

815	Contents	
816	1 Introduction	1
817	2 Preliminary	2
818	3 Bayesian flow for SO(3) generation	3
819	3.1 Theoretical challenge in directly building SO(3) Bayesian flow	3
820	3.2 Transforming SO(3) generation into hypersphere \mathbb{S}^3 generation	4
821	3.3 Bayesian flow on hypersphere \mathbb{S}^d	5
822	4 Rationalized all-atom protein design	5
823	4.1 Bayesian flow for each protein modality	6
824	4.2 The information shortcut problem	7
825	5 Related work	8
826	6 Experiments	8
827	6.1 Peptide design	8
828	6.2 Antibody design	9
829	6.3 Ablation study	9
830	7 Conclusion	9
831	A Proofs of propositions	22
832	B Detailed derivation of SO(3) Bayesian flow	23
833	B.1 Hypersphere and von Mises Fisher distribution	23
834	B.2 Bayesian update function h	23
835	B.3 Bayesian flow distribution $p_F(\theta_i \mathbf{x}; \alpha_{1:i})$	24
836	B.4 Discrete-time loss with n steps $L^n(\mathbf{x})$	24
837	C SO(3) toy data experiment	25
838	D Detailed training and sampling procedure	26
839	E Limitations and broader impacts	27
840	F Experiment details	28
841	F.1 Rationalized information flow implementation	28
842	F.2 Peptide design	29
843	F.3 Antibody design	29
844	F.4 Other implementation details	30
845	G Comparison between BFN and SDEs/ODEs	30

846	H More results	30
847	H.1 Error bars	30
848	H.2 Visualization	31

849	I Further discussion of information shortcut	32
-----	---	-----------

850 A Proofs of propositions

851 **Proposition 3.** Every distribution $p_{SO(3)}(\mathbf{R})$ supported on $SO(3)$ can be bijectively mapped to a
852 distribution $p_{\mathbb{S}^3}(\mathbf{q})$ supported on \mathbb{S}^3 which satisfies $p_{\mathbb{S}^3}(\mathbf{q}) = p_{\mathbb{S}^3}(-\mathbf{q})$.

853 *Proof.* Given each distribution $p_{SO(3)}(\mathbf{R})$ support on the $SO(3)$ group, we can build the correspond-
854 ing \mathbb{S}^3 measure $p_{SO(3)}(\mathbf{R})$ using the surjective homomorphism $\phi : SU(2) \rightarrow SO(3)$:

$$\forall \mathbf{q} \in \mathbb{S}^3, p_{\mathbb{S}^3}(\mathbf{q}) := p_{SO(3)}(\phi^{-1}(\mathbf{R}))/2 \quad (13)$$

855 Now we prove that $p_{\mathbb{S}^3}(\mathbf{q})$ satisfies all the probability distribution axioms and the antipodal symmetry:

- 856 1. Non-negativity: $\forall \mathbf{q} \in \mathbb{S}^3, p_{\mathbb{S}^3}(\mathbf{q}) = p_{SO(3)}(\phi^{-1}(\mathbf{R}))/2 \geq 0$
- 857 2. Normalization: $\int_{\mathbb{S}^3} p_{\mathbb{S}^3}(\mathbf{q}) d\mathbb{S}^3 = \frac{1}{2} \int_{\mathbb{S}^3} p(\mathbf{R}) d\mathbb{S}^3 = \frac{1}{2} \int_{SO(3)} p(\mathbf{R}) dSO(3) = 1$
- 858 3. Antipodal symmetry: $p_{\mathbb{S}^3}(\mathbf{q}) = \frac{1}{2} p_{SO(3)}(\phi^{-1}(\mathbf{R})) = p_{\mathbb{S}^3}(-\mathbf{q})$

859 For $p_{\mathbb{S}^3}(\mathbf{q}) = p_{\mathbb{S}^3}(-\mathbf{q})$, we can also build $p_{SO(3)}(\mathbf{R}) := 2p_{\mathbb{S}^3}(\mathbf{q}) = 2p_{\mathbb{S}^3}(-\mathbf{q}), \forall \mathbf{R} \in SO(3)$. We
860 can check that $p_{SO(3)}(\mathbf{R}) \geq 0$ and $\int_{SO(3)} p_{SO(3)}(\mathbf{R}) dSO(3) = 1$. Therefore, $p_{SO(3)}$ is a probability
861 density function on $SO(3)$. \square

862 **Proposition 4.** With Ψ as an antipodal equivariant function, i.e. $\Psi(-\theta^q) = -\Psi(\theta^q)$, and
863 uniform prior $p(\theta_0^q)$ on \mathbb{S}^3 , the marginal distribution defined by multiple Markov transitions
864 $p_{\Psi}(\theta_n^q) = p(\theta_0^q) \int \prod_{i=1}^n p_F(\theta_i^q | \Psi(\theta_{i-1}^q); \alpha_{1:i}) d\theta_{1:n-1}^q$ is antipodal invariant, which is equivalent to
865 a distribution $p_{\Psi}(\mathbf{R})$.

Proof.

$$\begin{aligned}
p_{\Psi}(-\theta_n^q) &= p(-\theta_0^q) \int p_{\Psi}(-\theta_{n:1}^q | -\theta_0^q) d\theta_{1:n-1}^q \\
&= p(-\theta_0^q) \int \prod_{t=0}^{n-1} p_{\Psi}(-\theta_{t+1}^q | -\theta_t^q) d\theta_{1:n-1}^q \\
&= p(-\theta_0^q) \int \prod_{t=0}^{n-1} p_F(-\theta_{t+1}^q | \Psi(-\theta_{i-1}^q); \alpha_{1:i}) d\theta_{1:n-1}^q \\
&= p(-\theta_0^q) \int \prod_{t=0}^{n-1} p_F(-\theta_{t+1}^q | -\Psi(\theta_{i-1}^q); \alpha_{1:i}) d\theta_{1:n-1}^q \\
&= p(-\theta_0^q) \int \prod_{t=0}^{n-1} p_F(\theta_{i-1}^q | \Psi(\theta_{i-1}^q); \alpha_{1:i}) d\theta_{1:n}^q \\
&= p(\theta_0^q) \int \prod_{t=0}^{n-1} p_F(\theta_i^q | \Psi(\theta_{i-1}^q); \alpha_{1:i}) d\theta_{1:n}^q \\
&= p_{\Psi}(\theta_n^q).
\end{aligned}$$

866 Therefore, the marginal distribution $p(\theta_n^q)$ is antipodal invariant. \square

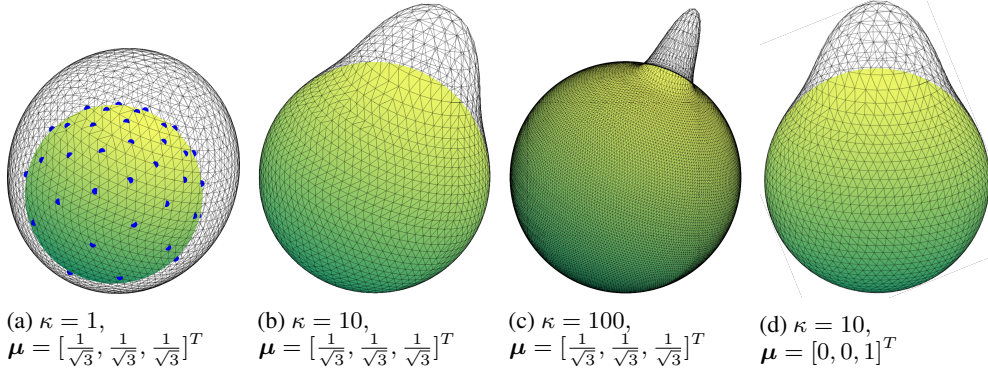


Figure 5: Visualization of the von Mises Fisher distribution with different parameters with $d = 2$. Those figures are from Frisch and Hanebeck [2023].

B Detailed derivation of SO(3) Bayesian flow

B.1 Hypersphere and von Mises Fisher distribution

Given an Euclidean space \mathbb{R}^d , the hypersphere \mathbb{S}^{d-1} located in \mathbb{R}^d is defined as:

$$\mathbb{S}^{d-1} = \{\mathbf{x} : \mathbf{x} \in \mathbb{R}^d, \|\mathbf{x}\|_2 = 1\} \quad (14)$$

The von Mises Fisher distribution [Mardia and Jupp, 2009] is a uni-modal distribution on \mathbb{S}^{d-1} with the location parameter $\boldsymbol{\mu}$ and concentration parameter κ . The parameters $\boldsymbol{\mu}$ and κ are analogous to the mean μ and variance $1/\sigma^2$ in the normal distribution: $\boldsymbol{\mu}$ represents the central location around which the distribution is concentrated, and κ serves as a measure of concentration. We provide a visualization of vMF distribution with different $\boldsymbol{\mu}$ and κ in Fig. 5. The probability density function of vMF distribution is:

$$f_p(\mathbf{x}; \boldsymbol{\mu}, \kappa) = vMF(\mathbf{x}|\boldsymbol{\mu}, \kappa) = C_p(\kappa) \exp(\kappa \boldsymbol{\mu}^T \mathbf{x}) = \frac{\kappa^\nu}{(2\pi)^{\nu+1} I_\nu(\kappa)} \exp(\kappa \boldsymbol{\mu}^T \mathbf{x}), \quad (15)$$

where $\|\boldsymbol{\mu}\| = \|\mathbf{x}\| = 1$, $\nu = d/2 - 1$, $\kappa \geq 0$, $I_\nu(\kappa)$ denotes the modified Bessel function of the first kind at order ν . The differential entropy of $vMF(\boldsymbol{\mu}, \kappa)$ is:

$$H(vMF(\boldsymbol{\mu}, \kappa)) = -\kappa \frac{I_{\nu+1}(\kappa)}{I_\nu(\kappa)} + (\nu + 1) \ln 2\pi + \kappa + \ln I_\nu(\kappa) - \nu \ln \kappa \quad (16)$$

We use vMF's entropy Eq. (16) to determine the accuracy schedule in the following sections.

B.2 Bayesian update function h

Sender distribution For each time step i , the sender corrupts the clean data $\mathbf{x} \sim p_{\text{data}}$ using the sender distribution p_S and the signal-to-noise ratio parameter α_i , gets the noised information $\mathbf{y} \sim p_S$, and send \mathbf{y} to the receiver. Here, we use the vMF distribution to instantiate p_S for hypersphere data $\mathbf{x} \in \mathbb{S}^{d-1}$:

$$p_S(\mathbf{y}|\mathbf{x}; \alpha) = vMF(\mathbf{y}|\mathbf{x}, \alpha) \quad (17)$$

Input distribution As for the receiver, its belief over the ground truth data \mathbf{x} is formulated by a distribution family with parameter $\boldsymbol{\theta}$, *e.g.*, Gaussian distribution with μ and σ for Euclidean space data $\mathbf{x} \in \mathbb{R}^1$, which can be expressed as:

$$p_I(\mathbf{x}|\boldsymbol{\theta}) = vMF(\mathbf{x}|\boldsymbol{\mu}, \kappa) \quad (18)$$

Bayesian update function For each time step i , the receiver updates its prior belief $\boldsymbol{\theta}_{i-1}$ according to the observed noised data \mathbf{y}_i with known accuracy α_i according to the Bayesian theorem. This process is formulated as the Bayesian update function $\boldsymbol{\theta}_i = h(\boldsymbol{\theta}_{i-1}, \mathbf{y}_i, \alpha_i)$. The Bayesian update

function h for von Mises Fisher distribution is derived as follows:

$$\begin{aligned}
p(\mathbf{x}|\mathbf{y}) &= p(\mathbf{y}|\mathbf{x}; \alpha)p(\mathbf{x}; \boldsymbol{\mu}_{i-1}, \kappa_{i-1})/p(\mathbf{y}) \\
&\propto p(\mathbf{y}|\mathbf{x}; \alpha)p(\mathbf{x}; \boldsymbol{\mu}_{i-1}, \kappa_{i-1}) \\
&= vMF(\mathbf{y}|\mathbf{x}, \alpha)vMF(\mathbf{x}|\boldsymbol{\mu}_{i-1}, \kappa_{i-1}) \\
&\propto \exp\{\alpha \mathbf{x}^T \mathbf{y} + \kappa_{i-1} \boldsymbol{\mu}_{i-1}^T \mathbf{x}\} \\
&= vMF(\mathbf{x}|\boldsymbol{\mu}_i, \kappa_i)
\end{aligned}$$

where $\kappa_i \boldsymbol{\mu}_i = \kappa_{i-1} \boldsymbol{\mu}_{i-1} + \alpha_i \mathbf{y}_i$. Therefore, the Bayesian update function is:

$$h(\{\boldsymbol{\mu}_{i-1}, \kappa_{i-1}\}, \mathbf{y}_i, \alpha_i) = \{\boldsymbol{\mu}_i, \kappa_i\}, \text{ where } \kappa_i \boldsymbol{\mu}_i = \kappa_{i-1} \boldsymbol{\mu}_{i-1} + \alpha_i \mathbf{y}_i \quad (19)$$

We define $\boldsymbol{\theta} \stackrel{\text{def}}{=} \kappa \boldsymbol{\mu}$ to compactly denote the parameter of the vMF distribution $\{\kappa, \boldsymbol{\mu}\}$.

B.3 Bayesian flow distribution $p_F(\boldsymbol{\theta}_i|\mathbf{x}; \alpha_{1:i})$

Given time index i and clean data \mathbf{x} , the Bayesian flow distribution for $\boldsymbol{\theta}_i$ is defined as the marginal distribution after conducting multiple Bayesian updates over noised data \mathbf{y}_i :

$$p_F(\boldsymbol{\theta}_i|\mathbf{x}; \alpha_{1:i}) = \int vMF(\mathbf{y}_1|\mathbf{x}, \alpha_1) \dots vMF(\mathbf{y}_i|\mathbf{x}, \alpha_i) \delta(\boldsymbol{\theta}_i - \sum_{j=1}^i \alpha_j \mathbf{y}_j) \quad (20)$$

For determining the accuracy schedule $\alpha_{1:i}$ according to entropy, we first formalize the entropy of the Bayesian flow:

$$H(t) \stackrel{\text{def}}{=} \int p_F(\boldsymbol{\theta}_i|\mathbf{x}; \alpha_{1:i}) H(p_I(\cdot|\boldsymbol{\theta}_i)), \text{ where } i = nt + 1 \quad (21)$$

The entropy of the Bayesian flow is dominated by the accuracy schedule $\alpha_{1:n}$. Although Eq. (21) is not tractable, we can evaluate its value at each step given the accuracy schedule $\alpha_{1:n}$. For synchronizing the entropy across each modality, we find the accuracy schedule that $\alpha_{1:n}$ such that the entropy of this Bayesian flow linearly decreases with respect to time:

$$H(t) = (1-t)H(0) + tH(1) \quad (22)$$

Although Eq. (22) is not analytically tractable, we can solve this equation using binary search to find each α_i at each step i .

B.4 Discrete-time loss with n steps $L^n(\mathbf{x})$

Receiver distribution Given $\boldsymbol{\theta}_i$ from the Bayesian flow distribution as input, the receiver uses the network Ψ to improve its belief considering interdependency across modalities and dimensions, which is referred to as the output distribution. We choose to parameterize the output distribution as delta distribution to directly predict the ground truth \mathbf{x} following [Graves et al., 2023]:

$$p_O(\mathbf{x}|\Psi(\boldsymbol{\theta}_i, t_i)) = \delta(\mathbf{x} - \Psi^{\mathbf{x}}(\boldsymbol{\theta}_i, t_i)) \quad (23)$$

To approximate the sender's distribution, the receiver's distribution is obtained by convolving the output distribution with the sender's distribution:

$$p_R(\mathbf{y}|\boldsymbol{\theta}_i, t_i, \alpha_i) = \mathbb{E}_{p_O(\mathbf{x}'|\Psi(\boldsymbol{\theta}_i, t_i))} p_S(\mathbf{y}|\mathbf{x}'; \alpha_i) = vMF(\mathbf{y}|\Psi^{\mathbf{x}}(\boldsymbol{\theta}_i, t_i), \alpha_i) \quad (24)$$

Discrete-time loss with n steps The discrete time loss is defined as the KL divergence between sender and receiver distribution:

$$L^n(\mathbf{x}) = n \int_{i \sim U\{1, n\}, p_F(\boldsymbol{\theta}_i|\mathbf{x}; \alpha_{1:i})} \mathbb{E} D_{KL}(p_S(\cdot|\mathbf{x}; \alpha_i) \parallel p_R(\cdot|\boldsymbol{\theta}_i, t_i, \alpha_i)) \quad (25)$$

$$= n \int_{i \sim U\{1, n\}, p_F(\boldsymbol{\theta}_i|\mathbf{x}; \alpha_{1:i})} \mathbb{E} \alpha_i \frac{I_{\nu+1}(\alpha_i)}{I_{\nu}(\alpha_i)} (1 - \text{dot}(\mathbf{x}, \Psi(\boldsymbol{\theta}_i, t_i))) \quad (26)$$

In fact, the term $(1 - \text{dot}(\mathbf{x}, \Psi(\boldsymbol{\theta}_i, t_i)))$ in Eq. (25) is proportional to the geodesic distance between the corresponding orientation matrix $\mathbf{R} = \phi(\mathbf{x})$ and $\hat{\mathbf{R}} = \phi(\hat{\mathbf{x}}) = \phi(\Psi(\boldsymbol{\theta}_i, t_i))$.

915 **Proposition 5.** *The training loss defined in Eq. (8) is proportional to the geodesic distance between*
 916 *the predicted orientation matrix $\hat{\mathbf{R}} = \phi(\Psi^{\mathbf{q}}(\boldsymbol{\theta}^{\mathcal{P}}, t))$ and the ground truth orientation matrix \mathbf{R} up to*
 917 *a constant.*

918 *Proof.* We first introduce the (w, \mathbf{v}) representation for unit quaternions. Specifically, for a unit
 919 quaternion $\mathbf{q} = [a, b, c, d]^T$, we define the scalar part as $w = a$ and the vector part as $\mathbf{v} = bi + cj + dk$,
 920 so that the quaternion can be written as $\mathbf{q} = w + \mathbf{v}$, where $\mathbf{i}, \mathbf{j}, \mathbf{k}$ are the standard basis elements
 921 corresponding to the three Cartesian axes. This representation is particularly convenient because the
 922 rotation angle ϑ of the rotation matrix $\mathbf{R} = \phi^{-1}(\mathbf{q})$ satisfies the relation

$$w = a = \cos(\vartheta/2) \quad [\text{Shoemake, 1985}] \quad (27)$$

923 Using the (w, \mathbf{v}) representation, the geodesic distance between two unit quaternions $\mathbf{q} = w_1 + \mathbf{v}_1$
 924 and $\hat{\mathbf{q}} = w_2 + \mathbf{v}_2$ on \mathbb{S}^3 is given by:

$$d_{\mathbb{S}^3}(\mathbf{q}, \hat{\mathbf{q}}) = \arccos(\langle \mathbf{q}, \hat{\mathbf{q}} \rangle) = \arccos(w_1 w_2 + \langle \mathbf{v}_1, \mathbf{v}_2 \rangle), \quad (28)$$

925 where $\langle \cdot, \cdot \rangle$ denotes the standard Euclidean inner product.

926 The geodesic distance between the corresponding rotation matrices $\mathbf{R} = \phi(\mathbf{q})$ and $\hat{\mathbf{R}} = \phi(\hat{\mathbf{q}})$ in the
 927 rotation group $\text{SO}(3)$ is the rotation angle of the relative rotation $\mathbf{R}_r = \mathbf{R}^\top \hat{\mathbf{R}}$. This angle can be
 928 computed via the quaternion representation of \mathbf{R}_r :

$$\mathbf{q}_r = \phi^{-1}(\mathbf{R}_r) = \mathbf{q}^{-1} \times \hat{\mathbf{q}} = [w_1 w_2 + \langle \mathbf{v}_1, \mathbf{v}_2 \rangle, w_1 \mathbf{v}_2 + w_2 \mathbf{v}_1 + \mathbf{v}_1 \times \mathbf{v}_2]. \quad (29)$$

929 The corresponding rotation angle ϑ of \mathbf{R}_r is then:

$$d_{\text{SO}(3)}(\mathbf{R}, \hat{\mathbf{R}}) = \vartheta = 2 \arccos(w_1 w_2 + \langle \mathbf{v}_1, \mathbf{v}_2 \rangle) = 2 d_{\mathbb{S}^3}(\mathbf{q}, \hat{\mathbf{q}}). \quad (30)$$

930 The proof is done. □

931 C SO(3) toy data experiment

932 In this section, we use the synthetic dataset in Brofos et al. [2021] to verify the effectiveness of the
 933 proposed SO(3) Bayesian flow over SDE/ODE based methods following [Bose et al., 2023].

934 **Toy dataset network** The antipodal equivariance of the toy dataset network is achieved by the
 935 following formula:

$$\Psi_{\text{toy}}^{\mathbf{q}}(\boldsymbol{\theta}^{\mathbf{q}}, t) = \text{Normalize}(\boldsymbol{\mu} \otimes \text{MLP}(\boldsymbol{\mu} \boldsymbol{\mu}^T, \kappa, t) + \boldsymbol{\mu}) \quad (31)$$

936 where \otimes denotes the element-wise product between two vectors, and $\text{Normalize}(\mathbf{q}) = \mathbf{q}/\|\mathbf{q}\|_2$.

937 **Metrics** The metrics W1 and W2 refer to the Wasserstein p -distance between the generated distri-
 938 bution and the ground truth test distribution with $p = 1$ and $p = 2$, respectively. These two metrics
 939 measure the distributional distance between the generated distribution and the ground truth data
 940 distribution.

941 **Results** We present the visualized generated results in Fig. 6 and the corresponding evaluation metrics
 942 in Tab. 5. As shown in Fig. 6, ProteoBayes produces higher-quality samples with fewer outliers
 943 compared to the SDE and ODE baselines. This observation is consistent with the improved W1 and
 944 W2 scores reported in Tab. 5, further demonstrating the effectiveness of the proposed SO(3) Bayesian
 945 flow.

946 We further validate the ineffectiveness of the Euler angle and axis-angle rotation representations
 947 in Fig. 6f and Fig. 6g, as discussed in the main text. Specifically, we employ the hypertorus
 948 BFN [Wu et al., 2025] to generate the Euler angle representation, and a combination of the hypertorus
 949 and hypersphere BFN to produce the axis-angle representation. As shown, both representations
 950 fail to capture the underlying SO(3) group structure, resulting in poor SO(3) generative modeling
 951 performances.

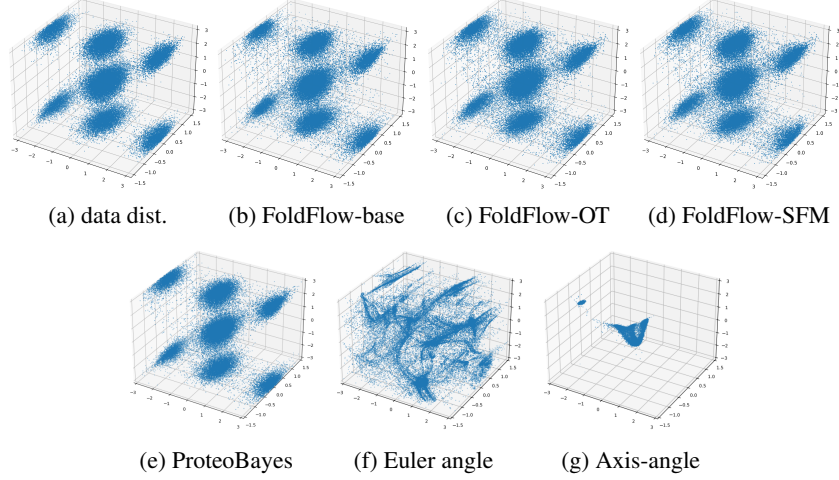


Figure 6: The data is visualized using the Euler-angle representation of the rotation matrices. (a)-(d) are from Bose et al. [2023]

Table 5: Toy dataset performance comparison. Baseline results are from [Bose et al., 2023].

Method	W1 ($\times 10^{-2}$, \downarrow)	W2 ($\times 10^{-1}$, \downarrow)
FoldFlow-base	5.39 ± 0.88	1.52 ± 0.27
FoldFlow-OT	4.96 ± 0.27	1.25 ± 0.12
FoldFlow-SFM	4.92 ± 1.56	1.26 ± 0.49
Simulated SDE	5.13 ± 1.36	1.33 ± 0.44
ProteoBayes	3.23 ± 0.68	0.99 ± 0.33

D Detailed training and sampling procedure

We describe the training loss functions used for the remaining modalities:

C_α position p training loss Sampling μ_{i-1}^p from p_F^p , the training loss for position p is calculated as the discrete time loss for continuous Euclidean space variable:

$$\mathcal{L}_p = \frac{n}{2} \left(1 - \sigma_1^{2/n} \right) \mathbb{E}_{i \sim U\{1, n\}, p_F^p(\mu_{i-1}^p | p; t_{i-1})} \frac{\|p - \Psi_{i-1}^p(\theta^p, t)\|^2}{\sigma_1^{2i/n}} \quad (32)$$

Sequence \mathcal{S} training loss Sampling θ_i^S from p_F^S , the training loss is the n -step discrete-time loss for discrete variable:

$$\mathcal{L}_S = n \mathbb{E}_{i \sim U\{1, n\}, p_F^S(\theta^S | \mathcal{S}; t_{i-1}), \mathcal{N}(\mathbf{y} | \alpha_i(K\mathbf{e}_S - \mathbf{1}), \alpha_i K\mathbf{I})} \ln \mathcal{N}(\mathbf{y} | \alpha_i(K\mathbf{e}_S - \mathbf{1}), \alpha_i K\mathbf{I}) - \sum_{d=1}^L \ln \left(\sum_{k=1}^K p_O^{(d)}(k | \theta^S; t_{i-1}) \mathcal{N}(y^{(d)} | \alpha_i(K\mathbf{e}_k - \mathbf{1}), \alpha_i K\mathbf{I}) \right) \quad (33)$$

where $\mathbf{I} \in \mathbb{R}^{K \times L \times L}$ and $\mathbf{1} \in \mathbb{R}^{K \times D}$.

Angle χ, ψ training loss Sampling θ^χ from p_F^χ , the training loss is the n -step discrete-time loss for periodic variable:

$$\mathcal{L}_\chi = n \mathbb{E}_{i \sim U\{1, n\}, p_F^\chi(\theta^\chi | \chi; \alpha_{1:i})} \alpha_i \frac{I_1(\alpha_i)}{I_0(\alpha_i)} (1 - \cos(\chi - \Psi(\theta_{i-1}^p, t_{i-1}))) \quad (34)$$

Note that for χ , we only calculate the loss for those non-padded values. Also, sampling θ^ψ from p_F^ψ , the training loss for ψ is:

$$\mathcal{L}_\psi = n \mathbb{E}_{i \sim U\{1, n\}, p_F^\psi(\theta^\psi | \psi; \alpha_{1:i})} \alpha_i \frac{I_1(\alpha_i)}{I_0(\alpha_i)} (1 - \cos(\psi - \Psi(\theta_{i-1}^p, t_{i-1}))) \quad (35)$$

Algorithm 1 Training

```
1: Require:  $n, K \in \mathbb{Z}, \sigma_1 \in \mathbb{R}^+, \beta_1 \in \mathbb{R}^+, \alpha_{1:n}^{\chi}, \alpha_{1:n}^{\psi}, \alpha_{1:n}^q \in \mathbb{R}^+, \lambda_q, \lambda_S, \lambda_p, \lambda_{\chi}, \lambda_{\psi}, \lambda_{bb} \in \mathbb{R}^+$   
2: Input: sequence  $\mathcal{S}$ , position  $\mathbf{p}$ , orientation  $\mathbf{R}$ , backbone torsion  $\psi$ , sidechain torsion  $\chi$ ,  $\chi$  mask  $\text{Mask}^{\chi}$   
3: Sample  $i \sim U\{1, n\}, t \leftarrow \frac{(i-1)}{n}$   
4: # randomly select one from the two quaternions  
5:  $\mathbf{q} \sim U\{\phi^{-1}(\mathbf{R})\}$   
6: # sampling from the Bayesian flow distribution of each modality  
7:  $\theta^q \sim p_F^q(\theta^q | \mathbf{R}; \alpha_{1:i}^q)$   
8:  $\theta^S \sim p_F^S(\theta^S | \mathcal{S}; t)$   
9:  $\mu^p \sim p_F^p(\mu^p | \mathbf{p}; t), \theta^p \leftarrow \{\mu^p\}$   
10:  $\mathbf{m}_i^{\chi} \sim p_F^{\chi}(\mathbf{m}_i^{\chi} | \chi; \alpha_{1:i}^{\chi}), \mathbf{c}_i^{\chi} \sim p_F^{\chi}(\mathbf{c}_i^{\chi} | \chi; \alpha_{1:i}^{\chi}),$   
11:  $\mathbf{m}_i^{\chi} \leftarrow \mathbf{m}_i^{\chi} \cdot \text{Mask}^{\chi}(\Psi^S), \mathbf{c}_i^{\chi}[1 - \text{Mask}^{\chi}(\Psi^S)] \leftarrow 1, \theta^{\chi} \leftarrow \{\mathbf{m}_i^{\chi}, \mathbf{c}_i^{\chi}\}$   
12:  $\mathbf{m}_i^{\psi} \sim p_F^{\psi}(\mathbf{m}_i^{\psi} | \psi; \alpha_{1:i}^{\psi}), \mathbf{c}_i^{\psi} \sim p_F^{\psi}(\mathbf{c}_i^{\psi} | \psi; \alpha_{1:i}^{\psi}), \theta^{\psi} \leftarrow \{\mathbf{m}_i^{\psi}, \mathbf{c}_i^{\psi}\}$   
13: # feed all protein modalities into the network  
14:  $\theta^P \leftarrow \{\theta^S, \theta^p, \theta^R, \theta^{\psi}, \theta^{\chi}\}$   
15:  $\Psi^S, \Psi^p, \Psi^R, \Psi^{\psi}, \Psi^{\chi} \leftarrow \Psi(\theta^P, t)$   
16: # calculate losses  
17:  $\mathcal{L}_q \leftarrow n \alpha_i^q \frac{I_{\nu+1}(\alpha_i^q)}{I_{\nu}(\alpha_i^q)} (1 - \text{dot}(\mathbf{q}, \Psi^q(\theta^P, i)))$   
18:  $\mathcal{L}_S \leftarrow n \mathbb{E}_{N(y | \alpha_i(K\mathbf{e}_S - 1), \alpha_i KI)} \ln \mathcal{N}(y | \alpha_i(K\mathbf{e}_S - 1), \alpha_i KI)$   
19:  $-\sum_{d=1}^L \ln \left( \sum_{k=1}^K p_O^{(d)}(k | \theta^S; t_{i-1}) \mathcal{N}(y^{(d)} | \alpha_i(K\mathbf{e}_k - 1), \alpha_i KI) \right)$   
20:  $\mathcal{L}_p \leftarrow \frac{n}{2} \left( 1 - \sigma_1^{2/n} \right) \frac{\|\mathbf{p} - \Psi_{i-1}^p(\theta^P, t)\|^2}{\sigma_1^{2i/n}}$   
21:  $\mathcal{L}_{\chi} \leftarrow n \alpha_i^{\chi} \frac{I_1(\alpha_i^{\chi})}{I_0(\alpha_i^{\chi})} (1 - \cos(\chi - \Psi^{\chi}(\theta_{i-1}^P, t_{i-1})))$   
22:  $\mathcal{L}_{\chi} \leftarrow \mathcal{L}_{\chi} \cdot \text{Mask}^{\chi}(\mathcal{S})$   
23:  $\mathcal{L}_{\psi} \leftarrow n \alpha_i^{\psi} \frac{I_1(\alpha_i^{\psi})}{I_0(\alpha_i^{\psi})} (1 - \cos(\psi - \Psi^{\psi}(\theta_{i-1}^P, t_{i-1})))$   
24: Optimize  $\lambda_q \mathcal{L}_q + \lambda_S \mathcal{L}_S + \lambda_p \mathcal{L}_p + \lambda_{\chi} \mathcal{L}_{\chi} + \lambda_{\psi} \mathcal{L}_{\psi}$ 
```

Algorithm 2 Sampling

```
Require:  $n, K \in \mathbb{Z}, \sigma_1 \in \mathbb{R}^+, \beta_1 \in \mathbb{R}^+, \alpha_{1:n}^{\chi}, \alpha_{1:n}^{\psi}, \alpha_{1:n}^q \in \mathbb{R}^+$   
# initialize the prior parameters  
 $\mu^q \sim U_{\mathbb{S}^3}, \kappa^q \leftarrow \mathbf{0}, \theta^q \leftarrow \{\mu^q, \kappa^q\}; \theta_0^S \leftarrow \frac{1}{K} \mathbf{1}_{1 \times L}; \mu^p \leftarrow \mathbf{0}, \theta^p \leftarrow \{\mu^p\}$   
 $\mathbf{m}^{\chi} \leftarrow U(0, 2\pi), \mathbf{c}^{\chi} \leftarrow \mathbf{0}, \theta^{\chi} \leftarrow \{\mathbf{m}^{\chi}, \mathbf{c}^{\chi}\}; \mathbf{m}^{\psi} \leftarrow U(0, 2\pi), \mathbf{c}^{\psi} \leftarrow \mathbf{0}, \theta^{\psi} \leftarrow \{\mathbf{m}^{\psi}, \mathbf{c}^{\psi}\}$   
for  $i \leftarrow 1, \dots, n; t \leftarrow \frac{i-1}{n}$  do  
  # use network to do inter-dependency modeling across dimensions of all modalities  
   $\theta^P \leftarrow \{\theta^S, \theta^p, \theta^R, \theta^{\psi}, \theta^{\chi}\}$   
   $\Psi^S, \Psi^p, \Psi^R, \Psi^{\psi}, \Psi^{\chi} \leftarrow \Psi(\theta^P, t)$   
  if  $i = n$  then  
    break  
  end if  
  # update each modality with Bayesian flow  
   $\theta^q \sim p_F^q(\theta^q | \Psi^R, \alpha_{1:i}^q)$   
   $\theta^S \sim p_F^S(\theta^S | \Psi^S; t)$   
   $\theta^p \leftarrow \mu^p \sim p_F^p(\mu^p | \mathbf{p}; t)$   
   $\mathbf{m}_i^{\chi} \sim p_F^{\chi}(\mathbf{m}_i^{\chi} | \chi; \alpha_{1:i}^{\chi}), \mathbf{c}_i^{\chi} \sim p_F^{\chi}(\mathbf{c}_i^{\chi} | \chi; \alpha_{1:i}^{\chi}),$   
   $\mathbf{m}_i^{\chi} \leftarrow \mathbf{m}_i^{\chi} \cdot \text{Mask}^{\chi}(\Psi^S), \mathbf{c}_i^{\chi}[1 - \text{Mask}^{\chi}(\Psi^S)] \leftarrow 1, \theta^{\chi} \leftarrow \{\mathbf{m}_i^{\chi}, \mathbf{c}_i^{\chi}\}$   
   $\mathbf{m}_i^{\psi} \sim p_F^{\psi}(\mathbf{m}_i^{\psi} | \psi; \alpha_{1:i}^{\psi}), \mathbf{c}_i^{\psi} \sim p_F^{\psi}(\mathbf{c}_i^{\psi} | \psi; \alpha_{1:i}^{\psi}), \theta^{\psi} \leftarrow \{\mathbf{m}_i^{\psi}, \mathbf{c}_i^{\psi}\}$   
end for  
Return  $\Psi^S, \Psi^p, \Psi^R, \Psi^{\psi}, \Psi^{\chi}$ 
```

963 E Limitations and broader impacts

964 **Limitations** Despite the promising results of ProteoBayes, we acknowledge that the binding affinities
965 and complex stability assessed by Rosetta serve as an initial screening step. Real-world properties of
966 the generated candidates still require validation through costly wet-lab experiments. However, this
967 challenge is shared across the research community.

968 **Broader impacts** Our work contributes to the advancement of unified, end-to-end protein design,
 969 potentially enabling more accurate generation of functional proteins. This has implications for
 970 therapeutic discovery, enzyme engineering, and synthetic biology. However, as with any generative
 971 biological technology, careful oversight is essential to prevent unintended misuse, such as the design
 972 of harmful or dual-use proteins.

973 F Experiment details

Algorithm 3 Invariant point attention with quaternion implementation (IPAq)

Require: $\{s_i\}, \{z_{ij}\}, \{T_i^{\mathcal{Q}}\}, N_{\text{head}} = 12, c = 16, N_{\text{query points}} = 4, N_{\text{point values}} = 8$

- 1: $q_i^h, k_i^h, v_i^h = \text{LinearNoBias}(s_i)$ $\{q_i^h, k_i^h, v_i^h \in \mathbb{R}^c, h \in \{1, \dots, N_{\text{head}}\}\}$
- 2: $q_i^{hp}, k_i^{hp} = \text{LinearNoBias}(s_i)$ $q_i^{hp}, k_i^{hp} \in \mathbb{R}^3, p \in \{1, \dots, N_{\text{query points}}\}$
- 3: $v_i^{hp} = \text{LinearNoBias}(s_i)$ $v_i^{hp} \in \mathbb{R}^3, p \in \{1, \dots, N_{\text{point values}}\}$
- 4: $b_{ij}^h = \text{LinearNoBias}(z_{ij})$
- 5: $w_C = \sqrt{\frac{2}{9N_{\text{query points}}}}$
- 6: $w_L = \sqrt{\frac{1}{3}}$
- 7: $a_{ij}^h = \text{softmax}_j \left(w_L \left(\frac{1}{\sqrt{c}} q_i^{h\top} k_j^h + b_{ij}^h - \frac{\gamma^h w_C}{2} \sum_p \|T_i^{\mathcal{Q}} \circ q_i^{hp} - T_j^{\mathcal{Q}} \circ k_j^{hp}\|^2 \right) \right)$
- 8: $\tilde{o}_i^h = \sum_j a_{ij}^h z_{ij}$
- 9: $o_i^h = \sum_j a_{ij}^h v_j^h$
- 10: $o_i^{hp} = (T_i^{\mathcal{Q}})^{-1} \circ \sum_j a_{ij}^h (T_j^{\mathcal{Q}} \circ v_j^{hp})$
- 11: $\tilde{s}_i = \text{Linear} \left(\text{concat}_{h,p}(\tilde{o}_i^h, o_i^h, o_i^{hp}, \|o_i^{hp}\|) \right)$
- 12: **return** $\{\tilde{s}_i\}$

974 We provide our code in <https://anonymous.4open.science/r/ProteoBayes-213D/>.

975 F.1 Rationalized information flow implementation

976 In fact, the rationalized information flow can be implemented in various ways using different paramete-
 977 r sizes or by mixing modules, with the only constraint being the input and output specifications
 978 of each module. Here we illustrate our instantiations for each module. Firstly, we explain the key
 979 component, *i.e.*, IPA with quaternion implementation.

980 **Invariant point attention with quaternion implementation** We illustrate the invariant point atten-
 981 tion [Jumper et al., 2021] implemented by the unit quaternion in Algorithm 3 using the orange color to
 982 denote the modification compared to the original IPA. Specifically, we only replace the original imple-
 983 mentation of the SE(3) transformation based on rotation matrix and translation $T_i = \{\mathbf{R}, \mathbf{t}\}$ with the
 984 quaternion-based one $T_i^{\mathcal{Q}} = \{\mathbf{q}, \mathbf{t}\}$. For a point with homogeneous representation $\mathbf{p} = [x, y, z, 0]^T$
 985 and $T_i^{\mathcal{Q}}$, the transformation is:

$$T_i^{\mathcal{Q}} \circ \mathbf{p} = \mathbf{q} \times \mathbf{p} \times (\mathbf{q})^{-1} + \mathbf{t} \quad (36)$$

986 where \times is the Hamilton product for two quaternions [Hazewinkel et al., 2006] and the inverse
 987 transformation $(T_i^{\mathcal{Q}})^{-1} = \{(\mathbf{q})^{-1}, -\mathbf{t}\}$. Such operations are antipodal invariant because $\mathbf{q} \times \mathbf{p} \times$
 988 $(\mathbf{q})^{-1} = (-\mathbf{q}) \times \mathbf{p} \times (-\mathbf{q})^{-1}$. Therefore, the IPAq module defined in Algorithm 3 is an antipodal
 989 invariant transformation.

990 **Sequence&Backbone Mixing Module implementation** The implementation of this module builds
 991 upon the graph attention-based architecture FramePred [Yim et al., 2023b], with the following
 992 modifications: 1) The initial node embedding is mixed with the noised sequence θ^S , the noised
 993 backbone torsion angles θ^ψ , and the timestep t via an MLP as follows:

$$\mathbf{h}_0' = \text{MLP}(\mathbf{h}_0, \text{embed}^S(\theta^S), \text{embed}^\psi(\theta^\psi), \text{embed}^t(t)) \quad (37)$$

994 where embed^S , embed^ψ , embed^t are embedders for sequence, angles, and time, respectively. 2) The
995 IPA module is modified as stated above and also in Algorithm 3. 3) The prediction of ψ is removed
996 and repositioned at the end of the Backbone & Side-chain Mixing module.

997 **Backbone&Sidechain Mixing Module implementation** To capture all-atom geometry during
998 backbone prediction, we introduce an additional multilayer perceptron (MLP) that embeds sidechain
999 features. This sidechain embedding is then passed to the BackboneUpdate module [Jumper et al.,
1000 2021], enabling refined updates to the backbone frame. The sidechain embedding module com-
1001 prises three linear layers with bias terms, followed by a LayerNorm layer. Conversely, to in-
1002 corporate backbone information into sidechain design, we employ another MLP-based module
1003 consisting of linear layers with bias terms—which jointly encodes the node embeddings from the
1004 Sequence&Backbone Mixing Module alongside the sidechain features.

1005 F.2 Peptide design

1006 **Metrics** Consistent with previous research [Kong et al., 2024], we generate 40 candidate structures
1007 per receptor for the sequence-structure co-design task and 10 candidates for each receptor-ligand pair
1008 in the binding conformation generation task. The evaluation metrics are detailed as follows:

1009 **ΔG** [Kong et al., 2024] The binding energy (in kcal/mol) calculated by Rosetta [Alford et al., 2017]
1010 to evaluate the binding affinity of the generated peptide using the function `InterfaceAnalyzer`.

1011 **Success rate** [Kong et al., 2024] The proportion of successful designs i.e., those with $\Delta G < 0$, out
1012 of all generated candidates.

1013 **DockQ** [Basu and Wallner, 2016] A comprehensive metric that evaluates the all-atom similarity at
1014 the interface between a candidate and the reference complex.

1015 **RMSD_{C_α}** [Kong et al., 2024] The root mean square deviation (RMSD) of the C_α coordinates between
1016 a candidate and the reference structure, measured in Ångströms (Å).

1017 **Validity** [Lin et al., 2024] refers to the proportion of peptides that are chemically valid, determined
1018 by whether the generated atomic bond lengths are within 0.5Å above and below the ideal value.

1019 **Diversity** [Lin et al., 2024] is measured by the average pairwise distance of the generated peptides
1020 using TM score and sequence similarity.

1021 **Novelty** [Lin et al., 2024] A generated peptide is considered novel if both its TM-score and sequence
1022 overlap with the reference are both below 0.5.

1023 Note that the diversity and novelty are evaluated combining the validity to avoid false positives.

1024 **$\text{RMSD}_{\text{atom}}$** [Kong et al., 2024] on all atoms to measure the quality of the all-atom geometry for the
1025 binding conformation generation task.

1026 **Baselines** For RFDiffusion, we set the number of cycles to one and disable the empirical force
1027 field refinement to ensure a fair comparison with other methods. For PepFlow, we use the official
1028 implementations and retrain the models on the same datasets using the default hyperparameters
1029 provided in its repository. For PepGLAD, most evaluation metrics are taken directly from the original
1030 paper, while the design metrics are computed using the official checkpoint available in its repository.

1031 F.3 Antibody design

1032 Following Ye et al. [2024], we generate 64 candidates for each receptor in the sequence-structure
1033 co-design task and report the average performance across these candidates.

1034 **Metrics** The metrics for antibody design include:

1035 **AAR** [Ye et al., 2024] The amino acid recovery rate, calculated as the number of residues in the
1036 generated CDR-H3 sequences that match the reference antibody;

1037 **RMSD** [Ye et al., 2024] The root-mean-square deviation, measured between the generated and natural
1038 antibodies using the C_α coordinates of the CDR-H3 region.

1039 **E_{total}** [Ye et al., 2024] The total energy computed using Rosetta’s full-atom score function with the
1040 default REF15 weight set, evaluated on the CDR-H3 regions.

1041 **ΔG** [Ye et al., 2024] The binding energy, analogous to that used in peptide design, but here evaluated
 1042 between the CDR-H3 region and the antigen.

1043 **Baselines** We directly borrow the baseline results reported in [Ye et al., 2024].

1044 F.4 Other implementation details

1045 **Computation resources** All experiments in this paper are conducted on a node with 8 NVIDIA A100
 1046 80GB. Each training task requires 4 GPUs for roughly 3 days.

1047 **Relaxation** For peptide design, we adopt the relaxation procedure described in Kong et al. [2024],
 1048 which optimizes the full all-atom structure. In contrast, for antibody design, we employ the protocol
 1049 from Ye et al. [2024], which performs relaxation exclusively on side-chain atoms while keeping the
 1050 backbone atoms fixed.

1051 **Hyperparameters** For all experiments, the network is configured with a node embedding size of
 1052 128 and an edge embedding size of 64. Node embeddings in the IPAq module have 128 dimensions,
 1053 and edge embeddings have 64 dimensions. The IPAq attention mechanism comprises 8 heads,
 1054 with 8 query-key points and 12 value points for geometric attention. Additionally, a sequence
 1055 transformer is integrated into the encoder, featuring 4 attention heads and 2 layers to enhance
 1056 contextual representation. The entire encoder architecture consists of 6 stacked IPAq blocks, enabling
 1057 deep and expressive modeling of the input molecular graph structure. Finally, the network consists of
 1058 7.02 million parameters. For optimization, we set the learning rate to 5×10^{-4} and use a batch size
 1059 of 40 per distributed node with Adam optimizer.

1060 G Comparison between BFN and SDEs/ODEs

1061 A recent study by Xue et al. [2024] explores potential connections between continuous-time Bayesian
 1062 Flow Networks and SDEs. However, we clarify that the BFNs are fundamentally distinct from
 1063 SDE/ODE with evidence as follows:

- 1064 1. The Bayesian flow distribution, as defined in Graves et al. [2023], does not involve differen-
 1065 tial terms, which are central to the formulation of SDEs and ODEs.
- 1066 2. The Bayesian flow formulation employed in this work is *discrete-time* and does not rely
 1067 on any discretization of continuous-time dynamics, in contrast to SDEs- and ODEs-based
 1068 approaches.
- 1069 3. In both the hypertorus Bayesian flows [Wu et al., 2025] and the hypersphere Bayesian flows
 1070 introduced in this paper, the accumulated accuracy parameters κ_i are inherently stochastic.
 1071 These stochastic components cannot be fully represented or captured within a conventional
 1072 SDE framework.
- 1073 4. The connection proposed in Xue et al. [2024] is not mathematically rigorous: it introduces
 1074 a truncation of the time domain from $[0, 1]$ to $[0, 1 - \eta]$, where η is a parameter, thereby
 1075 deviating from the standard continuous-time SDE formulation.

1076 H More results

1077 H.1 Error bars

Table 6: Peptide design task results with error bars representing standard deviations from three independent experiments using distinct random seeds.

Dataset	Method	Energy		Native Likeness		Valid (\uparrow)	Design	
		$\Delta G(\downarrow)$	Success(\uparrow)	DockQ(\uparrow)	RMSD $_{C_{\alpha}}(\downarrow)$		V&Div(\uparrow)	V&Novel(\uparrow)
PepBench	ProteoBayes	-28.63 \pm 0.39	72.92 \pm 0.79	0.740 \pm 0.037	2.28 \pm 0.036	0.998 \pm 0.00012	0.449 \pm 0.0021	0.728 \pm 0.0014

1078 We report the error bars for the protein design task, as shown in Tab. 6, based on three independent
 1079 experiments using different random seeds.

1080 **H.2 Visualization**

We provide the visualization of designed peptides and antibodies in Fig. 7 and Fig. 8.

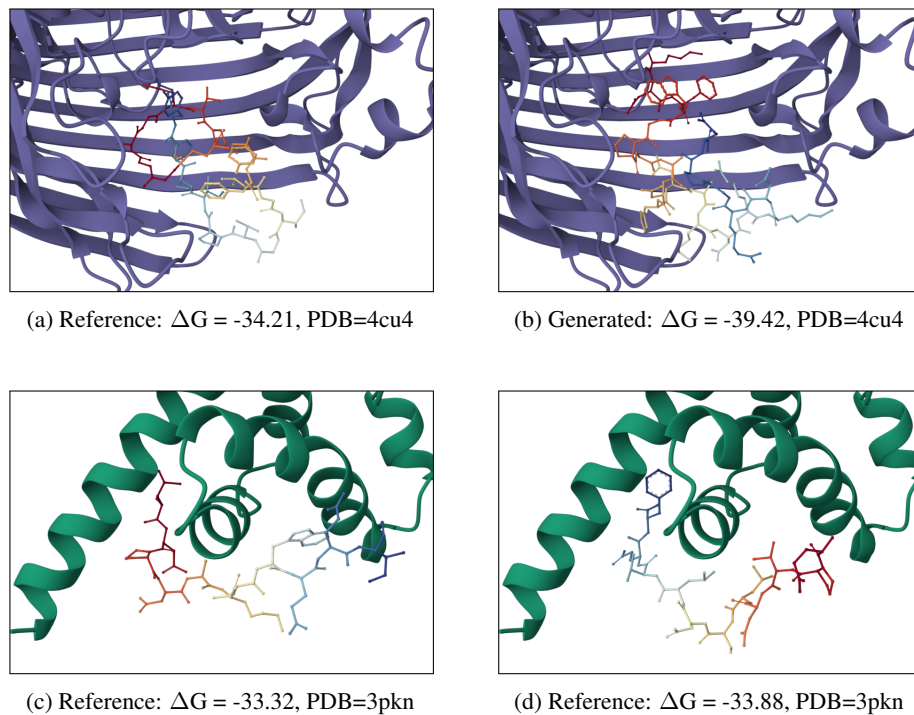


Figure 7: Visualization of designed peptides.

1081

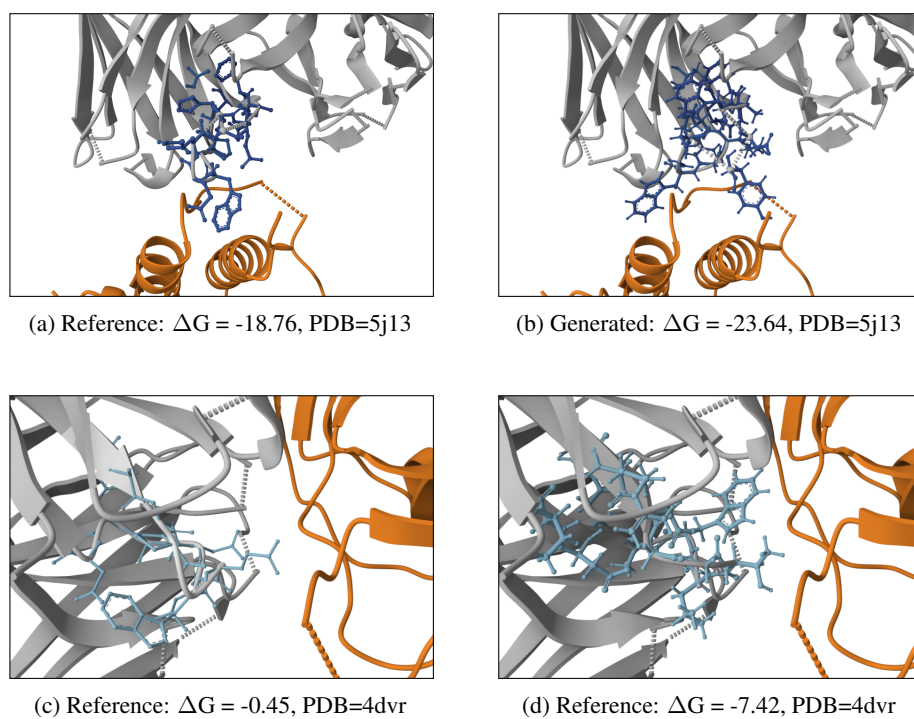


Figure 8: Visualization of designed antibodies.

1082 **I Further discussion of information shortcut**

1083 Protpardelle [Chu et al., 2024] tries to remedy the information leakage by noising all 37 unique atom
1084 position inputs instead of only the atoms corresponding to the sequence. However, we have proved
1085 in the main text that even noisy side-chain information can create an information shortcut for the
1086 sequence prediction. Therefore, the information shortcut still exists in Protpardelle.