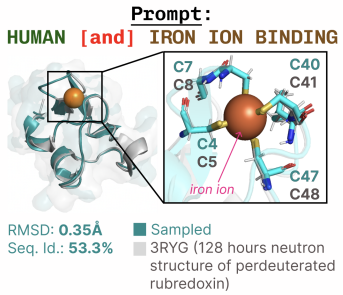

Figure 1: Function-conditioned samples can recapitulate sequence motifs and produce precise side-chain orientations at active sites, while maintaining low global sequence identity.

*Correspondence: `amyxlu@berkeley.edu`

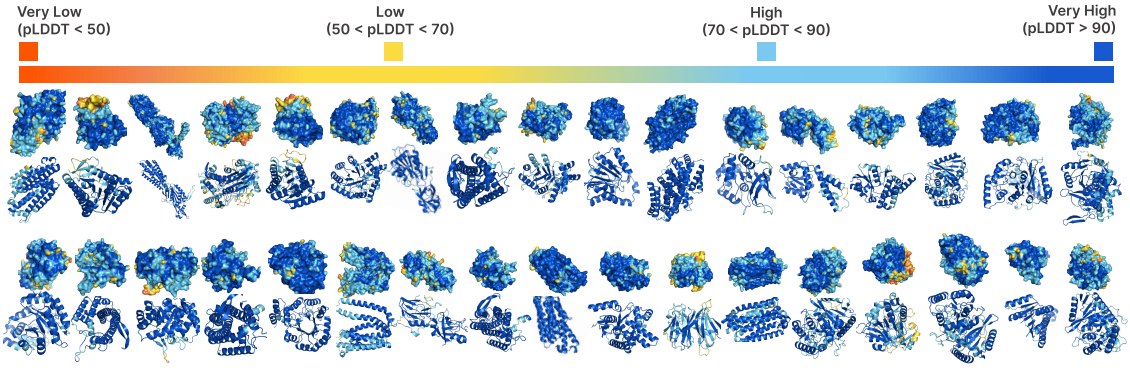

Figure 2: **PLAID unconditionally generates diverse, high-quality all-atom structures**, despite using only sequences for training the generative model.

design often require alternating between structure prediction and inverse-folding steps using an external model [5, 6]. **(3)** Evaluations often emphasize *in silico* oracle-based designability and structure-conditioning, with limited advancement in flexible controllability [1, 6]. **(4)** Methods that rely on experimentally-resolved structure databases [7] have a strong bias toward proteins that are crystallizable [1, 2, 8, 6, 5]. **(5)** Methods that ingest structure as inputs and/or rely on equivariance might face challenges in leveraging progress in hardware-aware neural network architectures for scalable training and inference [9, 10, 11].

Contributions To address these challenges, we introduce **PLAID (Protein Latent Induced Diffusion)**. Our principal demonstration is that multimodal generation can be achieved by learning the latent space of a predictor from a more abundant data modality (e.g., sequence) to a less abundant one (e.g., crystal structure). In particular, we focus on ESMFold [12] and all-atom structure generation, presenting a controllable diffusion model capable of **simultaneous sequence and all-atom protein structure generation while requiring only sequence inputs during training**.

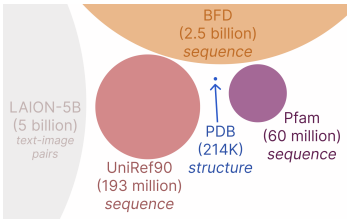

Figure 3: Size comparison of datasets drawn to scale. Sequence databases provide significantly more comprehensive coverage of the natural protein space than structural databases.

Because the training dataset can be defined by sequence databases rather than structural ones, this approach provides better coverage of the viable protein space traversed by evolution, enlarging experiment datasets available for training by 2 to 4 orders of magnitude (Figure 3). This also allows us to leverage structural information encoded in the *pretrained weights* rather than training data, and increases the availability of sequence annotations for controllable generation.

As a motivating demonstration, we examine compositional control across the axes of *function* and *organism*. Sequence databases also offer a wider range of annotation types, such as natural language abstracts. We show that PLAID can unconditionally generate diverse, high-quality samples (Figure 2). Function-conditioned samples can learn both the sequence motifs at active sites and the orientations of side-chains to be placed (Figure 1). We also demonstrate how motif scaffolding can be accomplished with this paradigm (Appendix Figure 17). The method is designed to be easily adaptable to expanding sequence datasets, leverage improved inference and training infrastructure for transformer-based models [13, 14, 11], and increasingly multimodal protein folding models, such as those that include nucleic acids and molecular ligand binding [15, 16].

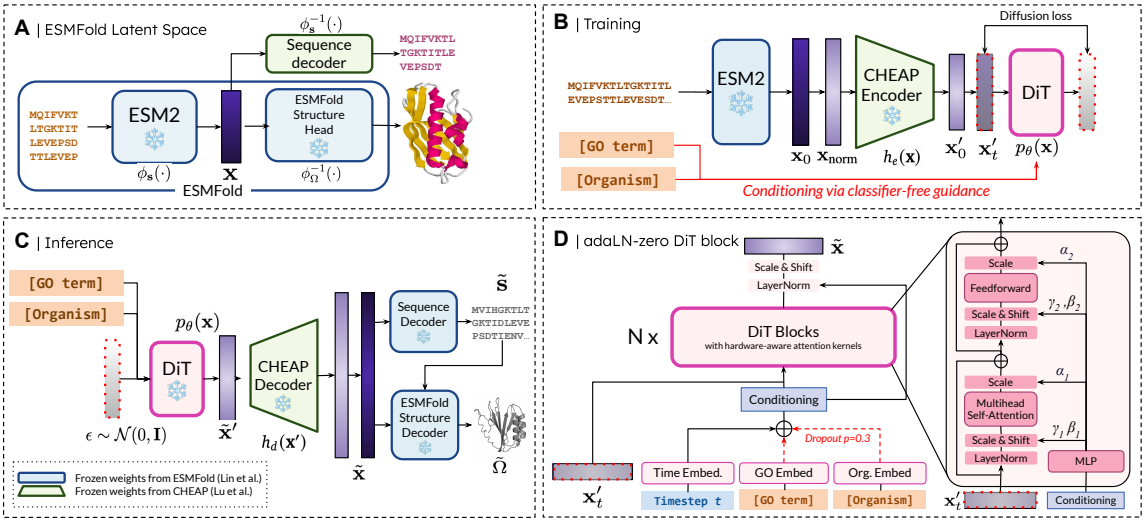

Figure 4: Overview of PLAID. **(A) ESMFold [12] latent space.** The latent space $p(\mathbf{x})$ represents a joint embedding of sequence and structure. **(B) Latent diffusion training.** Our goal is to learn and sample from $p_\theta(\mathbf{x})$, following the diffusion formulation [17]. To improve learning efficiency, we use the CHEAP [18] encoder $h_e(\cdot)$ to obtain a compressed embedding $\mathbf{x}' = h_e(\mathbf{x})$, so that the diffusion objective involves sampling from $p_\theta(h_e(\mathbf{x}))$. **(C) Inference.** To obtain both sequence and structure at inference time, we use the trained model to sample $\tilde{\mathbf{x}}' \sim p_\theta(\mathbf{x}')$, then uncompress using the CHEAP decoder to obtain $\tilde{\mathbf{x}} = h_d(\tilde{\mathbf{x}}')$. This embedding is decoded into the corresponding amino acid identities using a frozen sequence decoder trained in CHEAP [18]. The sequence of residue identities and $\tilde{\mathbf{x}}$ are used as input to the frozen structure decoder trained in ESMFold [12] to obtain the all-atom structure. **(D) DiT block architecture.** We use the Diffusion Transformer (DiT) [19] architecture with adaLN-zero DiT blocks to incorporate conditioning information. Classifier-free guidance is used to incorporate both the function (i.e., GO term) and organism class label embeddings [20]. Block architecture schematic adapted from Peebles and Xie [19].

2 Related Work

Generative Modeling for Proteins State-of-the-art diffusion models for designing protein structures have primarily focused on generating *novel backbone folds*, with controllability typically governed by secondary structure or used for scaffolding a known motif [1, 2, 4, 21]. Evaluation of these models focuses on fold stability and novelty, often involving oracle models [22, 12, 23, 24] for structure prediction and backbone-conditioned sequence design. However, to synthesize a protein, the sequence is required, and not all sampled structures have a corresponding sequence. To address this, "designability" is used as a metric to assess if a computationally-predicted sequence can be determined for folding into the proposed structure. Few mechanisms exist to enforce designability during training. Methods also exist for designing sequences [25, 26, 27, 28, 3], sometimes conditioned by the structure [29]. The structure can be constructed from these generations using a protein folding model, but the generative model itself does not produce atomic positions, increasing tool complexity and inference runtime.

Multimodal Sequence-Structure and All-Atom Generation All-atom generation can be framed as a multimodal problem, where the 1D protein sequence and 3D protein structure are jointly produced. Existing works [5, 6] often generate only one modality – structure or sequence – per diffusion step, relying on an external predictor to produce the other. Multiflow [8] enables co-generation without external tools, but does

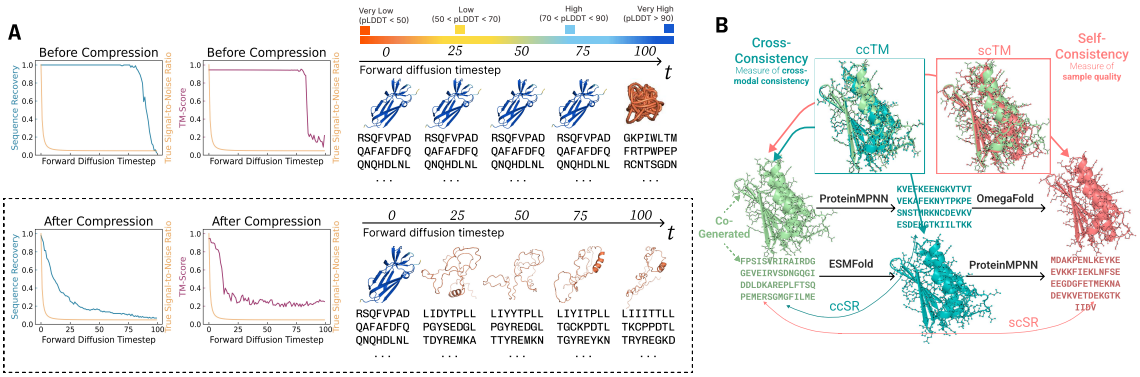

Figure 5: **(A)** When noise is added via a cosine schedule [35] (true signal-to-noise ratio (SNR) curve overlaid in orange) to the uncompressed latent space $\mathbf{x}$, the sequence and structure remain uncorrupted until the final forward diffusion timesteps, meaning most sampled timesteps are trivial for learning. After compression, noising in the compressed latent space $\mathbf{x}' = h_e(\mathbf{x})$ better aligns with the true SNR in the sequence and structure space, thereby improving the effectiveness of the diffusion task. **(B)** Schematic of **cross-consistency** and **self-consistency** metrics used to evaluate multimodal consistency and unimodal generation quality.

not produce side-chain atom positions. Some methods focus on specific protein subclasses, such as antibody design [30, 31]. While effective within these domains, antibodies represent a narrow subset of protein space, and such models often struggle to generalize across diverse protein families. Concurrently with this work, ESM3 [32] was developed to generate in a shared sequence-structure space and condition on InterPro [33], many of which are derived from GO terms [34]. However, the ESM3 tokenizer is trained on structure datasets rather than sequence databases, and cannot perform all-atom generation. PLAID can easily extend to the tokenized setting, as CHEAP [18] embeddings include a tokenized variant.

3 PLAID: Protein Latent Induced Diffusion

Notation A protein is composed of amino acids. A protein sequence $\mathbf{s} := \{r_i\}_{i=1}^{L}$ is often represented as a string of characters, with each character denoting the identity of an amino acid residue $r_i \in \mathcal{R}$, where $|\mathcal{R}| = 20$. Each unique residue r can be mapped to a set of atoms $\mathbf{r} := \{\mathbf{a}_j\}_{j=1}^{M_r}$, where each $\mathbf{a}_j \in \mathbb{R}^3$ is the 3D coordinate of an atom, and the number of atoms M_r may differ depending on the residue identity. A protein structure $\Omega := \{\mathbf{r}_i\}_{i=1}^{L}$ consists of all atoms in the protein and therefore implicitly contains $\mathbf{s}$.[1]

All-Atom Structure vs. Backbone-Only Structure The *all-atom structure* Ω requires knowledge of the amino acid identities at each position to specify the side-chain atoms. To reduce complexity, protein structure designers sometimes work with the backbone atoms $\Omega_{\text{backbone}} \subset \Omega$ only, which include only the N, C, and C_α atoms and are generally sufficient to define the protein fold.[2]

[1] In practice, to make use of array broadcasting, a standard M is selected for all residues, with an associated one-hot mask to specify which atoms are present for a given residue. We treat each structure as a matrix $\Omega \in \mathbb{R}^{L \times M \times 3}$. Following prior work [36, 12], we use the `atom14` representation where $M = 14$.

[2] Backbone-only structures induce $2(L-1)$ degrees of freedom arising from the ϕ and ψ angles (assuming that ω angles are held constant at $180°$). Depending on the residue identity, there may be 0 to 4 additional rotamer angles associated with the side chains. Therefore, *even when the sequence is known*, there may be up to $4L$ additional degrees of freedom necessary for all-atom structure prediction.

3.1 DEFINING p(SEQUENCE,STRUCTURE)

We begin with the motivation that sampling directly from $p(\mathbf{s}, \Omega)$ without implicitly factorizing it into $p(\Omega)p(\mathbf{s}|\Omega)$ (e.g., Protpardelle [5]) or $p(\mathbf{s})p(\Omega|\mathbf{s})$ (e.g., ProteinGenerator [6]) circumvents the difficulty in all-atom generation of not knowing which side-chain atoms to place; one can choose a latent manifold where residues do not need to be explicitly specified during iterative generation. Avoiding reliance on external prediction tools is computationally cheaper and avoids amplifying errors.

Our goal is to characterize a distribution $p(\mathbf{x})$ over $\mathcal{X}$ that encapsulates both sequence and structure information, such that there exist mappings $\mathbf{x} = \phi_{\mathbf{s}}(\mathbf{s})$ and $\mathbf{x} = \phi_{\Omega}(\Omega)$, where $\mathbf{x} \in \mathcal{X}$. In other words, both the sequence $\mathbf{s}$ and its corresponding structure Ω should map to the same latent embedding $\mathbf{x}$.

To achieve this, we follow the definition of a joint embedding of sequence and structure in Lu et al. [18]. By defining $\mathbf{x}$ as the latent space of a protein folding model $p(\Omega|\mathbf{s})$, we can establish these mappings. The trunk of the model provides $\mathbf{x} = \phi_{\text{ESM}}(\mathbf{s})$, and the structure module head provides $\Omega = \phi_{\text{SM}}(\mathbf{x})$. If we consider an implicit inverse function of the structure module such that $\mathbf{x} = \phi_{\text{SM}}^{-1}(\Omega)$, then we have:

$$\mathbf{x} = \phi_{\mathbf{s}}(\mathbf{s}) = \phi_{\Omega}(\Omega)$$

This satisfies our requirement for mappings that map both $\mathbf{s}$ and Ω to the same latent embedding $\mathbf{x}$.

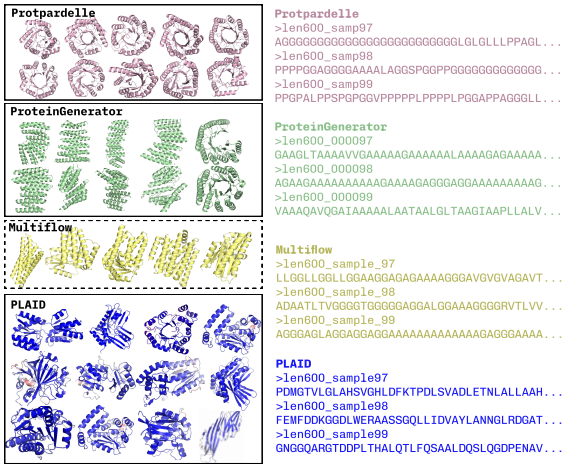

Figure 6: **Qualitative comparison of samples. (Left)** PLAID samples are structurally diverse, whereas baseline methods demonstrate mode collapse towards common structures such as TIM barrels and α-helix bundles at $L = 256$. **(Right)** Baseline methods generate sequences which have more repeats, especially at longer sequence lengths. Shown are sequences with $L = 600$, truncated for visualization.

Overview of ESMFold Briefly, ESMFold [12] has two main components: a protein language model component $\mathbf{x} = \phi_{\text{ESM}}(\mathbf{s})$ that captures evolutionary priors via the masked language modeling (MLM) loss, and a structure module component $\Omega = \phi_{\text{SM}}(\mathbf{x})$ that decodes these latent embeddings into a 3D structure. For the rest of this work, the "latent space of ESMFold" refers to the $\mathbf{x} \in \mathbb{R}^{L \times 1024}$ representation at the layer just prior to the structure module, where L is the length of a given protein. We choose this layer due to the observations in Lu et al. [18] (also see Section 3.2.1) that the pairwise input at inference time to the structure module is initialized to zeros, such that this sequence embedding contains all information necessary for structure prediction (Figure 4A and Appendix B).

3.2 SAMPLING ALL-ATOM STRUCTURE

Latent Generation Our goal is to learn $p_\theta(\mathbf{x}) \approx p(\mathbf{x})$, where θ is the set of parameters of the model learned through diffusion training (Figure 4C). Then, after training, we can sample $\tilde{\mathbf{x}} \sim p_\theta(\mathbf{x})$ (Figure 4C). To do so, we use diffusion models [17, 37] with some modifications (described in ablation Table 1). Classifier-free guidance is used to condition the model by GO term and organism classes, as further described in Section 3.3 and Appendix A.

Sequence Decoder To obtain the sequence, we need an implicit inverse mapping of ESM2 to get $\tilde{\mathbf{s}} = \phi_{\text{ESM}}^{-1}(\tilde{\mathbf{x}})$.[3] This mapping is easy to approximate by training a sequence decoder, since ESM2 was trained via the masked language modeling (MLM) loss. The sequence decoder ϕ_{ESM}^{-1} is trained and provided in Lu et al. [18], achieving a validation accuracy of 99.7% on a held-out partition of UniRef [38]. Note that $\tilde{\mathbf{s}}$ must be decoded first, as it determines the side-chain atoms to be placed in $\tilde{\Omega}$.

Structure Generation To obtain the structure from the sampled latent embedding, we use the frozen ESMFold structure module weights to compute $\tilde{\Omega} = \phi_{\text{SM}}(\tilde{\mathbf{x}}, \tilde{\mathbf{s}})$ (Figure 4C). Since the output of ϕ_{SM} is all-atom, the sampled $\tilde{\Omega}$ is also all-atom.

3.2.1 Latent Space Compression

In initial experiments, we found that directly learning $p_\theta(\mathbf{x})$ without compression performed poorly (results shown in Appendix Figure 12). We suspected that this might be due to the high dimensionality of $\mathbf{x} \in \mathbb{R}^{L \times 1024}$. For proteins with length $L = 512$ (the length cutoff used in this work), this corresponds to a high-resolution image synthesis problem similar to those encountered in image diffusion literature.[4] Therefore, we mirror works in this literature and perform diffusion in the latent space of an autoencoder, $\mathbf{x}' = h_e(\mathbf{x})$, such that the dimensions of $\mathbf{x}'$ are much smaller [41].

We use the CHEAP autoencoder [18], aiming to learn $p_\theta(\mathbf{x}') \approx p(\mathbf{x}')$, where $\mathbf{x}' = h_e(\mathbf{x})$. Noise is added to $\mathbf{x}'$ during the forward diffusion process and removed during denoising (Figure 4B). Based on results in Lu et al. [18], we choose a compressor that compresses from $1024 \to 32$ to balance the dimension of $\mathbf{x}'$ and reconstruction performance. We also downsample by $2\times$ along the length for better memory efficiency, allowing us to train up to longer sequences while maintaining parameter scalability. More information on CHEAP can be found in Lu et al. [18] and Appendix B.

At inference time, we begin by sampling the compressed latent variable $\tilde{\mathbf{x}}' \sim p_\theta(\mathbf{x}')$ and then decompress it to obtain $\tilde{\mathbf{x}} = h_d(\tilde{\mathbf{x}}')$. We then use frozen decoders to obtain the sequence $\tilde{\mathbf{s}} = \phi_{\text{ESM}}^{-1}(\tilde{\mathbf{x}})$ and the structure $\tilde{\Omega} = \phi_{\text{SM}}(\tilde{\mathbf{x}}, \tilde{\mathbf{s}})$. Figure 5A shows that when noise is added to the latent space, the sequence and structure remain unaltered until later timesteps. By adding noise in the compressed latent space, the resulting corruptions in sequence and structure space more closely match the true signal-to-noise ratio. Considering the importance of diffusion noise schedules for sample performance [42], we suspect that this factor contributes to the improved results observed when diffusing in the compressed latent space in our experiments.

3.3 Data and Training

Choice of Sequence Database The PLAID paradigm can be applied to any sequence database. As of 2024, sequence-only databases range in size from UniRef90 [38] (193 million sequences) to metagenomic datasets such as BFD [43] (2.5 billion sequences) and OMG [44] (3.3 billion sequences). We use Pfam because it provides more annotations for *in silico* evaluation and because protein domains are the primary units of structure-mediated functions. More information can be found in Appendix C.

[3] The embedding $\mathbf{x}$, defined just before the structure module, is actually a linearly projected version of the ESM2 embeddings. If we defined $\mathbf{x}$ as the ESM2 embeddings directly, we could use the decoder from ESM2's MLM training. However, since we use this modified embedding space, an approximation is necessary.

[4] While it may be possible to learn $p_\theta(\mathbf{x})$ without latent space compression, such as borrowing other techniques from high-resolution image synthesis such as cascaded diffusion [39] or specialized attention architectures [40], we found that compressing the latent space before diffusion training [41] was more effective than tuning diffusion hyperparameters in the original space.

Compositional Conditioning by Function and Organism Gene Ontology (GO) is a structured hierarchical vocabulary for annotating gene functions, biological processes, and cellular components across species [45, 46]. We examine all Pfam domains that have a Gene Ontology mapping, resulting in 2,219 GO terms compatible with our model. For domains with multiple associated GO term labels, the GO term that is least prevalent in our dataset is selected, to encourage the selected terms to be more specific.

We also examine all unique organisms in our dataset, identifying 3,617 organisms. Models are trained using classifier-free guidance [20]. The conditioning architecture is described in Figure 4D. More details can be found in Appendix A.

Architecture We use the Diffusion Transformer (DiT) [19] for the denoising task. This approach enables more flexible options for fine-tuning on mixed input modalities, as protein structure prediction models begin to incorporate complexes with nucleic acids and small-molecule ligands. It also better leverages transformer training infrastructure [47, 13, 48, 14, 9].

Table 1: Ablation results for metrics defined in Section 4.

	Configuration	ccTM	scTM	Ppl.	Seq. Div. %	Struct. Div.%
A	cosine noise sched. & pred. noise	0.54	0.55	16.97	**0.98**	0.86
B	A + v-diffusion	0.52	0.53	17.37	**0.98**	**0.89**
C	A + MinSNR	0.59	0.59	16.76	0.97	0.86
D	A +B + C + sigmoid noise sched.	0.56	0.58	16.88	0.92	0.86
E	D + self-conditioning	**0.70**	**0.65**	**15.38**	0.93	0.76
F	E + no cond drop	0.57	0.57	17.28	0.97	0.85

In early experiments, we found that allocating available memory to a larger DiT model was more beneficial than using triangular self-attention [22]. We train our models using the xFormers [47] implementation of [49], which provided a 55.8% speedup and a 15.6% reduction in GPU memory usage during our inference-time benchmarking experiments compared to a standard implementation using PyTorch primitives (see Appendix F). We train two versions of the model with 100 million and 2 billion parameters, respectively, both for 800,000 steps. More details are provided in Appendix A.

Diffusion Training and Inference-Time Sampling We use the discrete-time diffusion framework proposed by Ho et al. [17], employing 1,000 timesteps. To stabilize training and improve performance, we incorporate additional strategies: min-SNR reweighting [50], v-diffusion [51, 52], self-conditioning [53, 54], a sigmoid noise schedule [42], and exponential moving average (EMA) decay. Ablation results are shown in Table 1.

For sampling, unless otherwise noted, all results use the DDIM sampler [35, 37] with 500 timesteps. We use $c = 3$ as the conditioning strength for conditional generation; however, we find (Appendix Figure 16C) that sample quality is not strongly affected by this hyperparameter. We also find that DPM-Solvers [55] can achieve comparable results with $10\times$ fewer steps in scenarios where speed is a concern (see Appendix Figure 16), but in this work, we prioritize sample quality. More details on sampling methodology can be found in Appendix D, and Appendix F provides benchmarks on sampling speed.

4 EVALUATION

We outline metrics used to examine **unconditional generation** here. For evaluations that depend on hyperparameter settings, details are provided in Appendix E). For clarity, a schematic of cross- and self-consistency metrics is also shown in Figure 5B. For distributional conformity scores, each biophysical property is outlined in greater detail in Appendix E.1.

1. **Multimodal Cross-Consistency:** When the generated sequence is folded, does it match the generated structure? *[Cross-consistency TM-Score **(ccTM)**, cross-consistency RMSD **(ccRMSD)**.]* When the generated structure is inverse-folded into a sequence, does it match the generated sequence? *[Cross-consistency sequence recovery **(ccSR)**.]* What percentage of generated samples are designable? ***[ccRMSD < 2Å.]***

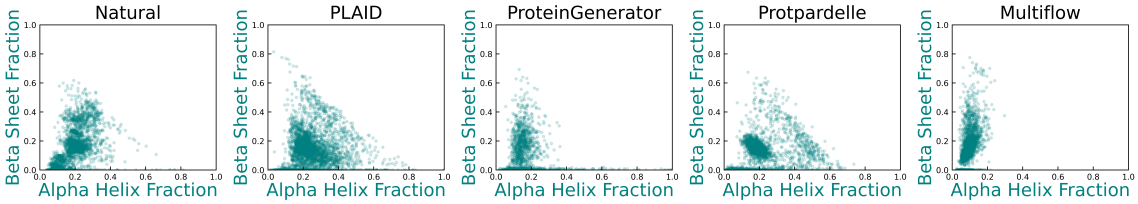

Figure 7: **Secondary structure composition analysis across protein generation methods.** Distribution of α-helix and β-sheet content in protein structures from natural proteins and different generation methods. Each point represents a single structure, with coordinates indicating the fraction of residues in α-helices (x-axis) and β-sheets (y-axis). Natural proteins show a characteristic distribution, which PLAID most closely approximates. In contrast, ProteinGenerator, Protpardelle, and Multiflow exhibit biases in their secondary structure distributions, with clustering in the high α-helix, low β-sheet region, especially regions where β-sheet fraction is zero.

2. **Unimodal Sample Quality**: What is the quality of generated sequence and structure when examined independently?
   (a) ***Structure.*** If we inverse-fold a generated structure into a sequence and fold the result with OmegaFold [56], is it consistent with the original? *[Self-consistency TM-Score **(scTM)**, self-consistency RMSD **(scRMSD)**.]*
   (b) ***Sequence.*** If we fold a generated sequence and inverse-fold the result, is it consistent with the original? *[Self-consistency sequence recovery **(scSR)**.]* Do generated sequences have low perplexity on next-token prediction models trained on natural proteins? *[Perplexity **(Ppl.)** under RITA XL [28].]*
3. **Distributional Conformity**: Do samples exhibit sensible biophysical parameters for real-world characterization? In other words, how similar are the distributions of biophysical properties between generated proteins and real proteins? Distributional similarity to natural proteins has been shown in Frey et al. [25] to be highly correlated with experimental expressibility. ***[Distributional conformity scores.]***
4. **Diversity**: After clustering by sequence using MMseqs [57] and by structure using Foldseek [58], how many distinct clusters can we observe? ***[# seq. clusters, # struct. clusters.]***
5. **Novelty**: Among designable samples, how similar are generated structures to their closest structural match to real proteins in PDB100 using `foldseek easy-search`? ***[Foldseek TMScore.]*** What about sequence identity to the closest `mmseqs easy-search` neighbor in UniRef90 [38] after pairwise alignment? ***[MMseqs seq id. %.]***

5 Experiments

5.1 Unconditional Generation

Following prior work demonstrating the effect of protein length on performance [1, 5, 8], we sample 64 proteins for each protein length between $\{64, 72, 80, ..., 512\}$, for a total of 3648 samples. Note that for completeness, we also compare against Multiflow [8] in (Figure 8 and Appendix Figure 15). However, *since Multiflow does not produce side chain positions*, it addresses a problem with fundamentally different complexity as PLAID, ProteinGenerator [6], and Protpardelle [5].

As shown in Figure 8, the degradation in PLAID's performance with increasing protein length is less pronounced than Multiflow and baseline all-atom methods. At longer lengths, PLAID better balances quality and diversity, possibly because the expanded dataset includes more samples of large proteins. Additionally, the flexible Diffusion Transformer [19] architecture and lengthwise downscaling [18] facilitates training on

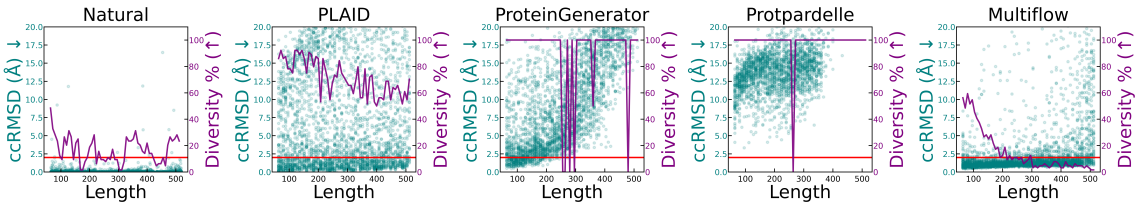

Figure 8: **Analysis of structural quality and diversity across protein lengths.** Comparison of natural proteins and different generation methods, showing structural quality (ccRMSD, teal points) and diversity (purple line, measured as ratio of unique structural clusters to total samples) for proteins of varying lengths (64-512 residues). The red line at 2Å indicates the designability threshold. While natural proteins and PLAID samples maintain consistent metrics across lengths, ProteinGenerator and Protpardelle show length-specific mode collapse, and Multiflow exhibits declining diversity at longer sequences. Analysis performed on 64 samples per length interval.

longer sequences. Baseline methods also exhibit pronounced mode collapse at specific lengths (Figure 8 and 6). Furthermore, secondary structure diversity is closer to the profile of natural proteins in PLAID versus baseline methods; Figure 7 shows that existing protein structure generation models often struggle to produce samples with high β-sheet content.

Table 2 compares the performance of different models across various consistency and quality metrics for all-atom protein generation, aggregated across all lengths. PLAID generates samples with high cross-modal consistency between generated sequences and structures, possibly due to sampling directly from $p(\mathrm{s}, \Omega)$ (Table 2). When examining unimodal quality, ProteinGenerator achieves the best self-consistency TM-score and pLDDT; it should be noted that ProteinGenerator uses RoseTTAFold [59] to produce structures at each step, and that the self-consistency TM-score for natural proteins is also imperfect. For sequence quality, all samples (including natural proteins) perform poorly on oracle-based metrics. PLAID comparably performs worse on scSR and perplexity under RITA XL [28]; however, as shown in Figure 6, some baseline sequence samples contain high levels of repeats, which sometimes lead to lower perplexity, despite being less biologically natural. Distributional conformity metrics (Table 3) offer an alternative for assessing sequence quality by examining biophysical patterns rather than token-level likelihoods.

Table 3 assesses diversity, novelty, and naturalness. We examine the diversity and quality trade-off by comparing the number of distinct designable sequence and structure clusters, where designability is defined as ccRMSD < 2Å. Among all-atom models, PLAID produces the highest number of distinct and designable samples in both sequence and structure space.

Table 2: Comparison of model performance across **consistency and quality metrics**. Bold values show best performance among all-atom generation models. pLDDT refers to the confidence score directly returned by the structure trunk of the generative model; for models which do not produce a pLDDT metric, N/A is used.

	Cross-Modal Consistency				Structure Quality			Sequence Quality	
	ccTM (↑)	ccRMSD (↓)	ccSR (↑)	% ccRMSD < 2Å (↑)	scTM (↑)	pLDDT (↑)	Beta sheet % (↑)	scSR (↑)	Ppl. (↓)
ProteinGenerator	0.58	11.86	**0.28**	0.08	**0.72**	**69.00**	0.04	0.40	**8.60**
Protpardelle	0.44	24.28	0.22	0.00	0.57	N/A	0.11	**0.44**	8.86
PLAID	**0.69**	**9.47**	0.26	**0.32**	0.64	59.46	**0.13**	0.27	14.61
Natural	*1.00*	*0.07*	*0.39*	*1.00*	*0.84*	84.51	0.13	0.39	7.40

Table 3: **Diversity, novelty, and distributional conformity** metrics across models. Bold values show best performance among all-atom generation models. Descriptions of each biophysical parameter for distributional conformity is described in Appendix E.1.

	Diversity			Novelty		Distributional Conformity (Wasserstein Distance)					
	# Des. (↑)	# Des. Seq. Clusts. (↑)	# Des. Struct. Clusts. (↑)	MMseqs Seq Id% (↓)	Foldseek TMScore (↓)	MW (↓)	Aroma-ticity (↓)	Dipeptide Instability Index (↓)	Iso-electricity (↓)	Hydro pathy (↓)	Charge at pH=7 (↓)
ProteinGenerator	309	309	309	0.57	**0.57**	9.54	0.07	14.55	1.42	0.31	6.12
Protpardelle	0	0	0	**0.56**	0.72	10.4	0.07	8.61	1.99	0.37	8.58
PLAID	**1171**	**809**	**522**	0.60	0.67	**0.62**	**0.01**	**1.98**	**0.49**	**0.28**	**2.71**
Natural	*3570*	*1362*	*600*	*0.81*	*0.87*	*0.0*	*0.0*	*0.0*	*0.0*	*0.0*	*0.0*

Novelty is measured by sequence similarity (sequence identity) and structural similarity (backbone TM-Score), with lower values indicating higher novelty. Although prior methods achieve the lowest sequence and structural similarities (i.e., higher novelty), this may be confounded by low-quality samples that artificially score high on novelty metrics.

The distribution of biophysical features for PLAID generations is closer to that of natural proteins, potentially due to the removal of biases toward structure in its training data. Across molecular weight (MW), ratio of aromatic amino acids, dipeptide-based instability index [60], hydropathy index (GRAVY) [61], isoelectric point, and charge at pH = 7, the distribution of PLAID samples is much closer to that of natural proteins than to those of baseline models. We consider distributional conformity to be an additional axis of real-world expressibility, as it has been shown in Frey et al. [25] to be heavily correlated with real-world expressibility. More information is in Appendix E.1.

5.2 Conditional Generation

Computational evaluation of function- and organism-conditioned generative models presents a conundrum: lower similarity is a favorable heuristic in machine learning, since it indicates that the generative model did not merely memorize the training data. From a bioinformatics perspective, however, conservation is key to function; taxonomic membership can be difficult to validate, given the high degree of similarity between homologs. In our experiments, we look for **high structural similarity to evaluate function conditioning** and **low sequence similarity to penalize exact memorization**.

Results shown in Figure 9 demonstrate that function-conditioned proteins possess known biological characteristics, such as conserved active site motifs and membrane hydrophobicity patterns. Despite high levels of conservation at catalytic sites, global sequence diversity is high, suggesting that the model has learned key biochemical features associated with the function prompt without direct memorization.

We further examine the Sinkhorn distance between function-conditioned generated latent embeddings and real proteins with this GO term annotation, randomly sampled from a *held-out* validation set that was unseen during training (Figure 10). This parallels the FID metric used in image generation, where one calculates the Fréchet Distance between the Inception embedding of a set of generated images and a set of random real images. Since we generate latent representations, we directly compare the distance between the sampled latent and heldout validation proteins in CHEAP [18] embedding space. We also use Sinkhorn Distance rather than Fréchet Distance to be more robust to smaller sample sizes, so that we can perform this analysis for GO term classes with fewer samples. It also assesses conditional generations independent of the sequence and structural decoders, and controls for memorization by comparing to a holdout dataset unseen during training. For comparison, the Sinkhorn distance between random real proteins from the validation set and the

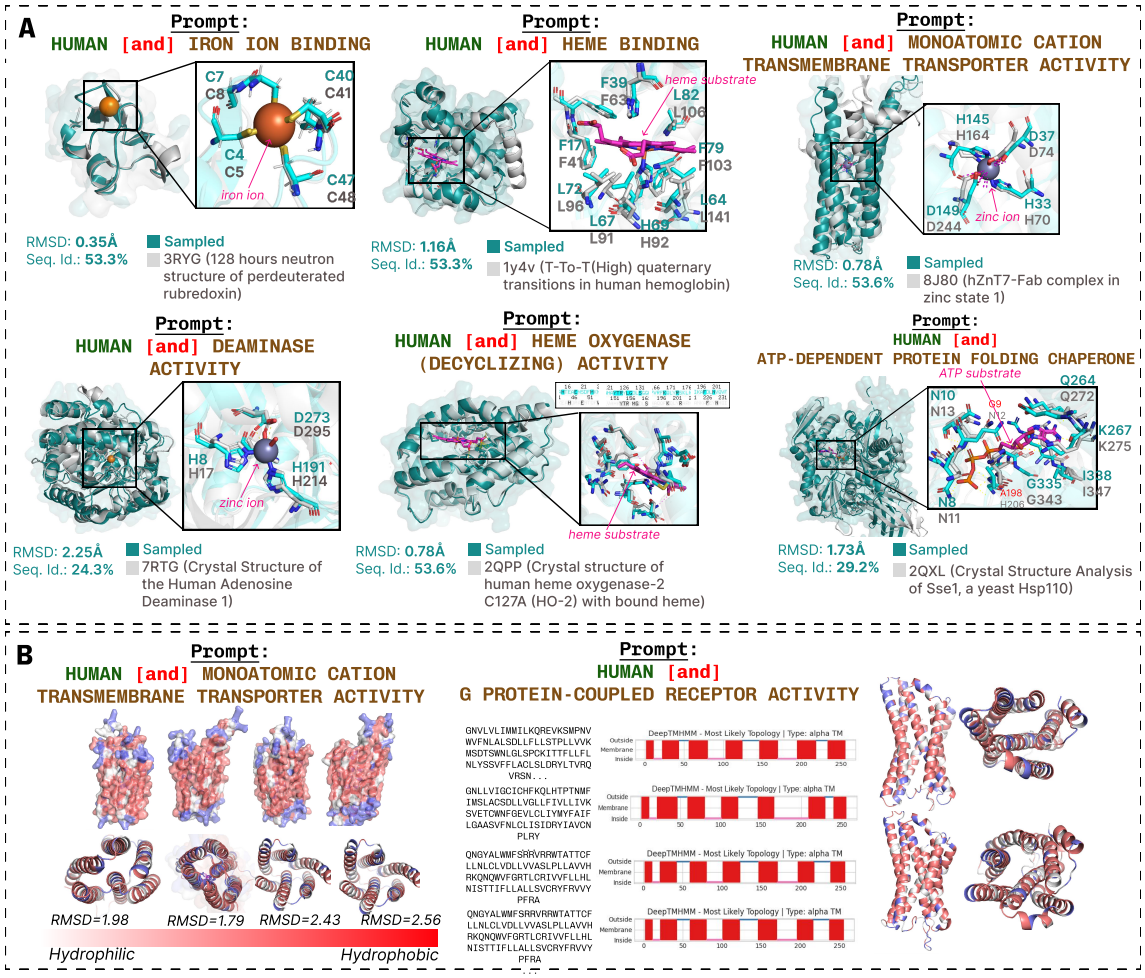

Figure 9: **PLAID enables function-guided protein generation while preserving critical structural motifs.** Additional examples provided in Appendix 13. **(A) Generated proteins capture sequence motifs and approximate side-chain orientations at active sites despite low sequence identity.** Each panel shows a PLAID-generated structure aligned with its closest PDB structural neighbor (identified via Foldseek) containing a bound ligand or substrate. RMSD and sequence similarity metrics are calculated globally. **(B) Generated membrane proteins exhibit biophysically realistic properties.** *(Left)* Generated transmembrane proteins display appropriate spatial distribution of hydrophobic and hydrophilic residues, with hydrophobic residues concentrated in membrane-spanning regions. Multiple independent samples demonstrate consistent recapitulation of these physical constraints. *(Right)* Generated G protein-coupled receptors (GPCRs) show the characteristic seven transmembrane helix architecture. DeepTMMHMM topology predictions confirm the expected transmembrane organization.

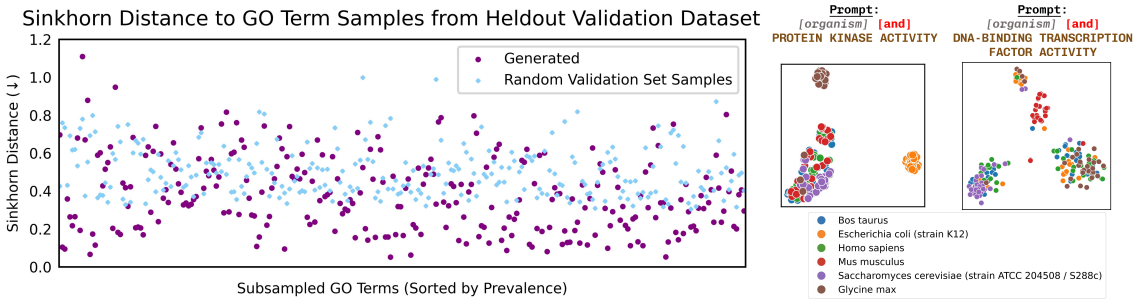

Figure 10: **(Left)** For each unique GO term in the validation set, we examine the Sinkhorn distance between generated samples and real proteins in the heldout subset in this class. For reference, we also calculate the Sinkhorn distance between random real proteins to the heldout subset. **(Right)** t-SNE reduction of generated embeddings, colored by the organism used for conditioning.

function-conditioned generations is also evaluated. Conditional generations generally have lower Sinkhorn distances to validation proteins with the same annotation than random samples, suggesting that the desired latent information is captured in the embedding. In Appendix Figure 16, we consider how conditioning scale might affect sample quality and possible GO term characteristics that might be influencing the difference in Sinkhorn distance between function-conditioned generations and random proteins.

To probe into organism conditioning abilities, Figure 10 shows t-SNE plots of generated embeddings colored by organism. Organisms that are more distantly related phylogenetically, such as *Glycine max* (i.e., soybean) and *E. coli*, form more distinct clusters than those more closely related evolutionarily, such as human and mouse. This suggests that function- and organism-conditioned samples have been imbued with desired characteristics. This embedding-level analysis provides an early investigation into organism-specific conditioning abilities. We observe, for human organism conditioned samples in Figure 9, that for 66.5% of samples, the closest `mmseqs easy-search` neighbor also come from *Homo sapiens*. We leave it as future work to do a more in-depth analysis of organism conditioning, since nearest-neighbor taxonomic analyses are very sensitive to search settings, and favors overrepresented species in the database. Moreover, assessing taxonomic origins for structures is complicated by the high structural conservation across kingdoms. A suitable follow up for this analysis could be for specialized organism-conditioning use cases, such as humanization or expression system specific protein characterization.

PLAID is fully compatible with motif scaffolding, and with binder design when trained on sequence complexes. Appendix Figure 17 demonstrates how motif scaffolding can be used with PLAID, by holding parts of the motif constant at each reverse diffusion step during latent generation. We focus on enabling *new* capabilities in all-atom generation in this work and leave further exploration of this capability as future work.

6 Discussion

We propose PLAID, a paradigm for multi-modal, controllable generation of proteins by diffusing in the latent space of a prediction model that maps single sequences to the desired modality. Our method is designed to leverage progress in **data availability, model scalability, and sequence-to-structure prediction capabilities**. To this end, we chose an implementation that makes use of fast attention kernels [47] for transformer-based architectures, and used GO terms as a proxy for the vast quantities of language annotation that are paired with sequence databases.

It is straightforward to expand PLAID to many downstream capabilities. Although we examine ESMFold [12] in this work, the method can be applied to any prediction model. There is rapid progress [16, 15, 62, 63, 64] in predicting complexes from sequence, and diffusing in the latent space of such models would allow using the frozen decoder to obtain more modalities than just all-atom structure. While we show a demonstration that motif scaffolding is possible in Appendix Figure 17, this capability can be greatly extended, including to binders and complexes. We use Pfam and GO terms as a proof-of-concept and focus on enabling new capabilities, though more "traditional" in-painting style tasks can also be used.

A limitation of PLAID is that performance can be bottlenecked by the prediction model from which the frozen decoders are derived. Here, we rely on the optimism that such models will continue to improve. With explicit fine-tuning for latent generation (e.g., training CHEAP and the structure decoder end-to-end), model performance can likely be improved. Furthermore, since the current structure decoder is deterministic, it does not sample alternative conformations. A solution is to use a decoder that returns a distribution over structural conformations instead; such a model might naturally be developed with progress in the field, or be explicitly fine-tuned. Additionally, the GO term one-hot encoding used here does not take into account the hierarchical nature of the Gene Ontology vocabulary, nor that a protein might have several relevant GO terms. This can be fixed by using a multi-class conditioning scheme instead. Finally, the classifier-free guidance scale can be separated for the organism and function conditions, since the two may require different guidance strengths in real-world use cases. These limitations will be examined in future work.

ACKNOWLEDGEMENTS

The authors thank Rob Alberstein, Sidney Lisanza, Pedro O. Pinheiro, Matthieu Kirchmeyer, Sai Pooja Mahajan, Simon Kelow, Andy Watkins, Dave Epstein, Ajay Jain, and other members of Prescient Design and BAIR for insightful discussions. We also thank Justin Wong, Hanlun Jiang, and Garyk Brixi for helpful feedback on the manuscript, anonymous reviewers for suggestions, and Alex Chu for clarifications on Protpardelle experiments. Experiments were completed in part on donated resources from NVIDIA to the Berkeley Artificial Intelligence Research (BAIR) Lab. AXL is funded in part by the NSERC PGS-D award. PA holds concurrent appointments as a Professor at UC Berkeley and as an Amazon Scholar; this paper describes work performed at UC Berkeley and is not associated with Amazon.

REFERENCES

[1] 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*, 620:1089–1100, 2023.

[2] 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.

[3] Sarah Alamdari, Nitya Thakkar, Rianne van den Berg, Alex Xijie Lu, Nicolo Fusi, Ava Pardis Amini, and Kevin K Yang. Protein generation with evolutionary diffusion: sequence is all you need. *bioRxiv*, pages 2023–09, 2023.

[4] Kevin E. Wu, Kevin K. Yang, Rianne van den Berg, James Y. Zou, Alex X. Lu, and Ava P. Amini. Protein structure generation via folding diffusion. *arXiv*, 2209.15611, 2022.

[5] Alexander E Chu, Lucy Cheng, Gina El Nesr, Minkai Xu, and Po-Ssu Huang. An all-atom protein generative model. *bioRxiv*, 2023.

[6] Sidney Lyayuga Lisanza, Jacob Merle Gershon, Sam Wayne Kenmore Tipps, Lucas Arnoldt, Samuel Hendel, Jeremiah Nelson Sims, Xinting Li, and David Baker. Joint generation of protein sequence and structure with RoseTTAFold sequence space diffusion. *bioRxiv*, 2023.

[7] Helen M Berman, John Westbrook, Zukang Feng, Gary Gilliland, Talapady N Bhat, Helge Weissig, Ilya N Shindyalov, and Philip E Bourne. The protein data bank. *Nucleic acids research*, 28(1):235–242, 2000.

[8] Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. *arXiv preprint arXiv:2402.04997*, 2024.

[9] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. *Advances in Neural Information Processing Systems*, 35: 16344–16359, 2022.

[10] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International conference on machine learning*, pages 8821–8831. Pmlr, 2021.

[11] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

[12] 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.

[13] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

[14] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE, 2020.

[15] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J Ballard, Joshua Bambrick, et al. Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature*, pages 1–3, 2024.

[16] Rohith Krishna, Jue Wang, Woody Ahern, Pascal Sturmfels, Preetham Venkatesh, Indrek Kalvet, Gyu Rie Lee, Felix S Morey-Burrows, Ivan Anishchenko, Ian R Humphreys, et al. Generalized biomolecular modeling and design with RoseTTAFold all-atom. *Science*, 384(6693):eadl2528, 2024.

[17] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[18] Amy X Lu, Wilson Yan, Kevin K Yang, Vladimir Gligorijevic, Kyunghyun Cho, Pieter Abbeel, Richard Bonneau, and Nathan Frey. Tokenized and continuous embedding compressions of protein sequence and structure. *bioRxiv*, pages 2024–08, 2024.

[19] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

[20] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[21] Yeqing Lin and Mohammed AlQuraishi. Generating novel, designable, and diverse protein structures by equivariantly diffusing oriented residue clouds. *Proceedings of the 40th International Conference on Machine Learning*, 2023.

[22] 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.

[23] 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*, 378:49–56, 2022.

[24] Chloe Hsu, Robert Verkuil, Jason Liu, Zeming Lin, Brian Hie, Tom Sercu, Adam Lerer, and Alexander Rives. Learning inverse folding from millions of predicted structures. *Proceedings of the 39th International Conference on Machine Learning*, 162:8946–8970, 2022.

[25] Nathan C Frey, Daniel Berenberg, Karina Zadorozhny, Joseph Kleinhenz, Julien Lafrance-Vanasse, Isidro Hotzel, Yan Wu, Stephen Ra, Richard Bonneau, Kyunghyun Cho, et al. Protein discovery with discrete walk-jump sampling. *arXiv preprint arXiv:2306.12360*, 2023.

[26] Noelia Ferruz, Steffen Schmidt, and Birte Höcker. ProtGPT2 is a deep unsupervised language model for protein design. *Nature Communications*, 13:4348, 2022.

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

[28] Daniel Hesslow, Niccoló Zanichelli, Pascal Notin, Iacopo Poli, and Debora Marks. RITA: a study on scaling up generative protein sequence models. *arXiv preprint arXiv:2205.05789*, 2022.

[29] Ivan Anishchenko, Samuel J Pellock, Tamuka M Chidyausiku, Theresa A Ramelot, Sergey Ovchinnikov, Jingzhou Hao, Khushboo Bafna, Christoffer Norn, Alex Kang, Asim K Bera, et al. De novo protein design by deep network hallucination. *Nature*, 600(7889):547–552, 2021.

[30] Karolis Martinkus, Jan Ludwiczak, Wei-Ching Liang, Julien Lafrance-Vanasse, Isidro Hotzel, Arvind Rajpal, Yan Wu, Kyunghyun Cho, Richard Bonneau, Vladimir Gligorijevic, et al. AbDiffuser: full-atom generation of in-vitro functioning antibodies. *Advances in Neural Information Processing Systems*, 36, 2024.

[31] Shitong Luo, Yufeng Su, Xingang Peng, Sheng Wang, Jian Peng, and Jianzhu Ma. Antigen-specific antibody design and optimization with diffusion-based generative models for protein structures. *Advances in Neural Information Processing Systems*, 35:9754–9767, 2022.

[32] Tomas Hayes, Roshan Rao, Halil Akin, Nicholas J Sofroniew, Deniz Oktay, Zeming Lin, Robert Verkuil, Vincent Q Tran, Jonathan Deaton, Marius Wiggert, et al. Simulating 500 million years of evolution with a language model. *bioRxiv*, pages 2024–07, 2024.

[33] Typhaine Paysan-Lafosse, Matthias Blum, Sara Chuguransky, Tiago Grego, Beatriz Lázaro Pinto, Gustavo A Salazar, Maxwell L Bileschi, Peer Bork, Alan Bridge, Lucy Colwell, et al. InterPro in 2022. *Nucleic acids research*, 51(D1):D418–D427, 2023.

[34] Gene Ontology Consortium. The gene ontology resource: 20 years and still GOing strong. *Nucleic Acids Research*, 47(D1):D330–D338, 2019.

[35] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

[36] Gustaf Ahdritz, Nazim Bouatta, Christina Floristean, Sachin Kadyan, Qinghui Xia, William Gerecke, Timothy J O'Donnell, Daniel Berenberg, Ian Fisk, Niccolò Zanichelli, et al. OpenFold: Retraining AlphaFold2 yields new insights into its learning mechanisms and capacity for generalization. *Nature Methods*, pages 1–11, 2024.

[37] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[38] Baris E Suzek, Yuqi Wang, Hongzhan Huang, Peter B McGarvey, Cathy H Wu, and UniProt Consortium. UniRef clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31(6):926–932, 2015.

[39] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *Journal of Machine Learning Research*, 23(47):1–33, 2022.

[40] Katherine Crowson, Stefan Andreas Baumann, Alex Birch, Tanishq Mathew Abraham, Daniel Z Kaplan, and Enrico Shippole. Scalable high-resolution pixel-space image synthesis with hourglass diffusion transformers. In *Forty-first International Conference on Machine Learning*, 2024.

[41] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[42] Ting Chen. On the importance of noise scheduling for diffusion models. *arXiv preprint arXiv:2301.10972*, 2023.

[43] Martin Steinegger, Milot Mirdita, and Johannes Söding. Protein-level assembly increases protein sequence recovery from metagenomic samples manyfold. *Nature methods*, 16(7):603–606, 2019.

[44] Andre Cornman, Jacob West-Roberts, Antonio Pedro Camargo, Simon Roux, Martin Beracochea, Milot Mirdita, Sergey Ovchinnikov, and Yunha Hwang. The omg dataset: An open metagenomic corpus for mixed-modality genomic language modeling. *bioRxiv*, pages 2024–08, 2024.

[45] TGO Consortium, Suzi A Aleksander, James Balhoff, Seth Carbon, JM Cherry, Harold J Drabkin, Dustin Ebert, Marc Feuermann, Pascale Gaudet, and Nomi L Harris. The gene ontology knowledgebase in 2023. *Genetics*, 224(1):iyad031, 2023.

[46] Michael Ashburner, Catherine A Ball, Judith A Blake, David Botstein, Heather Butler, J Michael Cherry, Allan P Davis, Kara Dolinski, Selina S Dwight, Janan T Eppig, et al. Gene ontology: tool for the unification of biology. *Nature genetics*, 25(1):25–29, 2000.

[47] Benjamin Lefaudeux, Francisco Massa, Diana Liskovich, Wenhan Xiong, Vittorio Caggiano, Sean Naren, Min Xu, Jieru Hu, Marta Tintore, Susan Zhang, Patrick Labatut, Daniel Haziza, Luca Wehrstedt, Jeremy Reizenstein, and Grigory Sizov. xFormers: A modular and hackable Transformer modelling library. `https://github.com/facebookresearch/xformers`, 2022.

[48] Reza Yazdani Aminabadi, Samyam Rajbhandari, Ammar Ahmad Awan, Cheng Li, Du Li, Elton Zheng, Olatunji Ruwase, Shaden Smith, Minjia Zhang, Jeff Rasley, et al. Deepspeed-inference: enabling efficient inference of transformer models at unprecedented scale. In *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–15. IEEE, 2022.

[49] Markus N Rabe and Charles Staats. Self-attention does not need O(n2) memory. *arXiv preprint arXiv:2112.05682*, 2021.

[50] Tiankai Hang, Shuyang Gu, Chen Li, Jianmin Bao, Dong Chen, Han Hu, Xin Geng, and Baining Guo. Efficient diffusion training via Min-SNR weighting strategy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7441–7451, 2023.

[51] Shanchuan Lin, Bingchen Liu, Jiashi Li, and Xiao Yang. Common diffusion noise schedules and sample steps are flawed. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 5404–5411, 2024.

[52] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

[53] Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*, 2022.

[54] Allan Jabri, David Fleet, and Ting Chen. Scalable adaptive computation for iterative generation. *arXiv preprint arXiv:2212.11972*, 2022.

[55] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022.

[56] Ruidong Wu, Fan Ding, Rui Wang, Rui Shen, Xiwen Zhang, Shitong Luo, Chenpeng Su, Zuofan Wu, Qi Xie, Bonnie Berger, et al. High-resolution de novo structure prediction from primary sequence. *bioRxiv*, 2022.

[57] Martin Steinegger and Johannes Söding. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature biotechnology*, 35(11):1026–1028, 2017.

[58] 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.

[59] Minkyung Baek, Frank DiMaio, Ivan Anishchenko, Justas Dauparas, Sergey Ovchinnikov, Gyu Rie Lee, Jue Wang, Qian Cong, Lisa N Kinch, R Dustin Schaeffer, et al. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373(6557):871–876, 2021.

[60] Kunchur Guruprasad, BV Bhasker Reddy, and Madhusudan W Pandit. Correlation between stability of a protein and its dipeptide composition: a novel approach for predicting in vivo stability of a protein from its primary sequence. *Protein Engineering, Design and Selection*, 4(2):155–161, 1990.

[61] Jack Kyte and Russell F Doolittle. A simple method for displaying the hydropathic character of a protein. *Journal of molecular biology*, 157(1):105–132, 1982.

[62] Chai Discovery, Jacques Boitreaud, Jack Dent, Matthew McPartlon, Joshua Meier, Vinicius Reis, Alex Rogozhnikov, and Kevin Wu. Chai-1: Decoding the molecular interactions of life. *bioRxiv*, pages 2024–10, 2024.

[63] Lihang Liu, Shanzhuo Zhang, Yang Xue, Xianbin Ye, Kunrui Zhu, Yuxin Li, Yang Liu, Wenlai Zhao, Hongkun Yu, Zhihua Wu, et al. Technical report of HelixFold3 for biomolecular structure prediction. *arXiv preprint arXiv:2408.16975*, 2024.

[64] Jeremy Wohlwend, Gabriele Corso, Saro Passaro, Mateo Reveiz, Ken Leidal, Wojtek Swiderski, Tally Portnoi, Itamar Chinn, Jacob Silterra, Tommi Jaakkola, et al. Boltz-1: Democratizing biomolecular interaction modeling. *bioRxiv*, pages 2024–11, 2024.

[65] Piotr Nawrot, Szymon Tworkowski, Michał Tyrolski, Łukasz Kaiser, Yuhuai Wu, Christian Szegedy, and Henryk Michalewski. Hierarchical transformers are more efficient language models. *arXiv preprint arXiv:2110.13711*, 2021.

[66] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022.

[67] JR Lobry and Christian Gautier. Hydrophobicity, expressivity and aromaticity are the major trends of amino-acid usage in 999 escherichia coli chromosome-encoded genes. *Nucleic acids research*, 22(15): 3174–3180, 1994.

APPENDIX

A ADDITIONAL TRAINING DETAILS

We train two variants of the model: a 2B version and a 100M version, both with the memory-efficient attention implementation in xFormers, using float32 precision. A learning rate of 1e-4 was used, with cosine annealing applied over 800,000 steps. The xFormers memory-efficient attention kernel requires input lengths to be a

multiple of 4. Since we also apply an upsampling factor of 2, the actual inference length must be a multiple of 4. During training, the maximum sequence length we use is 512, based on the distribution of sequences in Pfam and a shortening factor of 2 based on results in Lu et al. [18].

Following Ho and Salimans [20], with $p_{\text{uncond}} = 0.3$, the class label is replaced with the $\varnothing$ unconditional token. This is sampled separately for both function and organism. Note that not all data samples will have an associated GO term; we use the $\varnothing$ token for those cases as well. At inference time, to generate unconditionally (for either or both of function and/or organism), we use the $\varnothing$ token for conditioning.

B CHEAP Compression Details

Briefly, the CHEAP encoder and decoder use an Hourglass Transformer [65] architecture that downsamples lengthwise, as well as downprojects the channel dimension, to create a bottleneck layer, the output of which is our compressed embedding. The entire model is trained with the reconstruction loss $MSE(\mathbf{x}, \hat{\mathbf{x}})$. Results in Lu et al. [18] show that structural and sequence information in ESMFold latent spaces are in fact highly compressible, and despite using very small bottleneck dimensions, reconstruction performance can nonetheless be maintained when evaluated in sequence or structure space.

Based on reconstruction results in Lu et al. [18], we choose $\mathbf{x}' \in \mathbb{R}^{\frac{L}{2} \times 32}$ with $L = 512$, which balances reconstruction quality at a resolution comparable to the size of latent spaces in image diffusion models [41]. Dividing the length in half allows us to better leverage the scalability and performance of Transformers, while managing their $\mathcal{O}(L)$ memory needs.

The CHEAP module involves a channel normalization step prior to the forward pass through the autoencoder. We find that the distribution of embedding values is fairly "smooth" here (Figure 11). Though the original Rombach et al. [41] paper was trained with a KL constraint to a Gaussian distribution, we use the embedding output as is. CHEAP embeddings were also trained with a $\tanh$ layer at the output of the bottleneck; this allows us to clip our samples between $[-1, 1]$ at each diffusion iteration, as was done in original image diffusion works [17, 20, 35, 66]. We found in early experiments that being able to clip the output values was very helpful for improving performance. Without using the CHEAP compression prior to diffusion, sample quality was poor, even on short ($L = 128$) generations, as shown in Figure 12.

C Data

We use the September 2023 Pfam release, consisting of 57,595,205 sequences and 20,795 families. PLAID is fully compatible with larger sequence databases such as UniRef or BFD (roughly 2 billion sequences), which would offer even better coverage. We elect to use Pfam because sequence domains have more structure and functional labels, making it easier for *in silico* evaluation of generated samples. We also hold out about 15% of the data for validation.

Approximately 46.7% of the dataset ($N = 24,637,236$) is annotated with a GO term. Using the publicly available mapping as of July 1, 2024, we count all GO occurrences; for each Pfam entry with multiple GO entries, we pick the one with the fewest GO occurrences to encourage more descriptive and distinct GO labels.

The `Pfam-A.fasta` file available from the Pfam FTP server includes the UniRef code of the source organism from which the Pfam domain is derived. The UniRef code furthermore includes a 5-letter "mnemonic" to denote the organism. We examine all unique organisms in our dataset and find 3,617 organisms.

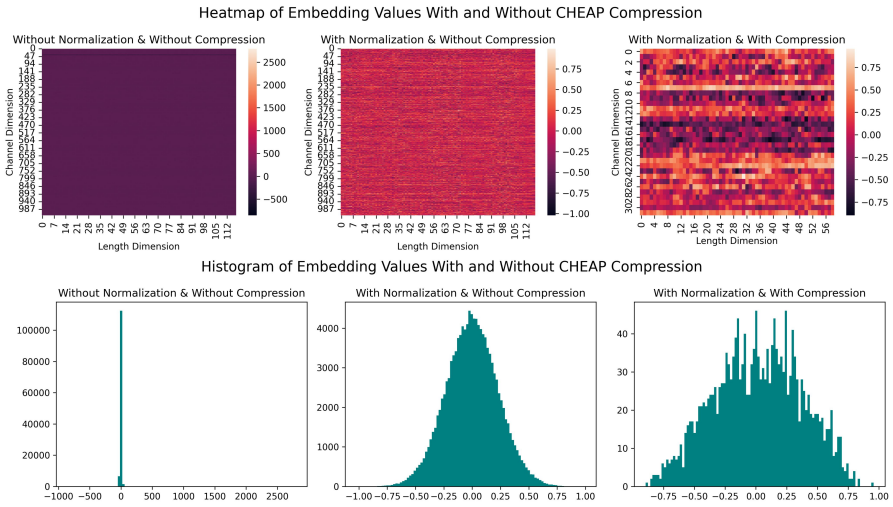

Figure 11: Visualizing the original ESMFold latent space before normalization, after per-channel normalization, and after compression. The value distribution of $p(\mathbf{x}')$ is fairly smooth and "Gaussian-like," making it amenable to diffusion.

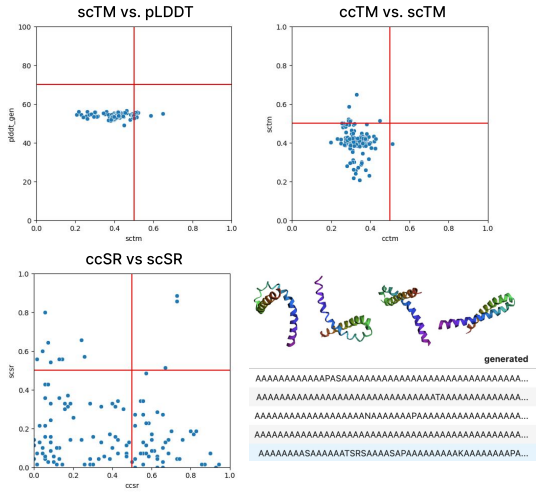

Figure 12: Results when running PLAID on the ESMFold latent space naively without CHEAP compression, for proteins of length 128. There is a tendency to generate repeated sequences, and quality is low compared to baselines.

D SAMPLING

Inference-time sampling hyperparameters provide the user with additional control over quality and sampling speed trade-offs. PLAID supports the DDPM sampler [17] and the DDIM sampler [35], as well as the

improved speed samplers from DPM++ [55]. We find that using the DDIM sampler with 500 timesteps using either the sigmoid or cosine schedulers works best during inference, and reasonable samples can be obtained using the DPM++2M-SDE sampler with only 20 steps. Experiments shown here use the DDIM sampler with the sigmoid noise schedule at 500 timesteps.

Note that the performance bottleneck is found mostly during the latent sampling and structure decoding (which depends on the number of recycling iterations [22, 12] used); however, these two processes can be easily decoupled and parallelized, which cannot be done in existing protein diffusion methods. Furthermore, it allows us to prefilter which latents to decode using heuristic methods, and decode only those latents to structure, which would boost performance for nearly the same computational cost. We do not empirically explore this in this paper to provide a fair comparison, and because the filtering criteria would vary greatly by downstream use.

E Evaluation Details

For all benchmarks and models, we use default settings provided in their open-source code. For ProteinMPNN [23], we use the `v_48_002` model with a sampling temperature of 0.1 and generate 8 sequences per protein, from which the best performing sequence is chosen. To calculate self-consistency, we fold sequences using OmegaFold [56] rather than ESMFold, again using default settings.

Though our models generate all-atom structure, we examine C_α RMSD rather than all-atom RMSD to avoid misattributing sequence generation underperformance to structure generation failures. Also, since there are usually differences in the sequences that are generated, different numbers of atoms make it difficult to assess all-atom RMSD.

For the hold-out natural reference dataset, we use sequences from Pfam and keep length distributions similar to that of the sampled proteins. Specifically, for each sequence bin between $\{64, 72, \ldots, 504, 512\}$, we take 64 natural sequences of that length. For the experiment in Figure 16D, we use the Sinkhorn Distance rather than the Fréchet Distance used commonly in images and video. Since not all functions have a large number of samples, we elected to use a metric that works better in low-sample settings.

Structure novelty is obtained by searching samples against PDB100 using Foldseek [58] `easy-search`. We examine the TM-score to the closest neighbor. For Foldseek and MMseqs experiments, all clustering experiments are performed by length. We use default settings for both tools. Though we report the average TM-Score to the top neighbor for Foldseek, we run `easy-search` in 3Di mode. For sequences, we use MMseqs2 [57] to see if sequences have a homolog in UniRef50, using default sensitivity settings. For samples with homologs, we further calculate the average sequence identity to the closest neighbor to assess novelty (Seq ID %).

E.1 Distributional Conformity to Biophysical Attributes

For Wasserstein Distance to the distribution of biophysical attributes, we examine the following:

- Molecular Weight (MW): the molecular weight calculated from residue identities specified by the sequence.
- Aromaticity: relative frequencies of phenylalanine, tryptophan, and tyrosine, from Lobry and Gautier [67].
- Instability Index: dipeptide-based heuristic of protein half-life, from Guruprasad et al. [60].
- Isoelectricity (pI): the pH at which a molecule has no net electrical charge.

- Hydropathy: based on the GRand AVerage of hYdropathy (GRAVY) metric, from Kyte and Doolittle [61].
- Charge at pH $= 7$: the charge of a given protein at pH $= 7$, i.e., neutral pH.

F SAMPLING SPEEDS

We examine the amount of time necessary for generating a simple sample. We first explore the time necessary to generate 100 sequences with $L = 600$. Multiflow and ProteinGenerator does not support batched generation in its default implementation, so in this experiment, we simply generate one sample at a time for a total of 100 samples. We report the amount of time per sample. For comparison, we also run an experiment where we only generate a single sample, such that none of the methods can make use of any improvements from batching.

Table 4: Time required to sample **proteins with 600 residues**. We assess time required both for sampling $N = 100$ samples in batches whenever possible, and when generating a single sequence. Experiments are run on Nvidia A100. Methods marked by (*) do not support batching in the default implementation.

	seconds/sample, batched		seconds/sample, unbatched	
	Sample Latent	Decode	Sample Latent	Decode
Protpardelle	11.21	-	17.16	-
Multiflow*	231.32	-	277.11	-
ProteinGenerator*	343.32	-	342.28	-
PLAID (100M)	1.64	15.12	27.63	1.07
PLAID (2B)	19.34	15.07	49.03	0.9

Table 5: Forward pass benchmark of vanilla multihead attention compared to the optimized xFormers implementation of memory-efficient attention [49] and FlashAttention-2 [13]. Though FlashAttention2 performed best in our benchmarks, a fused kernel implementation with key padding was not yet available at the time of writing. Since our data contained different lengths (as compared to most image diffusion use cases, or language use-cases that can make use of the implemented causal masking), we instead use the xFormers implementation. We expect that sampling speed results would improve once this feature is becomes available in the FlashAttention package.

Method	**Mean Time (s)**	**Mean Memory (GB)**
Standard Multihead Attention	0.0946 ± 9.23e-4	76.0 ± 0.409
xFormers Memory Efficient Attention	0.0519 ± 4.33e-05	64.0 ± 0.409
Flash Attention	0.0377 ± 1.91e-3	49.2 ± 0.783

G ADDITIONAL RESULTS

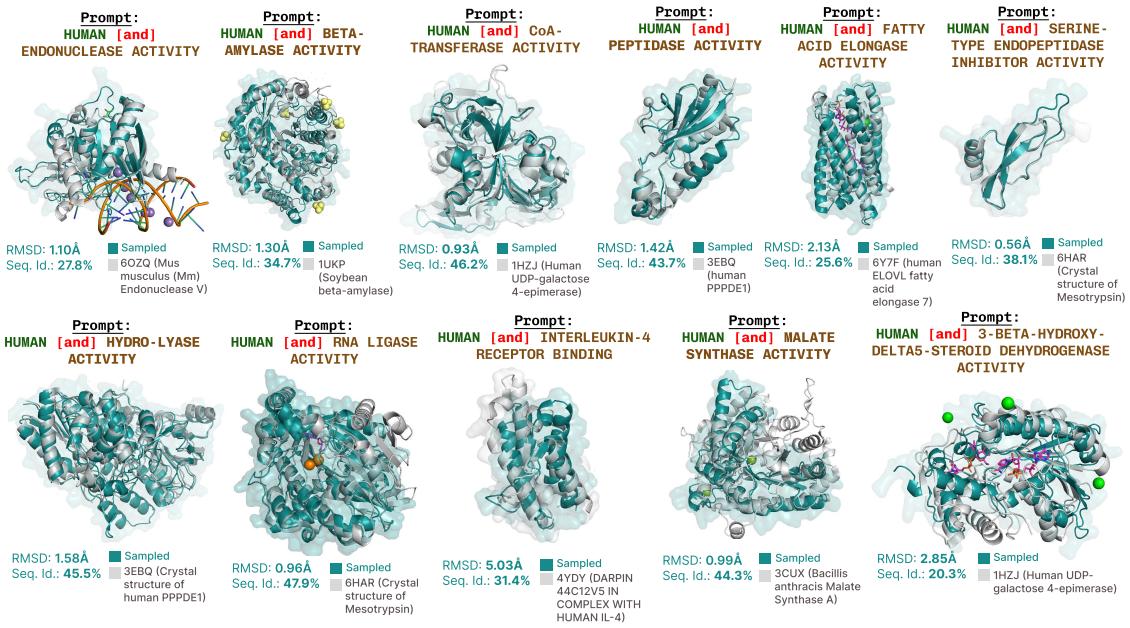

Figure 13: Additional examples of function-conditioned generations.

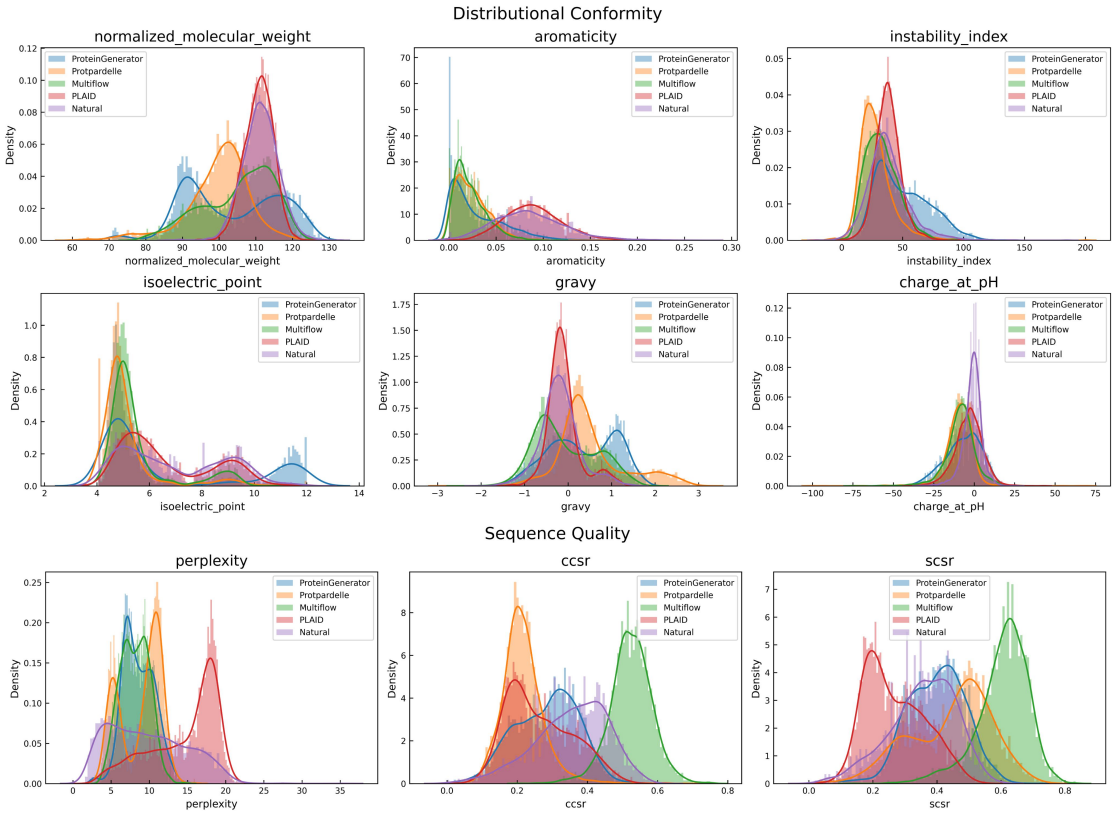

Figure 14: Examining histogram of metrics for nuanced comparison of how generated samples compare to that of natural proteins.

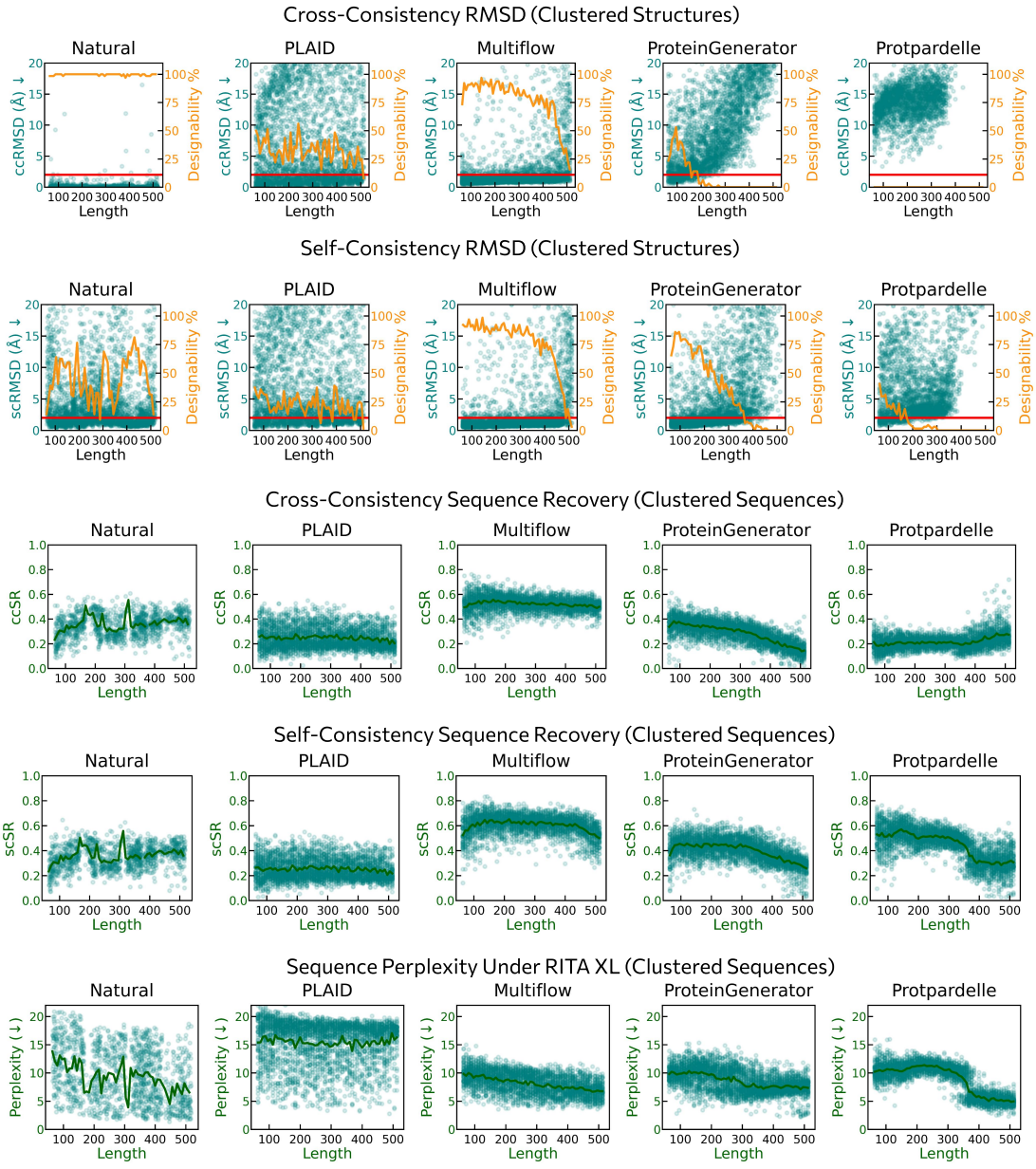

Figure 15: More comparison results between PLAID and baselines.

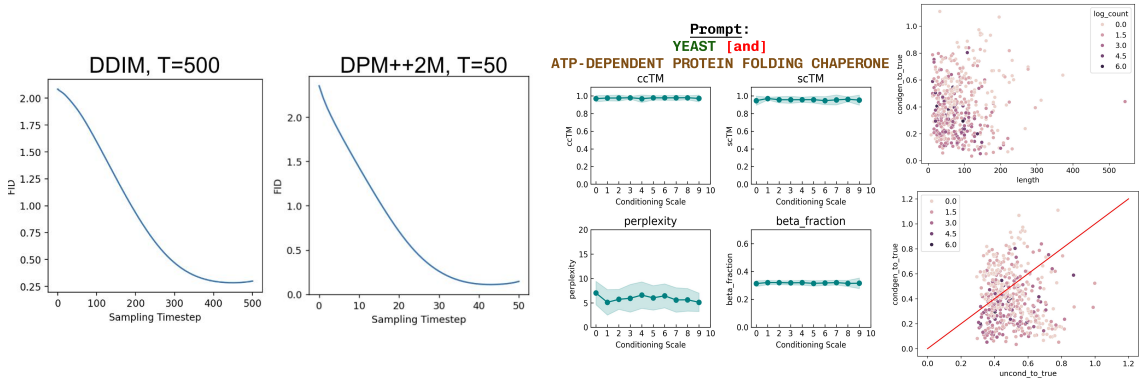

Figure 16: **(Left)** Frechet Distance between sampled protein and reference set of real protein, across sampling (reverse diffusion) timesteps, for the DDIM [35] sampler and the DPM++2M [55] sampler. For both, sample quality decreases steadily over time before plateauing. DPM++2M can achieve low FID results with only 10% of the original number of steps, but final results are still slightly worse. **(Center)** Examining the effect of conditioning scale on the output quality. **(Right)** Analyzing factors which may be contributing to a greater δ difference between the Sinkhorn distance of samples-to-real-functional-proteins vs random-proteins-to-real-functional-proteins.

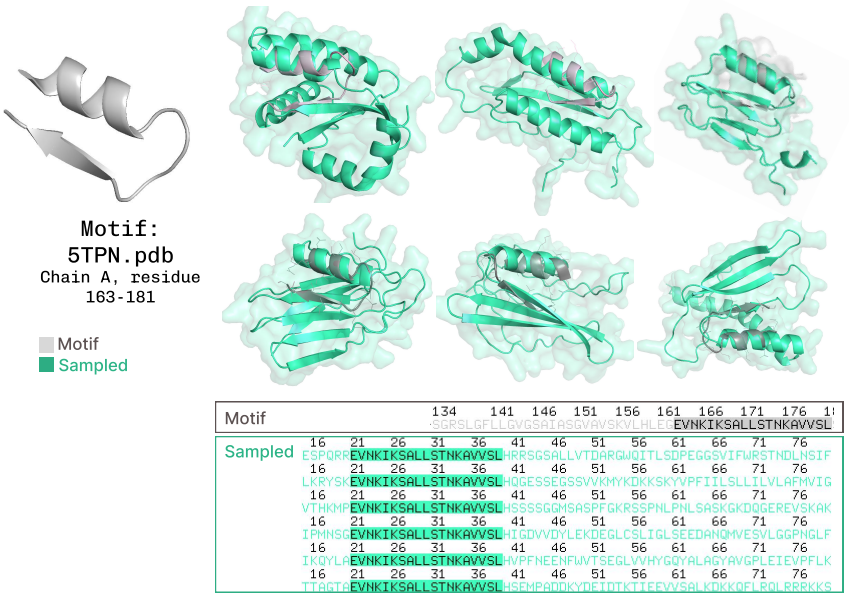

Figure 17: **Demo of motif scaffolding.** We use the same motif as the RFDiffusion [1] `design_motifscaffolding.sh` example for this experiment. The input motif is held constant at the user-prescribed location. Note that PLAID generates all-atom structure, whereas RFDiffusion does not position the sidechain atoms.

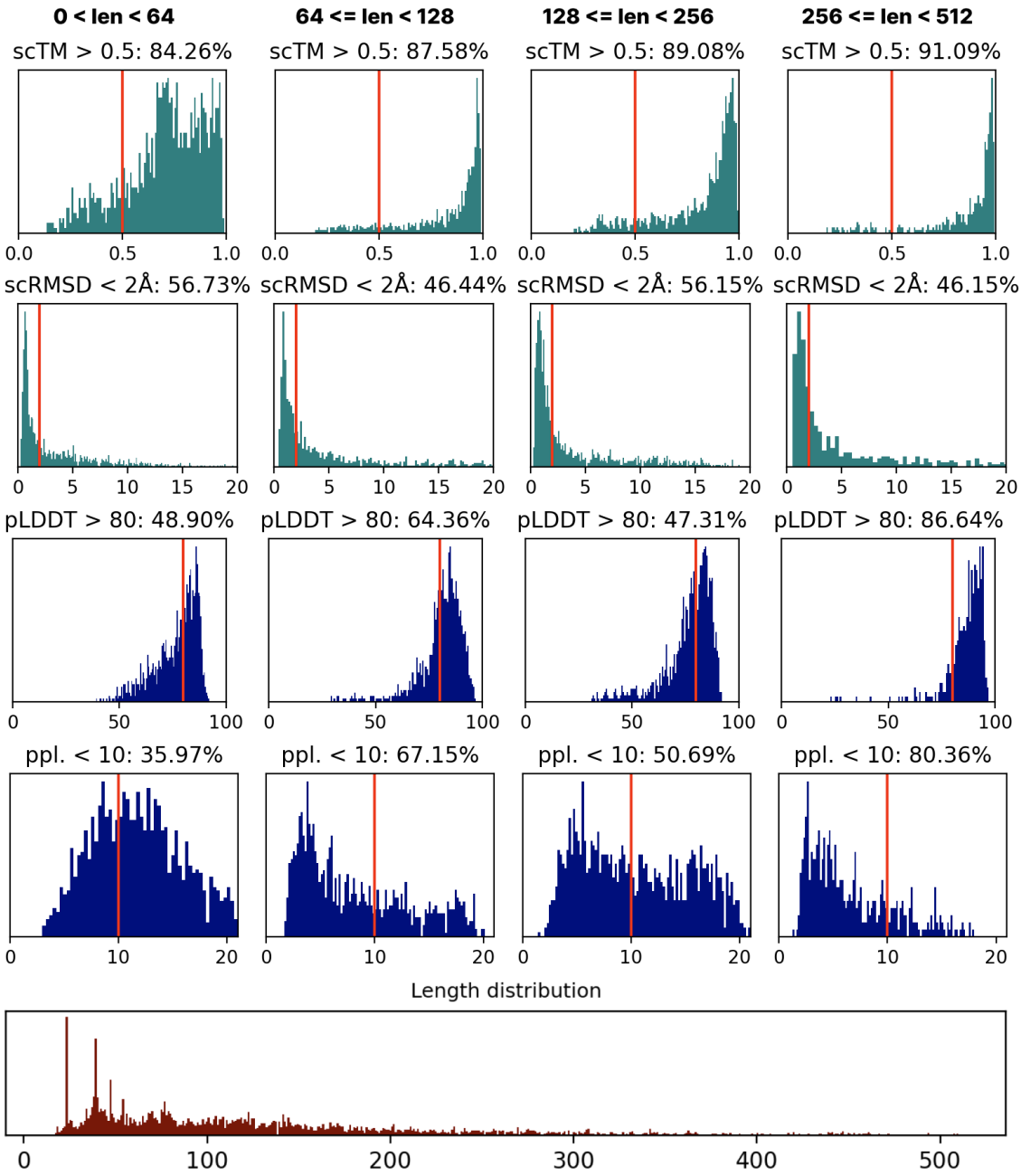

Figure 18: To examine the degree to which co-generation methods are overfitting to structure-based metrics, we examine properties on natural proteins.