Full-Atom Peptide Design via Riemannian–Euclidean Bayesian Flow Networks

Hao Qian[1], **Shikui Tu**[1*], **Lei Xu**[1, 2*]

[1] School of Computer Science, Shanghai Jiao Tong University
[2] Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Guangdong, China
tushikui@sjtu.edu.cn, leixu@sjtu.edu.cn

Abstract

Diffusion and flow matching models have recently emerged as promising approaches for peptide binder design. Despite their progress, these models still face two major challenges. First, categorical sampling of discrete residue types collapses their continuous parameters into one-hot assignments, while continuous variables (e.g., atom positions) evolve smoothly throughout the generation process. This mismatch disrupts the update dynamics and results in suboptimal performance. Second, current models assume unimodal distributions for side-chain torsion angles, which conflicts with the inherently multimodal nature of side-chain rotameric states and limits prediction accuracy. To address these limitations, we introduce PepBFN, the first Bayesian flow network for full-atom peptide design that directly models parameter distributions in fully continuous space. Specifically, PepBFN models discrete residue types by learning their continuous parameter distributions, enabling joint and smooth Bayesian updates with other continuous structural parameters. It further employs a novel Gaussian mixture–based Bayesian flow to capture the multimodal side-chain rotameric states and a Matrix Fisher-based Riemannian flow to directly model residue orientations on the $\mathrm{SO}(3)$ manifold. Together, these parameter distributions are progressively refined via Bayesian updates, yielding smooth and coherent peptide generation. Experiments on side-chain packing, reverse folding, and binder design tasks demonstrate the strong potential of PepBFN in computational peptide design.

Code — https://github.com/CMACH508/PepBFN

1 Introduction

Peptides, short chains of amino acids, can be engineered to specifically bind to protein targets. This specific binding modulates protein activity and can influence key biological processes (Craik et al. 2013; Henninot, Collins, and Nuss 2018; Wang et al. 2022). Compared to macromolecular biologics, such as proteins and RNAs, peptides exhibit lower immunogenicity and are less costly to produce (Giordano et al. 2014; Fosgerau and Hoffmann 2015; Davda et al. 2019). Relative to small molecule drugs, peptides typically offer greater safety and enhanced tolerance (Di 2015; Muttenthaler et al. 2021). Hence, peptides serve as an effective bridge between small molecules and biomacromolecules.

Traditional peptide design starts by analyzing the known crystal structures to understand their primary and secondary structures. Subsequently, methods such as alanine scanning (Alascan) (Lefèvre, Rémy, and Masson 1997) and the utilization of small, focused libraries (Quartararo et al. 2020) help to systematically develop the structure-activity relationship (SAR) (Fosgerau and Hoffmann 2015). However, the efficiency of these methods is significantly reduced by the combinatorial explosion of amino acid sequences. As experimental limitations persist, there is an increasing demand for computational approaches that enhance in silico peptide binder design (Bryant and Elofsson 2022; Swanson et al. 2022; Cao et al. 2022; Bhat et al. 2023; Chen et al. 2024).

Recently, deep generative models, such as auto-regressive models, diffusion models (Song et al. 2020; Song, Meng, and Ermon 2020; Dhariwal and Nichol 2021), and flow matching models (Lipman et al. 2022; Liu, Gong, and Liu 2022), have demonstrated their potential in peptide binder design (Li et al. 2024b; Lin et al. 2024; Kong et al. 2024; Wang et al. 2024). PepHAR (Li et al. 2024a), an auto-regressive model, efficiently generates peptide residues in a sequential manner within 3D space, guided by learned anchor hotspots. However, it tends to produce severe steric clashes between the generated peptide and the target. Diffusion and flow matching models, such as PepFlow (Li et al. 2024b) and PepGLAD (Kong et al. 2024), alleviate steric clashes through non-autoregressive generation.

Despite these advances, there still remain two limitations in existing models. Firstly, discrete residue types are sampled categorically, which collapses their continuous parameters into one-hot assignments and discards distributional information, while continuous structural variables are sampled smoothly. This mismatch leads to inconsistent samplings, causing sequence fluctuations and slow convergence, as also observed in previous studies on structure-based molecule design (Peng et al. 2023; Song et al. 2023; Qu et al. 2024). Secondly, existing approaches typically assume a unimodal distribution for side-chain angles, whereas their true distributions are inherently multimodal due to diverse rotameric states. This assumption negatively impacts the accuracy of side-chain prediction (see Sec. 5.1).

*Corresponding authors

To overcome these limitations, we introduce PepBFN, a Riemannian–Euclidean Bayesian flow network that jointly models sequences and structures in fully continuous parameter space. Bayesian Flow Networks (Graves et al. 2023) (BFNs) iteratively refine continuous parameter distributions via Bayesian inference, leading to smooth updates that resolve the mismatch problem. Moreover, unlike prior generative approaches that utilize unimodal distributions for modeling side-chain angles, we propose a novel Gaussian mixture-based Bayesian flow inference process. This design enables an effective representation of diverse rotameric states throughout the generation, leading to more accurate side-chain prediction. In addition, to model residue orientations, we introduce a new Riemannian Bayesian flow based on the Matrix Fisher distribution (Downs 1972; Khatri and Mardia 1977). As an exponential family distribution on the $\mathrm{SO}(3)$ manifold, Matrix Fisher enables tractable Bayesian updates, allowing residue orientations to be seamlessly integrated into the Bayesian flow framework.

Our main contributions are as follows:

- We propose PepBFN, the first Bayesian Flow Network for full-atom peptide binder design, which jointly models four key modalities, i.e., discrete residue types, continuous residue orientations, centroids, and side-chain torsions, within fully continuous parameter space.
- PepBFN incorporates a novel Gaussian mixture–based Bayesian flow for accurately capturing multimodal side-chain conformations, and a Matrix Fisher–based Riemannian flow for residue rotations, enabling tractable Bayesian updates on the $\mathrm{SO}(3)$ manifold.
- PepBFN achieves state-of-the-art performance on multiple benchmarks, including de novo peptide binder design, side-chain prediction, and reverse folding tasks, demonstrating its effectiveness as a unified framework for computational peptide design.

2 Related Works

2.1 Bayesian Flow Networks

Bayesian Flow Networks (BFNs) (Graves et al. 2023) are a recently proposed generative modeling framework that integrates Bayesian inference with neural networks. BFNs evolve parameter distributions through iterative Bayesian updates guided by a noise scheduler, rather than perturbing data directly as in diffusion models with predefined forward processes. BFNs demonstrate competitive performance across diverse domains, showing notable advantages in discrete settings such as protein language modeling (Atkinson et al. 2025) and molecule design (Song et al. 2024; Qu et al. 2024).

2.2 Generative Models for Protein and Peptide Binder Design

Generative models have recently emerged as powerful tools for protein and peptide binder design. Models such as PepFlow (Li et al. 2024b), SurfFlow (Wu et al. 2025) and PPFlow (Lin et al. 2024) leverage multi-modal flow matching to perform full-atom peptide generation conditioned on target protein structures. Additionally, diffusion-based models like PepGLAD (Kong et al. 2024) and DiffPepBuilder (Wang et al. 2024) explore joint diffusion processes to design peptide sequences and structures. Autoregressive approaches such as PepHAR (Li et al. 2024a) sequentially generate peptide residues in 3D space, guided by learned anchor hotspots.

3 Preliminaries

3.1 Problem Formulation

Protein–peptide complexes are denoted by $\{\mathcal{P}, \mathcal{G}\}$, where $\mathcal{P}$ and $\mathcal{G}$ represent the protein and peptide respectively. Each component is described as a sequence of local residue frames. For the i-th residue, we represent its geometry by the position of residue centroid (i.e., C_α position) $\mathbf{x}^{(i)} \in \mathbb{R}^3$ and an orientation matrix $\mathbf{o}^{(i)} \in \mathrm{SO}(3)$, consistent with the representation used in AlphaFold 3 (Abramson et al. 2024). The torsional angle set $\chi^{(i)} = \{\psi^{(i)}, \chi_1^{(i)}, \ldots, \chi_4^{(i)}\}$ includes the backbone angle $\psi^{(i)}$, which determines the position of the backbone oxygen atom, and up to four side-chain $\chi_{1-4}^{(i)}$ angles. The amino acid type is encoded as a one-hot vector $\mathbf{c}^{(i)} \in \mathbb{R}^{20}$. Thus, a protein or peptide with N residues can be expressed as $\{\mathcal{R}^{(i)}\}_{i=1}^{N}$, where each residue is defined by $\mathcal{R}^{(i)} = \{\mathbf{x}^{(i)}, \mathbf{o}^{(i)}, \chi^{(i)}, \mathbf{c}^{(i)}\}$.

Given a target protein pocket $\mathcal{P}$, PepBFN designs new peptides $\mathcal{G}$ that bind effectively to the target protein by jointly modeling their sequences and 3D structures.

3.2 Peptide Design via BFNs

We define $\boldsymbol{\theta}$ as the parameters of the input data distribution $p_I(\mathcal{G} \mid \boldsymbol{\theta})$. The peptide generation process is formulated as a Bayesian communication between a sender and a receiver. At each time step t_i, the sender generates a noisy peptide $\mathbf{y}_i$ by perturbing the clean peptide $\mathcal{G}$ with a known noise factor α_i, resulting in the *sender distribution*:

$$p_S(\mathbf{y}_i \mid \mathcal{G}; \alpha_i),$$

which resembles the idea of forward process in diffusion models. The receiver acts as the decoder, conditioned on the protein context $\mathcal{P}$ and previously inferred parameters $\boldsymbol{\theta}_{i-1}$, aiming to reconstruct the clean peptide $\hat{\mathcal{G}}$, yielding the *output distribution*:

$$p_O(\hat{\mathcal{G}} \mid \boldsymbol{\theta}_{i-1}, \mathcal{P}; t_i) = \Phi(\boldsymbol{\theta}_{i-1}, \mathcal{P}, t_i).$$

Here, Φ denotes a neural network conditioned on the protein target $\mathcal{P}$, the previous-step parameters $\boldsymbol{\theta}_{i-1}$, and the current time step t_i. Given the known noise factor α_i, the receiver is able to generate the noisy peptide $\mathbf{y}_i$ by injecting noise into its estimate $\hat{\mathcal{G}}$, thereby forming the *receiver distribution*:

$$p_R(\mathbf{y}_i \mid \boldsymbol{\theta}_{i-1}, \mathcal{P}; t_i) = \mathbb{E}_{\hat{\mathcal{G}} \sim p_O}\left[p_S(\mathbf{y}_i \mid \hat{\mathcal{G}}; \alpha_i)\right].$$

During generation, the receiver applies Bayesian inference to iteratively refine the peptide parameters $\boldsymbol{\theta}$ in closed form. Specifically, the *Bayesian update distribution* p_U is derived from the Bayesian update function h as:

$$p_U(\boldsymbol{\theta}_i \mid \boldsymbol{\theta}_{i-1}, \mathcal{G}, \mathcal{P}; \alpha_i) = \mathbb{E}_{\mathbf{y}_i \sim p_S}\Big[\delta\big(\boldsymbol{\theta}_i - h(\boldsymbol{\theta}_{i-1}, \mathbf{y}_i, \alpha_i)\big)\Big]. \quad (1)$$

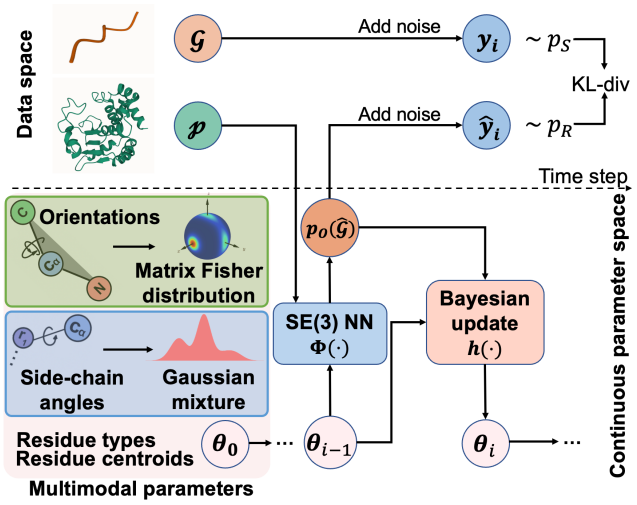


Figure 1: The overview of PepBFN.

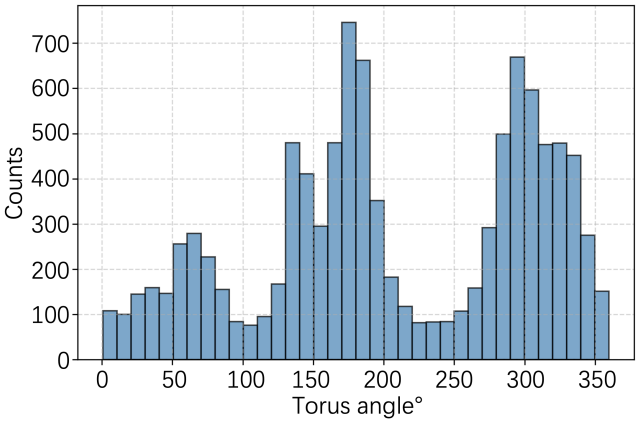


Figure 2: Distribution of peptide torus angles.

Here, $\delta(\cdot)$ denotes the Dirac delta distribution, and h maps the previous parameters $\boldsymbol{\theta}_{i-1}$ and noisy observation $\mathbf{y}_i$ to the updated parameters $\boldsymbol{\theta}_i$.

In addition, the *Bayesian flow distribution* p_F can be achieved to support simulation-free training as follows:

$$p_F(\boldsymbol{\theta}_i \mid \mathcal{G}, \mathcal{P}; t_i) = \mathbb{E}_{\boldsymbol{\theta}_{1:i-1} \sim p_U}\, p_U(\boldsymbol{\theta}_i \mid \boldsymbol{\theta}_{i-1}, \mathcal{G}, \mathcal{P}; \alpha_i) = p_U(\boldsymbol{\theta}_i \mid \boldsymbol{\theta}_0, \mathcal{G}, \mathcal{P}; \beta(t_i)), \quad (2)$$

where $\beta(t_i) = \sum_{j=1}^{i} \alpha_j$ represents the accumulated noise scheduler based on the additive property of the noise factors (Graves et al. 2023). The model is trained by minimizing the expected KL divergence between the sender and receiver distributions over n time steps:

$$L_n(\mathcal{G}, \mathcal{P}) = n\, \mathbb{E}_{i \sim U(1,n),\, \boldsymbol{\theta}_{i-1} \sim p_F}\, D_{\mathrm{KL}}(p_S \,\|\, p_R). \quad (3)$$

We summarize the key similarities and differences among different generative models in Table 5 in the Appendix.

Difficulties for Peptide Design under the BFN Framework A key theoretical challenge in applying BFNs to peptide design is the requirement for conjugate prior–posterior distributions. However, the commonly used distributions for side-chain angles (e.g., unimodal wrapped Gaussians (Zhang et al. 2023)) and residue orientations (e.g., Brownian motion on $\mathrm{SO}(3)$ (Leach et al. 2022)) violate this requirement, making principled training and inference difficult to implement in practice. Moreover, as we mentioned before, unimodal Gaussian distribution can not accurately model side-chain rotameric states. In this paper, we address these issues by devising two new BFNs for side-chain angles and residue orientations, respectively, and they both support conjugate-aligned formulations tailored to peptide design.

4 Methods

4.1 The Overview of PepBFN

As shown in Fig. 1, we construct PepBFN to jointly capture four key modalities involved in peptide design. Specifically, for residue orientations, we develop a Matrix Fisher-based Riemannian BFN, allowing for direct Bayesian update on the $\mathrm{SO}(3)$ manifold. For torsional angles, we propose a novel Gaussian mixture-based BFN on the toric manifold, enabling accurate modeling of multimodal angular distributions. Residue centroids are modeled based on unimodal Gaussian distributions, and residue types are modeled with categorical distributions, inspired by the original BFN paper (Graves et al. 2023). Together, these four components form a unified BFN framework for peptide binder design.

During generation, the SE(3) neural network takes the protein context $\mathcal{P}$ and previous parameters $\boldsymbol{\theta}_{i-1}$ as inputs, and predicts the denoised peptide $\hat{\mathcal{G}}$, which serves as parameters of the receiver distribution p_R. Samples drawn from p_R are combined with the previous parameters $\boldsymbol{\theta}_{i-1}$ through the Bayesian update operator $h(\cdot)$, producing the posterior parameters $\boldsymbol{\theta}_i$, which are then propagated to the next time step. The entire generative process is carried out in fully continuous parameter space, ensuring smooth and consistent parameter updates. To the best of our knowledge, PepBFN is the first Bayesian Flow Network specifically designed for the full-atom peptide design.

4.2 Gaussian Mixture-based BFN for Angles

Side-chain χ angles in proteins are often modeled using a single wrapped Gaussian distribution to account for rotational periodicity (Zhang et al. 2023). However, such unimodal approximations fail to capture their true conformational heterogeneity. Our analysis on prior distribution (see Fig. 2) shows that χ angles cluster around three distinct rotameric states corresponding to *gauche*$^+$, *trans*, and *gauche*$^-$[1]. To accurately capture the multimodal nature of side-chain torsion angles, we employ a novel Gaussian mixture-based BFN, which effectively learns multiple rotameric states throughout the time steps.

[1]Here, *gauche*$^-$ and *gauche*$^+$ denote staggered rotameric states where the torsion angle χ is near 300° (or equivalently -60°) and 60°, respectively. The *trans* state corresponds to another low-energy staggered conformation near 180°, commonly observed in rotamer libraries.

The input distribution is given by

$$\boldsymbol{\theta}^{ang} \overset{\text{def}}{=} \left\{\mu^k, \rho^k, \pi^k\right\}_{k=1}^K,$$
$$p_I(\chi \mid \boldsymbol{\theta}^{ang}) \overset{\text{def}}{=} \sum_{k=1}^K \pi^k \mathcal{N}\big(\chi \mid \mu^k, (\rho^k)^{-1}\big), \tag{4}$$

where $\pi^k \geq 0$, $\sum_{k=1}^K \pi^k = 1$ are the mixture weights, and ρ^k is the known precision for k-th Gaussian. The sender distribution is defined as a single Gaussian:

$$p_S(y_i^{ang} \mid \chi) = \mathcal{N}(y_i^{ang} \mid \chi, \mathcal{P}, \alpha_i; t_i), \tag{5}$$

where α_i is a noise factor at t_i time step.

Lemma 4.1 (Conjugacy of a Gaussian Mixture Prior with a Gaussian Likelihood)**.** *Let the prior $p(x)$ be a Gaussian mixture distribution and the likelihood $p(y \mid x)$ be a single Gaussian distribution. By Bayes' rule, the posterior $p(x \mid y)$ retains the Gaussian mixture form.*

Proposition 4.2 (Bayesian Flow for Gaussian Mixture Distribution)**.** *Given the input distribution defined in Eq. 4 and the sender distribution defined in Eq. 5, the Bayesian flow distribution p_F for the mean parameter μ_i^k of the k-th component at time step t_i is*

$$p_F(\mu_i^k \mid \chi, \mathcal{P}; t_i) = \mathcal{N}\left(\mu_i^k \,\middle|\, \frac{\beta(t_i)\chi + \mu_0^k \rho_0^k}{\rho_i^k}, \frac{\beta(t_i)}{(\rho_i^k)^2}\right), \tag{6}$$

where the k-th precision of input distribution $p_I(\cdot)$ at t_i time step is $\rho_i^k = \rho_0^k + \beta(t_i)$ and the precision scheduler is $\beta(t_i) = \sum_{m=1}^i \alpha_m$. The Bayesian flow distribution of the mixture weights π_i^k at time step t_i does not admit a closed-form expression and requires numerical simulation.

Proposition 4.2 provides an analytical expression for the mean parameters μ_i and a simulation-based procedure for the weight parameters π_i in the Bayesian flow distribution p_F. We next design the precision scheduler $\beta(\cdot)$ to ensure that the expected entropy of the input distribution $p_I(\cdot)$ under p_F decreases linearly over time as suggested in (Graves et al. 2023). Specifically, the expected entropy is defined as:

$$H(t_i) \overset{\text{def}}{=} \mathbb{E}_{p_F(\boldsymbol{\theta}_i^{ang} \mid \chi, \mathcal{P}; t_i)}\big[H(p_I(\cdot \mid \boldsymbol{\theta}_i^{ang}))\big]. \tag{7}$$

Proposition 4.3 (Time-Dependent Linear Decrease of the Expected Entropy Upper Bound)**.** *Given the expected entropy $H(t_i)$ defined in Eq. 7 and the Gaussian likelihood precision parameter $\alpha_i = \rho_0^{1-i/n} \cdot \rho_1^{i/n}\big[1 - (\frac{\rho_0}{\rho_1})^{1/n}\big]$ at the time step t_i, the upper bound of the expected entropy $H(t_i)$ decreases linearly with the time step t_i.*

Given the Gaussian likelihood precision parameter α_i defined in Proposition 4.3, the receiver distribution can be achieved:

$$p_R(y_i^{ang} \mid \boldsymbol{\theta}_{i-1}^{ang}, \mathcal{P}; t_i) = \mathbb{E}_{\hat{\chi} \sim p_O}\left[p_S(y_i^{ang} \mid \hat{\chi}; \alpha_i)\right],$$
$$\text{where} \quad p_O(\hat{\chi} \mid \boldsymbol{\theta}_{i-1}^{ang}, \mathcal{P}; t_i) = \Phi^{ang}(\boldsymbol{\theta}_{i-1}^{ang}, \mathcal{P}, t_i).$$

Here, Φ^{ang} denotes a neural network conditioned on protein target $\mathcal{P}$, parameters $\boldsymbol{\theta}_{i-1}^{ang}$ and time step t_i. Substituting into Eq. 3, the loss function becomes:

$$L_n^{ang}(\chi, \mathcal{P}) = n\, \mathbb{E}_{i \sim U(1,n), \boldsymbol{\theta}_{i-1}^{ang} \sim p_F} D_{\mathrm{KL}}(p_S \,\|\, p_R)$$
$$= \frac{n}{2}\, \mathbb{E}_{i \sim U(1,n), \boldsymbol{\theta}_{i-1}^{ang} \sim p_F} \alpha_i \|\chi - \hat{\chi}\|^2. \tag{8}$$

The integration of Gaussian mixture distributions into the BFN framework allows principled and stable modeling of multimodal torsion angles, well-suited for Bayesian inference. Please refer to Algorithm 4 in the Appendix for inference details.

4.3 Matrix Fisher-based BFN for Orientations

We represent the orientations of N residues in a peptide as $\mathbf{O} = \{\mathbf{o}^{(i)}\}_{i=1}^N \sim \mathcal{M}(\mathbf{O}; \boldsymbol{\theta}^{ori})$, where i-th residue orientation is denoted as a rotation matrix $\mathbf{o}^{(i)}$ and $\boldsymbol{\theta}^{ori}$ is the parameter of the Matrix Fisher distribution. Specifically, we define the input distribution as $p_I(\mathbf{O} \mid \boldsymbol{\theta}^{ori}) = \mathcal{M}(\mathbf{O}; \boldsymbol{\theta}^{ori}) = \frac{1}{c(\boldsymbol{\theta}^{ori})} \exp\left(\mathrm{tr}((\boldsymbol{\theta}^{ori})^\top \mathbf{O})\right)$, where $c(\boldsymbol{\theta}^{ori})$ is a normalizing constant and matrix $\boldsymbol{\theta}^{ori} \in \mathbb{R}^{3\times 3}$ (see Sec. 1 in the Appendix for details).

Lemma 4.4 (Conjugacy of Matrix Fisher Distributions)**.** *Let the prior distribution over rotation matrices $\mathbf{O} \in \mathrm{SO}(3)$ be a Matrix Fisher distribution given by*

$$p(\mathbf{O}) \propto \exp\left(\mathrm{tr}\left(\boldsymbol{\theta}_a^\top \mathbf{O}\right)\right),$$

and the likelihood of an observation $\mathbf{Y}^{ori} \in \mathrm{SO}(3)$ be

$$p(\mathbf{Y}^{ori} \mid \mathbf{O}) \propto \exp\left(\mathrm{tr}\left((\mathbf{O}\boldsymbol{\Lambda})^\top \mathbf{Y}^{ori}\right)\right),$$

where $\boldsymbol{\Lambda}$ denotes a diagonal matrix with identical diagonal entries. Then, the posterior distribution is also a Matrix Fisher distribution $\mathcal{M}(\mathbf{O} \mid \mathbf{Y}^{ori}; \boldsymbol{\theta}_b)$:

$$p(\mathbf{O} \mid \mathbf{Y}^{ori}) \propto \exp\left(\mathrm{tr}\left((\boldsymbol{\theta}_a + \mathbf{Y}^{ori}\boldsymbol{\Lambda})^\top \mathbf{O}\right)\right),$$

where

$$\boldsymbol{\theta}_b = \boldsymbol{\theta}_a + \mathbf{Y}^{ori}\boldsymbol{\Lambda}. \tag{9}$$

Solving the Bayesian update distribution $p_U(\boldsymbol{\theta}_i^{ori} \mid \boldsymbol{\theta}_{i-1}^{ori}, \mathcal{G}, \mathcal{P}; t_i)$ in closed form is non-trivial because the Matrix Fisher distribution family is not closed under operations in Bayesian update function $h(\cdot)$ (see Eq. 9). Here, we introduce an auxiliary variable $\mathbf{T}_i$, which resolves this difficulty and yields a new Matrix Fisher distribution that is both computationally tractable (see Proposition 4.5) and straightforward to sample from (see Sec. 1.3 in the Appendix).

Proposition 4.5 (Bayesian Flow for Matrix Fisher Distribution)**.** *Assume that the input distribution over residue orientations in dataset follows the Matrix Fisher distribution on $\mathrm{SO}(3)$, i.e., $p_I(\mathbf{O} \mid \boldsymbol{\theta}^{ori}) \overset{def}{=} \mathcal{M}(\mathbf{O}; \boldsymbol{\theta}^{ori})$. At time step t_i, we define the sender distribution as $p_S(\mathbf{Y}_i^{ori} | \mathbf{O}; \boldsymbol{\Lambda}_i) = \mathcal{M}(\mathbf{Y}_i^{ori}; \mathbf{O}\boldsymbol{\Lambda}_i)$. We introduce an auxiliary variable $\mathbf{T}_i$ defined as the $\mathrm{SO}(3)$ projection of $\mathbf{Y}_i^{ori}\boldsymbol{\Lambda}_i$, and the Bayesian flow distribution on $\mathbf{T}_i$ is given by*

$$p_F(\mathbf{T}_i \mid \mathbf{O}, \mathcal{P}; t_i) = \mathcal{M}(\mathbf{T}_i; \mathbf{O}\boldsymbol{\Lambda}_i^2). \tag{10}$$

Next, we define the receiver distribution as:

$$p_R(\mathbf{Y}_i^{ori} \mid \boldsymbol{\theta}_{i-1}^{ori}, \mathcal{P}; t_i) = \mathbb{E}_{\hat{\mathbf{O}} \sim p_O}\left[p_S(\mathbf{Y}_i^{ori} \mid \hat{\mathbf{O}}; \boldsymbol{\Lambda}_i)\right],$$
$$\text{where} \quad p_O(\hat{\mathbf{O}} \mid \boldsymbol{\theta}_{i-1}^{ori}, \mathcal{P}; t_i) = \Phi^{ori}(\mathbf{T}_{i-1}^{ori}, \mathcal{P}, t_i).$$

Here, Φ^{ori} denotes a neural network conditioned on protein target $\mathcal{P}$, parameters $\mathbf{T}_{i-1}$ and time step t_i. Since both Matrix Fisher distributions of p_S and p_R are isotropic around the rotation matrix $\mathbf{O}$, the loss function can be defined as:

$$L_n^{ori}(\mathbf{O}, \mathcal{P}) = n\,\mathbb{E}_{i\sim U(1,n),\,\mathbf{T}_{i-1}\sim p_F} D_{\mathrm{KL}}(p_S \,\|\, p_R)$$
$$= n\,\mathbb{E}_{i\sim U(1,n),\,\mathbf{T}_{i-1}\sim p_F} \lambda_i a(\lambda_i)\big(3 - \mathrm{tr}(\hat{\mathbf{O}}^\top \mathbf{O})\big), \quad (11)$$

where $a(\lambda_i)$ is a scalar depending only on λ_i. Please refer to Sec. 1 in the Appendix for more details.

For the inference process, we generate the final residue rotations $\mathbf{O}$ by sampling from the prior $\mathbf{T}_0 \sim \mathrm{Uniform}(\mathrm{SO}(3))$ and evolving through discrete time steps $i \in \{0, 1, \ldots, N\}$:

$$\mathbf{T}_0 \xrightarrow{\Psi^{ori}} \hat{\mathbf{O}} \xrightarrow{P_F} \mathbf{T}_1 \to \cdots \to \mathbf{T}_N \xrightarrow{\Psi^{ori}} \mathbf{O}_{\text{final}}.$$

4.4 Euclidean BFN for Centroids and Categorical BFN for Types

In this section, we briefly describe the construction of the Bayesian flow for Gaussian centroids and discrete residue types, mainly following the methodology of previous works (Graves et al. 2023; Qu et al. 2024).

The centroids (i.e., C_α positions) of N residues in a peptide, denoted as $\mathbf{X}$, can be characterized by Gaussian distribution, i.e., $\mathbf{X} = \{\mathbf{x}^{(i)}\}_{i=1}^N \sim p_I(\cdot \mid \boldsymbol{\theta}^{pos}) = \mathcal{N}(\mathbf{X} \mid \boldsymbol{\mu}^{pos}, (\rho^{pos})^{-1}\mathbf{I})$, where $\boldsymbol{\theta}^{pos} \overset{\text{def}}{=} \{\boldsymbol{\mu}^{pos}, \rho^{pos}\}$.

The amino acid types in a peptide can be denoted as $C = \{c^{(i)}\}_{i=1}^D$ where $c^{(i)} \in \{c \in \mathbb{Z}^+ \mid 1 \le c \le 20\}$. Here, we utilize a categorical distribution $\boldsymbol{\theta}^{type} = (\boldsymbol{\theta}^{(1)}, \cdots, \boldsymbol{\theta}^{(D)})$ with $\boldsymbol{\theta}^{(d)} = \big(\theta_1^{(d)}, \cdots, \theta_K^{(d)}\big) \in \Delta^{K-1}$, where $\theta_k^{(d)}$ is the probability assigned to class k for d-th residue type. Then the input distribution gives $p_I(C \mid \boldsymbol{\theta}^{type}) = \prod_{d=1}^D \theta^{(d)}_{c^{(d)}}$. Other key components for constructing a complete Bayesian flow, including the sender and receiver distributions, Bayesian flow distributions and loss functions are summarized in Sec. 3 in the Appendix.

The final centroids $\mathbf{X}_{\text{final}}$ and residue types $\mathbf{C}_{\text{final}}$ can be sampled by initializing from simple priors. Specifically, centroids are initialized with parameters $\boldsymbol{\theta}_0^{pos} = \mathbf{0} \in \mathbb{R}^{3\times 3}$, and residue types with $\boldsymbol{\theta}_0^{type} = \frac{1}{K}\mathbf{1}$. Both parameters follow similar Bayesian update trajectories:

$$\boldsymbol{\theta}_0^{pos} \xrightarrow{\Psi^{pos}} \hat{\mathbf{X}} \xrightarrow{P_F} \boldsymbol{\theta}_1^{pos} \to \cdots \to \boldsymbol{\theta}_N^{pos} \xrightarrow{\Psi^{pos}} \mathbf{X}_{\text{final}},$$
$$\boldsymbol{\theta}_0^{type} \xrightarrow{\Psi^{type}} \hat{\mathbf{C}} \xrightarrow{P_F} \boldsymbol{\theta}_1^{type} \to \cdots \to \boldsymbol{\theta}_N^{type} \xrightarrow{\Psi^{type}} \mathbf{C}_{\text{final}}.$$

4.5 Summary of PepBFN

By integrating all multimodal Bayesian flow objectives, the overall loss function is defined as follows:

$$\mathcal{L}_n^{all} = \lambda_1 \mathcal{L}_n^{pos} + \lambda_2 \mathcal{L}_n^{ori} + \lambda_3 \mathcal{L}_n^{type} + \lambda_4 \mathcal{L}_n^{ang}, \quad (12)$$

where $\lambda_1, \lambda_2, \lambda_3$, and λ_4 represent the respective weights of each component in the loss function. These four modality-specific modules are coupled through a single SE(3)-aware neural network to predict a denoised peptide $\hat{\mathcal{G}}$ under the Bayesian flow framework. We next evaluate this unified pipeline on side-chain packing, reverse folding, and binder co-design tasks. We provide complete training and sampling details of PepBFN in Sec. 4 in the Appendix.

	MAE (deg)↓				Correct % ↑
	χ_1	χ_2	χ_3	χ_4	
Rosetta	38.31	43.23	53.61	71.67	57.03
SCWRL4	30.06	40.40	49.71	53.79	60.54
DLPacker	22.44	35.65	58.53	61.70	60.91
AttnPacker	19.04	28.49	40.16	60.04	61.46
DiffPack	17.92	26.08	36.20	67.82	62.58
PepBFN_sc	**10.75**	**12.26**	**34.25**	**53.21**	**75.24**

Table 1: Evaluation of methods in side-chain packing task.

5 Experiments

In this section, we conduct a comprehensive evaluation of PepBFN across three tasks: (1) side-chain packing, (2) peptide reverse folding, and (3) sequence-structure co-design. Following previous works (Li et al. 2024b,a), we curated moderate-length sequences from PepBDB (Wen et al. 2019) and Q-BioLip (Wei et al. 2024). Our dataset consists of 158 complexes for testing, and 8,207 samples designated for training and validation.

5.1 Side-chain Packing

This task aims to predict the side-chain torsion angles of each peptide, given fixed backbone structures and sequences. For evaluation, each model generates 64 side-chain conformations per peptide to recover the target angles. Our model PepBFN_sc is trained under the same fixed backbone and sequence settings.

Metrics We evaluate performance using the mean absolute error (MAE) over the four predicted torsion angles. Given the inherent flexibility of side-chains, we additionally report the proportion of correct predictions whose deviations fall within 20° of the ground truth.

Baselines Two traditional energy-based methods: Rosetta Packer (Leman et al. 2020), SCWRL4 (Krivov, Shapovalov, and Dunbrack Jr 2009), and three learning-based models: DLPacker (Misiura et al. 2022), AttnPacker (McPartlon and Xu 2023), DiffPack (Zhang et al. 2023).

Results As shown in Table 1, our method consistently outperforms all baselines in the side-chain packing task. It achieves the lowest MAE for the four torsion angles. Furthermore, our method achieves the highest proportion of correctly predicted side-chains within 20° of the ground truth, which exceeds the second-best method by a substantial margin. A key insight is that our neural network directly takes K Gaussian parameters as expressive inputs and yields accurate predictions of the angles. Subsequent Bayesian updates then leverage these predictions to refine the posterior distribution while simultaneously preserving and sharpening its multi-peak structure. This integration of an expressive Gaussian mixture prior with principled Bayesian inference yields more stable and accurate torsion angle predictions than unimodal alternatives.

5.2 Peptide Reverse Folding

This task designs peptide sequences from backbone-only complex structures. Each model generates 64 sequences per

Method	AAR % ↑	Worst % ↑	Likeness error ↓	Hamming Diversity ↑
ProteinMPNN	53.28	45.99	2.70	15.33
ESM-IF	43.51	36.18	2.67	13.76
PepFlow	53.98	43.32	0.59	**26.40**
PepBFN_seq	**62.46**	**47.82**	**0.45**	13.21

Table 2: Evaluations of methods in reverse folding task.

peptide. We trained PepBFN_seq with fixed backbones to predict the sequences.

Baselines We use three baselines for peptide sequence design: ProteinMPNN (Dauparas et al. 2022), ESM-IF (Hsu et al. 2022), and PepFlow (partial sampling) (Li et al. 2024b).

Metrics The amino acid recovery rate (**AAR**) measures the sequence identity between the generated and the ground truth. **Worst** denotes, for each protein target, the lowest AAR among the generated peptides. **Likeness error** is assessed via the mean absolute errors of negative log-likelihood computed by ProtGPT2 (Ferruz, Schmidt, and Höcker 2022) between peptides in test set and the generated peptides, which captures how well the generated sequences conform to the native peptide distribution. **Hamming Diversity** is evaluated as the mean pairwise Hamming distance among the generated sequences.

Results As shown in Table 2, our method significantly outperforms all baselines in terms of AAR and the worst rate recovery across generated sequences, demonstrating its strong capability to recover peptide sequences under fixed backbone constraints. Moreover, our model achieves the smallest likeness error score compared to other methods, indicating that the generated sequences align most closely with native peptide distributions. The model exhibits comparatively low sequence diversity, which is expected since its high recovery rate restricts the exploration of sequence space. Overall, by modeling discrete data in continuous parameter space, our approach effectively captures the underlying data manifold, which in turn leads to improved performance.

5.3 Sequence-structure Co-design

This task involves generating both the sequence and the binding gesture of a peptide given its target protein. Each model takes the full-atom structure of the target protein as input and outputs peptide binder structures. For evaluation, we generate 64 peptides for each target protein.

Metrics (1) **Geometry**. **RMSD (Root-Mean-Square Deviation)** computes the C_α distances between the generated peptide structures and the native structures after alignment. **SSR (Secondary Structure Ratio)** calculates the overlap in secondary structures between the generated and native peptides. **BSR (Binding Site Rate)** quantifies the similarity in peptide-target interactions by evaluating the overlap of the binding sites. (2) **Energy**. **Affinity** evaluates the fraction of peptides that exhibit lower binding energies compared to the native peptide. **Stability** assesses the percentage of generated complexes that exhibit lower total energy

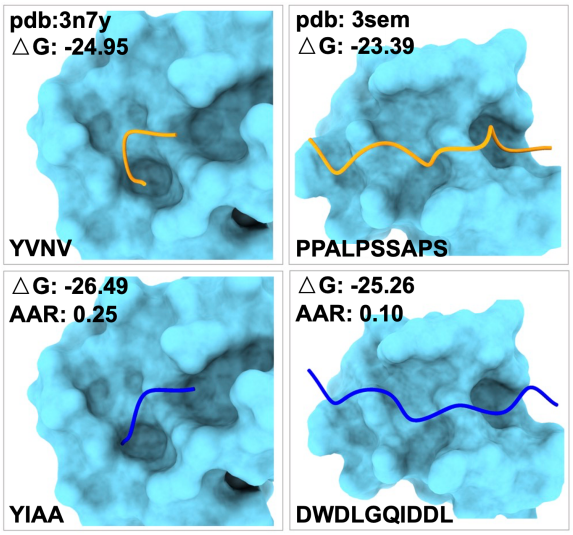


Figure 3: Two examples of PepBFN-generated peptides with improved binding affinities. Top row: native peptides; bottom row: peptides generated by our method.

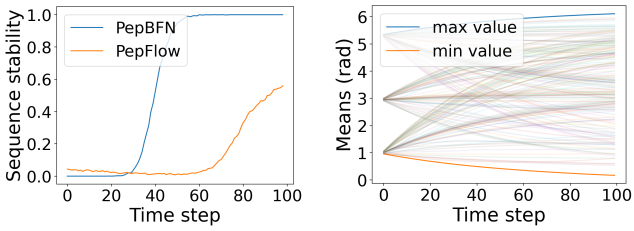
than native complexes, using the Rosetta energy function (Chaudhury, Lyskov, and Gray 2010). (3) **Novelty** is measured as the ratio of novel peptides, defined by two criteria: (a) TM-score ≤ 0.5 (Zhang and Skolnick 2005; Xu and Zhang 2010) and (b) sequence identity ≤ 0.5. (4) **Diversity** quantifies the structural and sequence variability among generated peptides for each target, computed as pairwise $(1 - \text{TM-score}) \times (1 - \text{sequence identity})$.

Baselines We evaluate PepBFN against five powerful peptide design models. **RFDiffusion** (Watson et al. 2023) leverages pre-trained weights from RoseTTAFold (Baek et al. 2021) to generate protein backbone structures through a denoising diffusion process. The peptide sequences are subsequently reconstructed using **ProteinMPNN** (Dauparas et al. 2022). **ProteinGenerator** enhances RFDiffusion by incorporating joint sequence-structure generation (Lisanza et al. 2023). **PepFlow** (Li et al. 2024b) generates full-atom peptides using a flow matching framework. **PepGLAD** (Kong et al. 2024) utilizes equivariant latent diffusion networks to generate full-atom peptide structures. **PepHAR** (Li et al. 2024a) generates peptide residues autoregressively, based on a learned prior distribution for hotspot residues.

Results As shown in Table 3, PepBFN effectively generates diverse and novel peptides with strong binding affinities. Specifically, PepBFN achieves the best binding affinity (22.26%) and binding site rate (86.97%), indicating effective modeling of peptide–protein interactions. In addition, it also achieves the highest novelty (79.79%) and diversity (32.47%). These improvements can be attributed to the smooth and expressive generative process of BFN, which accurately models the underlying parameter manifold and facilitates diverse yet plausible peptide generation. Interestingly, RFDiffusion exhibits better stability, probably because it was trained on a large corpus of proteins with more stable motifs (Li et al. 2024b). Compared with PepFlow and

Experiments	Geometry			Energy		Design	
	RMSD Å	SSR % ↑	BSR % ↑	Affinity % ↑	Stability % ↑	Novelty % ↑	Diversity % ↑
RFDiffusion	4.17	63.86	26.71	16.53	**26.82**	53.74	25.39
ProteinGenerator	4.35	29.15	24.62	13.47	23.84	52.39	22.57
PepFlow	2.07	83.46	86.89	21.37	18.15	50.26	20.23
PepGLAD	3.83	80.24	19.34	10.47	20.39	75.07	32.10
PepHAR	2.68	**84.91**	86.74	20.53	16.62	79.11	29.58
PepBFN	3.58	80.40	**86.97**	**22.26**	17.56	**79.79**	**32.47**

Table 3: Evaluation of six different methods for the peptide de novo design task.



(a) (b)

Figure 4: (a) Sequence stability of peptides during generation in peptide binder design task, measured by the fraction of sequences that no longer change. (b) Trajectories of Gaussian mixture component means for side-chain torsion angles.

	MAE (deg) ↓				Correct % ↑
	χ_1	χ_2	χ_3	χ_4	
PepBFN_uni	24.50	32.44	56.22	60.35	59.54
PepBFN_sc	**10.75**	**12.26**	**34.25**	**53.21**	**75.24**

Table 4: Unimodal Gaussian vs Gaussian mixture in side-chain packing task.

PepHAR, our method achieves lower binding energies despite exhibiting slightly higher RMSD values. This suggests that achieving strong binding affinity does not require minimal RMSD, as moderate deviations, particularly outside the binding interface, do not significantly disrupt critical binding interactions. As shown in Fig. 3, our method can generate peptides with different binding gestures while maintaining better binding affinities.

Together, experiments across three distinct tasks highlight the advantages of fully continuous parameterization for multimodal peptide modeling and underscore the effectiveness of the PepBFN framework.

6 Ablation Studies

6.1 Fast Convergence and High Sequence Stability

We first investigate whether modeling continuous-discrete data distributions in fully continuous parameter space can alleviate the aforementioned data-type mismatch. As shown in Fig. 4a, PepBFN converges significantly faster and achieves higher sequence stability than PepFlow, which directly models peptide distributions in hybrid continuous-discrete data space. This finding confirms that a fully continuous parameterization helps resolve the mismatch issue and facilitates faster, more stable generation.

6.2 Effective Gaussian Mixture-based Modeling

To further evaluate the role of Gaussian mixture distributions in modeling side-chain torsions, we replace the mixture with a single Gaussian (PepBFN_uni) while keeping all other components unchanged. Both PepBFN_uni and PepBFN_sc are trained with fixed backbones and sequences (without side-chain contexts) under the Bayesian flow framework. As shown in Table 4, replacing the Gaussian mixture distributions with a single Gaussian significantly reduces side-chain prediction accuracy. In addition, as shown in Fig. 4b, although no explicit periodicity constraint was applied, the initialization, sampling, and Bayesian update mechanism naturally constrain all Gaussian mixture component means within the $[0, 2\pi]$ interval, implicitly preserving the periodic nature of torsion angles.

7 Conclusion

We introduce PepBFN, the first Bayesian Flow Network designed for full-atom peptide generation. Our Gaussian mixture-based Bayesian flow framework effectively captures the multimodal nature of side-chain torsion angle distributions. Besides, by jointly modeling residue orientations, torsional angles, centroids, and amino acid types in fully continuous parameter space, PepBFN enables smooth parameter updates and exhibits stable peptide sequence generation. Evaluated across three peptide design benchmarks, PepBFN consistently outperforms existing methods. While this work establishes the potential of Bayesian flow networks for peptide design, there remains room for improvement. For instance, incorporating backbone generation in earlier steps and modeling sequence and side-chain generations in later steps may reduce the complexity of the task and further enhance performance. Overall, PepBFN establishes a principled and unified Bayesian flow framework for peptide design, providing a foundational tool to advance peptide engineering.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 62172273), the Science and Technology Commission of Shanghai Municipality (Grant No. 24510714300), and the Shanghai Municipal Science and Technology Major Project, China (Grant No. 2021SHZDZX0102).

References

Abramson, J.; Adler, J.; Dunger, J.; Evans, R.; Green, T.; Pritzel, A.; Ronneberger, O.; Willmore, L.; Ballard, A. J.; Bambrick, J.; et al. 2024. Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature*, 1–3.

Atkinson, T.; Barrett, T. D.; Cameron, S.; Guloglu, B.; Greenig, M.; Tan, C. B.; Robinson, L.; Graves, A.; Copoiu, L.; and Laterre, A. 2025. Protein sequence modelling with Bayesian flow networks. *Nature Communications*, 16(1): 3197.

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. 2021. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373(6557): 871–876.

Bhat, S.; Palepu, K.; Hong, L.; Mao, J.; Ye, T.; Iyer, R.; Zhao, L.; Chen, T.; Vincoff, S.; Watson, R.; et al. 2023. De Novo Design of Peptide Binders to Conformationally Diverse Targets with Contrastive Language Modeling. *bioRxiv*, 2023–06.

Bryant, P.; and Elofsson, A. 2022. EvoBind: in silico directed evolution of peptide binders with AlphaFold. *bioRxiv*, 2022–07.

Cao, L.; Coventry, B.; Goreshnik, I.; Huang, B.; Sheffler, W.; Park, J. S.; Jude, K. M.; Marković, I.; Kadam, R. U.; Verschueren, K. H.; et al. 2022. Design of protein-binding proteins from the target structure alone. *Nature*, 605(7910): 551–560.

Chaudhury, S.; Lyskov, S.; and Gray, J. J. 2010. PyRosetta: a script-based interface for implementing molecular modeling algorithms using Rosetta. *Bioinformatics*, 26(5): 689–691.

Chen, T.; Dumas, M.; Watson, R.; Vincoff, S.; Peng, C.; Zhao, L.; Hong, L.; Pertsemlidis, S.; Shaepers-Cheu, M.; Wang, T. Z.; et al. 2024. PepMLM: target sequence-conditioned generation of therapeutic peptide binders via span masked language modeling. *ArXiv*, arXiv–2310.

Craik, D. J.; Fairlie, D. P.; Liras, S.; and Price, D. 2013. The future of peptide-based drugs. *Chemical biology & drug design*, 81(1): 136–147.

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. 2022. Robust deep learning–based protein sequence design using ProteinMPNN. *Science*, 378(6615): 49–56.

Davda, J.; Declerck, P.; Hu-Lieskovan, S.; Hickling, T. P.; Jacobs, I. A.; Chou, J.; Salek-Ardakani, S.; and Kraynov, E. 2019. Immunogenicity of immunomodulatory, antibody-based, oncology therapeutics. *Journal for immunotherapy of cancer*, 7: 1–9.

Dhariwal, P.; and Nichol, A. 2021. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34: 8780–8794.

Di, L. 2015. Strategic approaches to optimizing peptide ADME properties. *The AAPS journal*, 17: 134–143.

Downs, T. D. 1972. Orientation statistics. *Biometrika*, 59(3): 665–676.

Ferruz, N.; Schmidt, S.; and Höcker, B. 2022. ProtGPT2 is a deep unsupervised language model for protein design. *Nature communications*, 13(1): 4348.

Fosgerau, K.; and Hoffmann, T. 2015. Peptide therapeutics: current status and future directions. *Drug discovery today*, 20(1): 122–128.

Giordano, C.; Marchiò, M.; Timofeeva, E.; and Biagini, G. 2014. Neuroactive peptides as putative mediators of antiepileptic ketogenic diets. *Frontiers in neurology*, 5: 63.

Graves, A.; Srivastava, R. K.; Atkinson, T.; and Gomez, F. 2023. Bayesian flow networks. *arXiv preprint arXiv:2308.07037*.

Henninot, A.; Collins, J. C.; and Nuss, J. M. 2018. The current state of peptide drug discovery: back to the future? *Journal of medicinal chemistry*, 61(4): 1382–1414.

Hsu, C.; Verkuil, R.; Liu, J.; Lin, Z.; Hie, B.; Sercu, T.; Lerer, A.; and Rives, A. 2022. Learning inverse folding from millions of predicted structures. In *International conference on machine learning*, 8946–8970. PMLR.

Huber, M. F.; Bailey, T.; Durrant-Whyte, H.; and Hanebeck, U. D. 2008. On entropy approximation for Gaussian mixture random vectors. In *2008 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, 181–188. IEEE.

Khatri, C.; and Mardia, K. V. 1977. The von Mises–Fisher matrix distribution in orientation statistics. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 39(1): 95–106.

Kong, X.; Jia, Y.; Huang, W.; and Liu, Y. 2024. Full-atom peptide design with geometric latent diffusion. *arXiv preprint arXiv:2402.13555*.

Krivov, G. G.; Shapovalov, M. V.; and Dunbrack Jr, R. L. 2009. Improved prediction of protein side-chain conformations with SCWRL4. *Proteins: Structure, Function, and Bioinformatics*, 77(4): 778–795.

Leach, A.; Schmon, S. M.; Degiacomi, M. T.; and Willcocks, C. G. 2022. Denoising Diffusion Probabilistic Models on SO(3) for Rotational Alignment. In *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*.

Lefèvre, F.; Rémy, M.-H.; and Masson, J.-M. 1997. Alanine-stretch scanning mutagenesis: a simple and efficient method to probe protein structure and function. *Nucleic acids research*, 25(2): 447–448.

Leman, J. K.; Weitzner, B. D.; Lewis, S. M.; Adolf-Bryfogle, J.; Alam, N.; Alford, R. F.; Aprahamian, M.; Baker, D.; Barlow, K. A.; Barth, P.; et al. 2020. Macromolecular modeling and design in Rosetta: recent methods and frameworks. *Nature methods*, 17(7): 665–680.

Li, J.; Chen, T.; Luo, S.; Cheng, C.; Guan, J.; Guo, R.; Wang, S.; Liu, G.; Peng, J.; and Ma, J. 2024a. Hotspot-Driven Peptide Design via Multi-Fragment Autoregressive Extension. *arXiv preprint arXiv:2411.18463*.

Li, J.; Cheng, C.; Wu, Z.; Guo, R.; Luo, S.; Ren, Z.; Peng, J.; and Ma, J. 2024b. Full-Atom Peptide Design based on Multi-modal Flow Matching. *arXiv preprint arXiv:2406.00735*.

Lin, H.; Zhang, O.; Zhao, H.; Jiang, D.; Wu, L.; Liu, Z.; Huang, Y.; and Li, S. Z. 2024. PPFLOW: Target-Aware Peptide Design with Torsional Flow Matching. *bioRxiv*, 2024–03.

Lipman, Y.; Chen, R. T.; Ben-Hamu, H.; Nickel, M.; and Le, M. 2022. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*.

Lisanza, S. L.; Gershon, J. M.; Tipps, S.; Arnoldt, L.; Hendel, S.; Sims, J. N.; Li, X.; and Baker, D. 2023. Joint generation of protein sequence and structure with RoseTTAFold sequence space diffusion. *bioRxiv*, 2023–05.

Liu, X.; Gong, C.; and Liu, Q. 2022. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*.

McPartlon, M.; and Xu, J. 2023. 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.

Misiura, M.; Shroff, R.; Thyer, R.; and Kolomeisky, A. B. 2022. DLPacker: Deep learning for prediction of amino acid side chain conformations in proteins. *Proteins: Structure, Function, and Bioinformatics*, 90(6): 1278–1290.

Muttenthaler, M.; King, G. F.; Adams, D. J.; and Alewood, P. F. 2021. Trends in peptide drug discovery. *Nature reviews Drug discovery*, 20(4): 309–325.

Peng, X.; Guan, J.; Liu, Q.; and Ma, J. 2023. Moldiff: Addressing the atom-bond inconsistency problem in 3d molecule diffusion generation. *arXiv preprint arXiv:2305.07508*.

Qu, Y.; Qiu, K.; Song, Y.; Gong, J.; Han, J.; Zheng, M.; Zhou, H.; and Ma, W.-Y. 2024. MolCRAFT: structure-based drug design in continuous parameter space. *arXiv preprint arXiv:2404.12141*.

Quartararo, A. J.; Gates, Z. P.; Somsen, B. A.; Hartrampf, N.; Ye, X.; Shimada, A.; Kajihara, Y.; Ottmann, C.; and Pentelute, B. L. 2020. Ultra-large chemical libraries for the discovery of high-affinity peptide binders. *Nature communications*, 11(1): 3183.

Song, J.; Meng, C.; and Ermon, S. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*.

Song, Y.; Gong, J.; Xu, M.; Cao, Z.; Lan, Y.; Ermon, S.; Zhou, H.; and Ma, W.-Y. 2023. Equivariant flow matching with hybrid probability transport for 3d molecule generation. *Advances in Neural Information Processing Systems*, 36: 549–568.

Song, Y.; Gong, J.; Zhou, H.; Zheng, M.; Liu, J.; and Ma, W.-Y. 2024. Unified generative modeling of 3d molecules with bayesian flow networks. In *The Twelfth International Conference on Learning Representations*.

Song, Y.; Sohl-Dickstein, J.; Kingma, D. P.; Kumar, A.; Ermon, S.; and Poole, B. 2020. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*.

Swanson, S.; Sivaraman, V.; Grigoryan, G.; and Keating, A. E. 2022. Tertiary motifs as building blocks for the design of protein-binding peptides. *Protein Science*, 31(6): e4322.

Wang, F.; Wang, Y.; Feng, L.; Zhang, C.; and Lai, L. 2024. Target-Specific De Novo Peptide Binder Design with DiffPepBuilder. *Journal of Chemical Information and Modeling*.

Wang, L.; Wang, N.; Zhang, W.; Cheng, X.; Yan, Z.; Shao, G.; Wang, X.; Wang, R.; and Fu, C. 2022. Therapeutic peptides: current applications and future directions. *Signal transduction and targeted therapy*, 7(1): 48.

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. 2023. De novo design of protein structure and function with RFdiffusion. *Nature*, 620(7976): 1089–1100.

Wei, H.; Wang, W.; Peng, Z.; and Yang, J. 2024. Q-biolip: A comprehensive resource for quaternary structure-based protein–ligand interactions. *Genomics, Proteomics and Bioinformatics*, 22(1): qzae001.

Wen, Z.; He, J.; Tao, H.; and Huang, S.-Y. 2019. PepBDB: a comprehensive structural database of biological peptide–protein interactions. *Bioinformatics*, 35(1): 175–177.

Wu, F.; Zhou, Z.; Jin, S.; Zeng, X.; Leskovec, J.; and Xu, J. 2025. Surface-based Molecular Design with Multi-modal Flow Matching. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, KDD '25, 3192–3203. New York, NY, USA: Association for Computing Machinery. ISBN 9798400714542.

Xu, J.; and Zhang, Y. 2010. How significant is a protein structure similarity with TM-score= 0.5? *Bioinformatics*, 26(7): 889–895.

Zhang, Y.; and Skolnick, J. 2005. TM-align: a protein structure alignment algorithm based on the TM-score. *Nucleic acids research*, 33(7): 2302–2309.

Zhang, Y.; Zhang, Z.; Zhong, B.; Misra, S.; and Tang, J. 2023. Diffpack: A torsional diffusion model for autoregressive protein side-chain packing. *Advances in Neural Information Processing Systems*, 36: 48150–48172.