An All-Atom Generative Model for Designing Protein Complexes

Ruizhe Chen [*1 3] **Dongyu Xue** [*† 3] **Xiangxin Zhou** [2 3]
Zaixiang Zheng [3] **Xiangxiang Zeng** [1] **Quanquan Gu** [3]

Abstract

Proteins typically exist in complexes, interacting with other proteins or biomolecules to perform their specific biological roles. Research on single-chain protein modeling has been extensively and deeply explored, with advancements seen in models like the series of ESM and AlphaFold2. Despite these developments, the study and modeling of multi-chain proteins remain largely uncharted, though they are vital for understanding biological functions. Recognizing the importance of these interactions, we introduce APM (All-**A**tom **P**rotein Generative **M**odel), a model specifically designed for modeling multi-chain proteins. By integrating atom-level information and leveraging data on multi-chain proteins, APM is capable of precisely modeling inter-chain interactions and designing protein complexes with binding capabilities from scratch. It also performs folding and inverse-folding tasks for multi-chain proteins. Moreover, APM demonstrates versatility in downstream applications: it achieves enhanced performance through supervised fine-tuning (SFT) while also supporting zero-shot sampling in certain tasks, achieving state-of-the-art results. We released our code at https://github.com/bytedance/apm.

1 Introduction

The application of AI technology in protein design has become a prominent research direction across biology, materials science, and artificial intelligence (Notin et al., 2024). The existing works can be categorized into two distinct approaches: general protein foundation models and protein design models for specific functions. The former includes methods such as protein folding models (Jumper et al., 2021; Lin et al., 2023; Baek et al., 2021), inverse-folding models (Dauparas et al., 2022; Hsu et al., 2022; Zheng et al., 2023), co-design models (Shi et al., 2023; Campbell et al., 2024), and protein language models (Lin et al., 2023). These works are not specifically designed for any particular protein design task but aim to learn the general distribution of protein sequences and structures from extensive protein data. The latter approach focuses on the design of proteins with explicit biological activities, such as antibodies (Kong et al., 2023a), binding peptides (Li et al., 2024), and enzymes (Song et al., 2024).

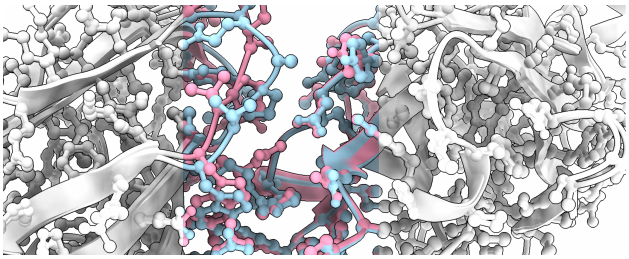

Figure 1. Interactions cause minor atom-level protein structure changes in the binding surface (middle colored part). Blue indicates the isolated structure, pink indicates the binding structure.

General protein foundation models have demonstrated impressive performance across a broad range of tasks. However, these approaches focus solely on modeling single-chain proteins. In contrast, when dealing with proteins involved in specific functions, the target proteins usually appear in the form of complexes. Furthermore, in multi-chain protein modeling, inter-chain interactions that occur at the atom-level play a crucial role (Figure 1). This necessitates incorporating models with atom-level information to enable precise learning of these interactions, which is fundamental for the effective modeling of multi-chain proteins.

To bridge this gap, we propose a novel method: APM (All-Atom Protein Generative Model). APM facilitates the generation of multi-chain protein complexes with all-atom structures and can be applied to various tasks involving multi-chain protein complexes, including generation, folding, inverse-folding, and specific functional protein designs. To develop such a generative model that can be used for designing bioactive complexes, we identify three core chal-

[*]Equal contribution [†]Project Lead [1]College of Computer Science and Electronic Engineering, Hunan University [2]School of Artificial Intelligence, University of Chinese Academy of Sciences [3]ByteDance Seed (this work was done during Ruizhe Chen and Xiangxin Zhou's internship at ByteDance Seed). Correspondence to: Quanquan Gu <quanquan.gu@bytedance.com>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

lenges: **multi-chain protein modeling**, **all-atom representation**, and **sequence-structure dependency**.

Multi-Chain Protein Modeling. Some efforts have attempted to adapt single-chain models for multi-chain protein tasks by using a poly-G pseudo sequence to connect different chains, treating them as a single chain, including AlphaFold2 (Jumper et al., 2021), ESMFold (Lin et al., 2023), and Linker-Tuning (Zou et al., 2023). This enables compatibility with multi-chain data but constrains the structural connectivity to head-to-tail linking, which is not representative of natural complex formations. In this work, we adopt a native method for modeling multi-chain proteins through both data integration and modeling strategies. For data, we use a mixture of single and multi-chain data in the training of APM. We believe that intra-chain modeling will benefit from the extensive amount of single-chain data. For modeling, our efforts include improving the model design and introducing conditional generation tasks. Key changes in model design focus on encoding more information without altering the overall model structure, such as introducing inter-chain or intra-chain attention, thereby maintaining consistency across single and multi-chain proteins.

All-Atom Representation. In protein design with all-atom structures, the fundamental challenge lies in how to effectively represent atomic structures as different amino acid types have distinct atomic types, numbers, and basic structures. When the protein sequence is not determined, the representation of its atom-level structure directly influences the modeling approach. Chu et al. (2024b) utilized an ensemble-based method to model the sidechain coordinates of all amino acid types simultaneously. Martinkus et al. (2024) represented sidechain structures by merging non-rotatable atoms into virtual atoms. Qu et al. (2024) followed the method used in AlphaFold3 (Abramson et al., 2024) to model all-atom coordinates directly. We choose to enhance residue-level information with the sidechain for all-atom protein representation that includes amino acid type, backbone structure, and the sidechain conformation parameterized by four torsion angles. This approach maintains computational efficiency while supplying atom-level information for modeling inter-chain interactions.

Sequence-Structure Dependency. The strong dependency between protein sequence and structure is the foundation for the success of folding and inverse-folding models. However, in the joint generation of protein structure and sequence, this dependency is disrupted during the independent noising process of each modality. This issue hampers effective learning of the dependency between sequence and structure. In APM, two strategies are implemented to enhance the dependency between the sequence and structure modalities. First, we decoupled the noising process for sequences and structures so that the noising level for each modality does not completely align, minimizing disruption of their dependency. Second, there is a 50% probability of performing a folding/inverse-folding task, compelling the model to learn the dependencies from both directions.

Finally, APM has demonstrated its capability in modeling multi-chain proteins and generating bioactive complexes. It achieved state-of-the-art (SOTA) performance in antibody design and binding peptide design. Besides, APM also exhibited exceptional performance in conventional single-chain protein-related tasks.

We highlight our main contributions as follows:

- APM natively supports the modeling of multi-chain proteins without the need to use pseudo sequence to connect different chains;
- APM generates proteins with all-atom structures efficiently by utilizing an innovative integrated model structure;
- Experiments related to general protein demonstrate that APM is capable of generating tightly binding protein complexes, as well as performing multi-chain protein folding and inverse folding tasks;
- Experiments in specific functional protein design tasks show that APM outperforms the SOTA baselines in antibody and peptide design with higher binding affinity.

2 Related Work

Protein Foundation Models. The breakthrough achievements in protein structure prediction, marked by AlphaFold series (AlphaFold1-3 (Senior et al., 2020; Jumper et al., 2021; Abramson et al., 2024) and RoseTTAFold (Baek et al., 2021; Krishna et al., 2024), have revolutionized the field of protein science. With these developments, protein language models (Rives et al., 2019; Madani et al., 2020) have emerged as powerful tools. The series of ESM (Rives et al., 2019; Lin et al., 2023; Hayes et al., 2025), trained on large-scale protein sequence data, have demonstrated remarkable capabilities in protein understanding and generation. Meanwhile, certain methods considered protein design workflow in two stages: RFdiffusion (Watson et al., 2023) tackles backbone structure generation using diffusion models, while ProteinMPNN (Dauparas et al., 2022) specializes in sequence design through message-passing neural networks. FrameFlow (Yim et al., 2024; 2023b) and FoldFlow (Bose et al., 2023) present developments applying SE(3) flow matching approaches to protein structure generation. FoldFlow2 (Huguet et al., 2024) further demonstrates the integration of protein language models for structure generation. Besides, Chroma (Ingraham et al., 2023) introduces a unified approach to protein design through a generative model that can directly sample novel protein structures and sequences while being conditioned to target specific properties and functions The field has also seen approaches like Multiflow (Campbell et al., 2024), ProteinGenerator (Lisanza et al., 2024), and Protpardelle (Chu et al., 2024a),

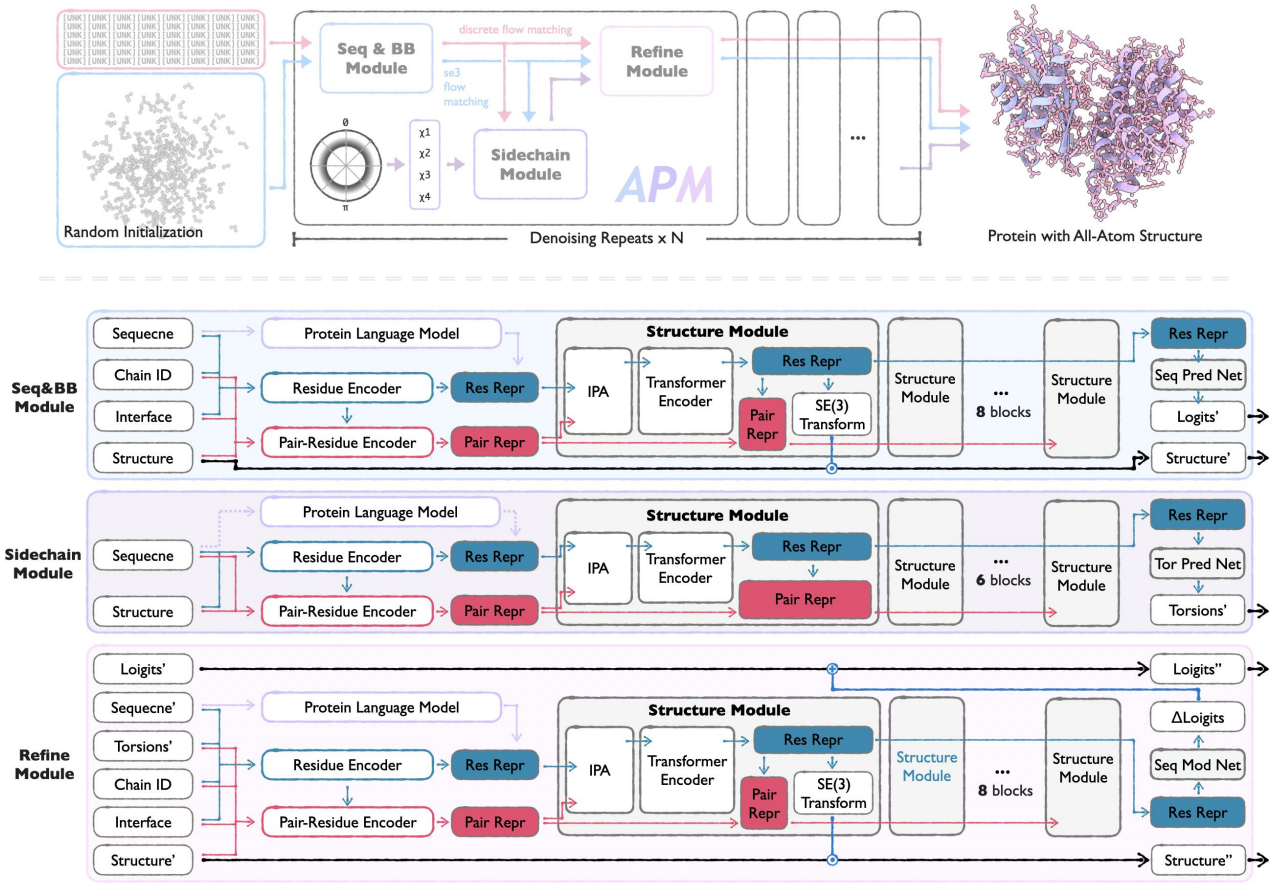

Figure 2. Overview of APM. APM consists of three modules: (1) A flow-matching based `Seq&BB` Module for generating backbone structure and sequence simultaneously; (2) a `Sidechain` Module for generating the all-atom structure based on the previous module's generation; (3) A `Refine` Module adjusts the sequence and structure with all-atom information. The iterative denoising process enables the generation of multi-chain proteins with all-atom structure. The detailed architecture of each module are presented below.

which enable generation of both sequence and structure. More recently, SaProt (Su et al., 2024a;b;c) and DPLM series (Zheng et al., 2023; Wang et al., 2024a;b; Hsieh et al., 2025) have further advanced protein token modeling by incorporating structural information into the pre-training process, enabling better understanding of protein sequences.

Functional Protein Design. Target-specific protein design has made remarkable advances recently. In antibody design, approaches like HERN (Jin et al., 2022), DiffAb (Luo et al., 2022), MEAN (Kong et al., 2023a), and dyMEAN (Kong et al., 2023b) have demonstrated the ability to generate functional antibodies. Besides, Wu & Li (2024), Zhu et al. (2024), and Gao et al. (2023) introduced pre-trained protein language models as sequence priors to improve antibody design. For peptide design, methods such as PPFlow (Lin et al., 2024), PepFlow (Li et al., 2024), PepGLAD (Kong et al., 2024) and CpSDE (Zhou et al., 2025) focus on designing bioactive peptides.

3 APM

In this section, we present APM, an all-atom generative protein model for designing bioactive complexes with the all-atom structure. We first define how we represent the all-atom structure in Section 3.1. Then we introduce the model architecture of APM in Section 3.2, and in Section 3.3, we introduce the learning objective and training process. Finally, we introduce the sampling method in Section 3.4.

3.1 Representation for Protein All-Atom Structure

In this study, we divide the goal of our approach into two parts: **1**, **foundational modeling of intra-chain sequences and structures** (determining what constitutes a plausible protein sequence and structure); **2**, **modeling of inter-chain interactions** (understanding how proteins interact with each other). Residue-level information is generally sufficient for intra-chain modeling of protein sequences and structures. Methods such as AlphaFold2 (Jumper et al., 2021), ESMFold (Lin et al., 2023), and ProteinMPNN (Dauparas et al., 2022), which leverage residue-level information, have

demonstrated high-quality protein structure modeling. However, these models often require an additional relaxation step to resolve atomic clashes on the sidechains due to the lack of finer detail modeling. Therefore, we enhance residue-level information by incorporating sidechain conformations into the protein representation.

While modeling the all-atom coordinates provides the most detailed view of interactions, it considerably raises complexity and limits the ability to model longer proteins, especially for multi-chain proteins. Sidechains of amino acids are not entirely free in their structures but have some conformations. Each amino acid has up to four rotatable bonds in its sidechain while maintaining a largely consistent atomic structure between these bonds. Therefore, combining amino acid type with sidechain torsion angles offers a comprehensive representation of sidechain conformation.

Ultimately, we adopted a representation that includes **amino acid type**, **backbone structure**, and **sidechain torsion angles**. This approach maintains computational efficiency while providing richer information for modeling inter-chain interactions.

Notations. The all-atom protein structure is represented as a collection of amino acid types, backbone frames, and sidechain torsion angles (Jumper et al., 2021; Lehninger et al., 2005). A multi-chain protein complex $\mathcal{P}$ is composed of K chains and $N = \sum_k k_N$ residues in total. For the k-th chain, the amino acid sequence is denoted as $\mathbf{S}_k = [S_k^{k_1}, S_k^{k_2}, \ldots, S_k^{k_N}]$, where $S_k^{k_i} \in \mathcal{A}$ and $\mathcal{A}$ is the set of 20 standard amino acids. Meanwhile, the backbone structure of this chain is characterized by rigid frames $\mathbf{T}_k = \left[T_k^{k_1}, T_k^{k_2}, \ldots, T_k^{k_N}\right]$, where each $T_k^{k_i} \in \mathrm{SE}(3)$ consists of a rotation $R_k^{k_i} \in \mathrm{SO}(3)$ and a translation vector $\mathbf{x}_k^{k_i} \in \mathbb{R}^3$, mapping rigid transformations from ideal peptide geometry (Engh & Huber, 2006). The sidechain torsion angles are denoted as $\chi_k = [\chi_k^{k_1}, \chi_k^{k_2}, \ldots, \chi^{k_N}]$, where $\chi_k^{k_i} \in [0, 2\pi)^4$ corresponds to the torsions of rotatable bonds in the sidechain of i-th residue. For **brevity**, we slightly abuse the notation such that $\mathbf{S} = \bigcup_k \mathbf{S}_k$, $\mathbf{T} = \bigcup_k \mathbf{T}_k$, $\chi = \bigcup_k \chi_k$, where chain indices (*i.e.*, k) are omitted hereafter unless needed.

3.2 An Integrated Architectural Design of APM

3.2.1 Overall Architecture

To implement an All-Atom Protein Generative Model, we designed the APM consisting of three distinct modules: the `Seq&BB` Module, the `Sidechain` Module, and the `Refine` Module (Figure 2). The `Seq&BB` Module is a flow-matching-based protein generative model that handles the co-generation of sequence and structure at the residue level. The `Sidechain` Module serves as an all-atom completion model, predicting the sidechain conformations for proteins generated by the `Seq&BB` Module. The `Refine` Module is an All-Atom Protein Refinement model, refining the generated proteins to make them more akin to natural proteins while resolving structural clashes.

When using APM for protein generation, the final result is progressively generated from noise to data, with timestep t from 0 to 1. Notably, the `Sidechain` Module and `Refine` Module are activated only after timestep $t \geq \mathcal{T}$ ($\mathcal{T} = 0.8$ here). We believe that the quality of proteins produced by the `Sidechain` Module with t far from data time (1) is insufficient to support high-quality predictions by the `Sidechain` Module and causes the meaningless in the refinement by the `Refine` Module.

The motivation for designing this integrated architecture, rather than using a single model to directly generate proteins with all-atom structures lies in the incompatibility of training the sidechain prediction model with the sequence and structure flow-matching model. Two key reasons prevent us from doing this: **(1)** sidechain prediction requires real sequences and structures to obtain accurate sidechain conformation labels, while the flow-matching model uses noised sequences and structures as input; **(2)** while it is possible to generate sidechain conformations using a flow-matching approach rather than a packing model (one-step prediction), this would require providing a noised sidechain conformation, χ_t, which still contains amino acid type information, **leading to sequence information leakage** (details in Appendix C). This is evidenced by the rapid convergence of amino acid type loss during training. During sampling, the absence of truly noised torsion angles, χ_t, significantly degrades the model's performance in the inference phase.

With the generation of sequence & backbone structures separated from sidechain conformations, an additional module is necessary to allow all-atom information to influence the design of the backbone structure accordingly. For this purpose, we developed the third module of APM, the `Refine` Module. It receives outputs from both the `Seq&BB` Module and `Sidechain` Module, making it all-atom aware. Based on this comprehensive information, the `Refine` Module further optimizes the sequence and backbone structure to ensure the overall structure more closely resembles natural proteins.

3.2.2 Sub-module Architectures

The core structure of the three submodules is essentially the same. We use stacked structure modules derived from AlphaFold2 as the trunk of APM. Each structure module is composed of IPA (Jumper et al., 2021) and a Transformer Encoder, which is employed to update residue information and pair-residue information. The differences among the submodules lie in the encoding of the input and the distinctions in the output, driven by the various modeling tasks. Apart from this, the `Seq&BB` Module and the `Refine` Module maintain consistent model sizes, whereas the `Sidechain` Module has fewer structure module blocks and a smaller hidden dimension. We believe that predict-

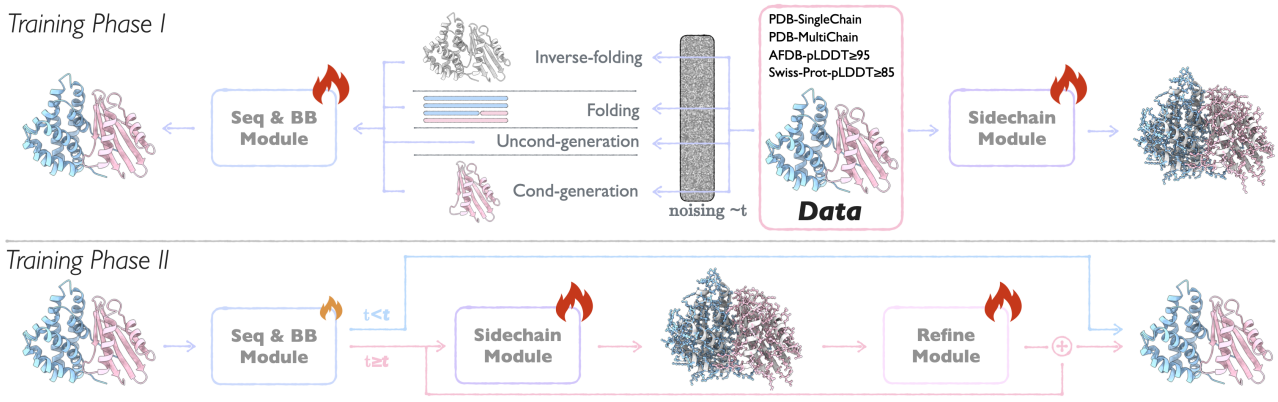

Figure 3. The two-phase of the training process of APM. In training phase I, the `Seq&BB` Module and `Sidechain` Module are trained separately. In training phase II, the three modules form the integral APM, and are trained in an iterative paradigm. In any phase, the training data is a mixture of PDB (Berman et al., 2000) single/multi-chain proteins, Swiss-Prot proteins, and AFDB (Varadi et al., 2022) proteins.

ing sidechain conformations is a relatively simple task, and utilizing a smaller model for enhancing efficiency.

3.2.3 Integration of Protein Language Model

A robust understanding of protein sequences requires a large-scale model trained on tens of millions of sequence data (like the 3B-parameters ESM2 in ESMFold, which is trained on 65 million unique sequences, is responsible for sequence understanding), or alternatively, using MSA as the sequence representation (like AlphaFold2/3). APM is trained on protein data with structural information, yet the available volume of such data is not sufficient to support learning the intricacies of protein sequence understanding. To address this, we integrated protein language models (PLMs) into all modules to enhance protein sequence understanding.

We utilized ESM2-650M, the widely adopted protein language model, to represent the input sequences. Drawing from ESMFold's approach, we used learnable weights to aggregate the representations from each layer of the protein language model, yielding the final amino acid encodings. It's important to note that ESM2 is only trained on single-chain data. Therefore, when encoding multi-chain proteins with ESM2, each chain is encoded individually.

3.3 Training of APM

In order to train APM with the integrated architecture, we designed a two-phase training approach (Figure 3). In phase I, `Seq&BB` Module and `Sidechain` Module are trained separately. In phase II, the three modules are joint-trained in an iterative paradigm. All the details refer to Appendix B.

3.3.1 Training of Seq&BB Module

`Seq&BB` Module is the foundation model in APM to generate the sequence and backbone structure trained in a flow-matching manner with tasks of unconditional generation, conditional generation, folding, and inverse folding. The primary learning objective is reconstructing either sequence or structure, or both, from a noisy state. As we decouple the noising processes of the two modalities, we denote the noised sequence as $\mathbf{S}_{t_S}$ and the noised structure as $\mathbf{T}_{t_T}$, where $t_S, t_T \sim \mathcal{U}(0,1)$ represent intermediate time steps. We also denote the original sequence and structure as $\mathbf{S}_1$ and $\mathbf{T}_1$. Then the learning objectives of `Seq&BB` Module are: $p_{\texttt{Seq\&BB}}(\mathbf{S}_1, \mathbf{T}_1 | \mathbf{S}_{t_S}, \mathbf{T}_{t_T}, t_S, t_T)$ for unconditional generation; $p_{\texttt{Seq\&BB}}(\mathbf{S}_1 | \mathbf{S}_{t_S}, \mathbf{T}_1, t_S)$ for inverse-folding; $p_{\texttt{Seq\&BB}}(\mathbf{T}_1 | \mathbf{S}_1, \mathbf{T}_{t_T}, t_T)$ for folding.

For each of the three tasks, there is a conditional version for multi-chain data, in which part of the target modalities is set as the noiseless state. The flow-matching loss $\mathcal{L}_{\text{flow-matching}}$ is defined over sequence and backbone structure as:

$$\mathcal{L}_{\text{flow-matching}} = \mathcal{L}_{\text{discrete}} + \mathcal{L}_{\text{SE(3)}}$$

It consists of two components. For the sequence, $\mathcal{L}_{\text{discrete}}$ measures the cross-entropy between the predicted sequence distribution $p_{\texttt{Seq\&BB}}(\hat{\mathbf{S}}_1 | \mathbf{S}_{t_S}, \mathbf{T}_{t_T})$ against the true sequence $\mathbf{S}_1$. For the structure, it is the mean squared error between the vector fields calculated from the noisy structure $\mathbf{T}_{t_T}$ to the generated structure $\hat{\mathbf{T}}_1$ and to the true structure $\mathbf{T}_1$. For the complete mathematical formulation, refer to Appendix A.

The refinement by the `Refine` Module can be considered as a posterior correction, where the generated protein $(\hat{\mathbf{S}}_1, \hat{\mathbf{T}}_1)$ at each sampling step is corrected to $(\tilde{\mathbf{S}}_1, \tilde{\mathbf{T}}_1)$ before being noised for the next step. We hope to ensure that the direction of correction at each step is as consistent as possible. By achieving this, the corrections at each step can accumulate, leading to improved performance. This consistency necessitates that the predicted $(\hat{\mathbf{S}}_1, \hat{\mathbf{T}}_1)$ at each (t_S, t_T) is aligned, which in turn requires the `Seq&BB` Module to maintain a smoother generative trajectory. To facilitate this, we incorporated a consistency loss, $\mathcal{L}_{\text{consistency}}$, into the `Seq&BB` Module to minimize the variations between predictions for adjacent t (details in Appendix B.1).

The final training loss of Seq&BB Module is defined as:

$$\mathcal{L}_{\texttt{Seq\&BB}} = \mathcal{L}_{\text{flow-matching}} + 0.3 \times \mathcal{L}_{\text{consistency}}$$

The training loss is consistent in two phases, the only difference is the learning rate.

3.3.2 Training of Sidechain Module

The learning objective of Sidechain Module is to predict the sidechain torsions, χ, given a protein with sequence and backbone structure. In the training phase I, the learning objective is **packing**, $p_{\text{Sidechain}}(\chi|\mathbf{S}_1, \mathbf{T}_1)$, which is no different from the normal packing model. While in phase II, the learning objective is switched to $p_{\text{Sidechain}}(\chi|\hat{\mathbf{S}}_1, \hat{\mathbf{T}}_1)$, which means the **reconstruction** of the ground truth sidechain from the predicted $(\hat{\mathbf{S}}_1, \hat{\mathbf{T}}_1)$. Besides, we also want Sidechain Module to keep the ability of packing, so there is a 50% probability that packing will continue to be used as the learning objective in phase II.

Training loss of Sidechain Module is also different for each learning objective. For **packing**, the loss consists of supervised torsion angle loss and all-atom Frame Aligned Point Error (FAPE) loss (Jumper et al., 2021), is defined as:

$$\mathcal{L}_{\text{Packing}} = \mathcal{L}_{\chi} + \mathcal{L}_{\text{FAPE}}$$

For **reconstruction**, we only maintain torsion angle loss, $\mathcal{L}_{\chi}$, as the input protein sequence and structure may not match the ground truth, resulting in the inappropriateness for calculating error on all frames (details in Appendix B.2).

3.3.3 Training of Refine Module

The Refine Module $p_{\text{Refine}}(\tilde{\mathbf{S}}_1, \tilde{\mathbf{T}}_1|\hat{\mathbf{S}}_1, \hat{\mathbf{T}}_1, t_S, t_T, \hat{\chi})$ is tasked to predict the real protein based on the generated one, $(\hat{\mathbf{S}}_1, \hat{\mathbf{T}}_1)$, with the all-atom level information formed with the predicted sidechain torsions $\hat{\chi}$. Thus, we define a correction loss on the corrected protein $(\tilde{\mathbf{S}}_1, \tilde{\mathbf{T}}_1)$ as the learning objective as:

$$\mathcal{L}_{\text{corr}} = -\log p(\mathbf{S}_1|\tilde{\mathbf{S}}_1) + \|\tilde{\mathbf{x}}_1 - \mathbf{x}_1\|^2 + \|\tilde{R}_1 - R_1\|^2$$

Besides, we also incorporate auxiliary objectives in the training of Refine Module, including backbone FAPE loss, $\mathcal{L}_{\text{BB-FAPE}}$, and residue distogram prediction loss, $\mathcal{L}_{\text{dist}}$ (details in Appendix B.3). Finally, the training loss of Refine Module is defined as: $\mathcal{L}_{\text{Refine}} = \mathcal{L}_{\text{corr}} + 0.25 \times \mathcal{L}_{\text{BB-FAPE}} + 0.25 \times \mathcal{L}_{\text{dist}}$.

3.3.4 Training in Phase II

In the second phase of the training, we did not train the three modules simultaneously as each module requires a distinct t range. Instead, we employed an iterative approach for training these three modules. Given that the modules of Seq&BB and Sidechain were already trained in phase I, they are only trained for 2 steps in each iteration in phase II, whereas the Refine module requires 8 steps of training. Each cycle comprises 12 steps, with the steps of 2-2-8 respectively.

The ultimate training loss of APM is defined as the expectation over timesteps and each residue in the protein:

$$\mathcal{L} = \mathbb{E}_{t\sim\mathcal{U}[0,1]}\left[\mathcal{L}_{\text{Seq\&BB}} + \mathcal{L}_{\text{Packing}} + \mathcal{L}_{\text{Refine}}\right]$$

3.4 Sampling Strategy

For structure sampling, we directly use the structure predicted by Seq&BB Module or the Refine Module corrected one as the model output if it is activated.

For sequence sampling, the strategy is different. To fully leverage the Protein Language Model, we update all the residues at each inference step and only keep the residues located in the positions with top $\max(\log(\text{prob}))$. For the top K positions (where K is the number of amino acids to be unmasked at the current t), we sample the amino acid types based on the corresponding logits with a carefully designed strategy composed of temperature annealing sampling and $\arg\max$. The remaining positions are set to [MASK] token (details refer to Appendix D.1). This decoding strategy also led us to abandon the flow-matching training approach for the Sidechain Module, as the amino acid type at each position may change during the sampling process, making it inappropriate for the Sidechain Module to rely on the sequence from the previous step for prediction.

4 Experiments

4.1 Data Curation

Single-chain data is built from three sources: PDB (Berman et al., 2000), Swiss-Prot (Boeckmann et al., 2003), and AFDB (Varadi et al., 2022). For PDB samples, we followed the data processing flow in MultiFlow, resulting in 18684 samples. For Swiss-Prot samples, we selected the samples with a pLDDT (Jumper et al., 2021; Mariani et al., 2013) greater than 85, resulting in 140769 samples. For AFDB samples, we take a more rigorous filter, leaving samples with a pLDDT greater than 95, which resulted in 28041 samples. Finally, we got 187494 single-chain samples.

Multi-chain data is built from PDB Biological Assemblies (Rose et al., 2016). To prevent potential information leakage in downstream tasks, we discarded samples that met any of the following conditions: (1) the sample's PDB ID is present in SAbDab (Dunbar et al., 2014); (2) the sample contains at least one chain with length less than 30, which is considered a peptide (Kong et al., 2024; Tsaban et al., 2022). The last condition led to the removal of a substantial number of samples (12,163). In many cases, the peptides played crucial roles in stabilizing the complexes and only removing peptides is unreasonable. Consequently, we opted to exclude this subset of data entirely from training. We also removed samples with lengths exceeding 2048 or lacking cluster IDs. Finally, we got 11620 multi-chain samples.

Cropping. During the training process, we performed random cropping (Evans et al., 2021) on multi-chain samples with residues exceeding 384 to prevent out-of-memory.

Table 1. Performance comparison of protein folding (blue highlighted) and inverse-folding tasks (pink highlighted). For each metric, we report the average/median performance.

Method	RMSD ↓	TM ↑	scTM ↑	AAR(%) ↑	ppl ↓
ESMFold	2.84/1.19	0.93/0.97	-	-	-
ProteinMPNN	-	-	0.94/0.97	46.58/46.76	11.44/11.48
ESM3(1.4B)	**4.71/2.27**	0.83/**0.91**	0.94/0.97	49.50/49.42	**8.64/7.90**
MultiFlow*	15.64/16.08	0.53/0.49	0.94/0.96	37.74/37.59	10.86/10.94
APM	4.83/2.64	**0.86/0.91**	0.94/0.97	**50.44/50.41**	8.74/8.10

Table 2. Performance comparison of different methods for various protein lengths. We evaluate the methods on three different length ranges (100, 200, 300) using `scTM` and `scRMSD`.

Method	Length 100		Length 200		Length 300	
	scTM	scRMSD	scTM	scRMSD	scTM	scRMSD
NativePDBs	0.91	2.98	0.88	3.24	0.92	3.94
ESM3(1.4B)	0.72	13.80	0.63	21.18	0.59	25.5
MultiFlow*	0.86	4.73	0.86	4.98	0.86	6.01
ProteinGenerator	0.91	3.75	0.88	6.24	0.81	9.26
ProtPardelle	0.56	12.90	0.64	13.67	0.69	14.91
APM	**0.96**	**1.80**	**0.89**	**4.25**	**0.87**	**5.96**

Cropping was centered around the randomly selected inter-chain residue pair at the binding interface, retaining the 384 amino acids nearest to the pair. PLM encodes the sequence of cropped samples before cropping.

4.2 Single-Chain Protein Related Tasks

While APM is specifically designed for modeling multi-chain proteins, it also possesses the capabilities of those foundation models designed for single-chain proteins, including folding and inverse-folding. We validated the folding and inverse-folding capabilities of APM on a PDB date split used by MultiFlow. We compared it with specialized models, including ESMFold and PorteinMPNN, as well as co-design models capable of performing multiple tasks, including ESM3 and MultiFlow*(without distillation). We utilize `RMSD` and `TMscore` (Zhang & Skolnick, 2005) between predicted and ground truth structures to evaluate folding performance, and self-consistency (Trippe et al., 2022) `TMscore` (`scTM`), amino acid recovery (`AAR`) and perplexity to evaluate inverse-folding performance. The perplexity (`ppl`) is provided by ProGen2-base (Madani et al., 2020; Nijkamp et al., 2023). The results are shown in Table 1.

APM can also perform unconditional protein generation. Besides ESM3 and MultiFlow*, we compared two methods capable of all-atom design, ProteinGenerator and ProtPardelle. For this task, we followed the evaluation methods in ProteinBench (Ye et al., 2024) and presented the average `scRMSD` and `scTM` for proteins with lengths of 100-300 in Table 2. APM achieved competitive performance compared to other co-design methods in all three tasks.

Table 3. Performance comparison in multi-chain protein folding (blue highlighted) and inverse-folding tasks (pink highlighted).

Method	RMSD ↓	TM ↑	scTM ↑	AAR(%) ↑
Boltz-1 w/MSA	5.40/1.95	0.87/0.97	-	-
Boltz-1 w/oMSA	17.86/18.43	0.44/0.45	-	-
ProteinMPNN	-	-	0.90/0.96	46.17/46.37
APM	12.6/13.67	0.64/0.62	0.85/0.95	61.26/59.48

4.3 Multi-Chain Protein Related Tasks

4.3.1 Folding & Inverse-Folding

We also initially examined APM's capabilities in modeling multi-chain proteins through folding and inverse-folding tasks. In these tasks, we used samples missing cluster IDs that were dropped during training as the test set, and we also removed samples exceeding a length of 512. The final test set comprised 273 proteins with a number of chains of 2-6. Furthermore, in the two tasks, we only compared APM with two specialized models, Boltz-1 (Wohlwend et al., 2024) and ProteinMPNN, as there are almost no other models that support multi-chain proteins. For the inverse-folding task, we employed Boltz-1 with MSA to refold the predicted sequences for calculating `scTM`. As depicted in Table 3, folding for multi-chain proteins represents an extreme challenge. Even with the use of MSA, Boltz-1 exhibits a decline in prediction accuracy compared to single-chain proteins. Without MSA, achieving effective prediction becomes considerably more difficult. Although the performance of APM also degrades, it still surpasses that of Boltz-1 without MSA. Conversely, APM exhibits commendable performance in inverse-folding for multi-chain proteins, with the `scTM` nearly matching the folding performance of Boltz-1 when using reference sequences.

4.3.2 Multi-Chain Protein Generation

The most significant difference between the APM and previous methods is its ability to directly generate multi-chain protein complexes. These generated complexes do not require the starting amino acids of each chain to be spatially close to the previous chain. Instead, they have independent spatial positions, yet each chain possesses precise and complementary binding interfaces with the others. However, evaluating these generated complexes poses challenges. Calculating the self-consistency of the complexes does not accurately assess APM 's capabilities in complex generation as folding models cannot reliably predict structures. Evaluating each single-chain independently for self-consistency is also not entirely appropriate because single-chain proteins may undergo significant conformational changes upon binding, such as 1AKE to 4AKE, or some might only fold correctly in the presence of other proteins.

As APM has demonstrated its single-chain protein generation capability in Section 4.2. We focus on the binding affinity between each chain in multi-chain proteins in this task. We use ΔG to represent the binding strength between

Table 4. The inter-chain binding affinity between generated complexes. For each metric, we report the average/median value. APM_{BB} means using APM in a residue-level manner by only activating the `Seq&BB` Module. We additionally use ProteinMPNN to redesign sequences for Chroma.(marked with *)

Length	Model	ΔG_{RSC}	ΔG_{RAA}	RMSD
50-100	Chroma	133.64/46.51	-83.96/-86.66	1.33/1.22
	Chroma*	-27.53/-41.71	-78.41/-77.09	1.44/1.28
	APM	-72.44/-71/91	-112.65/-116.98	1.05/0.95
	APM_{BB}	-64.30/-67.30	-114.94/-114.45	1.06/1.03
100-100	Chroma	89.47/22.97	102.33/48.34	1.46/1.39
	Chroma*	-31.09/-31.51	-62.15/-59.36	1.40/1.30
	APM	-91.61/-94.54	-130.31-134.57	1.04/0.94
	APM_{BB}	-36.74/-69.30	-117.53/-118.13	1.17/1.12
100-200	Chroma	79.97/35.86	-59.32/-54.30	1.58/1.48
	Chroma*	-32.14/-31.79	-62.79/-59.74	1.58/1.39
	APM	-44.02/-39.42	-93.21/-73.09	1.35/1.21
	APM_{BB}	-3.42/-33.71	-85.79/-69.12	1.58/1.42

two chains of multi-chain proteins with lengths of 50-100, 100-100, and 100-200 (the generation of more chain combinations is shown in the Appendix D.3). We report two types of ΔG: ΔG_{RSC}, for sidechain-only relaxed complexes; and ΔG_{RAA}, for all-atom relaxed complexes (both relaxation and ΔG calculation are performed by pyRosetta (Alford et al., 2017; Chaudhury et al., 2010)). Additionally, we report the `RMSD` between the two structures. The average/median results are shown in Table 4. For comparison, we utilized Chroma (Ingraham et al., 2023) to sample unconditional complex with the same length combinations. Besides, we also performed the same task with only `Seq&BB` Module activated (APM_{BB}), which means generated multi-chain proteins at residue-level. As shown in Table 4, compared with using all-atom information, APM_{BB} achieved weaker binding strength and higher `RMSD`, which proves **the importance of the all-atom information in the inter-chain interactions modeling**.

By default, APM generates all chains simultaneously. Under this manner, we observed that for complexes with chain length combinations of 50-100 and 100-100, APM tends to generate multi-chain proteins in a "single-chain protein mode", leading to strong inter-chain interactions. Conversely, in complexes with chain lengths of 100-200, APM generates two relatively independent single-chain proteins that are bound tightly, resulting in normal binding energies (Figure 4). APM also supports an alternative generation manner called **chain-by-chain**, raised from the conditional generation task in APM's training. The **chain-by-chain** approach yields significantly different results, with each chain appearing to fold independently before ultimately binding together. Related results can be found in Appendix D.3.

4.4 Downstream Tasks

We further verify the capacity of APM on specific tasks including antibody design and peptide design, in both supervised fine-tuning (SFT) and zero-shot manner. Details about SFT refer to Appendices B.4, D.4 and D.5.

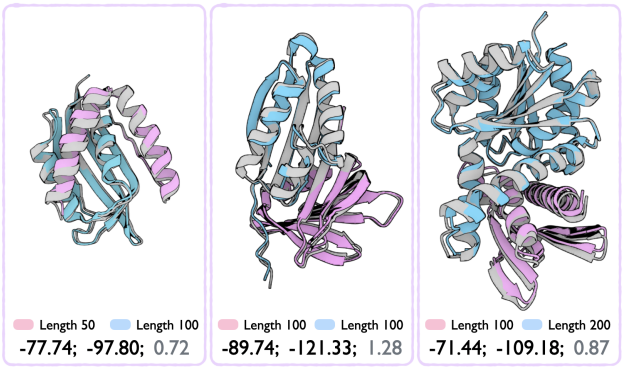

Figure 4. Showcases of the three length combinations. For each case, the gray structure represents APM's generated structure and the colored structure represents the backbone relaxed structure. Different chain is highlighted with different colors. We also report the two ΔG and the `RMSD` between the two structures.

4.4.1 Antibody CDR-H3 Co-Design

Setup. The design of Complementarity Determining Regions(CDRs) is a crucial step in developing potent therapeutic antibodies, especially CDR-H3. Following the data preprocessing pipeline introduced in (Ye et al., 2024; Zhou et al., 2024), we conduct training on the Structural Antibody Database (Dunbar et al., 2014) and perform evaluation on the RAbD benchmark (Adolf-Bryfogle et al., 2018).

We compare our model with four antigen-specific antibody design methods (**dyMEAN** (Kong et al., 2023b), **DiffAb** (Luo et al., 2022), **AbDPO** (Zhou et al., 2024) and its variant **AbDPO++**). Following previous works (Ye et al., 2024), we use multiple metrics to evaluate the quality of designed CDRs: `AAR` and C_α `RMSD` for generated sequence and backbone structure; `Total Energy (E)` and `Binding Energy` (ΔG) for atomic rationality and functionality, which are provided by pyRosetta.

Results. As indicated in Table 6, APM performs significantly superior to other methods in all metrics. With respect to accuracy related metrics, including `AAR` and `RMSD`, APM's superior performance demonstrates its capability in generating antibodies that resemble natives. On rational and functional metrics, APM's generated antibodies exhibit the highest rationality and binding capability. Additionally, antibodies generated in a zero-shot manner display excellent ΔG, proving APM's capacity in inter-chain interacted protein generation, while the abnormal `AAR`/`RMSD` highlight the different binding patterns between general proteins and antibodies (refer to Appendix D.4 for details).

4.4.2 Peptide Design

Setup. The design of functional and binding peptides plays a crucial role in pharmacological applications and targeted therapeutic development. To evaluate our APM's perfor-

Table 5. Comprehensive evaluation of peptide design methods across three key aspects: Functionality, Foldability, and Accuracy. The best results are highlighted in **bold**. 5 out of 93 ground truth samples exhibit ΔG greater than 0, are visualized in Appendix D.5.

Method	Functionality		Foldability			Accuracy	
	ΔG ↓	% <0 ↑	pLDDT ↑	ipTM ↑	Success ↑	DockQ ↑	% ≥0.8 ↑
GroundTruth	-24.54	*94.62*	88.31	0.94	100.00%	1.00	100.00
PPFlow	-8.56	16.72	55.72	0.57	13.01%	0.27	0.00
DiffPP	-12.40	38.17	55.10	0.57	16.55%	0.33	0.90
PepGLAD	-12.45	37.10	51.69	0.57	12.50%	0.35	0.00
RFDiffusion	-23.27	**78.58**	**69.65**	**0.73**	**46.28%**	0.28	0.00
APM_{SFT}	-19.90	69.34	60.36	0.66	29.22%	**0.40**	**11.29**
$APM_{zero\text{-}shot}$	**-23.71**	62.18	60.97	0.62	27.20%	0.24	0.12

Table 6. Performance comparison of antibody design methods on RAbD benchmark. The best results are shown in **bold**.

Method	AAR (%) ↑	RMSD ↓	E ↓	ΔG ↓
RAbD	100.00	0.00	-16.76	-15.33
dyMEAN	40.05	2.36	1239.29	612.75
DiffAb	35.04	2.53	495.69	489.42
AbDPO	31.29	2.79	270.12	116.06
AbDPO++	36.25	2.48	338.14	223.73
APM_{SFT}	**41.20**	**2.08**	**137.74**	91.64
$APM_{zero\text{-}shot}$	28.35	5.81	284.24	**81.12**

mance in receptor-targeted peptide design, we use the **PepBench** (Kong et al., 2024) dataset for training and validation and use the **LNR** (Tsaban et al., 2022) dataset as the test set. We compare our model with: **PepGLAD** (Kong et al., 2024), **PPFlow** (Lin et al., 2024), and **DiffPP** (Lin et al., 2024). We also include **RFDiffusion** (Watson et al., 2023), which utilizes ProteinMPNN for sequence design. The peptide candidates are comprehensively evaluated across three key aspects: **Functionality**, **Foldability**, and **Accuracy**. For functionality, we evaluate the binding energy (ΔG) and the proportion of candidates with ΔG below zero, % <0. For foldability, we use Boltz-1(wMSA) to fold sequences of generated peptides, then evaluate the folded structure in two confidence metrics: predicted Local Distance Difference Test (`pLDDT`) and interface predicted Template Modeling (`ipTM`) (Zhang & Skolnick, 2004; Xu & Zhang, 2010) score. We also report a comprehensive metric, defined as the proportion of candidates with both a `pLDDT` score $\geq$ 70 and an `ipTM` score $\geq$ 0.8, donated as Success. For accuracy, we evaluate the `DockQ` (Basu & Wallner, 2016; Mirabello & Wallner, 2024) score and the proportion of candidates achieving a `DockQ` score of at least 0.8 (% $\geq$0.8), which is the threshold considered as high-quality.

Results. As indicated in Table 5, APM exhibits competitive performance across all three key aspects. For functionality, APM generates peptides with an average binding energy of -19.90 and achieves negative ΔG in 69.34% samples, significantly outperforming other methods. The performance in foldability metrics demonstrates APM's superiority in sequences generation, with the high `pLDDT` and `ipTM`. For accuracy, APM stands out as nearly the only method capable of generating peptides with a `DockQ` score exceeding 0.8. Additionally, the peptides generated by APM in a zero-shot manner perform similarly to antibodies, with high affinity but do not resemble natural ones. This is reflected in the comparable performance in functionality & foldability and the significant degradation in accuracy. We also present more results on longer binder design in Appendix F.

5 Discussions

In this paper, we introduce APM, a generative model for protein complexes designing at all-atom level. APM is capable of generating tightly bound protein complexes, executing high-quality single-chain protein-related tasks, and achieving remarkable performance in specific functional protein design tasks. Despite APM's potential in AI-based functional protein design tasks, several limitations remain to be addressed. These limitations are primarily in these aspects: (1) the performance in folding tasks requires further improvement; (2) the functionality of `Refine` Module is relatively restricted; (3) the number of downstream tasks is limited. Future works are detailed in the Appendix G.

Impact Statement

Our work on multi-chain protein generation can be used in developing potent therapeutic macromolecues such as antibodies and accelerating the research process of drug discovery. Our method may be adapted to other scenarios of computer-aided design, such as small molecule design, material design, and chip design. It is also needed to ensure the responsible use of our method and refrain from using it for harmful purposes.

Acknowledgements

We thank anonymous reviewers for their insightful feedback. We would like to especially thank Dr. Hang Li for insightful discussions on the project and feedback on the manuscript that help shape this study. We thank Yi Zhou, Yuning Shen, Xinyou Wang, Yiming Ma, Fei Ye, and Lihao Wang for their valuable comments.

References

Abramson, J., Adler, J., Dunger, J., Evans, R., Green, T., Pritzel, A., Ronneberger, O., Willmore, L., Ballard, A. J., Bambrick, J., et al. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, 2024.

Adolf-Bryfogle, J., Kalyuzhniy, O., Kubitz, M., Weitzner, B. D., Hu, X., Adachi, Y., Schief, W. R., and Dunbrack Jr, R. L. Rosettaantibodydesign (rabd): A general framework for computational antibody design. *PLoS computational biology*, 2018.

Alamdari, S., Thakkar, N., van den Berg, R., Tenenholtz, N., Strome, B., Moses, A., Lu, A. X., Fusi, N., Amini, A. P., and Yang, K. K. Protein generation with evolutionary diffusion: sequence is all you need. *BioRxiv*, 2023.

Alford, R. F., Leaver-Fay, A., Jeliazkov, J. R., O'Meara, M. J., DiMaio, F. P., Park, H., Shapovalov, M. V., Renfrew, P. D., Mulligan, V. K., Kappel, K., et al. The rosetta all-atom energy function for macromolecular modeling and design. *Journal of chemical theory and computation*, 2017.

Baek, M., DiMaio, F., Anishchenko, I., Dauparas, J., Ovchinnikov, S., Lee, G. R., Wang, J., Cong, Q., Kinch, L. N., Schaeffer, R. D., et al. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 2021.

Basu, S. and Wallner, B. Dockq: a quality measure for protein-protein docking models. *PloS one*, 2016.

Bennett, N. R., Watson, J. L., Ragotte, R. J., Borst, A. J., See, D. L., Weidle, C., Biswas, R., Yu, Y., Shrock, E. L., Ault, R., et al. Atomically accurate de novo design of antibodies with rfdiffusion. *bioRxiv*, pp. 2024–03, 2025.

Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N., and Bourne, P. E. The protein data bank. *Nucleic acids research*, 2000.

Boeckmann, B., Bairoch, A., Apweiler, R., Blatter, M.-C., Estreicher, A., Gasteiger, E., Martin, M. J., Michoud, K., O'Donovan, C., Phan, I., et al. The swiss-prot protein knowledgebase and its supplement trembl in 2003. *Nucleic acids research*, 2003.

Bose, J., Akhound-Sadegh, T., Huguet, G., FATRAS, K., Rector-Brooks, J., Liu, C.-H., Nica, A. C., Korablyov, M., Bronstein, M. M., and Tong, A. Se (3)-stochastic flow matching for protein backbone generation. In *The Twelfth International Conference on Learning Representations*, 2023.

Campbell, A., Yim, J., Barzilay, R., Rainforth, T., and Jaakkola, T. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. In *ICML*, 2024.

Chaudhury, S., Lyskov, S., and Gray, J. J. Pyrosetta: a script-based interface for implementing molecular modeling algorithms using rosetta. *Bioinformatics*, 2010.

Chen, R. T. Q. and Lipman, Y. Flow matching on general geometries. In *ICLR*, 2024.

Chu, A. E., Kim, J., Cheng, L., El Nesr, G., Xu, M., Shuai, R. W., and Huang, P.-S. An all-atom protein generative model. *Proceedings of the National Academy of Sciences*, 2024a.

Chu, A. E., Kim, J., Cheng, L., El Nesr, G., Xu, M., Shuai, R. W., and Huang, P.-S. An all-atom protein generative model. *Proceedings of the National Academy of Sciences*, 2024b.

Dao, T., Fu, D. Y., Ermon, S., Rudra, A., and Ré, C. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

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

Dunbar, J., Krawczyk, K., Leem, J., Baker, T., Fuchs, A., Georges, G., Shi, J., and Deane, C. M. Sabdab: the structural antibody database. *Nucleic acids research*, 2014.

Eastman, P., Swails, J., Chodera, J. D., McGibbon, R. T., Zhao, Y., Beauchamp, K. A., Wang, L.-P., Simmonett, A. C., Harrigan, M. P., Stern, C. D., et al. Openmm 7: Rapid development of high performance algorithms for molecular dynamics. *PLoS computational biology*, 13(7): e1005659, 2017.

Engh, R. and Huber, R. Structure quality and target parameters. 2006.

Evans, R., O'Neill, M., Pritzel, A., Antropova, N., Senior, A., Green, T., Žídek, A., Bates, R., Blackwell, S., Yim, J., et al. Protein complex prediction with alphafold-multimer. *biorxiv*, 2021.

Fred Zhangzhi Peng, P. C. and contributors. Faesm: An efficient pytorch implementation of evolutionary scale modeling (esm). https://github.com/pengzhangzhi/faesm, 2024. Efficient PyTorch implementation of ESM with FlashAttention and Scalar Dot-Product Attention (SDPA).

Gao, K., Wu, L., Zhu, J., Peng, T., Xia, Y., He, L., Xie, S., Qin, T., Liu, H., He, K., et al. Pre-training antibody language models for antigen-specific computational antibody design. In *KDD*, 2023.

Guo, H.-B., Perminov, A., Bekele, S., Kedziora, G., Farajollahi, S., Varaljay, V., Hinkle, K., Molinero, V., Meister, K., Hung, C., et al. Alphafold2 models indicate that protein sequence determines both structure and dynamics. *Scientific reports*, 12(1):10696, 2022.

Hayes, T., Rao, R., Akin, H., Sofroniew, N. J., Oktay, D., Lin, Z., Verkuil, R., Tran, V. Q., Deaton, J., Wiggert, M., et al. Simulating 500 million years of evolution with a language model. *Science*, 2025.

Henikoff, S. and Henikoff, J. G. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences*, 89(22):10915–10919, 1992.

Hsieh, C.-Y., Wang, X., Zhang, D., Xue, D., Ye, F., Huang, S., Zheng, Z., and Gu, Q. Elucidating the design space of multimodal protein language models. *arXiv preprint arXiv:2504.11454*, 2025. URL https://arxiv.org/abs/2504.11454.

Hsu, C., Verkuil, R., Liu, J., Lin, Z., Hie, B., Sercu, T., Lerer, A., and Rives, A. Learning inverse folding from millions of predicted structures. *ICML*, 2022.

Huguet, G., Vuckovic, J., Fatras, K., Thibodeau-Laufer, E., Lemos, P., Islam, R., Liu, C.-H., Rector-Brooks, J., Akhound-Sadegh, T., Bronstein, M., et al. Sequence-augmented se (3)-flow matching for conditional protein backbone generation. *Advances in neural information processing systems*, 2024.

Ingraham, J. B., Baranov, M., Costello, Z., Barber, K. W., Wang, W., Ismail, A., Frappier, V., Lord, D. M., Ng-Thow-Hing, C., Van Vlack, E. R., Tie, S., Xue, V., Cowles, S. C., Leung, A., Rodrigues, J. a. V., Morales-Perez, C. L., Ayoub, A. M., Green, R., Puentes, K., Oplinger, F., Panwar, N. V., Obermeyer, F., Root, A. R., Beam, A. L., Poelwijk, F. J., and Grigoryan, G. Illuminating protein space with a programmable generative model. *Nature*, 2023. doi: 10.1038/s41586-023-06728-8.

Jin, W., Barzilay, D., and Jaakkola, T. Antibody-antigen docking and design via hierarchical structure refinement. In *ICML*, 2022.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. Highly accurate protein structure prediction with alphafold. *nature*, 2021.

Kong, X., Huang, W., and Liu, Y. Conditional antibody design as 3d equivariant graph translation. In *ICLR*, 2023a.

Kong, X., Huang, W., and Liu, Y. End-to-end full-atom antibody design. In *ICML*, 2023b.

Kong, X., Jia, Y., Huang, W., and Liu, Y. Full-atom peptide design with geometric latent diffusion. In *NeurIPS*, 2024.

Krishna, R., Wang, J., Ahern, W., Sturmfels, P., Venkatesh, P., Kalvet, I., Lee, G. R., Morey-Burrows, F. S., Anishchenko, I., Humphreys, I. R., et al. Generalized biomolecular modeling and design with rosettafold all-atom. *Science*, 2024.

Lefranc, M.-P., Pommié, C., Ruiz, M., Giudicelli, V., Foulquier, E., Truong, L., Thouvenin-Contet, V., and Lefranc, G. Imgt unique numbering for immunoglobulin and t cell receptor variable domains and ig superfamily v-like domains. *Developmental & Comparative Immunology*, 2003.

Lehninger, A. L., Nelson, D. L., and Cox, M. M. *Lehninger principles of biochemistry*. Macmillan, 2005.

Li, J., Cheng, C., Wu, Z., Guo, R., Luo, S., Ren, Z., Peng, J., and Ma, J. Full-atom peptide design based on multimodal flow matching. In *ICML*, 2024.

Lin, H., Zhang, O., Zhao, H., Jiang, D., Wu, L., Liu, Z., Huang, Y., and Li, S. Z. Ppflow: Target-aware peptide design with torsional flow matching. In *ICML*, 2024.

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

Lipman, Y., Chen, R. T., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. In *ICLR*, 2023.

Lisanza, S. L., Gershon, J. M., Tipps, S. W., Sims, J. N., Arnoldt, L., Hendel, S. J., Simma, M. K., Liu, G., Yase, M., Wu, H., et al. Multistate and functional protein design using rosettafold sequence space diffusion. *Nature biotechnology*, 2024.

Luo, S., Su, Y., Peng, X., Wang, S., Peng, J., and Ma, J. Antigen-specific antibody design and optimization with diffusion-based generative models for protein structures. *NeurIPS*, 2022.

Madani, A., McCann, B., Naik, N., Keskar, N. S., Anand, N., Eguchi, R. R., Huang, P.-S., and Socher, R. Progen: Language modeling for protein generation. *arXiv preprint arXiv:2004.03497*, 2020.

Mariani, V., Biasini, M., Barbato, A., and Schwede, T. lddt: a local superposition-free score for comparing protein structures and models using distance difference tests. *Bioinformatics*, 2013.

Martinkus, K., Ludwiczak, J., Liang, W.-C., Lafrance-Vanasse, J., Hotzel, I., Rajpal, A., Wu, Y., Cho, K., Bonneau, R., Gligorijevic, V., et al. Abdiffuser: full-atom generation of in-vitro functioning antibodies. *NeurIPS*, 2024.

McPartlon, M. and Xu, J. An end-to-end deep learning method for protein side-chain packing and inverse folding. *Proceedings of the National Academy of Sciences*, 120 (23):e2216438120, 2023.

Meng, E. C., Goddard, T. D., Pettersen, E. F., Couch, G. S., Pearson, Z. J., Morris, J. H., and Ferrin, T. E. Ucsf chimerax: Tools for structure building and analysis. *Protein Science*, 2023.

Mirabello, C. and Wallner, B. Dockq v2: Improved automatic quality measure for protein multimers, nucleic acids, and small molecules. *Bioinformatics*, 2024.

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

Notin, P., Rollins, N., Gal, Y., Sander, C., and Marks, D. Machine learning for functional protein design. *Nature biotechnology*, 2024.

Qu, W., Guan, J., Ma, R., Zhai, K., Wu, W., and Wang, H. P (all-atom) is unlocking new path for protein design. *bioRxiv*, 2024.

Rafailov, R., Sharma, A., Mitchell, E., Manning, C. D., Ermon, S., and Finn, C. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36: 53728–53741, 2023.

Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., Guo, D., Ott, M., Zitnick, C. L., Ma, J., and Fergus, R. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *PNAS*, 2019. doi: 10.1101/622803. URL https://www.biorxiv.org/content/10.1101/622803v4.

Rose, P. W., Prlić, A., Altunkaya, A., Bi, C., Bradley, A. R., Christie, C. H., Costanzo, L. D., Duarte, J. M., Dutta, S., Feng, Z., et al. The rcsb protein data bank: integrative view of protein, gene and 3d structural information. *Nucleic acids research*, 2016.

Sehnal, D., Bittrich, S., Deshpande, M., Svobodová, R., Berka, K., Bazgier, V., Velankar, S., Burley, S. K., Koča, J., and Rose, A. S. Mol* Viewer: modern web app for 3D visualization and analysis of large biomolecular structures. *Nucleic Acids Research*, 49(W1):W431–W437, 05 2021. ISSN 0305-1048. doi: 10.1093/nar/gkab314. URL https://doi.org/10.1093/nar/gkab314.

Senior, A. W., Evans, R., Jumper, J., Kirkpatrick, J., Sifre, L., Green, T., Qin, C., Žídek, A., Nelson, A. W., Bridgland, A., et al. Improved protein structure prediction using potentials from deep learning. *Nature*, 577(7792): 706–710, 2020.

Shi, C., Wang, C., Lu, J., Zhong, B., and Tang, J. Protein sequence and structure co-design with equivariant translation. In *ICLR*, 2023.

Song, Y. and Dhariwal, P. Improved techniques for training consistency models. In *ICLR*, 2024.

Song, Z., Zhao, Y., Shi, W., Jin, W., Yang, Y., and Li, L. Generative enzyme design guided by functionally important sites and small-molecule substrates. *arXiv preprint arXiv:2405.08205*, 2024.

Steinegger, M. and Söding, J. Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature biotechnology*, 2017.

Su, J., Han, C., Zhou, Y., Shan, J., Zhou, X., and Yuan, F. Saprot: Protein language modeling with structure-aware vocabulary. In *ICLR*, 2024a.

Su, J., Li, Z., Han, C., Zhou, Y., He, Y., Shan, J., Zhou, X., Chang, X., Jiang, S., Ma, D., et al. Saprothub: Making protein modeling accessible to all biologists. *bioRxiv*, 2024b.

Su, J., Zhou, X., Zhang, X., and Yuan, F. Protrek: Navigating the protein universe through tri-modal contrastive learning. *bioRxiv*, 2024c.

Trippe, B. L., Yim, J., Tischer, D., Baker, D., Broderick, T., Barzilay, R., and Jaakkola, T. Diffusion probabilistic modeling of protein backbones in 3d for the motif-scaffolding problem. *arXiv preprint arXiv:2206.04119*, 2022.

Tsaban, T., Varga, J. K., Avraham, O., Ben-Aharon, Z., Khramushin, A., and Schueler-Furman, O. Harnessing protein folding neural networks for peptide–protein docking. *Nature communications*, 2022.

Varadi, M., Anyango, S., Deshpande, M., Nair, S., Natassia, C., Yordanova, G., Yuan, D., Stroe, O., Wood, G., Laydon, A., et al. Alphafold protein structure database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic acids research*, 2022.

Wallace, B., Dang, M., Rafailov, R., Zhou, L., Lou, A., Purushwalkam, S., Ermon, S., Xiong, C., Joty, S., and Naik, N. Diffusion model alignment using direct preference optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8228–8238, 2024.

Wang, X., Zheng, Z., Ye, F., Xue, D., Huang, S., and Gu, Q. Diffusion language models are versatile protein learners. In *ICML*, 2024a.

Wang, X., Zheng, Z., Ye, F., Xue, D., Huang, S., and Gu, Q. Dplm-2: A multimodal diffusion protein language model. *arXiv preprint arXiv:2410.13782*, 2024b.

Watson, J. L., Juergens, D., Bennett, N. R., Trippe, B. L., Yim, J., Eisenach, H. E., Ahern, W., Borst, A. J., Ragotte, R. J., Milles, L. F., et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976): 1089–1100, 2023.

Wohlwend, J., Corso, G., Passaro, S., Reveiz, M., Leidal, K., Swiderski, W., Portnoi, T., Chinn, I., Silterra, J., Jaakkola, T., and Barzilay, R. Boltz-1: Democratizing biomolecular interaction modeling. *bioRxiv*, 2024.

Wu, F. and Li, S. Z. A hierarchical training paradigm for antibody structure-sequence co-design. *NeurIPS*, 2024.

Xu, J. and Zhang, Y. How significant is a protein structure similarity with tm-score= 0.5? *Bioinformatics*, 2010.

Ye, F., Zheng, Z., Xue, D., Shen, Y., Wang, L., Ma, Y., Wang, Y., Wang, X., Zhou, X., and Gu, Q. Proteinbench: A holistic evaluation of protein foundation models. *arXiv preprint arXiv:2409.06744*, 2024.

Yim, J., Campbell, A., Foong, A. Y., Gastegger, M., Jiménez-Luna, J., Lewis, S., Satorras, V. G., Veeling, B. S., Barzilay, R., Jaakkola, T., et al. Fast protein backbone generation with se (3) flow matching. *arXiv preprint arXiv:2310.05297*, 2023a.

Yim, J., Campbell, A., Foong, A. Y., Gastegger, M., Jiménez-Luna, J., Lewis, S., Satorras, V. G., Veeling, B. S., Barzilay, R., Jaakkola, T., et al. Fast protein backbone generation with se (3) flow matching. *arXiv preprint arXiv:2310.05297*, 2023b.

Yim, J., Campbell, A., Mathieu, E., Foong, A. Y. K., Gastegger, M., Jimenez-Luna, J., Lewis, S., Satorras, V. G., Veeling, B. S., Noe, F., Barzilay, R., and Jaakkola, T. Improved motif-scaffolding with SE(3) flow matching. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=fa1ne8xDGn.

Zambaldi, V., La, D., Chu, A. E., Patani, H., Danson, A. E., Kwan, T. O., Frerix, T., Schneider, R. G., Saxton, D., Thillaisundaram, A., et al. De novo design of high-affinity protein binders with alphaproteo. *arXiv preprint arXiv:2409.08022*, 2024.

Zhang, Y. and Skolnick, J. Scoring function for automated assessment of protein structure template quality. *Proteins: Structure, Function, and Bioinformatics*, 2004.

Zhang, Y. and Skolnick, J. Tm-align: a protein structure alignment algorithm based on the tm-score. *Nucleic acids research*, 2005.

Zhang, Y., Zhang, Z., Zhong, B., Misra, S., and Tang, J. Diffpack: A torsional diffusion model for autoregressive protein side-chain packing. *NeurIPS*, 2024.

Zheng, Z., Deng, Y., Xue, D., Zhou, Y., Ye, F., and Gu, Q. Structure-informed language models are protein designers. In *ICML*, 2023.

Zhou, X., Xue, D., Chen, R., Zheng, Z., Wang, L., and Gu, Q. Antigen-specific antibody design via direct energy-based preference optimization. In *NeurIPS*, 2024.

Zhou, X., Li, M., Xiao, Y., Li, J., Xue, D., Zheng, Z., Ma, J., and Gu, Q. Designing cyclic peptides via harmonic sde with atom-bond modeling. In *International Conference on Machine Learning*, 2025.

Zhu, T., Ren, M., and Zhang, H. Antibody design using a score-based diffusion model guided by evolutionary, physical and geometric constraints. In *ICML*, 2024.

Zou, S., Li, H., Mo, S., Cheng, X., Xing, E., and Song, L. Linker-tuning: Optimizing continuous prompts for heterodimeric protein prediction. *arXiv preprint arXiv:2312.01186*, 2023.

A Model

A.1 SE(3) Flow Matching

Flow Matching (FM) (Lipman et al., 2023) offers an efficient framework for learning continuous normalizing flows by directly learning a time-dependent vector field that transforms samples from a prior distribution to a target data distribution, eliminating the need for expensive likelihood evaluations or ODE solving during training.

Consider a prior distribution p_0 and a target distribution p_1. FM learns a time-dependent vector field $\mathbf{v}_t : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d$ that guides the transformation through a continuous-time flow $\psi_t : \mathbb{R}^d \to \mathbb{R}^d$, governed by the ordinary differential equation (ODE):

$$\frac{d}{dt}\psi_t(\mathbf{x}) = \mathbf{v}_t(\psi_t(\mathbf{x})), \quad \psi_0(\mathbf{x}) = \mathbf{x} \tag{1}$$

where samples $\mathbf{x} \sim p_0$ are drawn from the prior distribution and transformed to follow the target distribution p_1 at $t = 1$.

To enable tractable training, FM introduces an *interpolant* $\phi_t(\mathbf{x}_0, \mathbf{x}_1)$ that defines a smooth path between pairs of points $\mathbf{x}_0 \sim p_0$ and $\mathbf{x}_1 \sim p_1$. The conditional vector field $\mathbf{u}_t$ is derived as the time derivative of this interpolant. The conditional flow matching objective then becomes:

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[0,1],\mathbf{x}_0\sim p_0,\mathbf{x}_1\sim p_1}\left[\|\mathbf{v}_\theta(\mathbf{x}_t, t) - \mathbf{u}_t(\mathbf{x}_t|\mathbf{x}_0, \mathbf{x}_1)\|^2\right] \tag{2}$$

where $\mathbf{x}_t = \phi_t(\mathbf{x}_0, \mathbf{x}_1)$. After training, new samples are generated by solving:

$$\frac{d}{dt}\mathbf{x}_t = \mathbf{v}_\theta(\mathbf{x}_t, t), \quad \mathbf{x}_0 \sim p_0 \tag{3}$$

When applying this framework to SE(3), we need to consider both translations and rotations. For translations in $\mathbb{R}^3$, we employ linear interpolation:

$$\phi_t^{\text{trans}}(\mathbf{x}_0^i, \mathbf{x}_1^i) = (1-t)\mathbf{x}_0^i + t\mathbf{x}_1^i, \quad \mathbf{x}_0^i, \mathbf{x}_1^i \in \mathbb{R}^3 \tag{4}$$

with the corresponding conditional vector field:

$$\mathbf{u}_t^{\text{trans}}(\mathbf{x}_t^i|\mathbf{x}_0^i, \mathbf{x}_1^i) = \mathbf{x}_1^i - \mathbf{x}_0^i = \frac{\mathbf{x}_1^i - \mathbf{x}_t^i}{1-t} \tag{5}$$

For rotations in SO(3), following (Chen & Lipman, 2024; Campbell et al., 2024; Yim et al., 2023a), we utilize geodesic interpolation on the manifold. During training, we use a linear schedule:

$$\phi_t^{\text{rot}}(R_0^i, R_1^i) = \exp_{R_0^i}(t \cdot \log_{R_0^i}(R_1^i)), \quad R_0^i, R_1^i \in \text{SO}(3) \tag{6}$$

where $\exp(\cdot)$ and $\log(\cdot)$ denote the exponential and logarithm maps on SO(3). During inference, we employ an exponential schedule $\kappa(t) = e^{-ct}$ with $c = 10$:

$$\phi_t^{\text{rot}}(R_0^i, R_1^i) = \exp_{R_0^i}((1-e^{-ct})\log_{R_0^i}(R_1^i)) \tag{7}$$

The conditional vector field in the tangent space $T_{R_t}\text{SO}(3)$ takes different forms during training and inference. During training, following the linear schedule, it is given by:

$$\mathbf{u}_t^{\text{rot}}(R_t^i|R_0^i, R_1^i) = \frac{\log_{R_t^i}(R_1^i)}{1-t} \tag{8}$$

while during inference, with the exponential schedule, it becomes:

$$\mathbf{u}_t^{\text{rot}}(R_t^i|R_0^i, R_1^i) = c\log_{R_t^i}(R_1^i) \tag{9}$$

For the choice of distributions, we consider the geometric properties of SO(3). We use the uniform distribution over SO(3) during training and sampling.

For training, we define separate loss terms that jointly guide the learning of the vector field. The translation loss follows the Euclidean Flow Matching objective:

$$\mathcal{L}_{\text{trans}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[0,1],\mathbf{x}_0,\mathbf{x}_1}\left[\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{v}_\theta^{\text{trans}}(\mathbf{x}_t^i,t) - \mathbf{u}_t^{\text{trans}}(\mathbf{x}_t^i|\mathbf{x}_0^i,\mathbf{x}_1^i)\|^2\right] \tag{10}$$

For rotations, following the Riemannian geometry of $\mathrm{SO}(3)$, we define the loss using the geodesic distance on the Lie algebra:

$$\mathcal{L}_{\text{rot}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[0,1],R_0,R_1}\left[\frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{v}_\theta^{\text{rot}}(R_t^i,t) - \mathbf{u}_t^{\text{rot}}(R_t^i|R_0^i,R_1^i)\right\|_{\text{SO}(3)}^2\right] \tag{11}$$

where $R_t^i = \phi_t^{\text{rot}}(R_0^i, R_1^i)$ represents the interpolated rotation at time t, and $\mathbf{v}_\theta^{\text{rot}}(R_t^i,t) \in \mathfrak{so}(3)$ is the predicted velocity in the Lie algebra. The complete $\mathrm{SE}(3)$ Flow Matching objective combines both terms:

$$\mathcal{L}_{\text{SE}(3)}(\theta) = \mathcal{L}_{\text{trans}}(\theta) + \mathcal{L}_{\text{rot}}(\theta) \tag{12}$$

During inference, we solve the ODE using the exponential schedule for rotations while maintaining the linear schedule for translations.

A.2 Discrete Flow Matching

While continuous Flow Matching effectively handles continuous data in $\mathbb{R}^3$ and $\mathrm{SO}(3)$, discrete data such as amino acid sequences require a different approach. We adopt Discrete Flow Matching (Campbell et al., 2024), and define a path from a masked token distribution to the data distribution. Let $\mathbf{S}_1 = [S_1^1, ..., S_1^N]$ be a sequence from the data distribution, and M denote the mask token. The interpolant between $\mathbf{S}_1$ and the fully masked sequence is a categorical distribution defined via the Kronecker delta. We define the i-th token $\mathbf{S}_t^i$ at intermediate time t as:

$$p_{t|1}(\mathbf{S}_t^i|\mathbf{S}_1^i) = t\delta\{\mathbf{S}_t^i, \mathbf{S}_1^i\} + (1-t)\delta\{\mathbf{S}_t^i, M\} \tag{13}$$

where $\delta\{a,b\} = 1$ if $a = b$ and 0 otherwise. This interpolant linearly mixes the clean sequence and the mask state over time. The loss function is the cross-entropy between the predicted and data distributions, written as:

$$\mathcal{L}_{\text{discrete}}(\theta) = \mathbb{E}_{\substack{t\sim\mathcal{U}[0,1]\\ \mathbf{S}_t\sim p_{t|1}(\cdot|\mathbf{S}_1)}}\left[-\log p_{1|t}^\theta(\mathbf{S}_1|\mathbf{S}_t)\right] \tag{14}$$

where $\mathbf{S}_t \sim p_{t|1}(\cdot|\mathbf{S}_1)$ samples a corrupted sequence at time t using the above conditional interpolant, $p_{1|t}^\theta(\mathbf{S}_1|\mathbf{S}_t)$ is the neural network's predicted distribution given the corrupted sequence $\mathbf{S}_t$.

A.3 Sub-module Architectures

The detailed structure is shown in Figure 2. In `Sidechain` Module, the protein language model is only activated in training phase II, and the protein language model encoding is weighted with a learnable zero-initialized parameter before merging into residue representation.

The resdiue level and pair-residue level information are encoded in 384 and 192 dim in `Seq&BB` Module and `Refine` Module, while the dims are 256 and 128 in `Sidechain` Module. For model size, the overall APM contains 127M parameters, of which `Seq&BB` Module contains 52M, `Sidechain` Module contains 22M, and `Refine` Module contains 54M parameters.

`Refine` Module is initialized to apply zero change to the `Seq&BB` Module generated protein to ensure that `Refine` Module does not undergo negative optimization.

B Training Details

B.1 Loss for `Seq&BB` Module

Flow-matching Loss. The detailed flow-matching loss, $\mathcal{L}_{\text{flow-matching}}$, refers to Appendix A.1 and Appendix A.2.

Consistency Loss. For arbitrary t for each modality (t_S for sequence, t_T for backbone structure), we can get the corresponding noised data ($\mathbf{S}_{t_S}$ means noised sequence with t_S, $\mathbf{T}_{t_T}$ means noised structure with t_T) and predict the final sample by the `Seq&BB` Module in APM, $\text{APM}_{\text{Seq\&BB}}$:

$$\hat{L}_1, \hat{\mathbf{T}}_1 = \text{APM}_{\text{Seq\&BB}}(\mathbf{S}_{t_S}, \mathbf{T}_{t_T}, t_S, t_T) \tag{15}$$

We can also get the noised data from the adjacent t, $S_{t_S+\Delta t}$ and $T_{t_T+\Delta t}$, and predict the final sample:

$$\hat{L}_1', \hat{\mathbf{T}}_1' = \text{APM}_{\text{Seq\&BB}}(\mathbf{S}_{t_S+\Delta t}, \mathbf{T}_{t_T+\Delta t}, t_S+\Delta t, t_T+\Delta t) \tag{16}$$

Consistency loss is defined as the gap between the two predictions on the two modalities. Since the predictions from the t closer to 1 are more accurate, we set the predictions from $t+\Delta t$ as a teacher. For sequence, the gap is the KL divergence between the two predicted logits:

$$\mathcal{L}_{\text{consistency\_S}} = \text{KL}\Big(\log\big(\text{softmax}(\hat{L}_1)\big), \log\big(\text{softmax}(\hat{L}_1').\text{detach}()\big)\Big) \tag{17}$$

For structure, the gap is the MSE between the two predicted structures:

$$\mathcal{L}_{\text{consistency\_T}} = \text{MSE}\Big(\text{trans}(\hat{\mathbf{T}}_1), \text{trans}(\hat{\mathbf{T}}_1').\text{detach}()\Big) + \text{MSE}\Big(\text{Mat2Vec}\big(\text{rot}(\hat{\mathbf{T}}_1)\big), \text{Mat2Vec}\big(\text{rot}(\hat{\mathbf{T}}_1')\big).\text{detach}()\Big) \tag{18}$$

Considering that the quality of the APM's predictions is not high when t approaches 0, it is unreasonable to demand consistency at this point. Thus, we scale the consistency loss with respect to t, reducing the impact of consistency loss on model training when t is small. Additionally, we also followed the construction method for consistency loss proposed by Song & Dhariwal (2024). Finally, the consistency loss used in `Seq&BB` Module is defined as:

$$\mathcal{L}_{\text{consistency}} = t_S^2 \times (\sqrt[2]{\mathcal{L}_{\text{consistency\_S}}{}^2 + c_S^2} - c_S) + t_T^2 \times (\sqrt[2]{\mathcal{L}_{\text{consistency\_T}}{}^2 + c_T^2} - c_T) \tag{19}$$

$$c_S = 0.00054 * \sqrt[2]{\dim_S}, \quad c_T = 0.00054 * \sqrt[2]{\dim_T} \tag{20}$$

B.2 Loss for `Sidechain` Module

We followed AlphaFold2 (Jumper et al., 2021) to build the loss for `Sidechain` Module, $\mathcal{L}_\chi$ and $\mathcal{L}_{\text{FAPE}}$. $\mathcal{L}_\chi$ is built according to `Algorithm 27` in the supplementary information of AlphaFold2 and $\mathcal{L}_{\text{FAPE}}$ is built according to `Algorithm 28`.

B.3 Loss for `Refine` Module

Correction Loss. The correction loss, $\mathcal{L}_{\text{corr}}$, is similar to `Seq&BB` Module's flow-matching loss.

Auxiliary Loss. The auxiliary loss consists of backbone FAPE loss, $\mathcal{L}_{\text{BB-FAPE}}$, and residue distogram prediction loss, $\mathcal{L}_{\text{dist}}$. $\mathcal{L}_{\text{BB-FAPE}}$ is the simplified version of $\mathcal{L}_{\text{FAPE}}$ which only considers the backbone atoms.

$\mathcal{L}_{\text{dist}}$ is the gap between the real residue-pair distance and the generated residue-pair distance. For a protein with the length of N, the distance between any two residues i and j is donated as d_{ij}. The $\mathcal{L}_{\text{dist}}$ is defined as:

$$\mathcal{L}_{\text{dist}} = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} ||d^{ij} - \hat{d}^{ij}||^2 \tag{21}$$

B.4 Training

training phase I. In training phase I, the `Seq&BB` Module was trained on 64×H100 GPUs with 257,000 steps, with a learning rate of 1e-4. The `Sidechain` Module was trained on 8×H100 GPUs, accumulating a total of 836,901 steps, also with a learning rate of 1e-4.

training phase II. In training phase II, APM was trained on 64×H100 GPUs with 235,000 steps . The learning rate for the `Seq&BB` Module is set to 1e-5.

SFT. In the SFT phase, we used 8×H100 GPUs to fine-tune APM for antibody design and peptide design. The SFT phase lasted for 1200 epochs for every task. The learning rate for each module is set to 5e-5 and the training cycle is adjusted to 10-1-1.

Protein Language Model. We used a drop-in replacement for the ESM protein language model implementation named FAESM (Fred Zhangzhi Peng & contributors, 2024). We thank the authors for providing an efficient FlashAttention-based (Dao et al., 2022) implementation, which significantly accelerated the training speed.

C Sidechain Torsion Angles Distribution

We calculated the sidechain torsion angles for each amino acid in 22,281 single-chain proteins sourced from the PDB, with protein lengths ranging from 50 to 20,000. Subsequently, we analyzed the distribution of each sidechain torsion angle for each type of amino acid. As shown in Figure 5, it is evident that the number and distribution of sidechain torsion angles differ among various types of amino acids. There are exceptions, such as phenylalanine (**F**), tyrosine (**Y**), and tryptophan (**W**), which have similar sidechain torsion angle distributions due to their structural similarity. In the BLOSUM62 matrix (Henikoff & Henikoff, 1992), the substitution scores between these three amino acids are positive, indicating that they can be substituted with each other to some extent. Therefore, the sidechain torsion angles retain substantial information about the amino acid types even being noised.

D Experimental Details

D.1 Sequence Sampling

The predicted sequence $\hat{\mathbf{S}}_1$ is sampled from the predicted logits $\hat{L}_1$. If $\text{APM}_{\text{Refine}}$ is activated, the $\hat{L}_1$ comes from two modules, `Seq&BB` Module and `Refine` Module:

$$\hat{L}_1 = \begin{cases} \text{APM}_{\text{Seq\&BB}}(\mathbf{S}_{t_S}, \mathbf{T}_{t_T}, t_S, t_T), & \text{if } t_S < 0.8 \\ 0.8 \times \text{APM}_{\text{Seq\&BB}}(\mathbf{S}_{t_S}, \mathbf{T}_{t_T}, t_S, t_T) + 0.2 \times \text{APM}_{\text{Refine}}(\hat{\mathbf{S}}_1, \hat{\mathbf{T}}_1, t_S, t_T, \hat{\chi}), & \text{if } t_S \geq 0.8 \end{cases} \tag{22}$$

Then, the $\hat{\mathbf{S}}_1$ is sampled following:

$$\hat{S}_1 = \begin{cases} \text{Categorical}(\text{Softmax}(\hat{L}_1/\mathcal{T})), & \text{if } t_S < 0.85 \\ \arg\max(\hat{L}_1), & \text{if } t_S \geq 0.85 \end{cases} \tag{23}$$

where the temperature $\mathcal{T}$ follows an exponential decay schedule:

$$\mathcal{T} = \mathcal{T}_{\max} \times \exp(-\lambda \times t_S) \tag{24}$$

with hyperparameters $\mathcal{T}_{\max} = 30$ and decay rate $\lambda = 30$. When noising the $\hat{S}_1$ for the next step, we sort all the positions with their scores and only keep the amino acid with the top K scores, for a protein with the length of N, $K = \text{int}(t_S \times N)$. The score, $\mathcal{S}^i$, for any position i is defined as :

$$\mathcal{S}^i = \log(\text{Softmax}(\hat{L}_1^i))[\hat{\mathbf{S}}_1^i] + (1 - t_S) \times \log(\log(\mathcal{R} + 10^{-8}) + 10^{-8}), \quad \mathcal{R} \sim \mathcal{N}(0, 1) \tag{25}$$

where $\log(\log(\mathcal{R} + 10^{-8}) + 10^{-8})$ is a random term used to avoid decoding sequences in local optima. We denote the score of the Kth highest as $\mathcal{S}_K$, then the sequence for the next sampling step is defined as:

$$\hat{\mathbf{S}}_{t_S+\Delta t} = \{\hat{\mathbf{S}}^i_{t_S+\Delta t} | i \in [1, N]\}, \qquad \hat{\mathbf{S}}^i_{t_S+\Delta t} = \begin{cases} \hat{\mathbf{S}}^i_1, & \text{if } \mathcal{S}^i \geq \mathcal{S}_K \\ \texttt{[MASK]}, & \text{if } \mathcal{S}^i < \mathcal{S}_K \end{cases} \tag{26}$$

D.2 Statistical Validation on Folding

We conducted folding with APM and ESM3 using 20 random seeds. For RMSD, ESM3 shows a marginally better mean (4.708±0.094 vs 4.828±0.077) with statistical significance ($p < 0.05$). For TM-score, APM achieves better performance (0.856±0.002 vs 0.828±0.002) with statistical significance ($p < 0.05$). The detailed results are shown below with the format of (average±std).

Table 7. Folding performance comparison between ESM3 and APM

Method	RMSD ↓	TM ↑
ESM3 (1.4B)	4.708±0.094	0.828±0.002
APM	4.828±0.077	0.856±0.002

D.3 Multi-Chain Protein

MULTI-CHAIN PROTEIN GENERATION WITHOUT ALL-ATOM

During the phase II of APM training, the loss for the `Seq&BB` Module is computed directly based on its own output, rather than relying on the output from the `Refine` Module and then back-propagating to `Seq&BB` Module. As a result, once the

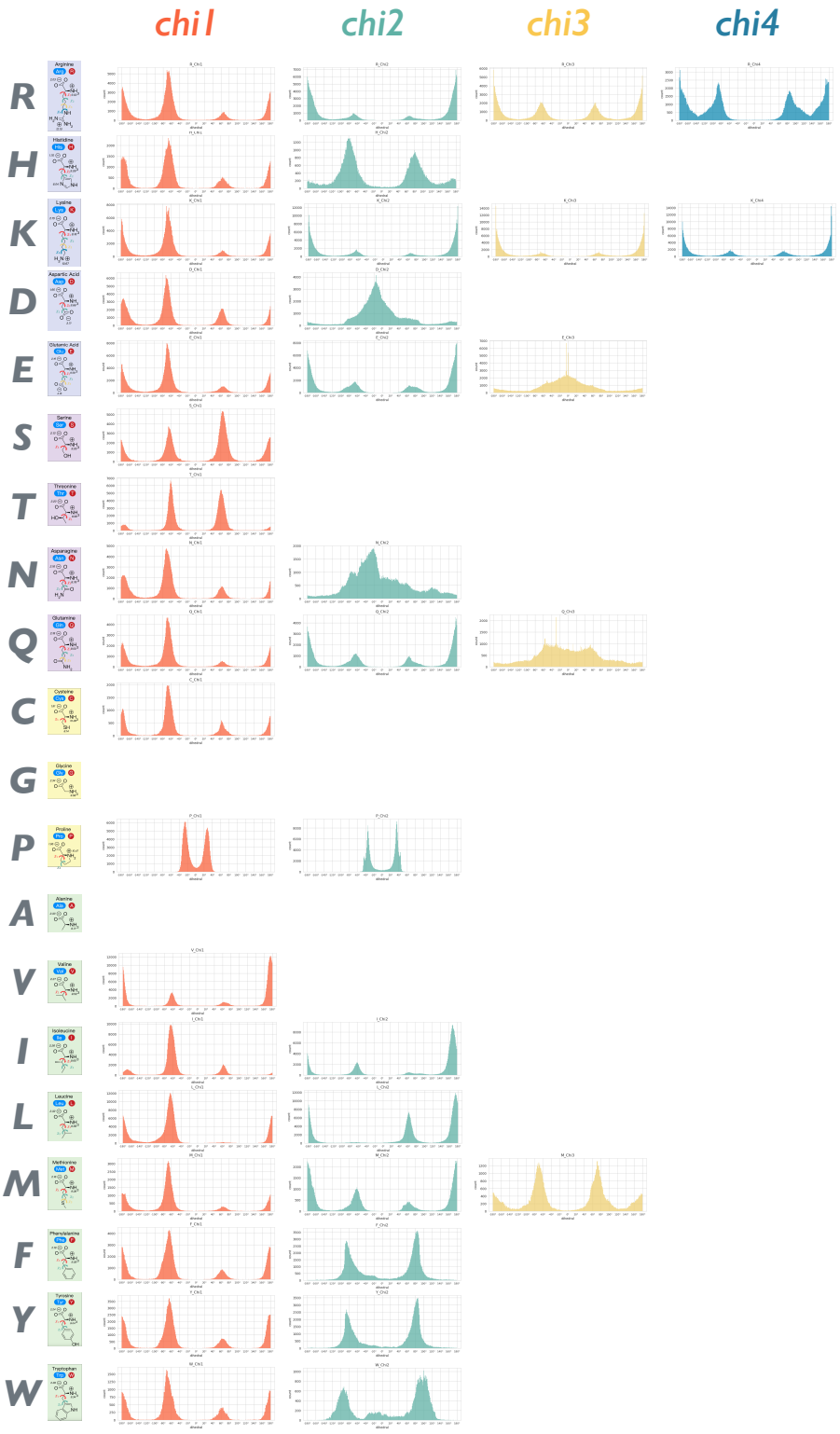

Figure 5. The distribution of four sidechain torsion angles in all amino acid types.

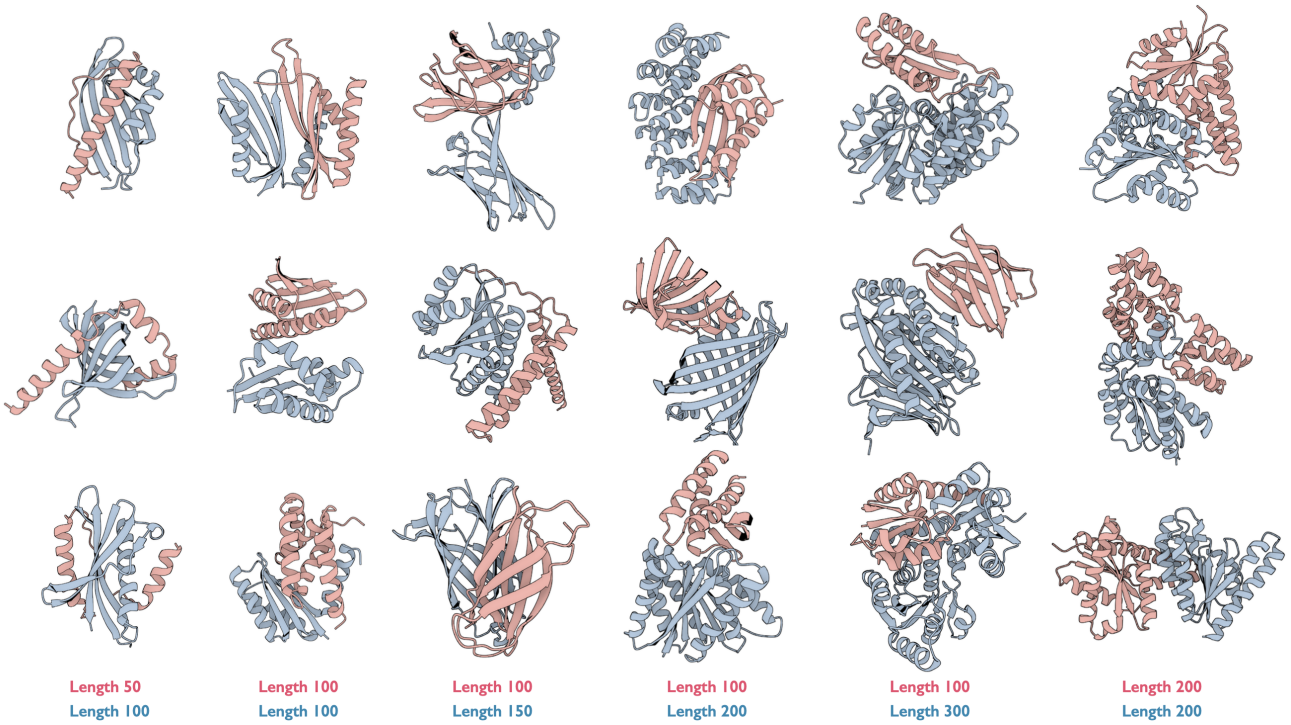

Figure 6. APM generated proteins with chain lengths of 50-100, 100-100, 100-150, 100-200, 100-300, and 200-200.

entire training process is completed, the `Seq&BB` Module can be used independently, allowing for protein generation at the residue level. To verify the importance of all-atom information in multi-chain protein design, we conducted an ablation study by generating multi-chain proteins at the residue level using only the `Seq&BB` Module, and report the results in the main text (Table 4). Although the backbone structure tends to stabilize when the `Sidechain` Module and `Refine` Module are activated during the last 20% of steps, the two different versions of the model still exhibit significant energy differences. When the design is carried out at the residue level, the inter-chain binding strength significantly decreases. Additionally, the difference in ΔG and structure before and after all-atom relaxation becomes more pronounced, indicating that information at the amino acid level alone is insufficient for modeling multi-chain interactions, as opposed to the complete APM. Furthermore, for proteins with lengths of 100-200, without using all-atom information, ΔG of the model-generated structures yields a mean value close to 0 and a median of -33. This suggests the presence of clashes at the binding interfaces, a situation not observed in the complete APM. These findings underscore the importance of all-atom information in modeling inter-chain protein interactions.

More Chain Length Combinations

Length Combinations. In the main text, we have reported the results of multi-chain protein generation with chain length combinations of 50-100, 100-100, and 100-200. Here, we present metrics for additional chain length combinations, including 50-200, 100-150, 100-300, and 200-200 in Table 8. Furthermore, we show more cases of each length combination in Figure 6.

Table 8. ΔG and the `RMSD` before and after all-atom relaxation for more chain length combinations. The trend of affinity change is consistent with that in the main text.

Length	ΔG_{RSC}	ΔG_{RAA}	RMSD
100-150	-63.34/-53.04	-102.74/-90.41	1.28/1.10
50-200	-42.38/-63.71	-117.19/-110.37	1.19/1.10
200-200	-28.41/-49.67	-90.10/-83.55	1.61/1.37
100-300	-23.43/-28.67	-73.03/-60.93	1.78/1.46

Chain Numbers. Theoretically, APM supports the generation of protein complexes composed of any number of chains. However, we observed that APM's performance declines when the generated complex contains more than two chains. This is evident in the increased presence of unstructured backbones or abnormally high ratios of α-helix/β-sheet structures (over

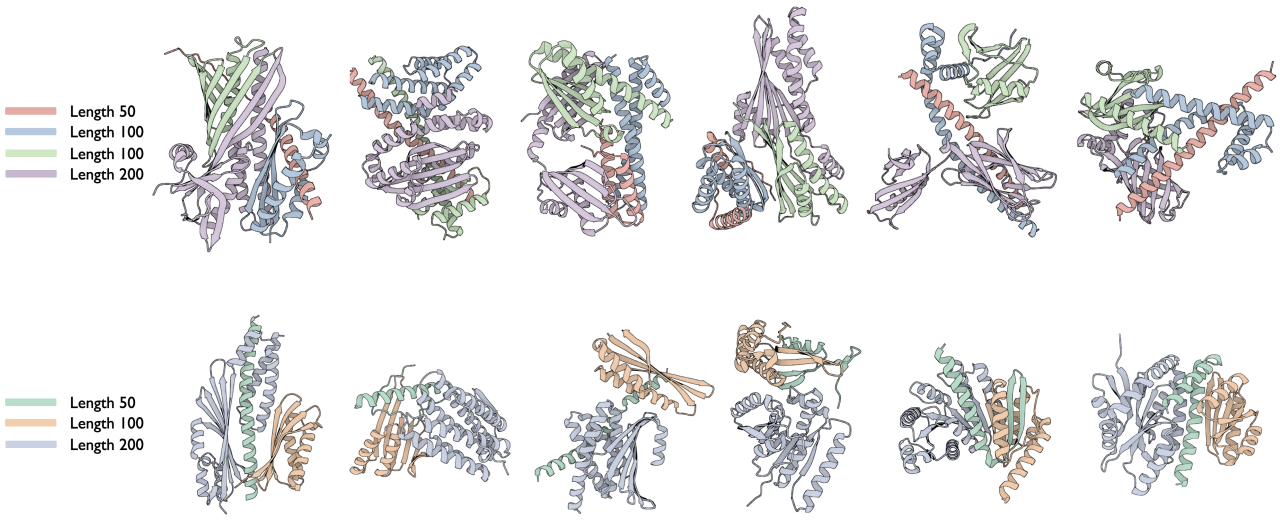

Figure 7. APM generated proteins with more than 2 chains. The top row, generated protein complexes composed of 4 chains; the bottom row, generated protein complexes composed of 3 chains. The length of each chain is highlighted with a unique color.

90% of single secondary structure) in the generated complexes. Interestingly, these structures still exhibit no obvious clashes in the binding interface. We do not apply detailed metric calculations since ΔG only computes the binding strength between two components. Instead, we present some cases for proteins composed of more chains in Figure 7.

We attribute the degradation in performance of APM with more chains to two main reasons: **1)** The majority of the complex samples consist of two chains, and complexes with more chains are significantly less; **2)** With more chains, the overall length of the protein increases, which also results in reduced model generation quality.

CHAIN-BY-CHAIN GENERATION

By default, APM generates all the chains within a multi-chain protein simultaneously. However, we have also implemented an iterative generation manner, called "**chain-by-chain**". APM supports this manner because its training task of performing conditional generation, which means generating the remaining chains based on one or more chains of a multi-chain protein. During the "chain-by-chain" generation, after completing a chain, we need to translate the generated parts. This is necessary because APM requires initialization at the origin when generating structures to meet the requirements of `SO(3)` invariance. Therefore, we need to move the generated parts away from the origin to make space for the next chain to be generated (we also tried not translating the parts, but found that the subsequently generated chains tended to wrap uniformly around the preceding chains).

The location where the generated parts are moved to determines where the binding interfaces of the complex appear, and this process can be manually specified. To avoid bias introduced by the manual specification of binding interfaces, we randomly select an amino acid on the generated part to serve as the binding site (we calculated the distance of each amino acid to the protein's center, then sorted them based on these distances, then we randomly selected from amino acids whose distances are ranked between the 33rd and 66th percentile). Subsequently, we translate the generated parts until the coordinate of the chosen binding site amino acid is at the origin, and continue to translate a small distance with the same direction (default 1Å). APM then begins generating the next chain.

Table 9. ΔG and the `RMSD` before and after all-atom relaxation for generated multi-chain proteins in "chain-by-chain" manner. Both of the two metrics show significant differences.

Length	ΔG_{RSC}	ΔG_{RAA}	RMSD
50-100	312.66/-14.05	-54.46/-54.45	1.64/1.44
100-100	235.52/-13.77	-31.79/-26.71	2.05/1.70
100-200	1073.51/-11.01	-40.29/-46.05	2.47/1.87
50-200	1012.53/-6.84	-10.96/-50.67	2.79/2.12
100-300	323.44/-10.94	-58.24/-53.81	2.42/2.00

APM shows significant differences in generating multi-chain proteins in the chain-by-chain manner compared to generating

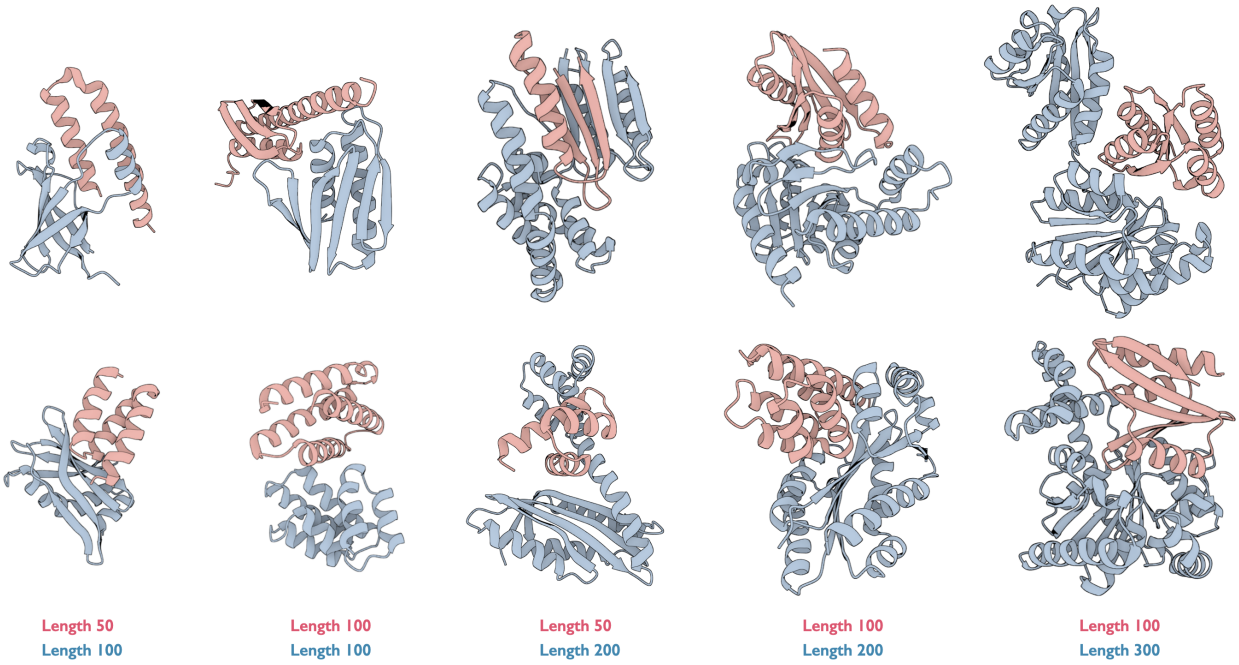

Figure 8. Samples generated by APM in the "chain-by-chain" manner. The generation order is from short chain to long chain, the length of each chain is highlighted with a unique color.

all chains simultaneously. As shown in Table 9, there are three main differences observed in multi-chain proteins generated in the chain-by-chain manner:

- The binding strength between chains shows a considerable decrease;
- The structural differences before and after relaxation become more pronounced;
- Without performing relaxation, structural clashes between chains are observed.

These variations may arise due to several reasons:

- In the chain-by-chain manner, the chains are more independent from another chain, which leads to reduced binding strength;
- The binding interface also impacts binding strength, and a randomly chosen binding interface may not be optimal;
- Once a chain is generated, its structure remains unchanged, which might result in some local structural clashes that the model cannot entirely resolve.

The samples generated in in chain-by-chain manner are shown in Figure 8.

D.4 Antibody Design

Data. Following (Ye et al., 2024; Zhou et al., 2024), we use the Structural Antibody Database (Dunbar et al., 2014) under IMGT (Lefranc et al., 2003) scheme as the dataset. We collected antigen-antibody complexes with both heavy and light chains and protein antigens and discarded the duplicate data with the same CDR-L3 and CDR-H3 sequence. The remaining complexes are used to cluster via MMseqs2 (Steinegger & Söding, 2017) with 40% sequence similarity as the threshold based on the CDR-H3 sequence of each complex. We then select the clusters that do not contain complexes in RAbD benchmark (Adolf-Bryfogle et al., 2018) and split the complexes into training and validation sets with a ratio of 9:1 (1786 and 193 complexes respectively). The test set consists of 55 eligible complexes from the RAbD benchmark.

Methods. We follow the evaluation pipeline and results in Ye et al. (2024). We generate 64 candidates for each antigen. As our method directly generates all-atom structures, no additional sidechain packing tools are required.

Evaluation. Besides the traditional metrics like AAR and RMSD, we mainly focus on the quality in terms of generated CDR-H3's rationality and functionality towards the specific antigens. We utilize pyRosetta to perform the sidechain-only relaxation and calculate energy terms. Baseline methods only design the backbone structure, while energy calculations require the all-atom structure. Therefore, sidechain packing is indispensable. To avoid performance bias introduced by packing methods and conduct a fair comparison, we keep backbone structures fixed while applying relaxation to sidechain

only.

Additionally, we also followed the procedure in the actual experiments, where both sidechain and backbone undergo relaxation before energy calculation, even though the structure at this point differs from the one designed by the model. The results are shown in Table 10, APM still achieves the best performance.

Table 10. Performance comparison of antibody design methods on RAbD benchmark. (↑/↓) indicate whether higher or lower values are better. The best results are shown in **bold**.

Method	E ↓	ΔG ↓
dyMEAN	72.76	36.43
DiffAb	14.56	2.29
APM_{SFT}	**-3.72**	**-4.08**
$\text{APM}_{\text{zero-shot}}$	10.38	-1.77

Zero-shot. We observed that antibodies generated in a zero-shot manner exhibited stronger binding energy compared to those generated using the SFT model. However, there is a significant decrease in performance in terms of accuracy/similarity to natural antibodies. This phenomenon is expected since APM is trained to generate proteins capable of binding to other chains, allowing it to produce binding-capable antibodies without SFT. However, because we excluded all antibody data from the training set, APM generates antibodies following the patterns of general proteins. As a result, the generated CDR-H3, in both sequence and structure, differs from the natural ones, highlighting the differences between antibodies and general proteins. Moreover, as we do not specify binding sites when designing proteins by using APM, the CDR-H3 designed without knowledge of antibody binding patterns might randomly bind antigens or antibody light chains. To further illustrate this phenomenon, we selected some typical samples for visualization, as shown in the Figure 9.

D.5 Peptide Design

Data. The training and evaluation datasets are derived from PepBench (Kong et al., 2024) and LNR (Tsaban et al., 2022). Following Lin et al. (2024); Kong et al. (2024), we extract receptor pockets based on their spatial distances to the peptides. For data structure compatibility, we drop peptide-receptor samples that have non-standard amino acids during preprocessing.

Methods. For baselines, we follow the official open-source code for training and sampling. We directly use the official checkpoint and scripts of PepGLAD[1] to sample candidates. For PPFlow and DiffPP, we carefully use their official code[2] for data preprocessing and follow their training instructions to obtain checkpoints at 200k steps. We also include RFDiffusion as a comparison method. Following the official guidelines[3], we generate peptide structures using 50 diffusion timesteps and employ ProteinMPNN for sequence design. Additionally, we define the 6 amino acids on the receptor that are closest to the peptide as hot-spot residues, as we extract receptor pockets for other methods.

Evaluation. Following Kong et al. (2024), we generate 40 peptide candidates for each sample using all methods. Note that the length of each generated peptide is predefined to match its corresponding ground-truth sequence. We evaluate the binding capabilities of generated peptide candidates from multiple perspectives.

- **Functionality.** We follow the approach used by Kong et al. (2024). Both relaxation and energy calculations are performed using pyRosetta. The evaluation proceeds by identifying the best candidate for each receptor, and reporting the median performance values across all receptors. Note that both the backbone and sidechain are applied relaxation. This procedure could achieve lower binding energies ($\Delta G < 0$) rather than the hundreds in antibody designing. It should be noted that this procedure may fail to achieve perfect relaxation in specific cases. For instance, 5 out of 93 ground truth samples still retained slight clashes, resulting in a ΔG slightly above 0(illustrated in Figure 10).
- **Foldability.** We only fold generated sequences with lengths greater than or equal to 10 residues (46 peptide-receptor pairs in total). As indicated in the documentation of Boltz-1 and AlphaFold3, the `ipTM` scores may not be reliable for sequences that are too short. To ensure reliability, we only consider peptide-receptor pairs where the ground-truth samples achieve successful confidence scores in Boltz-1 (`pLDDT`≥ 70 and `ipTM`≥ 0.8.), filtering out cases where even the ground truth structures fail to meet the confidence threshold. The two threshold values are adopted from the official output documentation[4] of AlphaFold3 and paper (Abramson et al., 2024). Considering the computational cost and MSA retrieval time, we select the top 16 sequences ranked by ΔG and perform folding 8 times for each peptide-receptor pair. For each sequence, we select the best folding result, then average these results across all 16

[1] https://github.com/THUNLP-MT/PepGLAD
[2] https://github.com/EDAPINENUT/ppflow
[3] https://github.com/RosettaCommons/RFdiffusion
[4] https://github.com/google-deepmind/alphafold3/blob/main/docs/output.md

sequences to obtain the final confidence score.

- **Accuracy.** We measure the interface structural accuracy using the DockQ score, which provides quality measures to quantify different aspects between generated and reference structures. Concretely, we utilize DockQv2[5] from their official codebase. Due to extreme structural conflicts observed in some samples generated by baseline methods, we only calculate DockQ score for samples with $\Delta G \leq 0$.

Visualization

- **Generated Structure.** We present the peptides designed by different methods in Figure 11. The showcased examples include peptides with diverse secondary structures (loops, helices, and sheets). As observed in the visualization, our method demonstrates the ability to understand and generate appropriate secondary structures while maintaining reasonable interactions with the receptor.
- **Folded Structure.** As shown in Figure 12, we visualize the folded structures of sequences generated by different methods using Boltz-1. The structures are colored according to the pLDDT confidence, where blue regions indicate high confidence. We borrow the color bar from AlphaFold server website[6]. Receptors are shown in gray with 20% transparency for better visualization. The visualization demonstrates that our method generates sequences capable of folding into stable structures with high confidence scores, indicate the quality of the generated sequences.

E Extended Related Works

Sidechain Prediction. Accurate prediction of sidechain conformation is crucial for protein design. Recent deep learning approaches have significantly advanced this field. DiffPack (Zhang et al., 2024) employs a torsional diffusion model that learns the joint distribution of sidechain torsion angles by diffusing and denoising in torsional space. It autoregressively generates the four torsion angles. AttnPacker (McPartlon & Xu, 2023) directly incorporates backbone 3D geometry to simultaneously compute all sidechain coordinates without relying on discrete rotamer libraries or conformation search.

Motif-Scaffolding. Motif-scaffolding, the design of proteins that incorporate specific functional motifs emerged as a powerful approach in functional protein design. Structure-based methods like RFDiffusion (Watson et al., 2023) and FrameFlow (Yim et al., 2023a) enable the generation of backbone scaffolds that can accommodate predefined motifs while maintaining overall structural stability. Sequence-based approaches including EvoDiff (Alamdari et al., 2023), DPLM (Wang et al., 2024a), and ESM3 (Hayes et al., 2025) present capabilities by designing sequences that fold into structures compatible with functional motifs. These methods collectively provide a comprehensive toolkit for designing proteins with specific functional properties.

Protein Structure Refinement. Structure refinement is essential for optimizing protein designs to achieve native-like stability and function. Physics-based methods such as Rosetta relax (Alford et al., 2017) and OpenMM minimization (Eastman et al., 2017) remain widely used for local refinement of protein structures. These refinement methods play a crucial role in the protein design pipeline, helping to bridge the gap between computational designs and experimentally viable proteins.

F Binder Design

Settings. We further explore APM's zero-shot capabilities in binder design, focusing specifically on longer protein binders rather than short peptides. Following previous works (Watson et al., 2023; Zambaldi et al., 2024), we selected several important targets: Interleukin-7 Receptor-α(IL-7RA), SARS-CoV-2 spike protein receptor-binding domain (SC2RBD), MDM2, Programmed Death-1 (PD1), Programmed Death-Ligand 1 (PD-L1), and CD3-epsilon (CD3E). We extracted the corresponding target chains and reference binders from PDB: 3di3, 6m0j, 1ycr, 4zqk, 4z18, and 1xiw, respectively. The evaluation settings remain consistent with peptide design experiments. However, due to uncertainty about which amino acids should be defined as hot-spot residues for these targets, we did not define hot-spot residues for RFDiffusion nor extract pockets for APM. Consequently, we do not report DockQ for this task. Additionally, due to computational resource constraints, we folded only the top 8 sequences rather than 16 as in previous experiments. The results of average metrics are presented in Table 11, with the 'Success' representing the proportion of samples (out of 8) that achieve both pLDDT > 80 and ipTM > 0.8. We present the generated binders across different methods and their corresponding folded structures with the highest pLDDT sequences in Figure 13.

Discussion. Overall, APM is comparable to RFDiffusion. Although these metrics have been proven by many studies to be predictive of wet lab experimental results (Zambaldi et al., 2024; Bennett et al., 2025), the actual effectiveness still requires validation through wet lab experiments. We observed that both APM and RFdiffusion encounter cases where some samples exhibit lower pLDDT and ipTM scores. The pLDDT score can vary significantly along a protein chain. This means

[5] https://github.com/bjornwallner/DockQ

[6] https://alphafoldserver.com/

Table 11. Performance comparison of binder design. APM_{MPNN} represents using ProteinMPNN to redesign sequences.

Target	Method	ΔG ↓	% <0 ↑	pLDDT	ipTM	Success
3di3	GroundTruth	-23.79	-	95.26	0.85	-
	RFDiffusion	-50.49	82.50%	87.83	0.30	0%
	APM_{MPNN}	-80.09	92.50%	83.39	0.29	12.5%
	APM	-80.10	95.00%	78.91	0.38	12.5%
6m0j	GroundTruth	-20.11	-	81.55	0.15	-
	RFDiffusion	-56.10	67.50%	70.90	0.45	0%
	APM_{MPNN}	-88.99	65.00%	67.82	0.40	12.5%
	APM	-96.47	67.50%	69.50	0.48	12.5%
1ycr	GroundTruth	-25.24	-	90.42	0.93	-
	RFDiffusion	-39.47	100%	78.49	0.81	25.0%
	APM_{MPNN}	-33.27	90.00%	71.10	0.70	50%
	APM	-37.94	90.00%	66.28	0.67	25.0%
4zqk	GroundTruth	-39.36	-	94.03	0.87	-
	RFDiffusion	-29.35	87.50%	75.79	0.39	0%
	APM_{MPNN}	-43.33	77.50%	79.10	0.36	0%
	APM	-45.27	90.00%	80.18	0.39	0%
4z18	GroundTruth	-40.89	-	92.08	0.76	-
	RFDiffusion	-18.69	57.50%	67.39	0.30	0%
	APM_{MPNN}	-63.37	55.00%	74.28	0.34	0%
	APM	-54.24	55.00%	69.28	0.35	0%
1xiw	GroundTruth	-71.69	-	92.64	0.95	-
	RFDiffusion	-56.99	95.00%	77.22	0.76	62.5%
	APM_{MPNN}	-46.96	82.50%	72.08	0.70	12.5%
	APM	-43.25	85.00%	73.27	0.62	12.5%

the folding model can be very confident in the structure of some regions of the protein, but less confident in other regions. We hypothesize that the low pLDDT scores stem from the complexity of long binders. Specifically, certain regions may be naturally highly flexible or intrinsically disordered, leading the folding model to assign low pLDDT scores to these residues (as indicated in (Guo et al., 2022)). Regarding ipTM, we speculate that the lower scores may result from the larger binding interfaces typical of long binders, which often involve multiple contact points or complex features such as convex or polar epitopes, or hydrophobic regions (Zambaldi et al., 2024). These structural complexities and biological properties can contribute to lower ipTM scores.

Future Directions. As suggested in (Zambaldi et al., 2024; Bennett et al., 2025), pLDDT and ipTM are predictive of binding success. We would like to discuss potential approaches to improve long binder design. APM was originally developed as a general-purpose model for complex modeling rather than a task-specific one, which presents challenges in the context of long binder design. This can be reframed as a question of how to adapt a general model into a domain-specialized one. Recent work (Bennett et al., 2025), provides valuable practical directions. The authors successfully transformed RFdiffusion into an antibody-specific model by fine-tuning it on antibody-antigen complex structures, demonstrating that domain-specific data can significantly enhance performance. Similarly, a feasible approach to enhance APM for long binder design would be to use a curated dataset of long binder-target complexes, potentially sourced from PDB or synthetic data. Besides, post-training techniques offer another strategy to optimize the model for generating high-confidence designs. As demonstrated in (Zambaldi et al., 2024; Bennett et al., 2025), pLDDT and ipTM correlate with binding success. Building on this insight, we could implement preference optimization focused on these confidence metrics. Applying DPO-like (Rafailov et al., 2023; Wallace et al., 2024) algorithms, we can then train the model to favor high-confidence designs while avoiding low-confidence ones.

G Future Works

Model Scaling. We chose not to incorporate triangular attention in APM, which is considered a key feature in the success of AlphaFold2/3, because we aim to scale the model in the future to observe whether scaling laws exist in our model. Triangular attention significantly restricts our ability to scale the model size.

Pair Information from PLM. In APM, PLM plays a crucial role by providing the model with a robust understanding of protein sequences. However, we only utilized the representations of individual amino acids from the PLM, and not the pair-level information (pair-level information refers to the attention matrix in PLMs). Pair-level information has been proven

to provide significant benefits for protein structure learning. The reason we did not use pair-level information is to accelerate the encoding process of sequences by PLM (especially in representing multi-chain data). We used a PLM implemented with flash attention, which prevented us from obtaining complete pair-level information. In the future, we will attempt to resolve the encoding speed issue with PLM and use the original PLM implementation to gain the access to pair-level information.

Refine Module. The `Refine` Module is designed to refine the structure and sequence generated by the `Seq&BB` Module, which can be seen as a form of relaxation that allows modifications to the types of amino acids. Currently, our `Refine` Module primarily aims to make the generated proteins resemble real proteins more closely. In the future, we will attempt to incorporate more biological/physical constraints (e.g., force fields) into the `Refine` Module to achieve better performance.

Interchain Hotspot Residue Assignment. Hotspot residues, the amino acids playing crucial roles in interactions, also be considered as regions where interactions occur, serving as important parameters in describing the binding proteins. However, the current version of APM does not support specifying hotspot residues during the generation of complexes or functional proteins. Instead, APM autonomously determines the regions where interactions occur. This design was made to avoid the impact of assigning different hotspot residues on the model's performance during general multi-chain protein generation. Consequently, this design leads to differences in the binding patterns of proteins generated directly using APM (in a zero-shot manner) compared to natural samples. In the future, we will address this issue to allow APM to support the assignment of hotspot residues, thereby enhancing APM's performance in a zero-shot manner.

Downstream Tasks. We will validate APM 's capability in designing proteins with biological functions in more downstream tasks.

H Visualization

All protein visualizations in this paper were completed using ChimeraX (Meng et al., 2023) (Figure 1, Figure 2, Figure 3, Figure 9, Figure 10, Figure 11, Figure 12, Figure 13) and Protein Viewer (Sehnal et al., 2021) (the remaining visualizations).

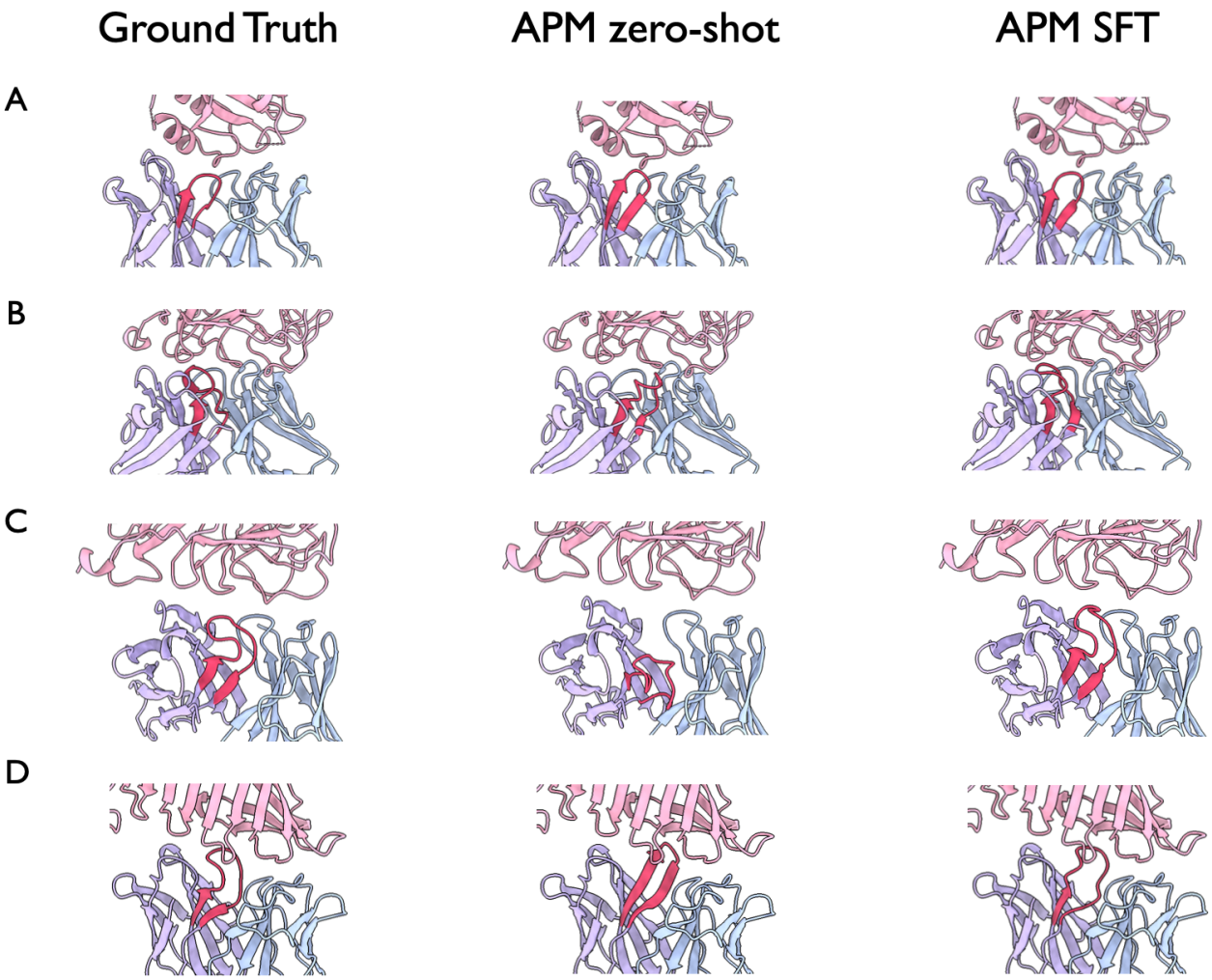

Figure 9. The CDR-H3s generated in a zero-shot manner display distinct patterns compared to natural ones. The antigen is represented in pink, the antibody heavy chain in purple, and the antibody light chain in blue, with the CDR-H3 highlighted in red. The differences between CDR-H3 generated in a zero-shot manner and those from natural antibodies can be categorized as follows: **A**. No obvious difference, the generated CDR-H3 closely resembles that of a natural antibody. **B**. The generated CDR-H3 interacts with the antigen but binds at a different position compared to the reference antibody. **C**. The generated CDR-H3 binds to the antibody light chain. **D**. While the generated CDR-H3 binds correctly with the antigen, its structure predominantly consists of beta-sheets, unlike the common looped structures in natural antibody CDRs. All the aforementioned differences disappear after undergoing SFT. The CDR-H3s generated by APM_{SFT} closely resemble that of the natural ones.

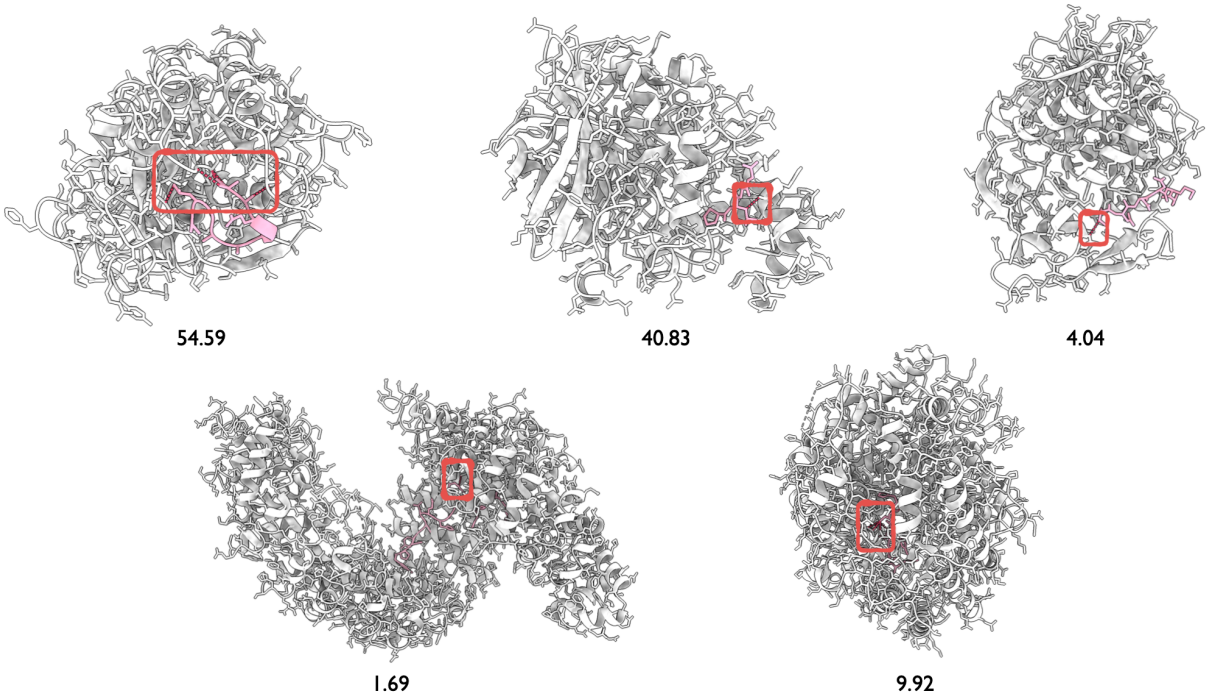

Figure 10. 5 ground truth samples with ΔG exceeding 0. The receptor is shown in white, with the peptide highlighted in red. Slight clashes are marked by red boxes. Additionally, ΔG for each sample are listed below them.

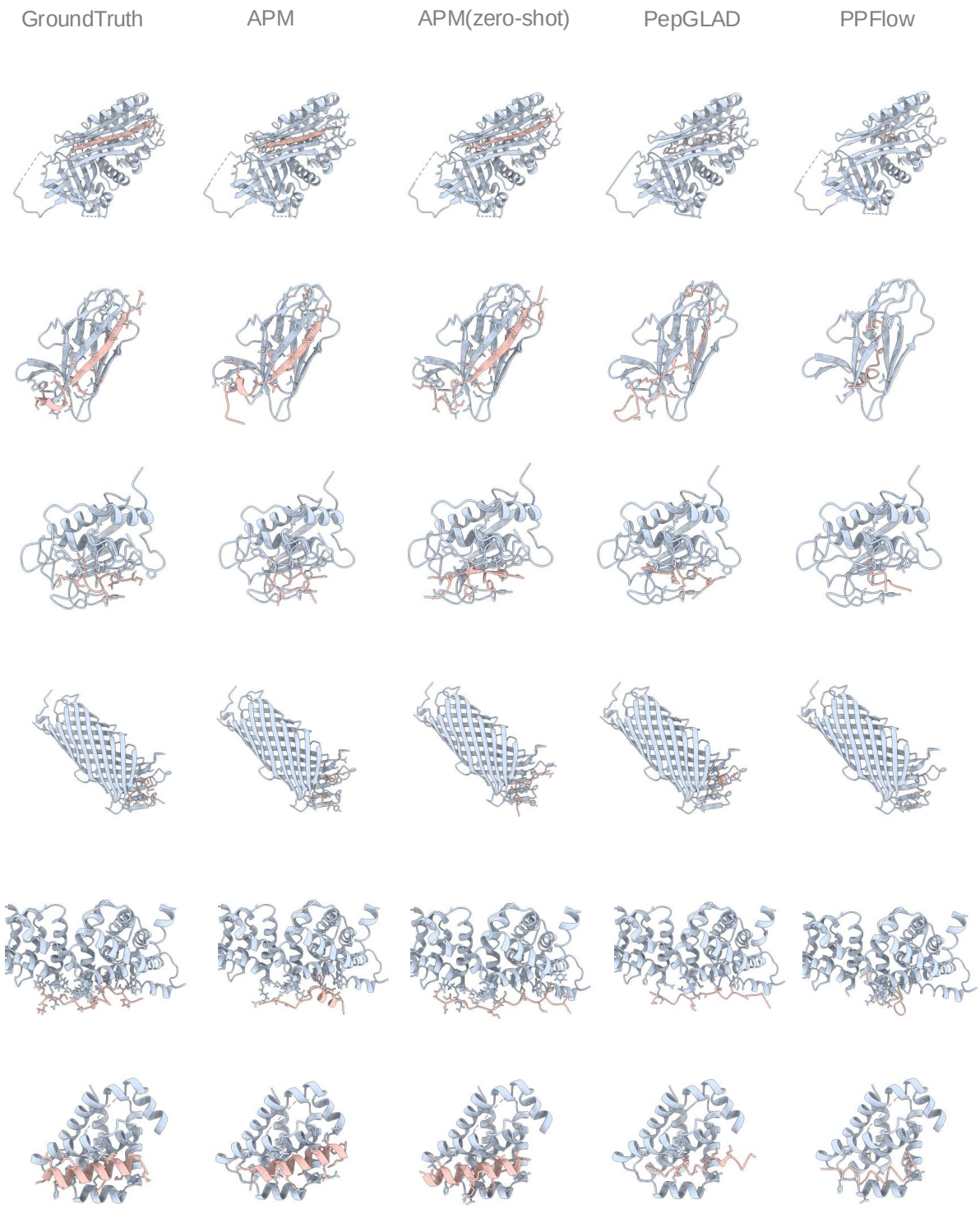

Figure 11. Visualization of peptides generated by different methods. The blue regions represent the given receptors, while the pink regions show the generated peptides, with all-atom structures displayed at the interface regions. From left to right: Ground truth structures, APM, APM zero-shot, PepGLAD, and PPFlow. The PDB IDs for the six cases from top to bottom are: `1jrr`, `2cnz`, `3ayu`, `4dcb`, `5frs`, and `6qg8`.

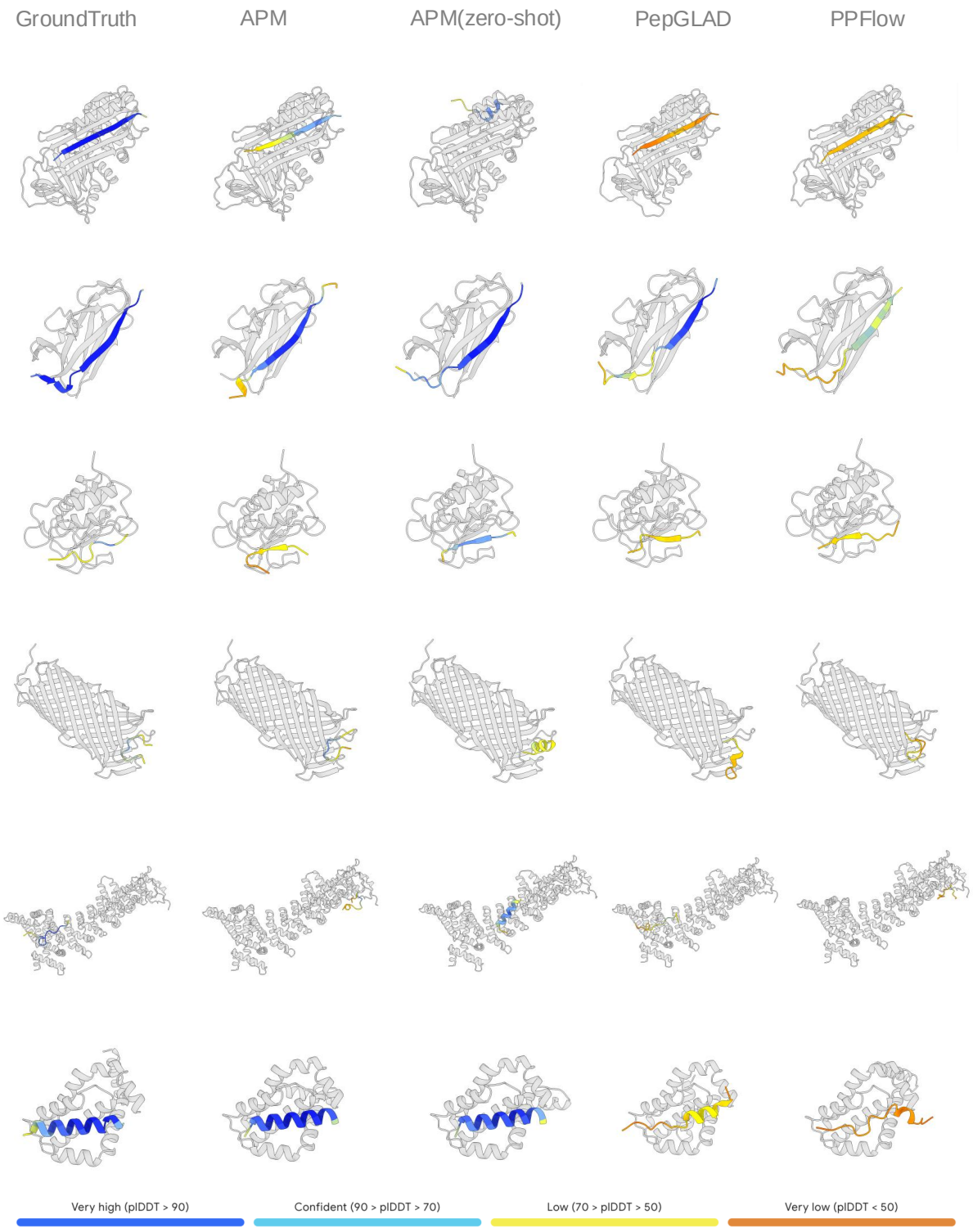

Figure 12. Folded structures of sequences generated by different methods using Boltz-1. The structures are colored according to the AlphaFold-style pLDDT confidence scheme. From left to right: Ground truth structures, APM, APM zero-shot, PepGLAD, and PPFlow. The PDB IDs for the six cases from top to bottom are: `1jrr`, `2cnz`, `3ayu`, `4dcb`, `5frs`, and `6qg8`.

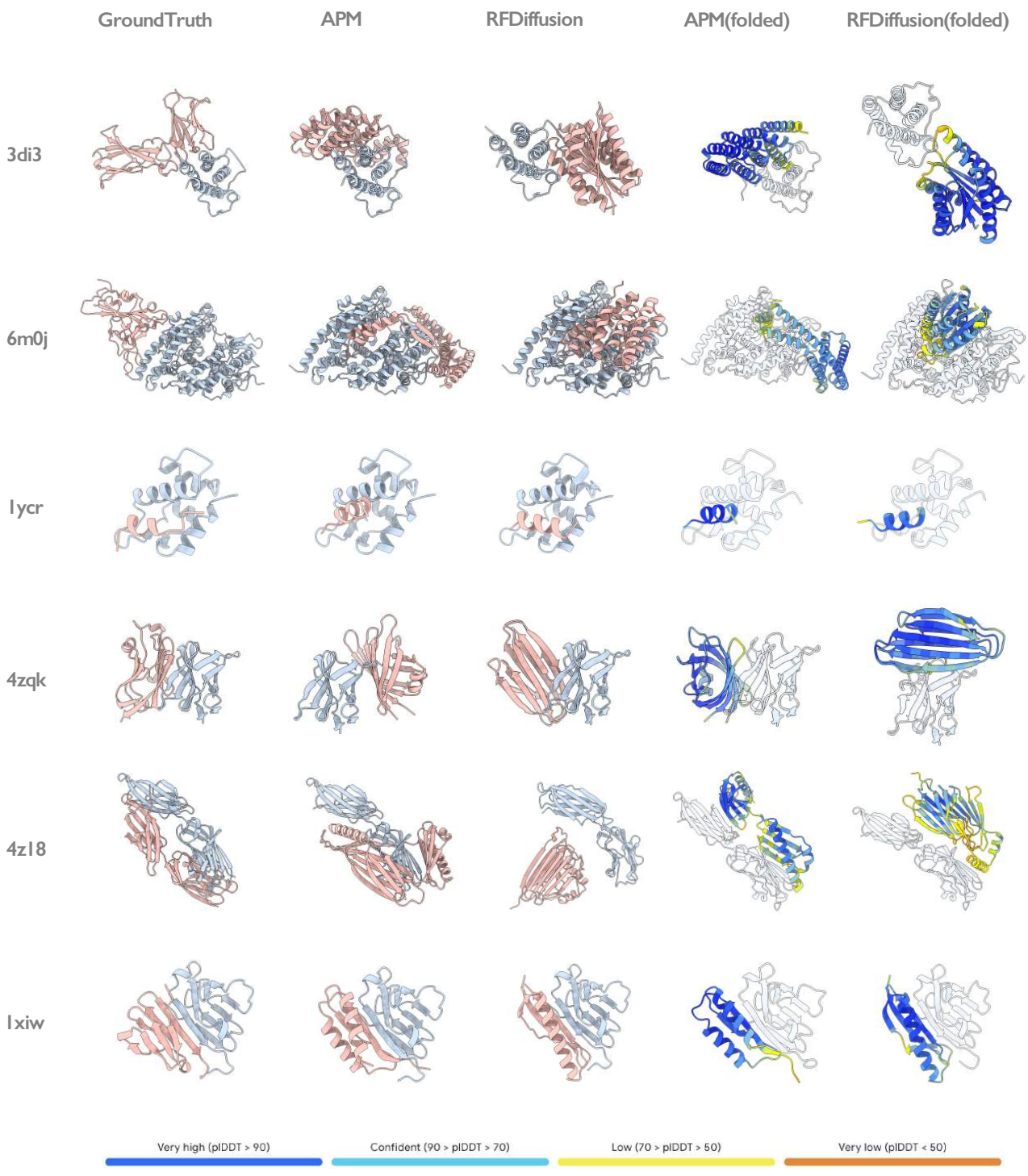

Figure 13. Visualization of binder design results across six protein targets. From left to right: (1) GroundTruth: native complex structures, (2) APM: structures generated by APM, (3) RFDiffusion: structures generated by RFDiffusion, (4) APM(folded): highest pLDDT APM sequences folded with Boltz, (5) RFDiffusion(folded): highest pLDDT sequences folded with Boltz. The blue regions represent the given targets, while the pink regions show the generated binders. For folded structures, targets are rendered transparent to highlight binders, with binders colored according to AlphaFold's pLDDT scheme. Each row represents a different PDB target: `3di3`, `6m0j`, `1ycr`, `4zqk`, `4z18`, and `1xiw`.