Scalable Normalizing Flows Enable Boltzmann Generators for Macromolecules

Joseph C. Kim[1,2] **David Bloore**[1] **Karan Kapoor**[1] **Jun Feng**[1]
Ming-Hong Hao[1] **Mengdi Wang**[2]
[1]Ensem Therapeutics, Waltham, MA
[2]Princeton University, Princeton, NJ

Abstract

The Boltzmann distribution of a protein provides a roadmap to all of its functional states. Normalizing flows are a promising tool for modeling this distribution, but current methods are intractable for typical pharmacological targets; they become computationally intractable due to the size of the system, heterogeneity of intramolecular potential energy, and long-range interactions. To remedy these issues, we present a novel flow architecture that utilizes split channels and gated attention to efficiently learn the conformational distribution of proteins defined by internal coordinates. We show that by utilizing a 2-Wasserstein loss, one can smooth the transition from maximum likelihood training to energy-based training, enabling the training of Boltzmann Generators for macromolecules. We evaluate our model and training strategy on villin headpiece HP35(nle-nle), a 35-residue subdomain, and protein G, a 56-residue protein. We demonstrate that standard architectures and training strategies, such as maximum likelihood alone, fail while our novel architecture and multi-stage training strategy are able to model the conformational distributions of protein G and HP35.

1 Introduction

The structural ensemble of a protein determines its functions. The probabilities of the ground and metastable states of a protein at equilibrium for a given temperature determine the interactions of the protein with other proteins, effectors, and drugs, which are keys for pharmaceutical development. However, enumeration of the equilibrium conformations and their probabilities is infeasible. Since complete knowledge is inaccessible, we must adopt a sampling approach. Conventional approaches toward sampling the equilibrium ensemble rely on Markov-chain Monte Carlo or molecular dynamics (MD). These approaches explore the local energy landscape adjacent a starting point; however, they are limited by their inability to penetrate high energy barriers. In addition, MD simulations are expensive and scale poorly with system size.

In their pioneering work, Noé et al. [25] proposed a normalizing flow model [28], that is trained on the energy function of a many-body system, termed Boltzmann generators (BGs). The model learns an invertible transformation from a system's configurations to a latent space representation, in which the low-energy configurations of different states can be easily sampled. As the model is invertible, every latent space sample can be back-transformed to a system configuration with high Boltzmann probability, i.e., $p(\mathbf{x}) \propto e^{-u(\mathbf{x})/(kT)}$.

BGs in the literature have often struggled with even moderate-sized proteins, due to the complexity of conformation dynamics and scarcity of available data. Most works have focused on small systems like alanine dipeptide (22 atoms) [18, 23, 15]. To date, only two small proteins, BPTI and bromodomain, have been modeled by BGs. Noé et al. [25] trained a BG for BPTI, a 58 amino acid

The NeurIPS 2023 Workshop on New Frontiers of AI for Drug Discovery and Development (AI4D3 2023), New Orleans, LA, USA, 2023.

structure, at all-atom resolution. Unfortunately, the training dataset used is licensed by DESRES [31] and not open-source. No works since have shown success on proteins of similar size at all-atom resolution or reported results for BPTI. Mahmoud et al. [21] trained a BG for bromodomain, a 100 residue protein, with a SIRAH coarse-grained representation. However, drug design applications require much finer resolution than resolvable by SIRAH. See Appendix A.

The limited scope of flow model BG applications is due to the high computational expense of their training process. Their invertibility requirement limits expressivity when modeling targets whose supports have complicated topologies [3], necessitating the use of many transformation layers. Another hurdle in scaling BGs is that proteins often involve long-range interactions; atoms far apart in sequence can interact with each other. In this work, we present a new BG method for general proteins with the following contributions:

- We use a global internal coordinate representation with fixed bond-lengths and side-chain angles. From a global structure and energetics point-of-view, little information is lost with this representation. Such a representation not only reduces the number of variables but also samples conformations more efficiently than Cartesian coordinates [25, 21].
- The global internal coordinate representation is initially split into a backbone channel and a side-chain channel. This allows the model to efficiently capture the distribution of backbone internal coordinates, which most controls the overall global conformation.
- A new NN architecture for learning the transformation parameters of the coupling layers of the flow model which makes use of gated attention units (GAUs) [13] and a combination of rotary positional embeddings [34] with global, absolute positional embeddings for learning long range interactions.
- To handle global conformational changes, a new loss-function, similar in spirit to the *Fréchet Inception Distance (FID)* [9], is introduced to constrain the global backbone structures to the space of native conformational ensemble.

We show in this work that our new method can efficiently generate Boltzmann distributions and important experimental structures in two different protein systems. We demonstrate that the traditional maximum likelihood training for training flow models is insufficient for proteins, but our multi-stage training strategy can generate samples with high Boltzmann probability.

2 Background

2.1 Normalizing Flows

Normalizing flow models learn an invertible map $f : \mathbb{R}^d \mapsto \mathbb{R}^d$ to transform a random variable $\boldsymbol{z} \sim q_Z$ to the random variable $\boldsymbol{x} = f(\boldsymbol{z})$ with distribution

$$q_X(\boldsymbol{x}) = q_Z(\boldsymbol{z})|\text{det}(J_f(\boldsymbol{z}))|^{-1}, \tag{1}$$

where $J_f(\boldsymbol{z}) = \partial f / \partial \boldsymbol{z}$ is the Jacobian of f. We can parameterize f to approximate a target distribution $p(\boldsymbol{x})$. To simplify notation, we refer to the flow distribution as q_θ, where θ are the parameters of the flow. If samples from the target distribution are available, the flow can be trained via maximum likelihood. If the unnormalized target density $p(\boldsymbol{x})$ is known, the flow can be trained by minimizing the KL divergence between q_θ and p, i.e., $\text{KL}(q_\theta||p) = \int_X q_\theta(\boldsymbol{x}) \log(q_\theta(\boldsymbol{x})/p(\boldsymbol{x}))d\boldsymbol{x}$.

2.2 Distance Matrix

A protein distance matrix is a square matrix of Euclidean distances from each atom to all other atoms. Practitioners typically use $C\alpha$ atoms or backbone atoms only. Protein distance matrices have many applications including structural alignment, protein classification, and finding homologous proteins [11, 10, 41]. They have also been used as representations for protein structure prediction algorithms, including the first iteration of AlphaFold [30, 40, 12].

2.3 2-Wasserstein Distance

The 2-Wasserstein Distance is a measure of the distance between two probability distributions. Let $P = \mathcal{N}(\boldsymbol{\mu}_P, \boldsymbol{\Sigma}_P)$ and $Q = \mathcal{N}(\boldsymbol{\mu}_Q, \boldsymbol{\Sigma}_Q)$ be two normal distributions in $\mathbb{R}^d$. Then, with respect to

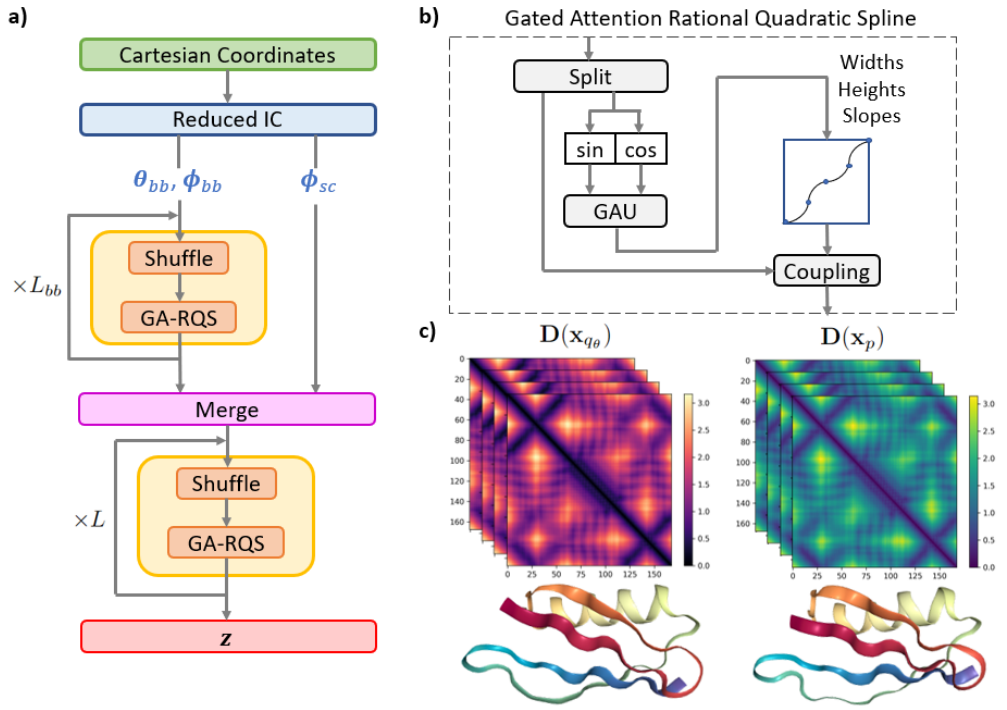

Figure 1: (a) Our split flow architecture. (b) Each transformation block consists of a gated attention rational quadratic spline (RQS) coupling layer. (c) Example structures of protein G from the flow q_θ (left) and from molecular dynamics simulation p (right). We also show sample distance matrices $\mathbf{D}(\mathbf{x}_{q_\theta})$ and $\mathbf{D}(\mathbf{x}_p)$.

the Euclidean norm on $\mathbb{R}^d$, the squared 2-Wasserstein distance between P and Q is defined as

$$W_2(P,Q)^2 = \|\boldsymbol{\mu}_P - \boldsymbol{\mu}_Q\|_2^2 + \text{trace}(\boldsymbol{\Sigma}_P + \boldsymbol{\Sigma}_Q - 2(\boldsymbol{\Sigma}_P\boldsymbol{\Sigma}_Q)^{1/2}). \tag{2}$$

In computer vision, the Fréchet Inception Distance (FID) [9] computes the 2-Wasserstein distance and is often used as an evaluation metric to measure generated image quality.

3 Scalable Boltzmann Generators

3.1 Problem Setup

BGs are generative models that are trained to sample from the Boltzmann distribution for physical systems, i.e., $p(\mathbf{x}) \propto e^{-u(\mathbf{x})/(kT)}$, where $u(\mathbf{x})$ is the potential energy of the conformation $\mathbf{x}$, k is the Boltzmann constant, and T is the temperature. A protein conformation is defined as the arrangement in space of its constituent atoms (Fig. 2), specifically, by the set of 3D Cartesian coordinates of its atoms. Enumeration of metastable conformations for a protein at equilibrium is quite challenging with standard sampling techniques. We tackle this problem with generative modeling. Throughout this work, we refer to p as the ground truth conformation distribution and q_θ as the distribution parameterized by the normalizing flow model f_θ.

3.2 Reduced internal coordinates

Energetically-favored conformational changes take place via rotations around single chemical bonds while bond vibrations and angle bending at physiologic temperature result in relatively small spatial perturbations [35]. Our focus on near ground and meta-stable states therefore motivates the use of internal coordinates: $N-1$ bond lengths d, $N-2$ bond angles θ, and $N-3$ torsion angles ϕ, where N is the number of atoms of the system (see Fig. 2). In addition, internal coordinate representation is translation and rotation invariant. See Appendix C.

A full internal coordinate system requires $3N-6$ dimensions where N is the number of atoms. Bond lengths hardly vary in equilibrium distributions while torsion angles can vary immensely. We treat non-backbone bond angles as constant, again replaced by their mean. Heterocycles in the sidechains of Trp, Phe, Tyr and His residues are treated as rigid bodies. Our final representation is

$$\mathbf{x} = [\boldsymbol{\theta}_{bb}, \boldsymbol{\phi}_{bb}, \boldsymbol{\phi}_{sc}],$$

where the subscripts bb and sc indicate backbone and sidechain, respectively. This reduces the input dimension and keeps the most important features for learning global conformation changes in the equilibrium distribution.

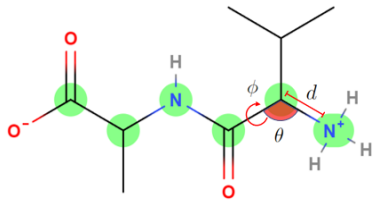

Figure 2: A two residue chain. Backbone atoms are highlighted green. Shown is an example of a bond length d, a bond angle θ, and a dihedral/torsion angle ϕ.

3.3 Training and evaluation

We train BGs with MD simulation data at equilibrium, i.e., the distribution of conformations is constant and not changing as with, for example, folding. We seed the simulation with energetically stable native conformations. BG training aims to learn to sample from the Boltzmann distribution of protein conformations. We compute the energy of samples generated by our model under the AMBER14 forcefield [2] and report their mean. In addition, in order to evaluate how well the flow model generates the proper backbone distribution, we define a new measure:

Definition 3.1 (Distance Distortion)**.** Let $\mathcal{A}_{bb}$ denote the indices of backbone atoms. Define $\mathbf{D}(\mathbf{x})$ as the pairwise distance matrix for the backbone atoms of $\mathbf{x}$. Define $\mathcal{P} = \{(i,j)|i,j \in \mathcal{A}_{bb}\}$. The distance distortion is defined as

$$\Delta D := \mathop{\mathbb{E}}_{\substack{\mathbf{x}_{q_\theta}\sim q_\theta \\ \mathbf{x}_p \sim p}} \left[\frac{1}{|\mathcal{P}|} \sum_{(i,j)\in\mathcal{P}} |\mathbf{D}(\mathbf{x}_{q_\theta})_{ij} - \mathbf{D}(\mathbf{x}_p)_{ij}| \right], \tag{3}$$

While $\mathbf{D}(\mathbf{x})$ is square, in practice, we only use the upper triangle with an offset of one as the matrix is symmetric and contains zeros on the diagonal.

3.4 Split Flow Architecture

We use Neural Spline Flows (NSF) with rational quadratic splines [7] having 8 bins each. The conditioning is done via coupling. Torsion angles $\boldsymbol{\phi}$ can freely rotate and are therefore treated as periodic coordinates [29].

The full architectural details are highlighted in Fig. 1. We first split the input into backbone and sidechain channels:

$$\mathbf{x}_{bb} = [\boldsymbol{\theta}_{bb}, \boldsymbol{\phi}_{bb}], \qquad \mathbf{x}_{sc} = [\boldsymbol{\phi}_{sc}].$$

We then pass the backbone inputs through $L_{bb} = 48$ gated attention rational quadratic spline (GA-RQS) coupling blocks. As all the features are angles in $[-\pi, \pi]$, we augment the features with their mapping on the unit circle. In order to utilize an efficient attention mechanism, we employ gated attention units (GAUs) [13]. In addition, we implement relative positional embeddings [32] on a global level so as to allow each coupling block to utilize the correct embeddings. The backbone latent embeddings are then concatenated with the side chain features and passed through $L = 10$ more GA-RQS coupling blocks.

3.5 Multi-stage training strategy

Normalizing flows are most often trained by maximum likelihood, i.e., minimizing the negative log likelihood (NLL)

$$\mathcal{L}_{\text{NLL}}(\theta) := -\mathbb{E}_{\mathbf{x}\sim p}[\log q_\theta(\mathbf{x})], \tag{4}$$

or by minimizing the reverse KL divergence (rKLD):

$$\mathcal{L}_{\text{KL}}(\theta) := \mathbb{E}_{\mathbf{x}\sim q_\theta}[\log(q_\theta(\mathbf{x})/p(\mathbf{x}))]. \tag{5}$$

In the BG literature, minimizing the rKLD is often referred to as "training-by-energy", as the expression can be rewritten in terms of the energy of the system. The rKLD suffers from mode-seeking behavior, which is problematic when learning multimodal target distributions. While minimizing the NLL is mass-covering, samples generated from flows trained in this manner suffer from high variance. In addition, for larger systems, maximum likelihood training often results in high-energy generated samples.

Noé et al. [25] used a convex combination of the two loss terms, in the context of BGs, in order to both avoid mode-collapse and generate low-energy samples. However, for larger molecules, target evaluation is computationally expensive and dramatically slows iterative training with the rKLD. In addition, during the early stages of training, the KL divergence explodes and leads to unstable training. One way to circumvent these issues is to train with the NLL loss, followed by a combination of both loss terms. Unfortunately, for larger systems, the KL term tends to dominate and training often get stuck at non-optimal local minima. In order to remedy these issues, we consider a sequential training scheme, whereby we smooth the transition from maximum likelihood training to the combination of maximum likelihood and rKLD minimization.

(1) As mentioned previously, we first train with maximum likelihood to convergence.

(2) Afterward, we train with a combination of the NLL and the 2-Wasserstein loss with respect to distance matrices of the backbone atoms:

$$\mathcal{L}_{\mathrm{W}}(\theta) \coloneqq \|\boldsymbol{\mu}_{q_\theta} - \boldsymbol{\mu}_p\|_2^2 + \mathrm{trace}(\boldsymbol{\Sigma}_{q_\theta} + \boldsymbol{\Sigma}_p - 2(\boldsymbol{\Sigma}_{q_\theta}\boldsymbol{\Sigma}_p)^{1/2}), \tag{6}$$

where

$$\boldsymbol{\mu}_p \coloneqq \mathbb{E}_{\mathbf{x}\sim p}[\mathbf{x}_{bb}], \quad \boldsymbol{\Sigma}_p \coloneqq \mathbb{E}_{\mathbf{x}\sim p}[(\mathbf{x}_{bb} - \boldsymbol{\mu}_p)(\mathbf{x}_{bb} - \boldsymbol{\mu}_p)^\top] \tag{7}$$

$$\boldsymbol{\mu}_{q_\theta} \coloneqq \mathbb{E}_{\mathbf{x}\sim q_\theta}[\mathbf{x}_{bb}], \quad \boldsymbol{\Sigma}_{q_\theta} \coloneqq \mathbb{E}_{\mathbf{x}\sim q_\theta}[(\mathbf{x}_{bb} - \boldsymbol{\mu}_{q_\theta})(\mathbf{x}_{bb} - \boldsymbol{\mu}_{q_\theta})^\top] \tag{8}$$

are mean and covariance, respectively, of the vectorized backbone atom distance matrices.

(3) As a third stage of training, we train with a combination of the NLL, the 2-Wasserstein loss, and the KL divergence. In our final stage of training, we drop the 2-Wasserstein loss term and train to minimize a combination of the NLL and the KL divergence.

4 Results

4.1 Protein Systems

Alanine dipeptide (ADP) is a two residue (22-atoms) common benchmark system for evaluating BGs [25, 18, 23]. We use the MD simulation datasets provided by [23] for training and validation.

HP35(nle-nle), a 35-residue double-mutant of the villin headpiece subdomain, is a well-studied structure whose folding dynamics have been observed and documented [1]. For training, we use the MD simulation dataset made publicly available by [1] and remove faulty trajectories and unfolded structures as done by [14].

Protein G is a 56-residue cell surface-associated protein from *Streptococcus* that binds to IgG with high affinity [4]. In order to train our model, we generated samples by running a MD simulation. The crystal structure of protein G, 1PGA, was used as the seed structure. The conformational space of Protein G was first explored by simulations with ClustENMD [17]. From 3 rounds of ClustENMD iteration and approximately 300 generated conformations, 5 distinctly diverse structures were selected as the starting point for equilibrium MD simulation by Amber. On each starting structure, 5 replica simulations were carried out in parallel with different random seeds for 400 ns at 300 K. The total simulation time of all the replicas was accumulated to 1 microsecond. Thus, 10^6 structures of protein G were saved over all the MD trajectories.

As a baseline model for comparison, we use Neural Spline Flows (NSF) with 58 rational quadratic spline coupling layers [7]. NSFs have been used in many recent works on BGs [19, 23, 21]. In particular, [23] utilized the NSF model (with fewer coupling layers) in their experiments with alanine dipeptide, a two residue system. In our experiments with ADP and HP35, we utilize 48 GA-RQS coupling layers for the backbone followed by 10 GA-RQS coupling layers for the full latent size. We also ensure that all models have a similar number of trainable parameters. We use a Gaussian base distribution for non-dihedral coordinates. For ADP and HP35, we use a uniform distribution for

Table 1: **Training BGs with different strategies.** We compute ΔD, energy $u(\cdot)$, and mean NLL of 10^6 generated structures after training with different training strategies with protein G and Villin HP35. ΔD is computed for batches of 10^3 samples. Means and standard deviations are reported. Statistics for $u(\cdot)$ are reported for structures with energy below the median sample energy. Results for ΔD and $u(\cdot)$ that are within tolerable range are bold-faced, while the best result for mean NLL is bold-faced. For reference, the energy for training data structures is -317.5 ± 125.5 kcal/mol for protein G and -1215.5 ± 222.2 kcal/mol for villin HP35. We compare our results against a Neural Spline Flows (NSF) baseline model.

		Training strategy					
System	Arch.	NLL	KL	W2.	ΔD (Å)	Energy $u(\mathbf{x})$ (kcal/mol)	$-\mathbb{E}_{p(\mathbf{x})}[\log q_\theta(\mathbf{x})]$
ADP	NSF	✓			0.09 ± 0.01	$(\mathbf{-1.19 \pm 0.61}) \times \mathbf{10^1}$	38.29 ± 0.19
	Ours	✓			0.08 ± 0.01	$(\mathbf{-1.18 \pm 0.65}) \times \mathbf{10^1}$	$\mathbf{36.15 \pm 0.15}$
		✓	✓		0.05 ± 0.01	$(\mathbf{-1.20 \pm 0.59}) \times \mathbf{10^1}$	38.66 ± 0.19
		✓		✓	$\mathbf{0.04 \pm 0.00}$	$(\mathbf{-1.06 \pm 0.74}) \times \mathbf{10^1}$	38.12 ± 0.03
	Ours	✓	✓	✓	$\mathbf{0.03 \pm 0.01}$	$(\mathbf{-1.31 \pm 0.52}) \times \mathbf{10^1}$	37.67 ± 0.09
Protein G	NSF	✓			2.92 ± 0.80	$(2.15 \pm 3.31) \times 10^{10}$	-263.46 ± 0.13
	Ours	✓			1.81 ± 0.14	$(9.47 \pm 15.4) \times 10^8$	$\mathbf{-310.11 \pm 0.08}$
		✓	✓		16.09 ± 1.14	$(2.86 \pm 0.62) \times 10^2$	-308.68 ± 0.08
		✓		✓	$\mathbf{0.18 \pm 0.01}$	$(2.68 \pm 4.31) \times 10^6$	-307.17 ± 0.01
	Ours	✓	✓	✓	$\mathbf{0.19 \pm 0.01}$	$(\mathbf{-3.04 \pm 1.24}) \times \mathbf{10^2}$	$\mathbf{-309.10 \pm 0.91}$
HP35	NSF	✓			0.81 ± 0.06	$(7.78 \pm 17.4) \times 10^7$	687.95 ± 1.92
	Ours	✓			0.65 ± 0.04	$(5.29 \pm 11.7) \times 10^6$	$\mathbf{651.90 \pm 2.88}$
		✓	✓		0.61 ± 0.04	$(6.46 \pm 14.3) \times 10^2$	678.38 ± 0.87
		✓		✓	$\mathbf{0.38 \pm 0.03}$	$(1.15 \pm 1.76) \times 10^7$	678.31 ± 1.55
	Ours	✓	✓	✓	$\mathbf{0.39 \pm 0.03}$	$(\mathbf{-4.66 \pm 3.52}) \times \mathbf{10^2}$	667.45 ± 2.04

dihedral coordinates. For protein G, we use a von Mises base distribution for dihedral coordinates; we noticed that using a von Mises base distribution improved training for the protein G system as compared to a uniform or truncated normal distribution.

4.2 Main Results

From Table 1, we see that our model has marginal improvements over the baseline model for ADP. This is not surprising as the system is extremely small, and maximum likelihood training sufficiently models the conformational landscape.

For both proteins, our model closely captures the individual residue flexibility as analyzed by the root mean square fluctuations (RMSF) of the generated samples from the various training schemes in Fig. 3(a). This is a common metric for MD analysis, where larger per-residue values indicate larger movements of that residue relative to the rest of the protein. Fig. 3(a) indicates that our model generates samples that present with similar levels of per-residue flexibility as the training data.

Table 1 displays ΔD, the mean energy, and the mean NLL of structures generated from flow models trained with different strategies. For each model (architecture and training strategy), we generate 3×10^6 conformations (10^6 structures over 3 random seeds) after training with either protein G or Villin HP35. Due to the cost of computing ΔD, we compute it for batches of 10^3 samples and report statistics (mean and standard deviation) over the 3×10^3 batches. Before we computed sample statistics for the energy u, we first filtered out the samples with energy higher than the median value. This was done to remove high energy outliers that are not of interest and would noise the reported mean and standard deviations. We also report the mean and standard deviation (across 3 seeds) for the average NLL. We see that our model is capable of generating low-energy, stable conformations for these two systems while the baseline method and ablated training strategies produce samples with energies that are positive and five or more orders of magnitude greater.

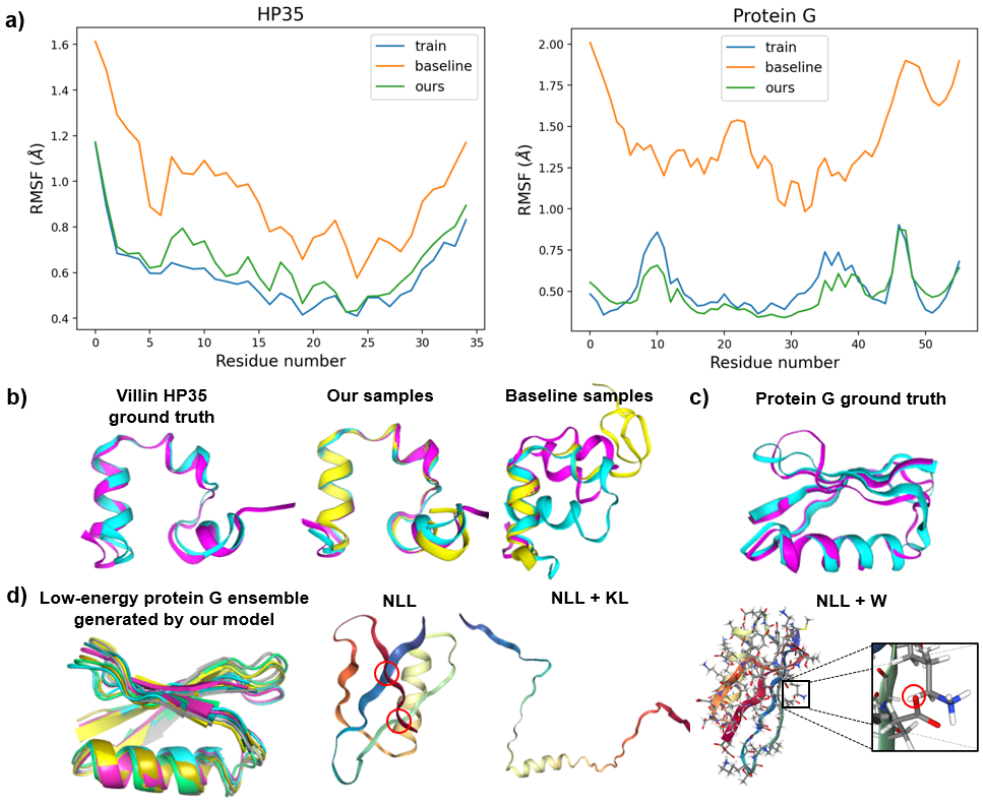

Figure 3: **Sample conformations generated by BG via different training strategies.** (a) Root mean square fluctuation (RMSF) computed for each residue (Cα atoms) in HP35 and protein G. Matching the training dataset's plot is desirable. (b) Examples of HP35 from ground truth training data, generated samples from our model, and generated samples from the baseline model. (c) Example of two metastable states from protein G training data. (d) Low-energy conformations of protein G generated by our model superimposed on each other. We also show some examples of pathological structures generated after training with different training paradigms: NLL (maximum likelihood), both NLL and KL divergence, and NLL and the 2-Wasserstein loss. Atom clashes are highlighted with red circles.

Table 1 highlights a key difference in the results for protein G and villin HP35. For villin, models trained by reverse KL and without the 2-Wasserstein loss do not result in completely unraveled structures. This is consistent with the notion that long-range interactions become much more important in larger structures. From Fig. 3(b), we see that Villin HP35 is not densely packed, and local interactions, e.g., as seen in α-helices, are more prevalent than long-range interactions/contacts. In addition, we see that our model generates diverse alternative modes of the folded villin HP35 protein that are energetically stable compared to the structures obtained from the baseline model.

Fig. 3(d) visualizes pathological structures of protein G generated via different training schemes. In Fig. 3(d, left), we see that minimizing the NLL generally captures local structural motifs, such as the α-helix. However, structures generated by training only with the NLL loss tend to have clashes in the backbone, as highlighted with red circles in Fig 3(d), and/or long range distortions. This results in large van der waals repulsion as evidenced by the high average energy values in Table 1.

In Fig. 3(d, middle), we see that structures generated by minimizing a combination of the NLL loss and the reverse KL divergence unravel and present with large global distortions. This results from large, perpetually unstable gradients during training. In Fig. 3(d, right), we see that training with a combination of the NLL loss and the 2-Wasserstein loss properly captures the backbone structural distribution, but tends to have clashes in the sidechains. Table 1 demonstrates that only our model with our multistage training strategy is able to achieve both low energy samples and proper global structures. The 2-Wasserstein loss prevents large backbone distortions, and thus, simultaneously

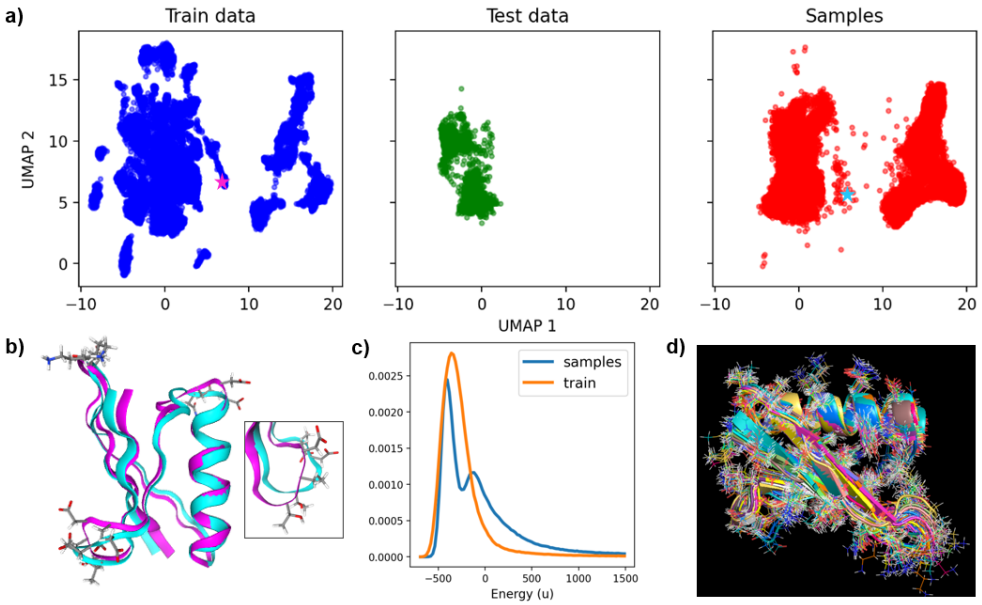

Figure 4: **BGs can generate novel sample conformations.** (a) Protein G 2D UMAP embeddings for the training data, test data, and 2×10^5 generated samples. (b) A representative example of generated structures by the BG model which was not found in training data (magenta) and the closest structure in the training dataset (cyan) by RMSD. Both structures are depicted as stars with their respective structural colors in (a). (c) Protein G energy distribution of training dataset (orange) and samples (blue) generated by our model. The second energy peak of the sampled conformations covers the novel structure shown in (b). (d) An overlay of high-resolution, lowest-energy all-atom structures of protein G generated by the BG model. This demonstrates that our model is capable of sampling low-energy conformations at atomic resolution.

minimizing the reverse KL divergence accelerates learning for the side chain marginals with respect to the backbone and other side chain atoms.

4.3 BGs can generate novel samples

One of the primary goals for development of BG models is to sample important metastable states that are unseen or difficult to sample by conventional MD simulations. Protein G is a medium-size protein with diverse metastable states that provide a system for us to evaluate the capability of our BG model. First, we visualize 2D UMAP embeddings [22] for the training data set, test dataset, and for 2×10^5 generated samples of protein G in Fig. 4(a). We see that the test dataset 4(a, middle), an independent MD dataset, covers far less conformational space than an equivalent number of BG samples as shown in Fig. 4(a, right).

Secondly, we computed, respectively, the energy distributions of the training set from MD simulations and sample set from the BG model as shown by Fig. 4(c). Unlike the training set, the BG sample energy distribution is bimodal. Analysis of structures in the second peak revealed a set of conformations not present in the training set. These new structures are characterized by a large bent conformation in the hair-pin loop which links beta-strand 3 and 4 of protein G. Fig. 4(b) compares representative structures of the discovered new structure (magenta) with the closest structure (by RMSD) in the training dataset (cyan). We also see vastly different sidechain conformations along the bent loops between two structures. Energy minimization on the discovered new structures demonstrated that these new structures are local-minimum metastable conformations. Thirdly, we carefully examined the lowest-energy conformations generated by the BG model. Fig. 4(d) showcases a group of lowest-energy structures generated by the BG model, overlaid by backbone and all-atom side chains shown explicitly. All of these structures are very similar to the crystal structure of protein G, demonstrating that the trained BG model is capable of generating protein structures with high quality at the atomic level.

References

[1] Kyle A. Beauchamp, Robert McGibbon, Yu-Shan Lin, and Vijay S. Pande. Simple few-state models reveal hidden complexity in protein folding. *The Proceedings of the National Academy of Sciences*, 109(44):17807–17813, 2012.

[2] D. A. Case, V. Babin, J. T. Berryman, R. M. Betz, Q. Cai, D.S. Cerutti, T.E. Cheatham III, T.A. Darden, R. E. Duke, H. Gohlke, A. W. Goetz, S. Gusarov, N. Homeyer, P. Janowski, J. Kaus, I. Kolossváry, A. Kovalenko, T. S. Lee, S. LeGrand, T. Luchko, R. Luo, B. Madej, K. M. Merz, F. Paesani, D. R. Roe, A. Roitberg, C. Sagui, R. Salomon-Ferrer, G. Seabra, C. L. Simmerling, W. Smith, J. Swails, R. C. Walker, J. Wang, R. M. Wolf, X. Wu, and P. A. Kollman. Amber 14, 2014.

[3] Rob Cornish, Anthony L. Caterini, George Deligiannidis, and Arnaud Doucet. Relaxing bijectivity constraints with continuously indexed normalising flows, 2019.

[4] Jeremy P. Derrick and Dale B. Wigley. The third igg-binding domain from streptococcal protein g. an analysis by x-ray crystallography of the structure alone and in a complex with fab. *Journal of Molecular Biology*, 243(5):906–918, 1994.

[5] Manuel Dibak, Leon Klein, Andreas Krämer, and Frank Noé. Temperature steerable flows and boltzmann generators. *Phys. Rev. Research*, 4(L042005), 2022.

[6] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. In *International Conference on Learning Representations*, 2017.

[7] Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural spline flows. In *NeurIPS*, 2019.

[8] Matej Grcić, Ivan Grubišić, and Siniša Šegvić. Densely connected normalizing flows, 2021.

[9] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper_files/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf`.

[10] Liisa Holm. Using dali for protein structure comparison. *Methods in Molecular Biology*, pp. 29–42, 2020.

[11] Liisa Holm and Chris Sander. Protein structure comparison by alignment of distance matrices. *Journal of Molecular Biology*, 233(1):123–138, 1993.

[12] Jie Hou, Tianqi Wu, Renzhi Cao, and Jianlin Cheng. Protein tertiary structure modeling driven by deep learning and contact distance prediction in casp13. *Proteins Structure Function Bioinformatics*, 87(12):1165–1178, 2019.

[13] Weizhe Hua, Zihang Dai, Hanxiao Liu, and Quoc V. Le. Transformer quality in linear time, 2022.

[14] Takashi Ichinomiya. Topological data analysis gives two folding paths in hp35(nle-nle), double mutant of villin headpiece subdomain. *Scientific Reports*, 12(1), 2022.

[15] Michele Invernizzi, Andreas Krämer, Cecilia Clementi, and Frank Noé. Skipping the replica exchange ladder with normalizing flows. *J. Phys. Chem. Lett.*, 13(50):11643—-11649, 2022.

[16] Bowen Jing, Gabriele Corso, Jeffrey Chang, Regina Barzilay, and Tommi Jaakkola. Torsional diffusion for molecular conformer generation. *arXiv preprint arXiv:2206.01729*, 2022.

[17] Burak T. Kaynak, She Zhang, Ivet Bahar, and Pemra Doruker. Clustenmd: efficient sampling of biomolecular conformational space at atomic resolution. *Bioinformatics*, 37(21):3956–3958, 2021.

[18] Jonas Köhler, Andreas Krämer, and Frank Noé. Smooth normalizing flows. In *Advances in Neural Information Processing Systems 34*, 2021.

[19] Jonas Köhler, Yaoyi Chen, Andreas Krämer, Cecilia Clementi, and Frank Noé. Flow-matching – efficient coarse-graining molecular dynamics without forces. *arXiv preprint arXiv:2203.11167*, 2022.

[20] Xuezhe Ma, Chunting Zhou, Xiang Kong, Junxian He, Liangke Gui, Graham Neubig, Jonathan May, and Luke Zettlemoyer. Mega: Moving average equipped gated attention, 2023.

[21] Amr H. Mahmoud, Matthew Masters, Soo Jung Lee, , and Markus A. Lill. Accurate sampling of macromolecular conformations using adaptive deep learning and coarse-grained representation. *Journal of Chemical Information and Modeling*, 62(7), 2022.

[22] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. Umap: Uniform manifold approximation and projection. *The Journal of Open Source Software*, 3(29):861, 2018.

[23] Laurence Illing Midgley, Vincent Stimper, Gregor N. C. Simm, Bernhard Schölkopf, and José Miguel Hernández-Lobato. Flow annealed importance sampling bootstrap. *arXiv preprint arXiv:2208.01893*, 2022.

[24] Toan Q. Nguyen and Julian Salazar. Transformers without tears: Improving the normalization of self-attention. In *IWSLT*, 2019.

[25] Frank Noé, Simon Olsson, Jonas Köhler, and Hao Wu. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365(6457), 2019.

[26] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2020.

[27] Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions, 2017.

[28] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1530–1538, Lille, France, 07–09 Jul 2015. PMLR. URL `https://proceedings.mlr.press/v37/rezende15.html`.

[29] Danilo Jimenez Rezende, George Papamakarios, Sebastien Racaniere, Michael Albergo, Gurtej Kanwar, Phiala Shanahan, and Kyle Cranmer. Normalizing flows on tori and spheres. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8083–8092. PMLR, 13–18 Jul 2020. URL `https://proceedings.mlr.press/v119/rezende20a.html`.

[30] Andrew W. Senior, Richard Evans, John Jumper, James Kirkpatrick, Laurent Sifre, Tim Green, Chongli Qin, Augustin Žídek, Alexander W. R. Nelson, Alex Bridgland, Hugo Penedones, Stig Petersen, Karen Simonyan, Steve Crossan, Pushmeet Kohli, David T. Jones, David Silver, Koray Kavukcuoglu, and Demis Hassabis. Protein structure prediction using multiple deep neural networks in the 13th critical assessment of protein structure prediction (casp13). *Proteins Structure Function Bioinformatics*, 87:1141–1148, 2019.

[31] David E. Shaw, Paul Maragakis, Kresten Lindorff-Larsen, Stafano Piana, Ron O. Dror, Michael P. Eastwood, Joseph A. Bank, John M. Jumper, John K. Salmon, and Willy Wriggers. Atomic-level characterization of the structural dynamics of proteins. *Science*, 330(6002): 341–346, 2010.

[32] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations, 2018.

[33] Florian Sittel, Thomas Filk, and Gerhard Stock. Principal component analysis on a torus: Theory and application to protein dynamics. *J. Chem. Phys.*, 147(244101), 2017.

[34] Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

[35] Nagarajan Vaidehi and Abhinandan Jain. Internal coordinate molecular dynamics: A foundation for multiscale dynamics. *J. Phys. Chem. B*, 119(4):1233–1242, 2015.

[36] Yihang Wang, Lukas Herron, and Pratyush Tiwary. From data to noise to data for mixing physics across temperatures with generative artificial intelligence. *The Proceedings of the National Academy of Sciences*, 119(32), 2022.

[37] Peter Wirnsberger, George Papamakarios, Borja Ibarz, Sébastien Racanière, Andrew J. Ballard, Alexander Pritzel, and Charles Blundell. Normalizing flows for atomic solids. *Machine Learning: Science and Technology*, 3(2), 2022.

[38] Hao Wu, Jonas Köhler, and Frank Noé. Stochastic normalizing flows. In *NeurIPS*, 2020.

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

[40] Jinbo Xu and Sheng Wang. Analysis of distance-based protein structure prediction by deep learning in casp13. *Proteins Structure Function Bioinformatics*, 87(12):1069–1081, 2019.

[41] Kun Zhu, Hong Su, Zhenling Peng, and Jianyi Yang. A unified approach to protein domain parsing with inter-residue distance matrix. *Bioinformatics*, 39(2), 2023.

A Related works

I. Boltzmann Generators

Bolzmann generators [25] are normalizing flows that approximate Boltzmann distributions. [25] utilized the fact that normalizing flows are tractable density models and introduced a notion of training by energy via reverse KL-divergence minimization. Recently, there has been a growing interest in Boltzmann generators. [5] proposed temperature steerable flows that generalized to families of ensembles across multiple temperatures, thereby increasing the range of thermodynamic states accessible for sampling. Unfortunately, this model tends to undersample significant local minimas for systems as small as alanine dipeptide. The authors believed that this was due to the limited expressivity of the flow model. [38] proposed stochastic normalizing flows, which combine flows with MCMC methods by introducing sampling layers between flow layers to improve model expressivity. Unfortunately, this method is computationally expensive as it involves many more target evaluations. In addition, stochastic normalizing flows tend to miss modes [23]. [18] introduced smooth normalizing flows, which are C^∞-smooth, thus making them more physically amenable. They also introduce force-matching as an added loss term. While they have impressive results and modal coverage for alanine dipeptide, they utilize a root-finding algorithm to approximate the inverse for their smooth flows, which becomes computationally prohibitive for higher-dimensional systems.

This work has focused on normalizing flows. However, diffusion models have also shown great promise as an alternative generative model for learning Boltzmann generators. [16] train a diffusion model to learn the Boltzmann distribution over the torsion angles of multiple drug-like molecules, while using cheminformatics methods for the bond lengths and angles. They perform energy-based training (similar in spirit to the reverse KL divergence in flow model training) via estimation of a score matching loss using samples generated by the model. However, this method does not scale well to larger molecules and inherits the same problem of unstable training at initialization.

II. Loss functions

[37] trained a flow model without MD samples by minimizing the KL divergence to approximate the Boltzmann distribution of atomic solids with up to 512 atoms. However, the KL-divergence suffers from mode-seeking behavior, which severely impairs training for multimodal target distributions. While the forward KL-divergence, i.e. maximum likelihood, is mass covering, the Monte Carlo approximations of such an objective have a very high variance in loss. To circumvent this, [23] trains a flow to approximate a target p by minimizing the alpha-divergence with $\alpha = 2$, which is estimated with annealed importance sampling (AIS) using the flow q as the base distribution and p^2/q as target. This method is notable in that it does not require any MD samples but still achieves impressive results for alanine dipeptide. Nonetheless, the AIS component is computationally expensive and scales poorly for larger systems.

III. Coarse-graining

Several works have attempted to scale flow-based Boltzmann generators to larger systems. [21] trained a flow model on coarse-grained protein representations which they then mapped back to full-atom representations using a language model. On a similar note, [19] trained a normalizing flow to represent the probability density for coarse-grained (CG) MD samples in order to learn the parameters of a CG model. Unfortunately, coarse-grain approaches tend to lose significant information compared to full-atom resolution for downstream applications. Importantly, both works note that using internal-coordinate representations do not scale well as small changes in torsion angles can lead to large global distortions. Our results indicate that this is not necessarily true, as we use a reduced internal-coordinate representation.

IV. Normalizing flow architectures

Our flow model, while novel, shares some similarities to previous works. DenseFlow [8] fuses a densely connected convolutional block with Nyström self-attention in modules with both cross-unit and intra-module couplings. This architecture is specifically designed for image data and utilizes a linear approximation for the self-attention mechanism. In contrast, we use gated attention and rotary positional embeddings in order to handle the sequential nature of proteins.

Multiscale flow architectures were first introduced by [6] In the protein domain, previous works also split the inputs into different channels [25, 18, 19]. However, they split the input dimensions into torsion, angle, and bond channels. In contrast, our model splits the input into separate backbone and sidechain channels to better capture the global distribution.

B Training by energy

Below, we show the connection between minimizing the reverse KL-divergence and minimizing the energy of generated samples.

$$\begin{aligned} KL(q_\theta||p) &= \mathbb{E}_{\mathbf{x}\sim q_\theta}\left[\log q_\theta(\mathbf{x}) - \log p(\mathbf{x})\right] \\ &= \mathbb{E}_{\mathbf{z}\sim q}\left[\log q(\mathbf{z}) - \log\left|\det(J_{f_\theta}(\mathbf{z}))\right| - \log p(f_\theta(\mathbf{z}))\right] \\ &= -H_{\mathbf{z}} + \log C + \mathbb{E}_{\mathbf{z}\sim q}\left[u(f_\theta(\mathbf{z})) - \log\left|\det(J_{f_\theta}(\mathbf{z}))\right|\right], \end{aligned}$$

where $H_{\mathbf{z}}$ is the entropy of the random variable $\mathbf{z}$ and $C = \int e^{-u(\mathbf{x})/(kT)} d\mathbf{x}$ is the normalization constant for the Boltzmann distribution $p(\mathbf{x}) \propto e^{-u(\mathbf{x})/(kT)}$. When minimizing the KL-divergence with respect to the parameters θ, the entropy term and the log normalization constant disappear as they are not dependent on θ:

$$\begin{aligned} \theta^* &= \underset{\theta}{\operatorname{argmin}} \cancel{-H_{\mathbf{z}}} + \cancel{\log C} + \mathbb{E}_{\mathbf{z}\sim q}\left[u(f_\theta(\mathbf{z})) - \log\left|\det(J_{f_\theta}(\mathbf{z}))\right|\right] \\ &= \underset{\theta}{\operatorname{argmin}}\, \mathbb{E}_{\mathbf{z}\sim q}\left[u(f_\theta(\mathbf{z})) - \log\left|\det(J_{f_\theta}(\mathbf{z}))\right|\right]. \end{aligned}$$

The expectation here is usually approximated with a Monte Carlo estimate, but a variety of different sampling procedures can be utilized. The log determinant Jacobian (ldj) term can be seen as promoting entropy, or exploration, of the sample space.

C Coordinate Transformation

I. Protein Structure

Protein structure refers to the three-dimensional arrangement of atoms in an amino acid-chain molecule. There are four distinct levels by which we can describe protein structure. The *primary structure* of a protein refers to the sequence of amino acids in the polypeptide chain. The *secondary structure* refers to regularly patterned local sub-structures on the actual polypeptide backbone chain. The two most common secondary structure motifs are α-helices and β-sheets. Tertiary structure refers to the overall three-dimensional structure created by a single polypeptide. Tertiary structure is primarily driven by non-specific hydrophobic interactions as well as long-range intramolecular forces. Quaternary structure refers to the three-dimensional structure consisting of two or more polypeptide chains that operate as a single functional unit.

II. Coordinate Representations

Boltzmann generators usually do not operate directly with Cartesian coordinates. The primary global conformational changes of a protein do not described efficiently by the atomic Cartesian coordinates. This is driven by the fact that chemical bonds are very stiff, and energetically-favored conformational changes take place via rotations around single chemical bonds [35]. A more commonly used alternative is internal coordinates. Internal coordinates are defined by bond lengths d, bond angles θ, and dihedral angles ϕ (Fig. S1).

In their seminal work, [25] introduced a coordinate transformation whereby the protein backbone atoms (primarily defined as the N, C_α, and C atoms) are mapped PCA coordinates while the rest of the atoms are mapped to internal coordinates. The motivation behind this mixed coordinate transformation is that protein conformations are highly sensitive to changes in backbone internal coordinates. This often results in unstable training and difficulty in generating natural, i.e., high Boltzmann probability, structures. Most works since have used full internal coordinate representations but experimented only with small systems, the most common of which is alanine dipeptide (22 atoms). [19] note that scaling Boltzmann generators to larger systems is difficult with internal coordinate representations.

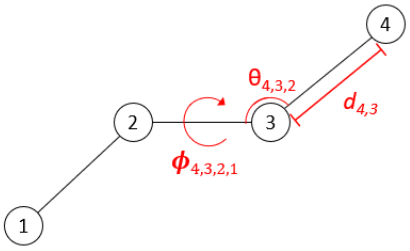

Fig. S1: An illustration of the definition of bond length, bond angle, and dihedral angle by four atoms. Subscripts indicate the atoms that define the value, where order is given by the bond graph connectivity. In internal coordinate system, the position or Cartesian coordinate of atom 4 is determined by atom 1,2 and 3 based on bond length, bond angle and dihedral angle.

In our work, rather than using a full internal coordinate representation, which would be $3N - 6$ dimensional (where N is the number of atoms in the system), we utilize a reduced internal coordinate representation. For training features, we use the dihedral angles and the bond angles for the 3 backbone atoms (N, C_α, C). For side-chain atoms, we use all rotatable dihedral angles around single bond. All bond lengths and bond angles other than the 3 defining backbone atoms and improper torsion angles are kept at their mean values calculated from input protein structures. By examining all protein structures generated, we confirmed that such a reduced internal coordinate system can represent all protein structures to very high accuracy and quality. Recent works have adopted similar approaches; [39] utilize only the backbone torsion and bond angles to represent various proteins, while Wang et al. [36] simply use the backbone torsion angles to represent the polypeptide AiB9.

D Training details and architecture

All models were trained on a single NVIDIA A100 GPUs with the Adam optimizer and a dropout factor of 0.1. For model that utilized GAU-RQS blocks, the dimensionalities of the Q, K, and V matrices were 32, 32, and 64, respectively. In addition, we utilized scaled normalization [24], the Laplace attention function [20], and SiLU activations [27]. For the gated attention units, we also use the T5 relative positional bias [26]. For the rational quadratic splines, we use a bin size of $K = 8$.

Data was all standard normalized. Dihedral angles were constrained to be within $[-\pi, \pi]$ and shifted as done by [33].

For the multi-stage training strategy, all models were trained for 200 epochs (12 hours) with the NLL loss, 50 epochs (8 hours) with NLL+W, 20 epochs (8 hours) with NLL+W+KL, and 10 epochs (3 hours) with NLL+KL. The approximate times are for protein G, which has 56 residues.

We do no hyperparameter tuning due to the lack of compute and time. Further implementation details are given in the code, which is available upon request.