RIGIDSSL: RIGIDITY-BASED GEOMETRIC PRETRAINING FOR PROTEIN GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Protein design stands as one of biology's most important frontiers, with the potential to transform medicine, advance human health, and drive sustainability. Protein generation, a central task in protein design, has been greatly accelerated by AI-driven models—such as FoldFlow, MultiFlow, and AlphaFlow that build on residue-wise rigidity–based modeling pioneered by AlphaFold2. Residue-wise rigid-body representations reduce structural dimensionality while enforcing chemical constraints, enabling more efficient and physically consistent protein structure generation than all-atom modeling. Despite these advances, existing models often underutilize the vast structural information available in large-scale protein datasets. This highlights the importance of pretraining, which can provide richer representations and improve generalization across diverse protein design tasks. More importantly, the challenge lies in how to fully exploit abundant, low-cost unlabeled protein datasets using unsupervised pretraining. We introduce RigidSSL, a rigidity-based pretraining framework for proteins. RigidSSL canonicalizes structures into an inertial frame, employs a two-phase workflow combining large-scale perturbations and molecular dynamics views, and applies a rigid-body flow matching objective with Invariant Point Attention to capture global geometry. This enables learning stable, geometry-aware representations that improve downstream protein generation. To evaluate the effectiveness of RigidSSL, we conduct quantitative experiments on the protein generation task. Empirically, RigidSSL outperforms previous state-of-the-art geometric pretraining algorithms, leading to improvements in unconditional generation across all metrics, including designability, novelty, and diversity, for length up to 800 residues.

1 INTRODUCTION

Proteins are large, complex biomolecules whose three-dimensional structures underpin their diverse biological roles, enabling precise molecular functions such as substrate specificity, binding affinity, and catalytic activity (LaPelusa and Kaushik, 2020). The ability to design proteins with tailored properties has the potential to revolutionize fields ranging from medicine, through the development of novel therapeutics and vaccines, to materials science, where proteins can serve as the basis for sustainable biomaterials (Listov et al., 2024; Miserez et al., 2023). Recent advances in deep generative models have opened new opportunities for protein design. Such data-driven models capture the statistical regularities of natural proteins and generalize beyond evolutionary constraints to propose novel structures. Representative approaches include RFDiffusion (Watson et al., 2023), AlphaFlow (Jing et al., 2024), and FoldFlow (Bose et al., 2023).

Notably, many protein generation methods adopt rigid-body modeling of protein backbones (Bose et al., 2023; Campbell et al., 2024; Jing et al., 2024; Watson et al., 2023), where each residue backbone (N–C_α–C) is treated as a rigid unit with fixed internal geometry. Invariant Point Attention (IPA), first introduced in AlphaFold2 (Jumper et al., 2021), is specifically designed to process rigid bodies through their translations and rotations, and has since become a widely used backbone transformer in structure-based protein modeling. Compared to all-atom approaches (Cho et al., 2025; Geffner et al., 2025a;b), rigid-body models offer three key advantages: (1) By constraining local geometry, they eliminate implausible distortions and reduce the dimensionality of conformational space, allowing models to better capture long-range structural relationships; (2) The reduced search space facilitates more efficient and stable sampling, yielding higher-fidelity backbones at lower computational cost,

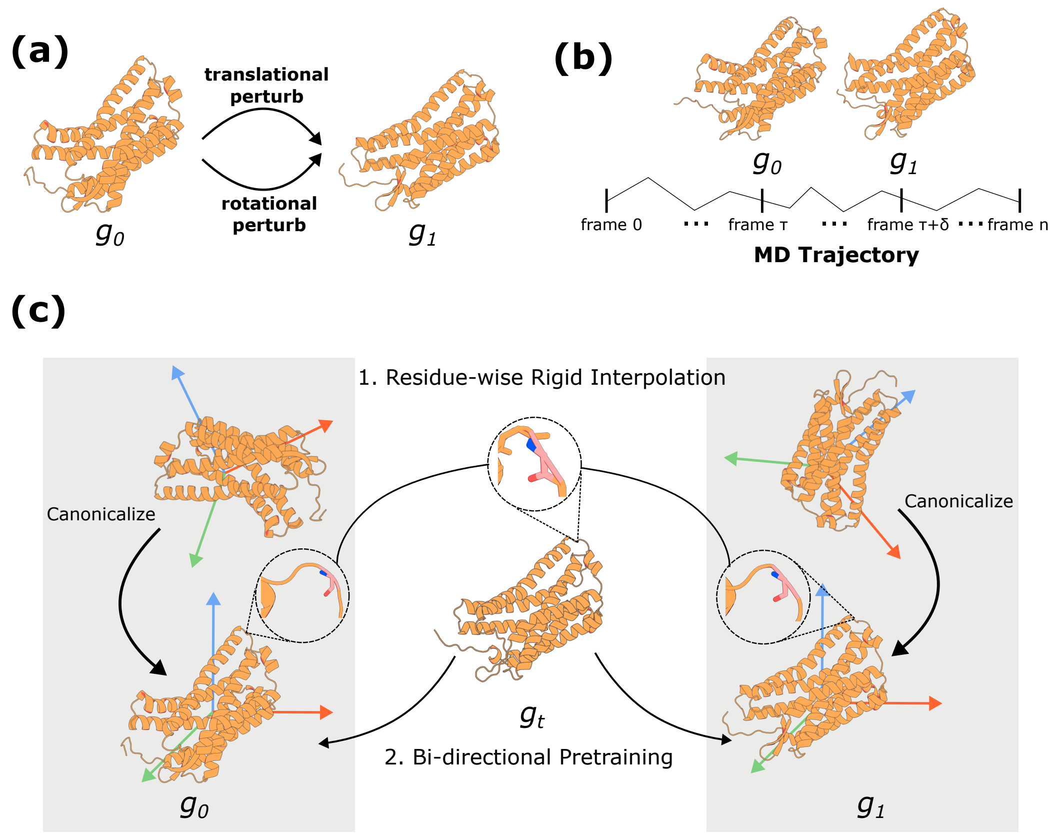

Figure 1: Overview of RigidSSL. (a) View construction in RigidSSL-Perturb: translational noise in $\mathbb{R}^3$ and rotational noise in $\mathrm{SO}(3)$ are applied to generate perturbations in the rigid–body motion group $\mathrm{SE}(3)$. (b) View construction in RigidSSL-MD: perturbed states are obtained by sampling alternative frames from MD trajectories. (c) Rigidity-based pretraining in RigidSSL: proteins are canonicalized into a reference frame, intermediate states are constructed via interpolation of translations and rotations for each rigid-reisude frame, and bi-directional flow matching is applied for pretraining. Details can be found in Section 3.

particularly in diffusion-based and autoregressive models; and (3) Powerful sequence design tools such as ProteinMPNN (Dauparas et al., 2022) can easily recover side chains, obviating the need to model them explicitly during structure generation. Together, these factors make rigid-body models especially effective for scalable protein generation, striking a balance between physical plausibility and computational efficiency.

Although large structural resources such as the AlphaFold Protein Structure Database (AFDB) are available, most protein generation models have relied primarily on experimentally resolved structures like the PDB (Bose et al., 2023; Campbell et al., 2024; Jing et al., 2024; Watson et al., 2023), with only limited exploration of AFDB subsets, as in FoldFlow2 (Huguet et al., 2024a). Trained end-to-end on these smaller datasets, such models are forced to learn both the fundamental rules of protein geometry and the complex task of *de novo* design simultaneously, often resulting in inefficiency, instability, and physically implausible outputs. Using AFDB for dedicated pretraining offers a way to decouple these challenges: by first learning robust geometric representations from large-scale data, models can acquire stronger inductive biases, improve sample efficiency, and establish a more stable foundation for downstream generation. Recent geometric pretraining efforts fall into three directions: (i) diffusion-based modeling (Guo et al., 2022), (ii) prediction of geometric features such as dihedral angles (Chen et al., 2023), and (iii) contrastive learning with perturbation-based views (Hermosilla and Ropinski, 2022; Zhang et al., 2022). However, these approaches rely on non-rigid local descriptors and fragmented substructures, which limit their ability to capture global protein geometry. Given the advantages of rigid-body representations in protein generation and the lack of large-scale pretraining that exploits AFDB, we raise the central question: *How can rigidity-based geometric pretraining better capture global geometry and generalize to protein design tasks?*

Our Contributions. To this end, we introduce RigidSSL, a rigidity-based pretraining paradigm with three key components: (1) We align each protein structure to its inertial frame, establishing a canonical reference frame across databases. This canonicalization removes arbitrary orientation differences and provides a consistent and generalizable basis for geometric pretraining, as shown in Figure 1 (c). (2) We design a two-phase pretraining workflow. In Phase 1, we employ perturbation-based view construction on the large-scale AFDB dataset of 542k structures (Varadi et al., 2022), simulating conformations in different energy regions. In a follow up Phase 2, we use 1.3k molecular dynamics trajectories to construct views that capture physically accurate conformational dynamics. The two-phase pipeline is shown in Figure 1 (a,b). (3) Building on these canonicalized views, we propose a rigid-body flow matching objective that uses the IPA for global geometry encoding. More concretely, we decompose residue-wise rigidity interpolation into a linear interpolation in $\mathbb{R}^3$ for translations and a spherical interpolation in $\mathrm{SO}(3)$ for rotations, and optimize a flow-matching surrogate loss to maximize the mutual information between views constructed in each phase, as illustrated in Figure 1 (c). We empirically demonstrate the effectiveness of RigidSSL on protein generation tasks, which outperforms previous state-of-the-art geometric pretraining algorithms, leading to improvements in unconditional generation across all metrics, including designability, novelty, and diversity, for length up to 800 residues.

Related Work. We briefly review the most related works here and include a more detailed discussion in Appendix A. Geometric representation learning has been extensively explored for both small molecules (Coors et al., 2018; Gasteiger et al., 2020; Schütt et al., 2018) and proteins (Fan et al., 2022; Jing et al., 2020; Wang et al., 2022), with methods categorized into SE(3)-invariant and SE(3)-equivariant models (Liu et al., 2023). Existing pretraining methods for small molecule optimize mutual information between or within modalities (Jiao et al., 2022; Liu et al., 2021; 2022a; Stärk et al., 2022; Zaidi et al., 2022). Protein-specific pretraining has evolved through diffusion-based methods (Guo et al., 2022), contrastive learning on substructures via GearNet (Zhang et al., 2022) and ProteinContrast (Hermosilla and Ropinski, 2022), and MSA-based strategies like MSAGPT (Chen et al., 2024). Finally, joint protein-ligand pretraining approaches, including CoSP (Gao et al., 2022) and MBP (Yan et al., 2023), model binding interactions directly.

2 Preliminiaries

Protein Structure Representation. A protein is composed of a sequence of residues, each containing three backbone atoms in the order N, C_α, and C. Accordingly, the protein backbone with L residues can be represented as $L \times [N, C_\alpha, C]$. Although there is rotational freedom around the N–C_α (ϕ) and C_α–C (ψ) bonds, following the common simplification used in AlphaFold2 (Jumper et al., 2021), we treat each residue as a rigid body. In this formulation, bond lengths and bond angles within the backbone are fixed to idealized values, and only torsional rotations (ϕ, ψ, ω) are allowed to vary. Because each residue is modeled as a rigid body, its configuration in 3D space can be described entirely by its position and orientation. Thus, a protein chain can be modeled as a sequence of rigid residues, where each residue is parameterized by a translation and a rotation. Formally, we represent a protein structure as a sequence of rigid transformations $g_{\text{raw}} = \{T_{\text{raw},i}\}_{i=1}^{L} = \{(\vec{t}_{\text{raw},i}, r_{\text{raw},i})\}_{i=1}^{L}$, where the subscript "raw" indicates the raw and uncanonicalized coordinates from the database, and each residue i is described by a translation vector $\vec{t}_i \in \mathbb{R}^3$ and a rotation matrix $r_i \in \mathrm{SO}(3)$, specifying its position and orientation in 3D space. Later, we will adopt a canonization process to align the protein structures to an invariant pose, *i.e.*, $g = \{T_i\}_{i=1}^{L} = \{(\vec{t}_i, r_i)\}_{i=1}^{L}$. We define $x_i := \vec{t}_i$ as the 3D coordinates of the C_α atom at residue i, so $\{x_i\}_{i=1}^{L}$ gives the C_α trace of the backbone. In what follows, we will use $\vec{t}_i$ and x_i interchangeably according to different contexts. Furthermore, each residue is also associated with an amino acid identity $A_i \in \{1, \ldots, 21\}$, representing the 20 standard amino acids plus one unidentified type. More details can be found in Appendix C.

Structure Encoder. Invariant Point Attention (IPA), first introduced in AlphaFold2 (Jumper et al., 2021), is a geometric attention mechanism designed to encode protein backbone representations. It is provably invariant under rigid Euclidean transformations due to the fact that the inner product of two rigidity bases stays constant under SE(3) transformations. We adopt IPA as the base model in both pretraining and downstream tasks. More details can be found in Appendix F.

Flow Matching. *Flow matching* has become an expressive generative model (Albergo and Vanden-Eijnden, 2022; Lipman et al., 2022; Liu et al., 2022b). It directly minimizes the discrepancy between a learnable vector field $v_t(x)$ and a target velocity field $u_t(x)$ that transports samples along a probability path from a simple prior distribution at $t = 0$ to a target distribution at $t = 1$. The training objective is

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[0,1],\, x\sim p_t(x)} \left[\|v_t(x) - u_t(x)\|^2\right], \tag{1}$$

where $v_t(x)$ is the model output and $u_t(x)$ is the ideal (but generally unknown) velocity field that induces the time-dependent distribution $p_t(x)$. However, this objective is often intractable because computing $u_t(x)$ requires knowledge of the exact marginal distributions and their dynamics.

To address this issue, *Conditional Flow Matching* (CFM) (Lipman et al., 2022) introduces conditioning on a known target sample x_1 to define a tractable conditional probability path from x_0 to x_1. The objective then becomes:

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[0,1],\, x_1\sim q(x_1),\, x\sim p_t(x|x_1)} \left[\|v_t(x) - u_t(x; x_1)\|^2\right], \tag{2}$$

where $q(x_1)$ denotes the marginal distribution of targets and $p_t(x \mid x_1)$ defines the interpolated distribution between x_0 and x_1 at time t. In this setting, $u_t(x; x_1)$ is a known velocity field (*e.g.*, from linear or spherical interpolation), making the training objective tractable. This formulation enables an efficient training of conditional generative models by leveraging path-based supervision.

3 METHOD: RIGIDSSL

Protein structures deposited in databases such as the PDB or AFDB are static, whether as single representative structures or averaged atomic positions. However, native proteins at physiological temperatures exhibit dynamics across multiple scales. These range from ultrafast atomic vibrations on the femtosecond timescale, to side-chain motions on the picosecond–nanosecond scale, and domain-level conformational changes on the microsecond–millisecond scale, collectively giving rise to a rich and highly variable energy landscape (Beach et al., 2005; Kovermann et al., 2016; Miller and Phillips, 2021; Nam and Wolf-Watz, 2023).

In this work, we propose RigidSSL to capture the fast, functionally neutral atomic vibrations in protein structures. It is a pretraining paradigm applicable to various downstream tasks. RigidSSL has three main components. The first is frame canonicalization. The second is a two-phase view construction. The third is an objective function based on rigidity-guided flow matching. Each component is detailed in Sections 3.1 to 3.3.

3.1 REFERENCE FRAME CANONICALIZATION

Protein structures in databases exist under arbitrary coordinate systems. To establish a canonical reference system, we align each protein structure to its reference inertial frame. From Section 2, recall that during modeling, we treat each protein structure g as a sequence of L rigid residues, where each residue is represented with a translation and rotation transformation, *i.e.*, $g_{\text{raw}} = \{T_{\text{raw},i}\}_{i=1}^{L} = \{(\vec{t}_{\text{raw},i}, r_{\text{raw},i})\}_{i=1}^{L}$. Recall that the subscript "raw" indicates the raw and uncanonicalized coordinates from the database. Based on this, the global alignment proceeds in two steps.

Translational Alignment. We align the protein to its center of mass, $\bar{x} = \frac{1}{L}\sum_{i=1}^{L} x_{\text{raw},i}$, which defines the origin of the inertial frame. Each residue's coordinate is shifted as $x_i = x_{\text{raw},i} - \bar{x}$, and the translation vector becomes $\vec{t_i} = \vec{t}_{\text{raw},i} - \bar{x}$.

Rotational Alignment. We align the protein to its principal axes, defined by the axes of the inertial frame. More concretely, we compute the inertia tensor $\hat{\mathbf{I}} = \frac{1}{L}\sum_{i=1}^{L}\left(\|x_i\|^2\mathbf{I}_3 - x_(x_i)^\top\right)$ where $\mathbf{I}_3$ is the 3×3 identity matrix. We then perform eigen-decomposition $\hat{\mathbf{I}} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top$, where columns of $\mathbf{V}$ contain the eigen-vectors forming the principal axes of inertia. Then, the aligned rotation matrix is computed as $r_i = r_{\text{raw},i} \cdot \mathbf{V}$.

The canonicalized protein structure is represented as $g = \{T_i\}_{i=1}^{L} = \{(\vec{t_i}, r_i)\}_{i=1}^{L}$. Frame alignment serves as a canonicalization step before RigidSSL-Perturb and RigidSSL-MD, as will be introduced next. By expressing all protein structures in a consistent reference frame, we ensure that the rotational and translational interpolation paths in SE(3) also reside in a consistent reference system, improving the generalizability of RigidSSL.

3.2 View Constructions in a Two-Phase Pretraining Framework

3.2.1 Phase 1: RigidSSL-Perturb

RigidSSL-Perturb adopts the massive AFDB (Varadi et al., 2022) with 542k unique protein structures. In each training iteration, RigidSSL-Perturb applies a simulated perturbation to each AFDB structure g^0 to generate a second view g^1, as shown in Figure 1(a). Concretely, for g^0 parameterized as a collection of *aligned* rigid bodies $\{T_i\}_{i=1}^L = \{(\vec{t_i}, r_i)\}_{i=1}^L$, RigidSSL-Perturb independently constructs a perturbed view for each rigid body $T_i = (\vec{t_i}, r_i)$ to obtain g^1. This residue-wise perturbation handles translation $\vec{t_i}$ and rotation r_i separately.

Translation Perturbation. ${x_i}^0$ is perturbed by adding a Gaussian noise in the Euclidean space:

$$\vec{t}_i^{\,1} = \vec{t}_i^{\,0} + \sigma \cdot \epsilon, \quad \epsilon \sim \mathcal{N}(0, \mathbf{I}_3). \tag{3}$$

Rotation Perturbation. To generate physically plausible structural variations in protein backbones, we employ the isotropic Gaussian distribution on the special orthogonal group $\mathrm{SO}(3)$ ($\mathcal{IG}_{SO(3)}$). This choice is guided by three key considerations: (1) protein dynamics fundamentally arise from thermal Brownian motion, which $\mathcal{IG}_{SO(3)}$ naturally models in the rotational domain (Yanagida et al., 2007), (2) rotations form a non-Euclidean manifold, requiring manifold-aware sampling to maintain geometric validity, and (3) since proteins explore their entire conformational landscape through continuous rotational changes, we need smooth, continuous sampling across all rotational states without singularities. Methodologically, $\mathcal{IG}_{SO(3)}$ is parametrized by a mean rotation $\mu \in \mathrm{SO}(3)$ and a concentration parameter $\epsilon \in \mathbb{R}_+$. The density function of this distribution is expressed as:

$$p(r; \mu, \epsilon^2) = \frac{1}{Z(\epsilon)} \exp\left(\frac{1}{\epsilon^2} \mathrm{trace}(\mu^T r)\right), \tag{4}$$

where $Z(\epsilon)$ is a normalization constant. We can decompose $\mathcal{IG}_{\mathrm{SO}(3)}$ into an axis distribution and an angle distribution when using the axis-angle representation for rotation, and more details are in Appendix D.2. To sample a rotation $r \sim \mathcal{IG}_{\mathrm{SO}(3)}(\mu, \varepsilon^2)$, we follow (Leach et al., 2022) as detailed in Appendix D.3. Then, the rotational rigid perturbation is obtained via:

$$r_i^{\,1} = r_i^{\,0} \cdot r, \quad r \sim \mathcal{IG}_{\mathrm{SO}(3)}(\mathbf{I}, \epsilon^2). \tag{5}$$

Note that perturbations are applied to canonicalized proteins to ensure consistent geometric effects, as detailed in Appendix D.4.

Summary. Building upon Equations (3) and (5), we define a perturbation function *Perturb*$_{\sigma,\epsilon}$: $SE(3) \to SE(3)$ that applies both translational and rotational noise to individual rigid body frames:

$$T_i^{\,1} = \textit{Perturb}_{\sigma,\epsilon}(T_i^{\,0}) = (\vec{t}_i^{\,0} + \sigma \cdot \epsilon, r_i^{\,0} \cdot r), \quad \text{where } \epsilon \sim \mathcal{N}(0, \mathbf{I}_3), r \sim \mathcal{IG}_{\mathrm{SO}(3)}(\mathbf{I}, \epsilon^2). \tag{6}$$

And the second view is assembled as $g^1 = \{T_i^1\}_{i=1}^L$.

3.2.2 Phase 2: RigidSSL-MD

As for phase two, RigidSSL-MD aims to learn about more realistic protein structural dynamics. It is trained on the ATLAS dataset (Vander Meersche et al., 2024), which contains 1.3k MD trajectories. MD simulations sample the energy landscape of proteins by numerically integrating Newton's equations of motion, producing trajectories that reflect structural fluctuations governed by physical force fields. To capture such fluctuations, we form two views g^0 and g^1 from time–separated snapshots within the same trajectory.

In a trajectory, the raw snapshot at time τ can be denoted as $F_{\mathrm{raw},\tau} = \{(\vec{t}^{\,\tau}_{\mathrm{raw},i}, r^{\tau}_{\mathrm{raw},i})\}_{i=1}^L$. We extract pairs $\big(F_{\mathrm{raw},\tau}, F_{\mathrm{raw},\tau+\delta}\big)$ with a fixed interval δ. The choice of δ controls the scale of conformational variation: smaller values capture only minor thermal vibrations, whereas larger values may reflect substantial conformational rearrangements. We set $\delta = 2$ ns to yield views that represent conformational fluctuations. Canonicalizing the raw snapshots gives $F^\tau, F^{\tau+\delta}$, yielding $g^0 = F^\tau = \{(\vec{t}_i^{\,\tau}, r_i^\tau)\}_{i=1}^L$ and $g^1 = F^{\tau+\delta} = \{(\vec{t}_i^{\,\tau+\delta}, r_i^{\tau+\delta})\}_{i=1}^L$.

3.3 Rigid Flow Matching for Multi-view Pretraining: RigidSSL

Our goal is to learn a generalizable latent space by maximizing the mutual information between the constructed views of a protein structure, *i.e.*, $\mathcal{L} = \text{MI}(g^0, g^1)$. It encourages the model to capture the underlying geometric and structural patterns that are preserved across different views.

To address this, we follow Liu et al. (2021) and adopt a surrogate objective based on conditional likelihoods:

$$\mathcal{L} = \log p(g^0 \mid g^1) + \log p(g^1 \mid g^0), \tag{7}$$

where $g^0 = \{\vec{t_i}^0, {r_i}^0\}$ and $g^1 = \{\vec{t_i}^1, {r_i}^1\}$ denote two rigid-body views of a protein, with translations t and rotations R . For later calculation, we convert the rotation matrix R into a quaternion q for interpolation between views. Thus, the objective becomes

$$\mathcal{L} = \log p(\{\vec{t_i}^0, {q_i}^0\} \mid \{\vec{t_i}^1, {q_i}^1\}) + \log p(\{\vec{t_i}^1, {q_i}^1\} \mid \{\vec{t_i}^0, {q_i}^0\}). \tag{8}$$

To optimize Equation (8), we adopt the Flow Matching framework (Albergo and Vanden-Eijnden, 2022; Lipman et al., 2022; Liu et al., 2022b). Unlike generic flows, our model respects the inductive bias that each residue behaves as a rigid body during denoising. The objective is to learn a velocity field that drives the system from g^0 to g^1 (and vice versa), via an intermediate state at time $\tau \in [0, 1]$. We decompose this into translation and rotation components, as described below.

LERP for Translation in $\mathbb{R}^3$. We interpolate the translations between residues using linear interpolation (LERP):

$$x_\tau = \text{LERP}(\vec{t}^{\,0}, \vec{t}^{\,1}, \tau) = \tau \vec{t}^{\,0} + (1 - \tau)\vec{t}^{\,1}, \tag{9}$$

where $\tau \in [0, 1]$ is the interpolation parameter.

SLERP for Rotation in SO(3). For rotations, we interpolate quaternions using spherical linear interpolation (SLERP):

$$q_\tau = \text{SLERP}(q^0, q^1, \tau) = \frac{\sin((1-\tau)\phi)q^0 + \sin(\tau\phi)q^1}{\sin(\phi)}, \tag{10}$$

where $\tau \in [0, 1]$ is the interpolation parameter and ϕ is the angle between q^0 and q^1.

Recall that we are using IPA as our backbone model, v_θ. We want v_θ to learn the true flow of both translation and rotation (quaternion) through flow matching, *i.e.*, $[\mathbf{u}_{\theta,\mathbb{R}^3}, \mathbf{u}_{\theta,\text{SO}(3)}] = v_\theta(x_\tau, q_\tau, \tau)$. Thus, the objective function for one direction $g^0 \to g^1$ is:

$$\mathcal{L}^{g^0 \to g^1} = \mathcal{L}_{\mathbb{R}^3} + \mathcal{L}_{\text{SO}(3)} = \left\|\vec{t_i}^1 - \vec{t_i}^0 - \mathbf{u}_{\theta,\mathbb{R}^3}\right\|^2 + \left\|\frac{d}{d\tau}\text{SLERP}({q_i}^0, {q_i}^1, \tau) - \mathbf{u}_{\theta,\text{SO}(3)}\right\|^2. \tag{11}$$

Final Objective. We apply the same formulation in the reverse direction, yielding the final loss:

$$\mathcal{L} = \mathcal{L}^{g^0 \to g^1} + \mathcal{L}^{g^1 \to g^0}. \tag{12}$$

This final objective is shared between RigidSSL-Perturb and RigidSSL-MD.

4 Experiments: Protein Structure Generation

One of the most widely acknowledged benefits of pretraining is the enhancement of model performance on downstream tasks. Models initialized with pretrained weights tend to achieve higher performance compared to models trained from scratch using only the task-specific data. This improvement is often attributed to the model's ability to learn general, transferable features or knowledge from the large-scale pretraining dataset. This knowledge, particularly features that are difficult to discern from smaller datasets, provides a significant advantage. In addition, pretraining effectively gives the model a "head-start" in the learning process. Instead of starting with random weights, the model begins fine-tuning with parameters already optimized for a related task or general data distribution. This may lead to significantly faster convergence during the fine-tuning phase, as the model requires fewer iterations to reach optimal performance for the downstream task.

We begin by pretraining the IPA module with RigidSSL following the scheme described in Appendix F. To evaluate the effectiveness of RigidSSL, we conduct experiments on unconditional protein structure generation and compare against four pretraining baselines: a randomly initialized model and three coordinate-aware geometric self-supervised learning methods that maximize mutual information (MI). Specifically, GeoSSL-InfoNCE and GeoSSL-EBM-NCE employ contrastive objectives that align positive pairs while repelling negatives (Liu et al., 2021; Oord et al., 2019), whereas GeoSSL-RR adopts a generative objective, leveraging intra-protein supervision by reconstructing one view from its counterpart in the representation space (Liu et al., 2021). A comparison of pretraining and downstream training configurations can be found in Table Table 1.

Table 1: Overview of pretraining and downstream training configurations. This table compares data sources, dataset sizes, protein lengths, training objectives, and computational resources.

Method	**Stage**	**Data Source**	**Data Size**	**Length**	**Objective**	**Compute & Time**
GeoSSL-InfoNCE	Pretraining	AFDB (UniProtKB)	432,194	60-512	Contrastive (InfoNCE)	4 days (4$\times$A100)
GeoSSL-EBM-NCE	Pretraining	AFDB (UniProtKB)	432,194	60-512	Contrastive (EBM-NCE)	4 days (4$\times$A100)
GeoSSL-RR	Pretraining	AFDB (UniProtKB)	432,194	60-512	Generative (Representation Reconstruction)	3 days (4$\times$A100)
RigidSSL-Perturb (Ours)	Pretraining_Phase1	AFDB (UniProtKB)	432,194	60-512	Generative (RigidSSL)	2.75 days (1$\times$H100)
RigidSSL-MD (Ours)	Pretraining_Phase2	ATLAS	1,390 traj.	60-512	Generative (RigidSSL)	1.88 days (1$\times$H100)
FrameDiff	Fine-tuning	PDB	20,312	60-512	SE(3) Diffusion	7 days (4$\times$H100)
FoldFlow-2	Fine-tuning	PDB	20,312	60-384	SE(3) Flow Matching	4 days (2$\times$A100)

4.1 Downstream: Unconditional Protein Structure Generation

We evaluate two state-of-the-art generative models for protein backbones, FrameDiff and FoldFlow-2, both built on invariant point attention and rigid-body representations. FrameDiff (Yim et al., 2023) generates novel backbones by defining a diffusion process directly on the manifold of residue-wise SE(3) frames. Built on invariant point attention (IPA) to update embeddings, it learns an SE(3)-equivariant score function that drives the reverse diffusion dynamics. FoldFlow-2 (Huguet et al., 2024b) is a sequence-augmented, SE(3)-equivariant flow-matching model. Its architecture uses IPA twice: first to encode structures into latent representations, and then to decode multimodal inputs into $SE(3)_0^N$ vector fields for backbone generation.

We evaluate RigidSSL by replacing the IPA module in FrameDiff and FoldFlow-2, followed by finetuning on monomer backbone generation using their original training strategies (see Sec. F). We then assess the generated samples in terms of designability, diversity, and novelty. **Designability** is a self-consistency metric that examines if there exist amino acid sequences that can fold into the generated structure. Specifically, we employ ProteinMPNN (Dauparas et al., 2022) to design sequences (i.e. reverse fold) for FrameDiff/FoldFlow-2 structures, which are then folded with ESMFold (Lin et al., 2023) to compare with the original structure. We quantify self-consistency through scRMSD and employ scRMSD< 2Å as the criterion for being designable. **Novelty** assesses whether the model can generalize beyond the training set and produce novel backbones dissimilar from those in PDB. We use FoldSeek (Van Kempen et al., 2024) to search for similar structures and report the highest TM-scores (measure of similarity between two protein structures) of samples to any chain in PDB. **Diversity** measures the structural difference between generated samples. We quantify diversity through the number of distinct structural clusters via clustering with MaxCluster (Herbert, A. and Sternberg, M., 2008) to hierarchically cluster backbones with a 0.5 TM-score threshold. We report diversity as the proportion of unique clusters: (number of clusters) / (number of samples).

To determine the optimal translation and rotation noise levels for constructing perturbed views in Phase 1, we conducted an ablation study (Appendix G) and selected a translation noise scale of $\sigma = 0.03$ and a rotation noise scale of $\mu = 0.5$. As shown in Table 2, compared with no pretraining or with GeoSSL-EBM-NCE, GeoSSL-InfoNCE, and GeoSSL-RR, FrameDiff pretrained with RigidSSL-Perturb achieves substantially higher performance in designability, novelty, and diversity. Specifically, relative to the unpretrained FrameDiff, RigidSSL-Perturb improves mean designability 10%, mean novelty by 6.1%, and mean diversity by 3.1% (as measured by pairwise TM). However, while RigidSSL-MD further improves mean diversity by 9.4% relative to the unpretrained model, it leads decreased performance in designability and novelty. Similarly, for FoldFlow-2, RigidSSL-Perturb improves mean designability by 42.9%, mean novelty by 4%, and mean diversity by 6.9% (as measured by MaxCluster) when compared with unpretrained. We hypothesize that the increased diversity across FrameDiff and FoldFlow-2 by RigidSSL-MD arises from the ATLAS dataset, as it offers an exhaustive and non-redundant coverage of the PDB conformational space. More discussion can be found in Section 5. Furthermore, we examine the distribution of secondary structures, as shown

Table 2: Comparison of Designability (fraction of proteins with scRMSD < 2.0Å), Diversity (avg. pairwise TMscore and MaxCluster diversity), Novelty (max. TM-score to PDB). Designability, Novelty, and Diversity metrics include standard errors. The best mean values are shown in **bold** and the second-best are <u>underlined</u>.

	Pretraining	Length	Designability	Novelty	Diversity	
			Fraction (↑)	avg. max TM (↓)	pairwise TM (↓)	MaxCluster (↑)
FrameDiff	None	100-300	<u>0.775 ± 0.100</u>	<u>0.555 ± 0.137</u>	0.565 ± 0.029	0.033
FrameDiff	GeoSSL-EBM-NCE	100-300	0.725 ± 0.420	0.615 ± 0.339	0.597 ± 0.058	0.033
FrameDiff	GeoSSL-InfoNCE	100-300	0.650 ± 0.091	0.613 ± 0.097	0.568 ± 0.017	0.033
FrameDiff	GeoSSL-RR	100-300	0.700 ± 0.109	0.579 ± 0.108	0.604 ± 0.026	<u>0.089</u>
FrameDiff	RigidSSL-Perturb (ours)	100-300	$\mathbf{0.875 \pm 0.096}$	$\mathbf{0.494 \pm 0.261}$	<u>0.534 ± 0.107</u>	0.033
FrameDiff	RigidSSL-MD (ours)	100-300	0.700 ± 0.102	0.657 ± 0.139	$\mathbf{0.471 \pm 0.013}$	$\mathbf{0.156}$
FoldFlow2	None	100-600	0.329 ± 0.013	0.810 ± 0.004	0.620 ± 0.002	0.183
FoldFlow2	GeoSSL-EBM-NCE	100-600	0.424 ± 0.014	0.790 ± 0.009	0.626 ± 0.002	0.225
FoldFlow2	GeoSSL-InfoNCE	100-600	0.333 ± 0.012	0.786 ± 0.005	0.631 ± 0.001	0.052
FoldFlow2	GeoSSL-RR	100-600	0.344 ± 0.012	0.787 ± 0.005	$\mathbf{0.601 \pm 0.003}$	0.137
FoldFlow2	RigidSSL-Perturb (ours)	100-600	$\mathbf{0.758 \pm 0.016}$	$\mathbf{0.770 \pm 0.003}$	0.650 ± 0.001	<u>0.252</u>
FoldFlow2	RigidSSL-MD (ours)	100-600	<u>0.584 ± 0.018</u>	<u>0.782 ± 0.004</u>	<u>0.613 ± 0.002</u>	$\mathbf{0.318}$

Table 3: Comparison of Clashscore and MolProbity scores for 700- and 800-residue proteins generated by FoldFlow-2 under different pretraining methods. The best values are shown in **bold** and the second-best are <u>underlined</u>.

Metric	**Length**	Unpretrained	GeoSSL-EBM-NCE	GeoSSL-RR	GeoSSL-InfoNCE	RigidSSL-Perturb	RigidSSL-MD
Clashscore (↓)	700	152.53	45.11	96.38	55.02	**21.36**	<u>37.16</u>
	800	172.36	61.70	123.16	74.98	**26.42**	<u>39.35</u>
MolProbity score (↓)	700	2.64	2.22	2.51	2.21	**1.82**	<u>2.05</u>
	800	2.69	2.38	2.61	2.34	**1.91**	<u>2.08</u>

in Figure 2. FoldFlow-2 pretrained with RigidSSL-MD exhibits the greatest secondary structure diversity, followed by RigidSSL-Perturb. While unpretrained and GeoSSL-EBM-NCE pretraining leads to predominantly α-helix structures, RigidSSL yields a substantially broader spectrum that include coils and mixed α-helix/β-sheet compositions.

4.2 Case Study: Long Chain Generation

To explore the potential of RigidSSL, we conducted a case study on the generation of long protein chains (details provided in F.2.2). Whereas the original FoldFlow-2 analysis was restricted to sequences of 100–300 residues, we extend the evaluation to ultra-long sequences of 700 and 800 residues under different pretraining methods. We visualize the best structure in terms of scRMSD among 50 generated samples in Figure 3. Notably, only FoldFlow-2 pretrained with RigidSSL-Perturb is able to produce self-consistent and thus designable structures.

To further understand the stereochemistry of the generated structures, we use MolProbity (Davis et al., 2007) to compute: (1) Clashscore, which is the number of serious steric overlaps ($>$0.4 Å) per 1000 atoms, and (2) the MolProbity score, which is a composite metric integrating clashscore, rotamer, and Ramachandran evaluations. As shown in Table 3, FoldFlow-2 pretrained with RigidSSL-Perturb achieves the best clashscore and MolProbity score, indicating that it effectively captures the biophysical constraints underlying long, complex protein structures despite being trained only on shorter sequences (60–384 residues). This demonstrates that RigidSSL-Perturb substantially improves the model's ability to maintain stereochemical accuracy at long sequence lengths, suggesting robust learning of global structural patterns that generalize beyond the training regime.

5 Discussion

Discussion on Self-Supervised RigidSSL-Perturb. In the self-supervised pretraining step in Phase 1, we employ data augmentation by adding noise to the original geometry to construct a perturbed geometry as the second view. This choice is motivated by three considerations. (1) Gaussian perturbation can be interpreted as a form of "masking" applied to atom positions in 3D Euclidean space. Analogous to how self-supervised learning in computer vision masks image patches or in NLP masks tokens, noise injection in geometry partially obscures atomic coordinates and encourages the model to infer consistent global features across corrupted inputs. (2) At small noise scales, this augmentation mimics systematic errors that arise during data collection or prediction. Protein

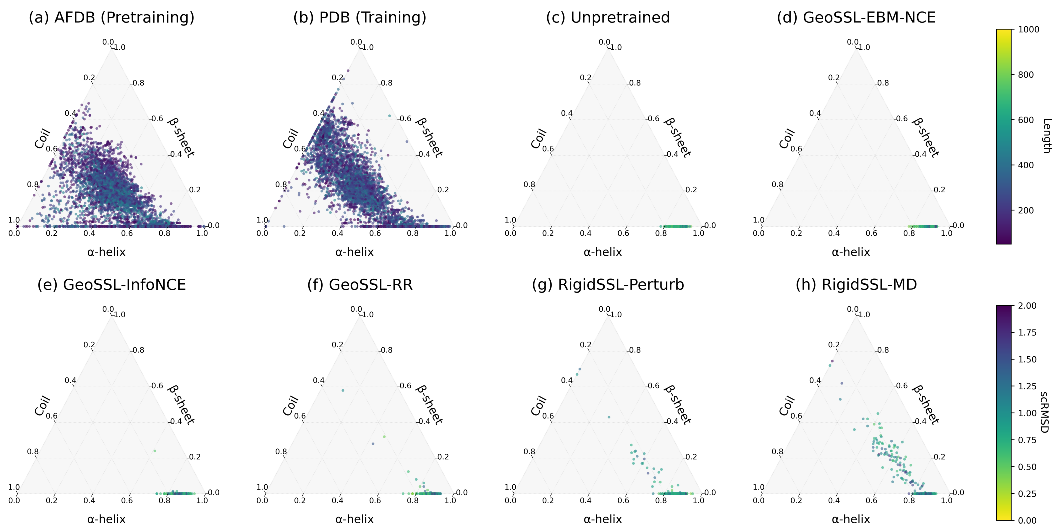

Figure 2: Distribution of secondary structure elements (α-helices, β-sheets, and coils) in protein structure database (a-b) and in designable proteins (scRMSD < 2.0) generated by FoldFlow-2 under different pretraining methods (c-h). Plots of the structure database are color-coded by sequence length, whereas those of the generated structures are color-coded by scRMSD.

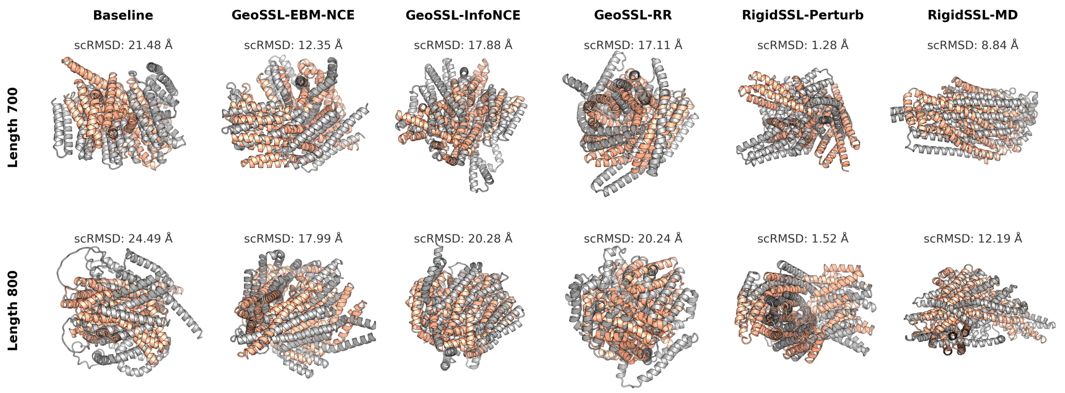

Figure 3: FoldFlow-2 generated structures (orange) compared against ProteinMPNN $\rightarrow$ ESMFold refolded structures (grey). Columns denote pretraining methods, and rows denote sequence lengths of 700 and 800.

structures from crystallography, cryo-EM, or computational databases such as AFDB, often normally contain small uncertainties in atom placement. Exposing the model to controlled perturbations helps it become robust to such imperfections. (3) Gaussian perturbations can also approximate natural conformational fluctuations around a stable equilibrium. Proteins are dynamic molecules, and their functional states often involve small deviations from the ground-state structure. Training with noisy perturbations introduces the model to this variability, allowing it to capture flexible yet physically plausible structural patterns.

As shown in Table 2, Phase 1 pretraining improves designability, indicating that the model's outputs shift toward a more reliable distribution, but this comes at the cost of reduced diversity. While random perturbation provides clear benefits for robustness and representation learning, it also introduces an inherent trade-off between designability and diversity. By enforcing invariance across noisy views, the model learns to emphasize stable, fold-defining features while disregarding minor structural deviations. This bias enhances designability, as downstream generators built on such representations tend to produce more foldable backbones. However, the same invariance can also suppress conformational variability, collapsing nearby structures into a single representation and thereby reducing diversity. In practice, noise perturbation prioritizes stability over exploration: it improves the reliability of designed proteins but may limit the diversity of generated folds. In Appendix G, we analyze how the noise scales in RigidSSL-Perturb affects the structures generated in downstream tasks.

Discussion on Supervised RigidSSL-MD. In the second phase, RigidSSL-MD incorporates molecular dynamics (MD) trajectories to build augmented views. While this improves structural diversity, it reduces designability compared to perturbation-based pretraining (Table 2). The trade-off arises because MD pretraining is not purely self-supervised: it depends on force-field energies and gradients, inheriting biases that may misalign with de novo design goals and even cause negative transfer, as seen in prior studies (Liu et al., 2023). At the same time, MD trajectories capture conformational dynamics across multiple metastable states, enriching flexibility and promoting greater diversity in the learned representations. However, curated MD datasets are far smaller and more expensive to obtain than large-scale static resources such as AFDB, limiting their statistical strength and scalability. Taken together, MD-based pretraining emphasizes flexibility and diversity at the expense of foldability, while perturbation-based SSL favors stability and designability, suggesting that the two approaches may be complementary depending on downstream priorities.

ICLR Paper Checklist

1. **The Use of Large Language Models**

   In this work, we used large language models (*e.g.*, ChatGPT) to (1) polish English expressions, (2) assist in formulating LaTeX equations, and (3) find relevant works.

2. **Ethics Statement**

   This research fully adheres to the ICLR Code of Ethics. The study does not involve human subjects or the use of personal or sensitive data. All datasets and code utilized and released conform to their respective licenses and terms of use. The contributions in this work are foundational and do not raise issues related to fairness, privacy, security, or potential misuse. We confirm that all ethical considerations have been thoroughly addressed.

3. **Reproducibility Statement**

   We are committed to making our work easy to reproduce. All necessary details for replicating our main experimental results, including data access, experimental setup, model configurations, evaluation metrics, and checkpoints, will be uploaded to GitHub after the paper is accepted. Users can follow our documentation and scripts to accurately reproduce the results, ensuring transparency and scientific rigor.

References

M. S. Albergo and E. Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *arXiv preprint arXiv:2209.15571*, 2022.

H. Beach, R. Cole, M. L. Gill, and J. P. Loria. Conservation of mus-ms enzyme motions in the apo- and substrate-mimicked state. *Journal of the American Chemical Society*, 127(25):9167–9176, June 2005. ISSN 0002-7863. doi: 10.1021/ja0514949.

A. J. Bose, T. Akhound-Sadegh, G. Huguet, K. Fatras, J. Rector-Brooks, C.-H. Liu, A. C. Nica, M. Korablyov, M. Bronstein, and A. Tong. Se (3)-stochastic flow matching for protein backbone generation. *arXiv preprint arXiv:2310.02391*, 2023.

A. Campbell, J. Yim, R. Barzilay, T. Rainforth, and T. Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. *arXiv preprint arXiv:2402.04997*, 2024.

B. Chen, Z. Bei, X. Cheng, P. Li, J. Tang, and L. Song. Msagpt: Neural prompting protein structure prediction via msa generative pre-training. *Advances in Neural Information Processing Systems*, 37:37504–37534, 2024.

C. Chen, J. Zhou, F. Wang, X. Liu, and D. Dou. Structure-aware protein self-supervised learning. *Bioinformatics*, 39(4):btad189, 2023.

H. Cheng, R. D. Schaeffer, Y. Liao, L. N. Kinch, J. Pei, S. Shi, B.-H. Kim, and N. V. Grishin. ECOD: An Evolutionary Classification of Protein Domains. *PLOS Computational Biology*, 10(12):e1003926, Dec. 2014. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1003926. URL `https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1003926`. Publisher: Public Library of Science.

Y. Cho, M. Pacesa, Z. Zhang, B. E. Correia, and S. Ovchinnikov. Boltzdesign1: Inverting all-atom structure prediction model for generalized biomolecular binder design. *bioRxiv*, pages 2025–04, 2025.

B. Coors, A. P. Condurache, and A. Geiger. Spherenet: Learning spherical representations for detection and classification in omnidirectional images. In *Proceedings of the European conference on computer vision (ECCV)*, pages 518–533, 2018.

J. Dauparas, I. Anishchenko, N. Bennett, H. Bai, R. J. Ragotte, L. F. Milles, B. I. Wicky, A. Courbet, R. J. de Haas, N. Bethel, et al. Robust deep learning–based protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022.

I. W. Davis, A. Leaver-Fay, V. B. Chen, J. N. Block, G. J. Kapral, X. Wang, L. W. Murray, W. B. Arendall, J. Snoeyink, J. S. Richardson, and D. C. Richardson. MolProbity: all-atom contacts and structure validation for proteins and nucleic acids. *Nucleic Acids Research*, 35(Web Server issue): W375–383, July 2007. ISSN 1362-4962. doi: 10.1093/nar/gkm216.

H. Fan, Z. Wang, Y. Yang, and M. Kankanhalli. Continuous-discrete convolution for geometry-sequence modeling in proteins. In *The Eleventh International Conference on Learning Representations*, 2022.

Z. Gao, C. Tan, L. Wu, and S. Z. Li. Cosp: Co-supervised pretraining of pocket and ligand. *arXiv preprint arXiv:2206.12241*, 2022.

J. Gasteiger, J. Groß, and S. Günnemann. Directional message passing for molecular graphs. *arXiv preprint arXiv:2003.03123*, 2020.

T. Geffner, K. Didi, Z. Cao, D. Reidenbach, Z. Zhang, C. Dallago, E. Kucukbenli, K. Kreis, and A. Vahdat. La-proteina: Atomistic protein generation via partially latent flow matching. *arXiv preprint arXiv:2507.09466*, 2025a.

T. Geffner, K. Didi, Z. Zhang, D. Reidenbach, Z. Cao, J. Yim, M. Geiger, C. Dallago, E. Kucukbenli, A. Vahdat, et al. Proteina: Scaling flow-based protein structure generative models. *arXiv preprint arXiv:2503.00710*, 2025b.

Y. Guo, J. Wu, H. Ma, and J. Huang. Self-supervised pre-training for protein embeddings using tertiary structures. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6801–6809, 2022.

Herbert, A. and Sternberg, M. MaxCluster: A tool for Protein Structure Comparison and Clustering, 2008. URL `https://www.sbg.bio.ic.ac.uk/maxcluster/`.

P. Hermosilla and T. Ropinski. Contrastive representation learning for 3d protein structures. *arXiv preprint arXiv:2205.15675*, 2022.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

G. Huguet, J. Vuckovic, K. Fatras, E. Thibodeau-Laufer, P. Lemos, R. Islam, C. Liu, J. Rector-Brooks, T. Akhound-Sadegh, M. Bronstein, et al. Sequence-augmented se (3)-flow matching for conditional protein generation. *Advances in neural information processing systems*, 37:33007–33036, 2024a.

G. Huguet, J. Vuckovic, K. Fatras, E. Thibodeau-Laufer, P. Lemos, R. Islam, C.-H. Liu, J. Rector-Brooks, T. Akhound-Sadegh, M. Bronstein, A. Tong, and A. J. Bose. Sequence-Augmented SE(3)-Flow Matching For Conditional Protein Backbone Generation, Dec. 2024b. URL `http://arxiv.org/abs/2405.20313`. arXiv:2405.20313 [cs].

R. Jiao, J. Han, W. Huang, Y. Rong, and Y. Liu. 3d equivariant molecular graph pretraining. *arXiv preprint arXiv:2207.08824*, 2022.

B. Jing, S. Eismann, P. Suriana, R. J. Townshend, and R. Dror. Learning from protein structure with geometric vector perceptrons. *arXiv preprint arXiv:2009.01411*, 2020.

B. Jing, B. Berger, and T. Jaakkola. AlphaFold Meets Flow Matching for Generating Protein Ensembles, Sept. 2024. URL `http://arxiv.org/abs/2402.04845`. arXiv:2402.04845 [q-bio].

J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, et al. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, 2021.

D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization, Jan. 2017. URL `http://arxiv.org/abs/1412.6980`. arXiv:1412.6980 [cs].

M. Kovermann, P. Rogne, and M. Wolf-Watz. Protein dynamics and function from solution state NMR spectroscopy. *Quarterly Reviews of Biophysics*, 49:e6, 2016. ISSN 1469-8994. doi: 10.1017/S0033583516000019.

A. LaPelusa and R. Kaushik. Physiology, proteins. 2020.

A. Leach, S. M. Schmon, M. T. Degiacomi, and C. G. Willcocks. Denoising Diffusion Probabilistic Models on SO(3) for Rotational Alignment. Mar. 2022. URL `https://openreview.net/forum?id=BY88eBbkpe5`.

Z. Lin, H. Akin, R. Rao, B. Hie, Z. Zhu, W. Lu, N. Smetanin, R. Verkuil, O. Kabeli, Y. Shmueli, A. dos Santos Costa, M. Fazel-Zarandi, T. Sercu, S. Candido, and A. Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, Mar. 2023. doi: 10.1126/science.ade2574. URL `https://www.science.org/doi/10.1126/science.ade2574`. Publisher: American Association for the Advancement of Science.

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

D. Listov, C. A. Goverde, B. E. Correia, and S. J. Fleishman. Opportunities and challenges in design and optimization of protein function. *Nature Reviews Molecular Cell Biology*, 25(8): 639–653, Aug. 2024. ISSN 1471-0080. doi: 10.1038/s41580-024-00718-y. URL `https://www.nature.com/articles/s41580-024-00718-y`. Publisher: Nature Publishing Group.

S. Liu, H. Wang, W. Liu, J. Lasenby, H. Guo, and J. Tang. Pre-training molecular graph representation with 3d geometry. *arXiv preprint arXiv:2110.07728*, 2021.

S. Liu, H. Guo, and J. Tang. Molecular geometry pretraining with se (3)-invariant denoising distance matching. *arXiv preprint arXiv:2206.13602*, 2022a.

S. Liu, W. Du, Y. Li, Z. Li, Z. Zheng, C. Duan, Z. Ma, O. Yaghi, A. Anandkumar, C. Borgs, J. Chayes, H. Guo, and J. Tang. Symmetry-informed geometric representation for molecules, proteins, and crystalline materials. *arXiv preprint arXiv:2306.09375*, 2023.

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

M. D. Miller and G. N. Phillips. Moving beyond static snapshots: Protein dynamics and the Protein Data Bank. *The Journal of Biological Chemistry*, 296:100749, May 2021. ISSN 0021-9258. doi: 10.1016/j.jbc.2021.100749. URL `https://pmc.ncbi.nlm.nih.gov/articles/PMC8164045/`.

A. Miserez, J. Yu, and P. Mohammadi. Protein-Based Biological Materials: Molecular Design and Artificial Production. *Chemical Reviews*, 123(5):2049–2111, Mar. 2023. ISSN 0009-2665. doi: 10.1021/acs.chemrev.2c00621. URL `https://doi.org/10.1021/acs.chemrev.2c00621`. Publisher: American Chemical Society.

K. Nam and M. Wolf-Watz. Protein dynamics: The future is bright and complicated! *Structural Dynamics*, 10(1):014301, Feb. 2023. ISSN 2329-7778. doi: 10.1063/4.0000179. URL `https://pmc.ncbi.nlm.nih.gov/articles/PMC9974214/`.

A. v. d. Oord, Y. Li, and O. Vinyals. Representation Learning with Contrastive Predictive Coding, Jan. 2019. URL `http://arxiv.org/abs/1807.03748`. arXiv:1807.03748 [cs].

A. Roy, E. Ward, I. Choi, M. Cosi, T. Edgin, T. S. Hughes, M. S. Islam, A. M. Khan, A. Kolekar, M. Rayl, I. Robinson, P. Sarando, E. Skidmore, T. L. Swetnam, M. Wall, Z. Xu, M. Yung, N. Merchant, and T. J. Wheeler. MDRepo—an open data warehouse for community-contributed molecular dynamics simulations of proteins. *Nucleic Acids Research*, 53(D1):D477–D486, Jan. 2025. ISSN 1362-4962. doi: 10.1093/nar/gkae1109. URL `https://doi.org/10.1093/nar/gkae1109`.

K. T. Schütt, H. E. Sauceda, P.-J. Kindermans, A. Tkatchenko, and K.-R. Müller. Schnet–a deep learning architecture for molecules and materials. *The Journal of Chemical Physics*, 148(24), 2018.

H. Stärk, D. Beaini, G. Corso, P. Tossou, C. Dallago, S. Günnemann, and P. Liò. 3d infomax improves gnns for molecular property prediction. In *International Conference on Machine Learning*, pages 20479–20502. PMLR, 2022.

M. Van Kempen, S. S. Kim, C. Tumescheit, M. Mirdita, J. Lee, C. L. Gilchrist, J. Söding, and M. Steinegger. Fast and accurate protein structure search with foldseek. *Nature biotechnology*, 42 (2):243–246, 2024.

Y. Vander Meersche, G. Cretin, A. Gheeraert, J.-C. Gelly, and T. Galochkina. Atlas: protein flexibility description from atomistic molecular dynamics simulations. *Nucleic acids research*, 52(D1): D384–D392, 2024.

M. Varadi, S. Anyango, M. Deshpande, S. Nair, C. Natassia, G. Yordanova, D. Yuan, O. Stroe, G. Wood, A. Laydon, A. Žídek, T. Green, K. Tunyasuvunakool, S. Petersen, J. Jumper, E. Clancy, R. Green, A. Vora, M. Lutfi, M. Figurnov, A. Cowie, N. Hobbs, P. Kohli, G. Kleywegt, E. Birney, D. Hassabis, and S. Velankar. AlphaFold Protein Structure Database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic Acids Research*, 50(D1):D439–D444, Jan. 2022. ISSN 0305-1048. doi: 10.1093/nar/gkab1061. URL `https://doi.org/10.1093/nar/gkab1061`.

L. Wang, H. Liu, Y. Liu, J. Kurtin, and S. Ji. Learning hierarchical protein representations via complete 3d graph networks. *arXiv preprint arXiv:2207.12600*, 2022.

J. L. Watson, D. Juergens, N. R. Bennett, B. L. Trippe, J. Yim, H. E. Eisenach, W. Ahern, A. J. Borst, R. J. Ragotte, L. F. Milles, B. I. M. Wicky, N. Hanikel, S. J. Pellock, A. Courbet, W. Sheffler, J. Wang, P. Venkatesh, I. Sappington, S. V. Torres, A. Lauko, V. De Bortoli, E. Mathieu, S. Ovchinnikov, R. Barzilay, T. S. Jaakkola, F. DiMaio, M. Baek, and D. Baker. De novo design of protein structure and function with RFdiffusion. *Nature*, 620(7976):1089–1100, Aug. 2023. ISSN 1476-4687. doi: 10.1038/s41586-023-06415-8. URL `https://www.nature.com/articles/s41586-023-06415-8`. Publisher: Nature Publishing Group.

J. Yan, Z. Ye, Z. Yang, C. Lu, S. Zhang, Q. Liu, and J. Qiu. Multi-task bioassay pre-training for protein-ligand binding affinity prediction. *arXiv preprint arXiv:2306.04886*, 2023.

T. Yanagida, M. Ueda, T. Murata, S. Esaki, and Y. Ishii. Brownian motion, fluctuation and life. *Bio Systems*, 88(3):228–242, Apr. 2007. ISSN 0303-2647. doi: 10.1016/j.biosystems.2006.08.012.

J. Yim, B. L. Trippe, V. De Bortoli, E. Mathieu, A. Doucet, R. Barzilay, and T. Jaakkola. Se (3) diffusion model with application to protein backbone generation. *arXiv preprint arXiv:2302.02277*, 2023.

S. Zaidi, M. Schaarschmidt, J. Martens, H. Kim, Y. W. Teh, A. Sanchez-Gonzalez, P. Battaglia, R. Pascanu, and J. Godwin. Pre-training via denoising for molecular property prediction. *arXiv preprint arXiv:2206.00133*, 2022.

Z. Zhang, M. Xu, A. Jamasb, V. Chenthamarakshan, A. Lozano, P. Das, and J. Tang. Protein representation learning by geometric structure pretraining. *arXiv preprint arXiv:2203.06125*, 2022.

APPENDIX

A RELATED WORKS

Geometric Representation for Small Molecules and Proteins. A wide range of representation learning models with invariance or equivariance properties have been proposed for both small molecules (Coors et al., 2018; Gasteiger et al., 2020; Schütt et al., 2018) and proteins (Fan et al., 2022; Jing et al., 2020; Wang et al., 2022), targeting tasks such as property prediction and molecular generation. As a comprehensive benchmark, Geom3D (Liu et al., 2023) offers a systematic roadmap for geometric representations of small molecules and proteins. It categorizes mainstream approaches into three major families: SE(3)-invariant models, SE(3)-equivariant models with spherical frame bases, and SE(3)-equivariant models with vector frame bases.

Geometric Pretraining for Small Molecules. The GraphMVP (Liu et al., 2021) proposes one contrastive objective (EBM-NCE) and one generative objective (variational representation reconstruction, VRR) to optimize the mutual information between the topological and conformational modalities. 3D InfoMax (Stärk et al., 2022) is a special case of GraphMVP, where only the contrastive loss is considered. GeoSSL (Liu et al., 2022a) proposes maximizing the mutual information between noised conformations using an SE(3)-invariant denoising score matching and a parallel work (Zaidi et al., 2022) is a special case of GeoSSL using only one denoising layer. 3D-EMGP (Jiao et al., 2022) is a parallel work, yet it is E(3)-equivariant, which needlessly satisfies the reflection-equivariant constraint in molecular conformation distribution.

Geometric Pretraining for Proteins. Several parallel efforts have been made toward geometric pretraining for proteins. Guo et al. (2022) focuses on maximizing mutual information between diffusion trajectory representations of structurally correlated conformers. GearNet (Zhang et al., 2022) introduces a contrastive self-supervised framework trained on AlphaFold Database v1, aiming to maximize the similarity between representations of substructures within the same protein through sparse edge message passing. Similarly, ProteinContrast (Hermosilla and Ropinski, 2022) proposes a contrastive learning approach trained on 476K PDB structures to learn effective 3D protein representations. More recently, MSAGPT (Chen et al., 2024) presents a novel pretraining strategy based on Multiple Sequence Alignments (MSAs), where the model learns to generate MSAs by capturing their statistical distribution, thereby improving structure prediction for proteins with few or no known homologs.

Binding Pretraining on Stable Positions. In addition to pretraining on proteins and small molecules separately, several studies have explored joint pretraining strategies that model protein–ligand interactions directly. CoSP (Gao et al., 2022) introduces a co-supervised framework that simultaneously learns 3D representations of protein pockets and ligands using a gated geometric message passing network, incorporating contrastive loss to align real pocket–ligand pairs and employing a chemical similarity-enhanced negative sampling strategy to improve performance. MBP (Yan et al., 2023) addresses the scarcity of high-quality 3D structural data by constructing a pretraining dataset called ChEMBL-Dock, including over 300K protein–ligand pairs and approximately 2.8 million docked 3D structures; MBP employs a multi-task pretraining approach, treating different affinity measurement types (e.g., IC50, Ki, Kd) as separate tasks and incorporating pairwise ranking within the same bioassay to mitigate label noise, thereby learning robust and transferable structural knowledge for binding affinity prediction.

B GROUP SYMMETRY AND EQUIVARIANCE

In this article, a 3D molecular graph is represented by a 3D point cloud. The corresponding symmetry group is SE(3), which consists of translations and rotations. Recall that we define the notion of equivariance functions in $\mathbf{R}^3$ in the main text through group actions. Formally, the group SE(3) is said to act on $\mathbf{R}^3$ if there is a mapping ϕ : SE(3) $\times \mathbf{R}^3 \to \mathbf{R}^3$ satisfying the following two conditions:

1. if $e \in$ SE(3) is the identity element, then
$$\phi(e, \boldsymbol{r}) = \boldsymbol{r} \quad \text{for } \forall \boldsymbol{r} \in \mathbf{R}^3.$$
2. if $g_1, g_2 \in$ SE(3), then
$$\phi(g_1, \phi(g_2, \boldsymbol{r})) = \phi(g_1 g_2, \boldsymbol{r}) \quad \text{for } \forall \boldsymbol{r} \in \mathbf{R}^3.$$

Then, there is a natural SE(3) action on vectors $\boldsymbol{r}$ in $\mathbf{R}^3$ by translating $\boldsymbol{r}$ and rotating $\boldsymbol{r}$ for multiple times. For $g \in$ SE(3) and $\boldsymbol{r} \in \mathbf{R}^3$, we denote this action by $g\boldsymbol{r}$. Once the notion of group action is defined, we say a function $f : \mathbf{R}^3 \to \mathbf{R}^3$ that transforms $\boldsymbol{r} \in \mathbf{R}^3$ is equivariant if:

$$f(g\boldsymbol{r}) = gf(\boldsymbol{r}), \quad \text{for } \forall \ \boldsymbol{r} \in \mathbf{R}^3.$$

On the other hand, $f : \mathbf{R}^3 \to \mathbf{R}^1$ is invariant, if f is independent of the group actions:

$$f(g\boldsymbol{r}) = f(\boldsymbol{r}), \quad \text{for } \forall \ \boldsymbol{r} \in \mathbf{R}^3.$$

C Protein Backbone Parameterization

Following AlphaFold2 (Jumper et al., 2021), we construct local coordinate frames from atomic coordinates in protein structures. Each residue frame is defined as a rigid transformation $(\vec{t}, r)$, where $\vec{t} \in \mathbb{R}^3$ is the translation vector and $r \in SO(3)$ is the rotation matrix. Frames are derived from the backbone atoms N, C$_\alpha$, and C, with coordinates x_1, x_2, and x_3, respectively. The translation vector $\vec{t}$ is set to the C$_\alpha$ position x_2, while the rotation matrix r is computed using Gram-Schmidt orthogonalization in Algorithm 1. As a result, each residue i in a protein $g = \{(\vec{t}_i, r_i)\}_{i=1}^{L}$ is thus associated with a orthonormal frame centered at its backbone atom.

Algorithm 1 Frame construction via Gram-Schmidt orthogonalization

```
Require: Three atomic positions $x_1, x_2, x_3 \in \mathbb{R}^3$
Ensure: Translation vector $\vec{t} \in \mathbb{R}^3$ and rotation matrix $r \in \mathbb{R}^{3\times3}$ and
1: $\vec{v}_1 \leftarrow x_3 - x_2$
2: $\vec{v}_2 \leftarrow x_1 - x_2$
3: $\vec{e}_1 \leftarrow \vec{v}_1 / \|\vec{v}_1\|$
4: $\vec{u}_2 \leftarrow \vec{v}_2 - \left(\vec{e}_1^{\top} \vec{v}_2\right) \vec{e}_1$
5: $\vec{e}_2 \leftarrow \vec{u}_2 / \|\vec{u}_2\|$
6: $\vec{e}_3 \leftarrow \vec{e}_1 \times \vec{e}_2$
7: $r \leftarrow [\vec{e}_1, \vec{e}_2, \vec{e}_3]$
8: $\vec{t} \leftarrow x_2$
9: return $(\vec{t}, r)$
```

D Rotation Perturbation in RigidSSL-Perturb

Recall that a structure g is modeled as a sequence of L rigid residues, with each residue represented by a translation and rotation transformation: $g = \{(\vec{t}_i, r_i)\}_{i=1}^{L}$. In RigidSSL-Perturb, we independently perturb each residue's translation $\vec{t}_i$ and rotation r_i. This section details the mathematical foundation, implementation, and implications of rotation perturbation.

D.1 Parametrizations of SO(3)

The rotation group SO(3) admits several equivalent representations. We review its group structure, Lie algebra, axis–angle parameterization, and the exponential and logarithm maps, which together provide the foundations for defining and sampling rotational perturbations in RigidSSL-Perturb.

D.1.1 Rotation Group and Manifold Structure

The special orthogonal group SO(3) $= R \in \mathbb{R}^{3\times3} \mid R^T R = I, \det(R) = 1$ forms a 3-dimensional compact Lie group manifold representing all possible rotations in 3D space. As a manifold, SO(3) is diffeomorphic to real projective space $\mathbb{RP}^3$, obtained by identifying antipodal points on the 3-sphere S^3.

D.1.2 Lie Algebra and Tangent Space

The Lie algebra $\mathfrak{so}(3)$ is the tangent space to SO(3) at the identity matrix I. It consists of all 3×3 skew-symmetric matrices: $\mathfrak{so}(3) = X \in \mathbb{R}^{3\times3} \mid X^T = -X$ Elements of $\mathfrak{so}(3)$ can be interpreted as

infinitesimal rotations or angular velocities. A remarkable isomorphism exists between $\mathfrak{so}(3)$ and $\mathbb{R}^3$ via the "hat map": $\wedge : \mathbb{R}^3 \to \mathfrak{so}(3), \quad \boldsymbol{\omega}^\wedge = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix}$
The inverse "vee map" is denoted $\vee : \mathfrak{so}(3) \to \mathbb{R}^3$.

D.1.3 Axis-Angle Representation of Rotation

Axis-angle rotation representations exists in two related forms:

1. Axis-angle representation: A *geometric* pair $(\mathbf{n}, \theta)$ where $\mathbf{n} \in S^2$ is a unit vector representing the rotation axis and $\theta \in [0, \pi]$ is the rotation angle. This representation lives in the product space $S^2 \times [0, \pi]$ with identifications: $(\mathbf{n}, 0) \sim (-\mathbf{n}, 0)$ and $(\mathbf{n}, \pi) \sim (-\mathbf{n}, \pi)$.
2. Axis-angle vector (rotation vector): A *algebraic* vector $\boldsymbol{\omega} = \theta\mathbf{n} \in \mathbb{R}^3$ where the direction specifies the rotation axis and the magnitude $|\boldsymbol{\omega}| = \theta$ specifies the rotation angle. This representation belongs to a ball of radius π in $\mathbb{R}^3$ with antipodal points on the boundary identified. The axis-angle vector is precisely the exponential coordinates in $\mathfrak{so}(3)$ under the isomorphism provided by the hat map. That is, for a rotation vector $\boldsymbol{\omega}$, the corresponding element in $\mathfrak{so}(3)$ is $\boldsymbol{\omega}^\wedge$.

D.1.4 Exponential Map

The exponential map $\exp : \mathfrak{so}(3) \to \mathrm{SO}(3)$ connects the Lie algebra to the Lie group: $\exp(X) = \sum_{k=0}^{\infty} \frac{X^k}{k!}$ For an axis-angle vector $\boldsymbol{\omega} = \theta\mathbf{n}$, Rodrigues' formula provides a closed-form expression for the corresponding rotation matrix: $\exp(\boldsymbol{\omega}^\wedge) = I + \frac{\sin\theta}{\theta}\boldsymbol{\omega}^\wedge + \frac{1-\cos\theta}{\theta^2}(\boldsymbol{\omega}^\wedge)^2$ When $\theta = 0$, this reduces to the identity matrix I. The exponential map is surjective but not injective globally and wraps around when $|\boldsymbol{\omega}| > \pi$.

D.1.5 Logarithm Map

The logarithm map $\log : \mathrm{SO}(3) \to \mathfrak{so}(3)$ is the (local) inverse of the exponential map. For a rotation matrix $R \in \mathrm{SO}(3)$, where $\mathrm{trace}(R) \neq -1$: $\log(R) = \frac{\theta}{2\sin\theta}(R - R^T)$ where $\theta = \cos^{-1}\left(\frac{\mathrm{trace}(R)-1}{2}\right)$. The logarithm map yields an axis-angle vector representation when composed with the vee map: $\boldsymbol{\omega} = (\log(R))^\vee$.

D.2 Isotropic Gaussian Distribution on SO(3) : $IG_{\mathrm{SO}(3)}$

With axis-angle representation of rotations, $IG_{\mathrm{SO}(3)}$ consists of two components:

- *Axis Distribution.* The axis distribution is uniform over the unit sphere S^2, with PDF:

$$p(\hat{\mathbf{n}}) = \frac{1}{4\pi} \tag{13}$$

  where $\hat{\mathbf{n}} \in S^2$ is a unit vector representing the axis of rotation.

- *Angle Distribution.* The angle distribution for $\omega \in [0, \pi]$ has the probability density function:

$$p(\omega) = \frac{1-\cos\omega}{\pi} \sum_{l=0}^{\infty} (2l+1) e^{-l(l+1)\varepsilon^2} \frac{\sin((l+\frac{1}{2})\omega)}{\sin(\frac{\omega}{2})} \tag{14}$$

  where ε is the concentration parameter of $\mathcal{IG}_{SO(3)}$. For small values of ε (where numerical convergence of the infinite sum becomes difficult), an approximation can be used:

$$p(\omega) = \frac{(1-\cos\omega)}{\pi}\sqrt{\pi\varepsilon^{-\frac{3}{2}}}e^{\frac{\varepsilon}{4}}e^{-\frac{(\frac{\omega}{2})^2}{\varepsilon}} \cdot \frac{\left[\omega - e^{-\frac{\pi^2}{\varepsilon}}\left((\omega-2\pi)e^{\frac{\pi\omega}{\varepsilon}} + (\omega+2\pi)e^{-\frac{\pi\omega}{\varepsilon}}\right)\right]}{2\sin\left(\frac{\omega}{2}\right)} \tag{15}$$

As $\varepsilon \to \infty$, $f(\omega)$ approaches the uniform distribution on SO(3):

$$f_{\text{uniform}}(\omega) = \frac{1-\cos\omega}{\pi} \tag{16}$$

D.3 Rotation Sampling from $IG_{\text{SO}(3)}$

We sample a rotation from $IG_{\text{SO}(3)}$ as follows:

1. *Decompose the problem.* A sample from $\mathcal{IG}_{\text{SO}(3)}(\mu, \varepsilon^2)$ can be decomposed as $r = \mu r_c$ where $r_c \sim \mathcal{IG}_{\text{SO}(3)}(I, \varepsilon^2)$. Therefore, we first sample from an identity-centered distribution and then rotate by μ.
2. *Sample the rotation axis.* Generate a random unit vector $\hat{\mathbf{n}} \in S^2$ uniformly by sampling a point (x, y, z) from the standard normal distribution $\mathcal{N}(0, I_3)$ and normalizing to unit length.
3. *Sample the rotation angle.* Sample $\omega \in [0, \pi]$ from the distribution $f(\omega)$ described above using inverse transform sampling. Specifically, numerically compute the CDF $F(\omega) = \int_0^\omega f(t)dt$, sample $u \sim \text{Uniform}[0,1]$, and compute $\omega = F^{-1}(u)$ through numerical inversion or interpolation.
4. *Construct the exponential coordinates.* Form the axis-angle vector $\boldsymbol{\omega} = \omega\mathbf{n} \in \mathbb{R}^3$. This vector represents elements in the Lie algebra $\mathfrak{so}(3)$ via the hat map: $\boldsymbol{\omega}^\wedge \in \mathfrak{so}(3)$, as detailed in Sec.D.1.
5. *Apply the exponential map.* Convert $\boldsymbol{\omega}$ to a rotation matrix using Rodrigues' formula:

$$r_c = \exp(\boldsymbol{\omega}^\wedge) = I + \frac{\sin\omega}{\omega}\boldsymbol{\omega}^\wedge + \frac{1-\cos\omega}{\omega^2}(\boldsymbol{\omega}^\wedge)^2 \tag{17}$$

   where $\boldsymbol{\omega}^\wedge$ is the skew-symmetric matrix:

$$\boldsymbol{\omega}^\wedge = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix}, \quad \boldsymbol{\omega} = (\omega_1, \omega_2, \omega_3) = \omega\mathbf{n} \tag{18}$$

6. *Apply the mean rotation.* Compute the final rotation as:

$$r = \mu r_c \tag{19}$$

D.4 Rotation Perturbation Effects in $\mathbb{R}^3$

When applying rotational perturbations sampled from $\mathcal{IG}_{SO(3)}$, the geometric effect in $\mathbb{R}^3$ varies with initial rotation despite consistent geodesic distances on SO(3). For rotations $R_a^0, R_b^0 \in$ SO(3) and point $p \in \mathbb{R}^3$, with perturbations of equal magnitude:

$$R_a^1 = R_a^0 \cdot \exp(\boldsymbol{\omega}_a), ; R_b^1 = R_b^0 \cdot \exp(\boldsymbol{\omega}_b) \tag{20}$$

where $|\boldsymbol{\omega}a| = |\boldsymbol{\omega}b| = \theta$ and $\boldsymbol{\omega}a, b \sim \mathcal{IG}SO(3)(\mathbf{I}, \epsilon^2)$, the resulting displacements:

$$\Delta_a = R_a^1 p - R_a^0 p = R_a^0(\exp(\boldsymbol{\omega}_a) - \mathbf{I})p \tag{21}$$

$$\Delta_b = R_b^1 p - R_b^0 p = R_b^0(\exp(\boldsymbol{\omega}_b) - \mathbf{I})p \tag{22}$$

generally have $|\Delta_a| \neq |\Delta_b|$, as displacement magnitude depends on the initial rotation and point position.

E BASE PROTEIN ENCODER

We adopt the IPA module from AlphaFold2 (Jumper et al., 2021) as the base protein encoder. Specifically, the edge embedding is initialized using a distogram, which discretizes the continuous pairwise distances into a one-hot vector representation. For any two residues i and j, the Euclidean distance $d_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|_2$ is categorized into one of N_{bins} discrete bins. The k-th component of the resulting embedding vector is determined by the indicator function $\mathbb{I}(\cdot)$:

$$\text{EdgeEmbed}(d_{ij})_k = \mathbb{I}(l_k < d_{ij} < u_k), \quad \text{for } k = 1, \dots, N_{\text{bins}}, \tag{23}$$

where (l_k, u_k) are the lower and upper bounds of the k-th distance bin. In our model, we set $N_{\text{bins}} = 22$, with bins linearly spaced between a minimum of 1×10^{-5} Å and a maximum of 20.0 Å. The final bin extends to infinity to capture all larger distances.

Next, like normal attention mechanisms, we generate the scalar and point-based query, key, and value features for each node using bias-free linear transformations:

$$\mathbf{q}_i^h, \mathbf{k}_i^h, \mathbf{v}_i^h = \mathbf{W}_1 \cdot \mathbf{s}_i \tag{24}$$

$$\mathbf{q}_i^{hp}, \mathbf{k}_i^{hp}, \mathbf{v}_i^{hp} = \mathbf{W}_2 \cdot \mathbf{s}_i \tag{25}$$

$$\mathbf{b}_{ij}^h = \mathbf{W_3} \cdot \mathbf{z}_{ij} \tag{26}$$

where $\mathbf{W}_1 \in \mathbb{R}^{(h \cdot d) \times d_s}$, $\mathbf{W}_2 \in \mathbb{R}^{(h \cdot p \cdot 3) \times d_s}$, $\mathbf{W}_3 \in \mathbb{R}^{h \times d_z}$, and $\mathbf{s}_i \in \mathbb{R}^{d_{\text{in}}}$ is the input node embedding. Here, h denotes the number of attention heads, and p is the number of reference points used per head in the 3D attention mechanism. Each point-based feature (e.g., $\mathbf{q}_i^{hp}$) is reshaped into $[h, p, 3]$, representing a set of 3D vectors per head that enable spatial reasoning in the invariant point attention module.

A key component then, is calculating the attention score:

$$a_{ij}^h = \text{softmax}_j \left(w_L \left(\frac{\mathbf{q}_i^\top \mathbf{k}_j^h}{\sqrt{d}} + b_{ij}^h - \frac{\gamma^h w_C}{2} \sum_p \left\| T_i \circ \mathbf{q}_i^{hp} - T_j \circ \mathbf{k}_j^{hp} \right\|^2 \right) \right), \tag{27}$$

where $T \circ y = (r \cdot y + x)$ representing the transformation.

As shown in 27, IPA introduces an additional bias term besides the bias b_{ij} from the pairwise feature, $\sum_p \left\| T_i \circ \mathbf{q}_i^{hp} - T_j \circ \mathbf{k}_j^{hp} \right\|^2$, which explicitly models the spatial relationship between nodes after applying the local transformations T_i and T_j. When the transformed point $T_i \circ \mathbf{q}_i^{hp}$ and $T_j \circ \mathbf{k}_j^{hp}$ differ significantly, it indicates a large spatial discrepancy between node i and node j, and the resulting attention score is correspondingly reduced.

We then calculate the attention score and update the node representation:

$$\mathbf{o}_i^h = \sum_j a_{ij}^h \, \mathbf{z}_{ij} \tag{9}$$

$$\mathbf{o}_i^h = \sum_j a_{ij}^h \, \mathbf{v}_j^h \tag{10}$$

$$\tilde{\mathbf{o}}_i^{hp} = T_i^{-1} \circ \left(\sum_j a_{ij}^h \left(T_j \circ \tilde{\mathbf{v}}_j^{hp} \right) \right) \tag{11}$$

$$\tilde{\mathbf{s}}_i = \text{Linear} \left(\text{concat}_{h,p} \left(\tilde{\mathbf{o}}_i^h, \mathbf{o}_i^h, \tilde{\mathbf{o}}_i^{hp}, \|\tilde{\mathbf{o}}_i^{hp}\| \right) \right) \tag{12}$$

In $\tilde{\mathbf{o}}_i^{hp} = T_i^{-1} \circ \left(\sum_j a_{ij}^h \left(T_j \circ \tilde{\mathbf{v}}_j^{hp} \right) \right)$, the model first applies the global transformation T_j to each neighboring point vector $\tilde{\mathbf{v}}_j^{hp}$, weighs them by the attention scores a_{ij}^h, and then aggregates the results in the global frame. The result is then transformed back to the local coordinate frame of node i using T_i^{-1}. This ensures that the update is the same in the local frame though a global transform is applied.

To make it compatible with our flow matching setting, after each IPA block, we embed the current time step t using sinusoidal positional embedding (Ho et al., 2020) denoted as F_t and add that to $\tilde{\mathbf{s}}_i$

to complete the full update:

$$\tilde{\mathbf{s}}_i = \tilde{\mathbf{s}}_i + F_t(t) \tag{28}$$

Finally, we map the updated node representation back to the translation and quarternion:

$$x_i, q_i = Linear(\tilde{\mathbf{s}}_i) \tag{29}$$

, where x_i and q_i represents the translation and rotation (quaternion) respectively.

F Experiment Details

F.1 Pretraining: Dataset

RigidSSL-Perturb. For the first phase of our pretraining strategy, we use version 4 of AFDB. Specifically, we utilize AFDB's coverage for the UniProtKB/Swiss-Prot section of the UniProt KnowledgeBase (UniProtKB), which comprises 542,378 proteins, demonstrating the large-scale nature of RigidSSL-Perturb. We filter sequences by length, retaining only those between 60 and 512 residues, yielding 432,194 proteins for training.

RigidSSL-MD. For the second pretraining phase, we utilize the ATLAS dataset, obtained from the MDRepo (Roy et al., 2025; Vander Meersche et al., 2024). This dataset contains 100 ns all-atom MD simulation triplicates for 1,390 protein chains, which were selected to provide an exhaustive and non-redundant sampling of the PDB's conformational space based on the Evolutionary Classification of Domains (ECOD) (Cheng et al., 2014). All simulations were performed with GROMACS using the CHARMM36m force field; each protein was solvated in a triclinic box with TIP3P water and neutralized with Na^+/Cl^- ions at a 150 mM concentration. We filter sequences by length, retaining only those between 60 and 512 residues. To generate training samples for RigidSSL-MD, we extracted 80 pairs of adjacent conformations separated by a 2 ns interval from each trajectory to reflect the conformational fluctuations of the protein structures.

F.2 Pretraining: Optimization

For training RigidSSL-Perturb and RigidSSL-MD, we use Adam optimizer (Kingma and Ba, 2017) with learning rate of 0.0001. RigidSSL-Perturb was trained for 2.75 days on one NVIDIA H100 GPU with batch size of 1. RigidSSL-MD was trained for 1.88 days on one NVIDIA H100 GPU with batch size of 1.

F.2.1 Downstream: Training

We trained FrameDiff follwoing the exact same dataset processing, hyperparameters for the network and for optimization as reported in the FrameDiff paper (Yim et al., 2023). The network is trained for 1 week on 4 NVIDIA H100 GPUs. To evaluate FrameDiff, we perform inference by sampling sequences from 100 to 300 residues long with a step size of 50.

We trained FoldFlow-2 follwoing the exact same dataset processing, hyperparameters for the network and for optimization as reported in the FoldFlow-2 paper (Huguet et al., 2024b). The network is trained for 4 days on 2 NVIDIA A100 80G GPUs. To evaluate FoldFLow-2, as it is a more capable model than FrameDiff, we perform inference by sampling sequences from 100 to 600 resiudes ong with a step size of 50.

F.2.2 Downstream: Inference

For the results reported in Table 2, we generate protein structures of lengths 100–300 residues using FrameDiff and 100–600 residues using FoldFlow-2 (in increments of 50), following the inference hyperparameters specified in the original works (Huguet et al., 2024b; Yim et al., 2023).

For the long-chain case study in Section 4.2, we apply the same inference procedure but extend the sequence lengths to 700 and 800 residues. The structure exhibiting the highest self-consistency is then selected for stereochemical evaluation using MolProbity (Davis et al., 2007).

Table 4: Comparison of Designability (fraction of proteins with scRMSD < 2.0Å), Novelty (max. TM-score to PDB), and Diversity (avg. pairwise TMscore) of structures generated by FoldFlow-2 pretrained with RigidSSL-Perturb under different noise scales. All metrics include standard errors.

Translation Noise	Rotation Noise	Length	Designability Fraction (↑)	Novelty avg. max TM (↓)	Diversity pairwise TM (↓)
0.01	0.5	100-600	0.336±0.012	0.768±0.010	0.635±0.002
0.03	0.5	100-600	0.758±0.016	0.770±0.003	0.650±0.001
0.05	0.5	100-600	0.589±0.014	0.769±0.005	0.654±0.002
0.5	0.5	100-600	0.382±0.015	0.748±0.005	0.649±0.001
0.5	0.75	100-600	0.660±0.014	0.763±0.004	0.644±0.001
0.5	1.0	100-600	0.352±0.025	0.772±0.008	0.614±0.003
1.0	0.75	100-600	0.460±0.016	0.773±0.005	0.663±0.001
2.0	0.75	100-600	0.347±0.014	0.797±0.006	0.624±0.002

G Ablation Studies

We present ablation studies of RigidSSL to assess how design choices in the view construction process affect its generalizability when applied to FoldFlow-2. Specifically, we investigate the impact of translational and rotational noise scales in RigidSSL-Perturb (Section 3.2.1). As shown in Figure 4, increasing either noise scale leads to more steric clashes and reduced bond validity. The resulting FoldFlow-2 models trained with these noise settings are reported in Table 4. Unless otherwise stated, all other experiments use RigidSSL-Perturb pretrained with a translation noise scale $\sigma = 0.03$ and a rotation noise scale $\mu = 0.5$.

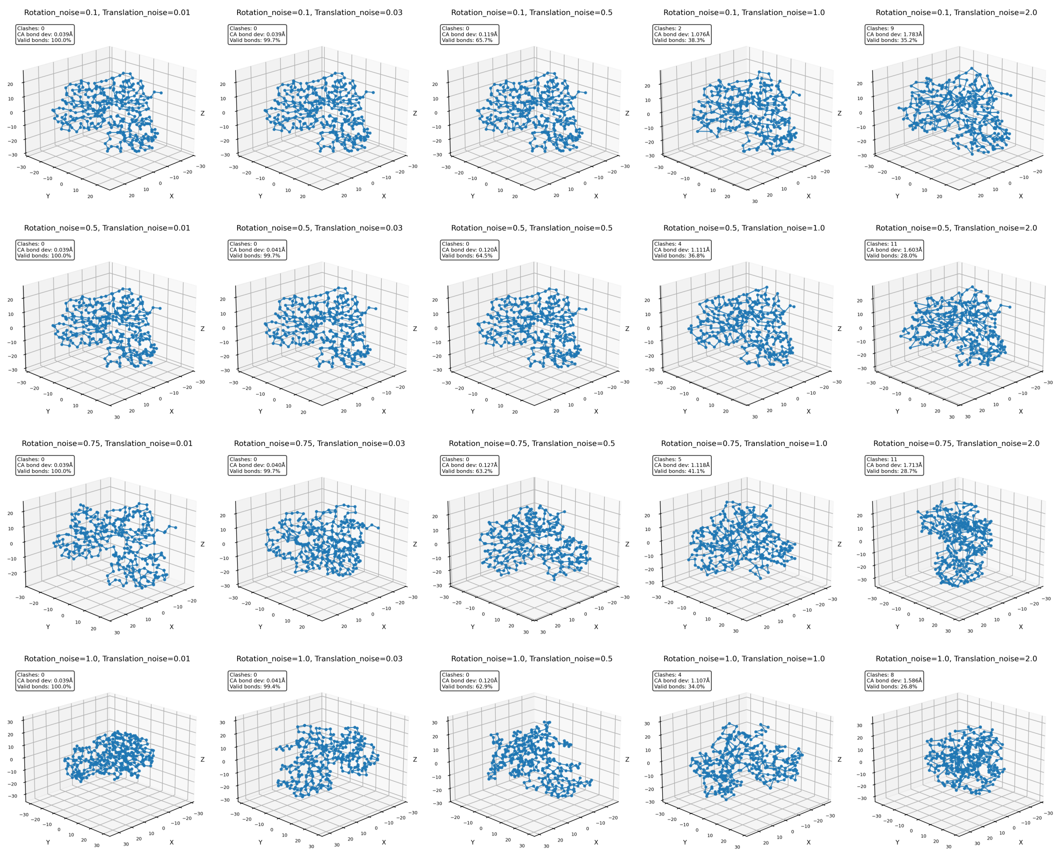

Figure 4: Impact of translation and rotation noise scale on protein structure validity