Accelerating 3D Molecule Generative Models with Trajectory Diagnosis

Zhilong Zhang[1,2*] **Yuxuan Song**[1,3*] **Yichun Wang**[4*] **JingJing Gong**[1] **Hanlin Wu**[1]
Dongzhan Zhou[5] **Hao Zhou**[1,5] & **Wei-Ying Ma**[1]
[1] Institute of AI Industry Research (AIR), Tsinghua University
[2] Qiuzhen College, Tsinghua University
[3] Department of Department of Computer Science and Technology, Tsinghua University
[4] Department of Industrial Engineering, Tsinghua University
[5] Shanghai Artificial Intelligence Laboratory

Abstract

Geometric molecule generative models have found expanding applications across various scientific domains, but their generation inefficiency has become a critical bottleneck. Through a systematic investigation of the generative trajectory, we discover a unique challenge for molecule geometric graph generation: generative models require determining the permutation order of atoms in the molecule before refining its atomic feature values. Based on this insight, we decompose the generation process into *permutation phase* and *adjustment phase*, and propose a geometric-informed prior and consistency parameter objective to accelerate each phase. Extensive experiments demonstrate that our approach achieves competitive performance with approximately 10 sampling steps, 7.5 $\times$ faster than previous state-of-the-art models and approximately 100 $\times$ faster than diffusion-based models, offering a significant step towards scalable molecular generation. Code is available at https://github.com/GenSI-THUAIR/MolTD

1 Introduction

Geometric generative models have achieved notable progress in various important scientific tasks, including protein folding [1], *de novo* drug design [2, 3], and crystal generation [4]. Previous progress are largely driven by the adoption of advanced generative models, particularly diffusion models, which could refine a cloud of atoms into precise molecular structures through iterative sampling. A notable example is AlphaFold3, which employs a diffusion-based network to predict the joint structures of molecular complexes [1].

However, the iterative sampling process of these advanced geometric generative models can lead to inefficiencies in practice. For instance, the AlphaFold3-level structure prediction system requires around 200 diffusion steps, which significantly contributes to the heavy inference pipeline and usually 5 sampling procedures are conducted with different seeds for a single structure prediction task to obtain comparable performance [5]. As application scenarios expand, generation inefficiency has become a critical bottleneck, especially given the growing demand for scalable inference to produce large-scale synthetic data [6] and to integrate generative models into real-time scientific workflows in drug discovery [7] and material design [4]. Therefore, enhancing the efficiency of geometric generative models has emerged as a key research direction.

Recent efforts have adapted techniques from general domains such as image generation to improve sampling efficiency in the molecular generation setting. However, even the most efficient methods

*Equal Contribution. Correspondence to Hao Zhou(zhouhao@air.tsinghua.edu).

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

still require approximately a hundred or more steps—for example, 90 steps in GOAT [8] and 200 steps in EquiFM [9]. In contrast, state-of-the-art image generation techniques only take as few as 1–2 steps to achieve desirable results [10, 11]. This discrepancy raises two key questions: 1) What are the fundamental bottlenecks limiting the efficiency of 3D molecular generative models? 2) Can we achieve a substantial improvement in efficiency, potentially reducing the required sampling steps by an order of magnitude?

The challenge of efficient molecule generation is fundamentally distinct from that of image generation due to the structural nature of the data. While images have a fixed spatial ordering (e.g., pixel positions remain constant), 3D molecular generative models that generate the structure holistically requires **determining the permutation order of atoms before refining their atomic feature values.** This permutation step is unique to 3D geometric generation and have not been adequately addressed by directly applying acceleration techniques designed for image domains.

In this paper, we first conduct a systematic and theoretical investigation of the generative trajectory to analyze this permutation component. Building on the insights from this analysis, we propose decomposing the geometric generation into two phases: **the reordering of permutations** and **the adjustment of atomic features**, and propose MOLTD (Molecule Trajectory Diagnosis) with novel acceleration methods for both phases. For the permutation phase, we introduce a geometric-informed prior to the sampling process with a corresponding accuracy scheduler which could efficiently reduce the procedure of reordering; For the adjustment phase, we demonstrate that previous acceleration approaches from general domains, such as consistency training [10], can be significantly beneficial for efficient generation when specifically adapted for geometric generation in the adjustment phase. To sum up, our paper makes the following contributions:

- To analyze the intrinsic properties and underlying challenges in efficient geometric generation, we propose a quantitative framework for analyzing the generative trajectory. This approach allows us to identify key considerations for developing improved methods while highlighting the fundamental differences between 3D molecular generation and general domains.
- We further introduce effective modifications targeting different decomposed phases of the generation process. For the permutation phase, a geometric-informed prior is adopted to accelerate the generation of stable structures. For the adjustment phase, we propose consistency parameter objective to improve the accuracy of atomic feature adjustment.
- We demonstrate the effectiveness of the proposed methods on two molecule datasets: QM9 [12] and GEOM-DRUG [13]. For the first time, our approach enables the generation of large molecules using approximately 10 steps while reaching new state-of-the-art performance on both datasets: on QM9, MOLTD achieves molecule stability of 93.16%, and on GEOM-DRUG it achieves Atom stability of 86.88% . This represents a significant improvement in generation efficiency: 7.5 $\times$ faster than flow matching models, 8.3 $\times$ faster than Bayesian Flow Network models, and nearly 100 $\times$faster than diffusion-based models.
- We demonstrate the broad applicability of MOLTD by generalizing to different tasks and generative backbones. In the task of structure-based drug design, MOLTD achieves state-of-the-art performance using only 25 NFEs — a 40$\times$ speedup over diffusion-based models. Ablation studies further confirm that our approach significantly accelerates both diffusion models and Bayesian Flow Networks.

2 Related Work

2.1 Efficient Generation in General Domains

Advancements have been made to enhance the sampling efficiency of generative models. For diffusion models, DDIM [14] introduced an adaptable diffusion process for faster inference, while DiffFlow [15] optimized the diffusion trajectory. Another line of works focuses on improving ODE solvers to reduce discretization error, for instance, DPM-Solver [16] introduced a high-order, training-free solver that reduces sampling steps to just 12. For flow matching models, recent efforts aim to improve the coupling between prior and target distributions. Concurrent works [17, 18] approximate optimal transport coupling to create a more straight trajectory, while Rectified Flow [19] employs iterative

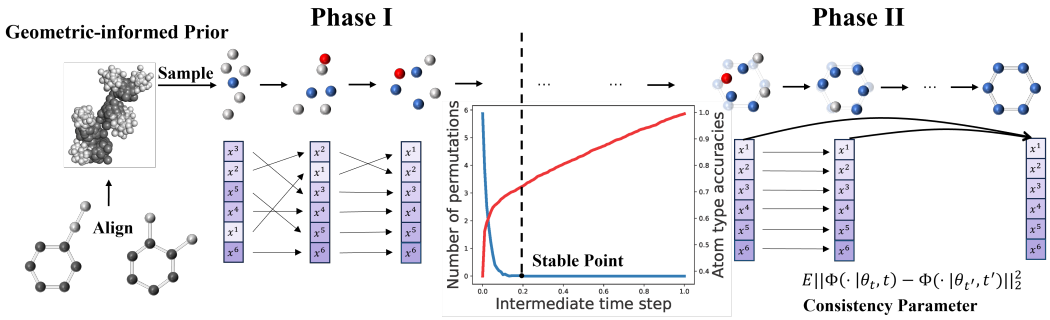

Figure 1: Illustration of MOLTD. To accelerate the generation in the first phase, we introduce a geometric-informed prior to the sampling process based on the aligned molecules in the dataset. For the second phase, we propose a consistency parameter objective to accelerate the adjustment of atomic features.

distillation to straighten it. Most recently, Consistency Models [10, 20] propose consistency training, which enables one-step generation without sacrificing quality. While these advancements primarily focus on general domains like image generation, they have inspired improvements in geometric generative models.

2.2 Advancements in 3D Molecule Generative Models and Sampling Techniques

Extensive prior work has focused on generating molecules as 2D graphs [21, 22, 23], but there is growing interest in 3D molecule generation. Autoregressive methods like G-Schnet and GSphereNet [24, 25] iteratively connecting fragments to build molecules, but they require complex action design. Another approach models molecules as atomic density grids, generating densities over voxelized 3D space [26].

Recent advances leverage diffusion models and flow matching for 3D molecular generation [27, 9], antibody design [28], and protein design [29]. However, these methods typically require hundreds of sampling steps. To accelerate the generation, EquiFM [9] introduced optimal transport objectives with adaptive ODE solvers, GOAT [8] applied optimal transport in a joint latent space, and GeoLDM [30] leveraged a latent space to reduce dimensionality. However, even the most efficient methods still require 90–200 steps, whereas state-of-the-art image generation achieves high quality in just 1–2 steps [10, 11].

3 Background

3.1 Notations and Definition

To differentiate between geometric representations and atomic property features, we represent 3D molecules using the tuple $\boldsymbol{g} = \langle \boldsymbol{x}, \boldsymbol{h} \rangle$. Here, $\boldsymbol{x} = (\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N) \in \mathcal{X}$ denotes the atomic coordinate matrix, and $\boldsymbol{h} = (\boldsymbol{h}^1, \ldots, \boldsymbol{h}^N) \in \mathbb{R}^{N \times d}$ represents the node feature matrix, which includes attributes such as atomic types and charges. Here $\mathcal{X} = \{\boldsymbol{x} \in \mathbb{R}^{N \times 3} : \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{x}^i = 0\}$ is the Zero Center-of-Mass (Zero CoM) space, which means the average of the N elements should be 0. For Bayesian Flow Networks, we use $\boldsymbol{\theta}_t$ to denote the parameter encoding the information of the molecule at time t. We use q to denote the data distribution and p_ϕ as learned distribution.

To analyze the generative trajectory, we introduce additional notations: $\pi \in \mathbb{R}^{N \times N}$ is the permutation of N elements, and $\mathbf{R} \in \mathbb{R}^{N \times 3}$ is a rotation matrix in 3D space, $V(\boldsymbol{x}; r)$ is an open ball centered at $\boldsymbol{x}$ with radius r.

3.2 General Formulation of Generation Process

Most advanced geometric generative models employ iterative sampling processes to progressively transform noisy latent variables into valid molecule. This approach is exemplified by diffusion models (DMs), flow-matching models (FMs), and Bayesian Flow Networks (BFNs) [27, 6, 31], and their sampling process can be unified as follows:

$$\boldsymbol{g}_t = \alpha_t \boldsymbol{g}_{t-1} + \beta_t \Phi(\boldsymbol{g}_{t-1}, t-1) + \gamma_t \boldsymbol{\epsilon}_t, t \in \{1, 2, \ldots, T\} \tag{1}$$

where $\boldsymbol{g}_t$ is the noisy latent at time step t, T denotes the total number of sampling steps, $\boldsymbol{\epsilon}_t$ is a standard Gaussain random vector, Φ represents the neural network, and $\alpha_t, \beta_t, \gamma_t$ are parameters to be instantiated according to specific generative model and sampling technique.

For diffusion-based models and flow-matching-based models [32, 33, 34], the iterative generation process aims to discretizing and solve the differential equation:

$$\frac{d\boldsymbol{g}_t}{dt} = \boldsymbol{v}_\theta(\boldsymbol{g}_t, t) \tag{2}$$

which transports the prior distribution $p(\boldsymbol{g}_0)$ to the target distribution $p(\boldsymbol{g}_T)$. Typically, the network Φ in Equation (1) is trained to approximate the velocity term $\boldsymbol{v}_\theta$ by estimating a conditioned version of the ground truth velocity. However, since discretization inevitably introduces errors into the sampling process, these models generally require a large number of sampling steps to effectively approximate the learned differential equations and generate high-quality samples [35, 16].

In contrast, Bayesian Flow Networks (BFNs) generate data without requiring discretization by performing Bayesian updates based on observed noisy random variables [36]. This approach provides a unified framework for handling diverse data modalities, including continuous, discretized, and discrete data. Each modality is addressed with a specifically adapted Bayesian update rule, as outlined in Equation (1). The network Φ is trained to reconstruct the ground truth molecules. In previous work [31], effectively approximating the data distribution with a BFNs-based model required hundreds of sampling steps. We include a more detailed introduction of BFNs in Appendix B.

4 Method

In this section, we systematically and theoretically investigate the generative trajectory of geometric generative models and introduce a decomposition framework for 3D molecule generation in Section 4.1. Building on insights gained from this analysis, we introduce the geometric-informed prior in Section 4.2 and the consistency parameter objective in Section 4.3 to accelerate the first and second phase, respectively. Proofs for the propositions presented in this section are provided in the Appendix D.

4.1 Decomposition of the Generative Trajectory

We focus on advanced geometric generative models, including the diffusion-based approach EDM [27], the flow-matching-based approach EquiFM [9], and the Bayesian flow network-based approach GeoBFN [31]. These methods generate a trajectory of noisy molecules $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_T$ through iterative sampling described in Section 3.2. We quantify the change of molecular geometry along this trajectory to investigate the characteristics of the generation process.

To evaluate the changes in molecular geometry, we compare the intermediate molecules $\{\boldsymbol{g}_i\}_{i=1}^{T-1}$ generated along the trajectory, against the final molecule $\boldsymbol{g}_T$ obtained at the end of the trajectory. We define two metrics to evaluate the geometric changes: $\mathcal{D}_{\text{structure}}(\boldsymbol{g}_i)$ measures structural changes at the molecular level, while $\mathcal{D}_{\text{type}}(\boldsymbol{g}_i)$ captures changes in atom types at the atomic level:

Definition 4.1. (Metrics for the change of geometry)

The structural difference between $\boldsymbol{g}_i$ and $\boldsymbol{g}_T$ is defined as

$$\mathcal{D}_{\text{structure}}(\boldsymbol{g}_i) = \frac{\|\pi_i^* - I\|_0}{N}, \tag{3}$$

where

$$\pi_i^*, \mathbf{R}_i^* = \underset{\pi, \mathbf{R}}{\operatorname{argmin}} \|\pi(\mathbf{R}\boldsymbol{x}_i) - \boldsymbol{x}_T\|_2 \tag{4}$$

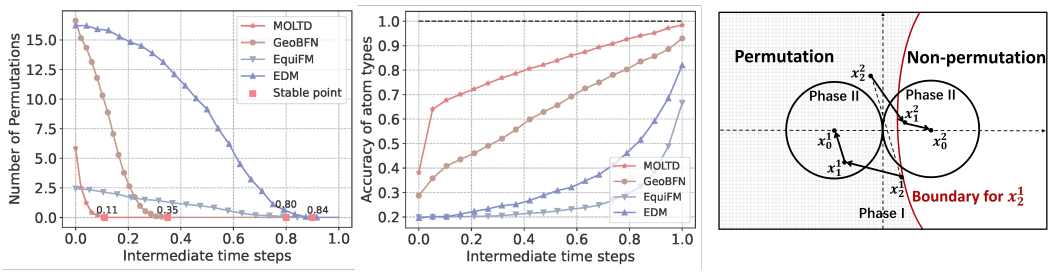

Figure 2: **Left:** The structural difference measured by $\mathcal{D}_{\text{structure}}$. The Stable point t_{stable} represent the dividing point in time between the first generation phase and the second phase. **Middle:** The change of atom types measured by $\mathcal{D}_{\text{type}}$. **Right:** Illustration of the two phases. In the trajectory of $(x_2^1, x_2^2) \to (x_1^1, x_1^2) \to (x_0^1, x_0^2)$, the (x_1^1, x_1^2) lies in the second phase and is aligned with (x_0^1, x_0^2), while (x_2^1, x_2^2) lies in the first phase where permutation reordering is required.

Here $\boldsymbol{x}_i$ is the molecular structure, $\|\pi_i^* - I\|_0$ is the number of permutation (i.e., transpositions) performed by π_i^*.

And the difference of atom types is defined as

$$\mathcal{D}_{\text{type}}(\boldsymbol{g}_i) = \frac{\sum_{j=1}^{N} \delta(\boldsymbol{h}_i^j - \boldsymbol{h}_T^j)}{N}, \tag{5}$$

where δ is the Dirac delta function.

We randomly sample 1000 generative trajectories and report the mean of these metrics. More implementation details about the investigation and additional experiments can be found in Appendix C.

The results of geometric changes along the generated trajectory are presented in the left and middle of Figure 2. We could observe the trajectory of generated molecules exhibits a two-phase pattern across all analyzed generative models, and these two phases are delineated by the *stable point*:

Definition 4.2. The stable point is a time step in the generative trajectory that distinguishes two phases:

$$t_{\text{stable}} = \underset{i}{\text{argmin}} \mathcal{D}_{\text{structure}}(\boldsymbol{g}_i), s.t. \mathcal{D}_{\text{structure}} \leq \epsilon \tag{6}$$

where ϵ is a small positive number, *e.g.*, $1e-3$.

For $t \leq t_{\text{stable}}$, $\boldsymbol{x}_t$ requires permutations to align with the final molecules, a phase we term as **the reordering of permutations**. This process highlights a unique challenge in molecular generation, distinguishing it from other generative tasks such as image generation. In general image generation, intermediate samples inherently follow a fixed permutation order consistent with the final output, i.e., $\pi(\boldsymbol{x}_i) = I$, where I represents the identity map. In contrast, the geometric generative models must first resolve the correct permutation mapping for intermediate molecular structures.

For $t > t_{\text{stable}}$, the molecular structure stabilizes, and the generative models **refine atomic features** including both atom coordinates and atom types to finalize the molecular geometry. In this phase, the molecule can be treated as general data, as $\pi(\boldsymbol{x}_t) = I$, allowing the application of advanced techniques from general domains to accelerate generation.

In fact, the existence of the two phases is theoretically justified as follows:

Theorem 4.3. *Let $x_1^i \in V(x_0^i; r)$, $i \in \{1, 2, \cdots, N\}$. The radius r is chosen as $r = \frac{1}{2}\min_{i \neq j} \|x_0^i - x_0^j\|_2$, where $i, j \in \{1, 2, \cdots, N\}$ and $j \neq i$. Then,*

$$\underset{\pi}{\text{argmin}} \|\pi(x_1) - x_0\|_2 = I_{N \times N},$$

where π is an $N \times N$ permutation matrix, and $I_{N \times N}$ is the identity matrix.

Theorem 4.3 establishes that if the two structures, x_1^1 and x_1^2, lie within the open ball of the corresponding target, as shown on the right side of Figure 2, then no further permutation is required. This explains why permutation for alignment is not needed in the second phase.

For the first phase, because of the significant difference between the prior and the target, permutation is inevitable at the beginning of generation. The following theorem provides the conditions under which permutation occurs. Since all permutations can be decomposed into products of pairwise permutations, in the following theorem, we focus on pairwise permutations between two atoms:

Theorem 4.4. *Assume $x_2^i \notin V(x_0^i; r)$, $i \in \{1, 2, \cdots, N\}$. For $j \in \{1, 2, \cdots, N\} \setminus \{i\}$, a permutation between the i-th and j-th rows is required if and only if x_2^j lies outside one sheet of the two-sheeted hyperboloid defined by the foci x_0^i and x_0^j, passing through x_2^i.*

We now apply the proposed tools to analyze the inefficiencies in advanced geometric generative models. Based on the results in the left and middle of Figure 2, we identify the permutation phase as a key bottleneck in the generation efficiency of current models. Both diffusion-based EDM and flow matching-based EquiFM require hundreds of sampling steps, and a large proportion of steps are spent in finding the stable structure in the first phase. In contrast, GeoBFN benefits from a smoother trajectory and the capability of unified modeling of different modalities, enabling it to reach a stable point earlier than previous methods, and thus significantly reduce the necessary sampling steps. However, GeoBFN still experiences a large number of permutations at the start of the first phase.

Building on this analysis, we gained valuable insights into accelerating each geometric generative model. In this study, we focus on optimizing GeoBFN, as it effectively disentangles the two phases of the generation process and exhibits the highest generation efficiency among the models analyzed. Although these methods are tailored for BFNs, we demonstrate that they can be readily generalized to other generative models, such as diffusion models, paving the way for efficient acceleration in large-scale molecular generation systems.

To address the challenges of each phase, we propose two techniques specifically designed for the first and second phases, respectively. These techniques are discussed in detail in the following sections. With the proposed methods, our approach achieves an earlier stable point, more accurate atom-type predictions, and largely reduces the redundant steps, as demonstrated in Figure 2.

4.2 Accelerating the Permutation-Reordering Phase with Geometric-Informed Prior

In this section, we focus on accelerating the first phase of generation by leveraging the intrinsic structural information present in the molecular dataset. In drug discovery, it is well recognized that molecules can be decomposed into stable and conserved substructures, such as scaffold-arm decomposition [37] and fragment-based decomposition [38]. Inspired by this fact, we propose utilizing the "representative structures" within the molecule dataset that capture the general characteristics of most molecular structures, as an effective starting point for efficient generation of stable structure. To this end, we extract representative structural information from the dataset, summarize it into a geometric-informed prior, and seamlessly incorporate it into the sampling process.

The extraction of representative structural information consists of two main steps:

- In order to extract fine-grained geometric information, we stratify the molecules in the dataset based on their number of nodes N, since molecules with different number of nodes show distinct structural patterns [27, 39]. We provide visualization in the Appendix G to support this claim.
- We perform Equivariant Optimal Transport (EOT) [9] to align the geometric structures within the stratified molecules. By eliminating variations due to rotation and translation while preserving the relative spatial configuration of atoms, we could extract representative structural features by simply taking the mean of all atomic features over the aligned molecules. The resulting representative structure is denoted as $\bar{\boldsymbol{g}}$.

A detailed description of the above process can be found in Algorithm 1.

The framework of BFN provides an efficient approach to utilize $\bar{\boldsymbol{g}}$ in the sampling process, without the need to train the model from scratch. First, as $\bar{\boldsymbol{g}}$ still lies in the sample space, we utilize the Bayesian update distribution to map it to the parameter space:

$$\bar{\boldsymbol{\theta}}_p = \mathop{\mathbb{E}}_{p_S(\boldsymbol{y}_p|\bar{\boldsymbol{g}},\alpha_p)} \delta(\bar{\boldsymbol{\theta}}_p - h(\boldsymbol{y}_p, \boldsymbol{\theta}_0, \alpha_p)) \tag{7}$$

where $\bar{\boldsymbol{\theta}}_p$ is the geometric-informed prior, p_S and h are sender distribution and Bayesian update function, basic components of BFNs that are defined in Appendix B. α_p is the accuracy schedule,

which determine the signal-to-noise ratio in the prior. We provide an ablation study on the choose of α_p in Appendix F. Finally, we perform standard Bayesian update in BFNs with $\bar{\boldsymbol{\theta}}_p$ as the prior. Notably, the sampling process of MOLTD starts with $\bar{\boldsymbol{\theta}}_p$, a random vector embedding intrinsic structural information, in contrast to the fixed, uninformative prior used in the original BFNs.

Based on our construction, the prior already specifies an orientation in 3D, and thus p_ϕ that starts from the prior $\boldsymbol{\theta}_p$ enjoys the rotational-equivariant property:

Proposition 4.5. *The density induced by the geometric-informed prior is rotational-equivariant:*

$$p_\phi(\boldsymbol{g}|\boldsymbol{\theta}_p) = p_\phi(\boldsymbol{R}\boldsymbol{g} \mid \boldsymbol{R}\boldsymbol{\theta}_p), \tag{8}$$

where $\boldsymbol{R}$ is any orthogonal matrix.

4.3 Accelerating the Adjustment Phase with Consistency Parameter Objective

In the adjustment phase, the permutation orders of the generated molecules are fixed, allowing us to adapt state-of-the-art acceleration techniques from domains such as image generation.

While consistency training [10] has shown remarkable success in accelerating generation tasks, it is primarily designed for continuous data like images. To address the challenges posed by the multi-modal nature of 3D molecules, we introduce a **consistency parameter objective**—a novel objective function that enforces consistency in the parameter space of BFNs. Our key insight is that the continuous parameter space of BFNs provides a smooth and structured representation of multi-modal molecular information, which in turn enhances the stability and effectiveness of consistency training in this domain.

The proposed consistency parameter objective is formulated as follows (we highlight the differences with standard consistency training using color blue):

$$\mathcal{L}_\phi(\boldsymbol{g}) = \sum_{i=1, t_i \geq t_{\text{stable}}}^{N-1} \mathbb{E}_q \|\Phi(\boldsymbol{\theta}_{t_i}, t_i) - \Phi^-(\boldsymbol{\theta}_{t_{i+1}}, t_{i+1})\|_2^2 \tag{9}$$

where Φ^- is neural network with stop-gradient, t_{stable} is the stable point, ensuring that the consistency parameter objective is only computed in the second phase. And $\{t_i\}_{i=1}^N$ denotes discretized time steps based on pre-defined curriculum [20]. More implementation details can be found in Appendix E.

Similar to standard consistency training, the goal of the consistency parameter objective is to accurately predict ground-truth molecule—but from the parameter space rather than the sample space. This objective is naturally aligned with the original training goal of BFNs, where the network Φ is optimized to perform accurate Bayesian updates by predicting the ground-truth molecule $\boldsymbol{g}$:

Proposition 4.6. *In the original formulation BFNs, the objective is upper bounded by the estimation error of ground truth molecules:*

$$\mathop{\mathbb{E}}_{p(\boldsymbol{g}), p_F(\boldsymbol{\theta}|\boldsymbol{g},t)} D_{KL}(p_s(\boldsymbol{y}|\boldsymbol{g},t)||p_R(\boldsymbol{y}|\boldsymbol{\theta},t)) \lesssim \mathop{\mathbb{E}}_{p(\boldsymbol{g}), p_F(\boldsymbol{\theta}|\boldsymbol{g},t)} d(\boldsymbol{g}, \Phi(\boldsymbol{\theta},t)) \tag{10}$$

where $d(\cdot,\cdot)$ is a measure of differences that depends on the data modality.

This insight enabled us to jointly optimize consistency parameter objective as well as the original BFN objective to stabilize the training dynamics.

5 Experiments

In this section, we justify the advantages of MOLTD with comprehensive experiments. We first introduce our experimental setup in Section 5.1. Then we report and analyze the evaluation results in Section 5.2. We also provide further ablation studies in Section 5.3 to investigate the effect of several model designs. The ablation on sampling steps is shown in the Appendix F.

5.1 Experiment Setup

Task and Datasets. Following the setting of prior works [24, 25, 40, 27, 41], we focus on molecular modeling and efficient generation, which measure the efficiency of the models to generate chemically

Table 1: Results on QM9 and DRUG datasets, including NFE, atom stability, molecule stability, validity, and validity×uniqueness×novelty (V&U%N). A higher number indicates a better generation quality. Metrics are calculated with 10000 samples generated from each model, we run the evaluation for 3 times and report the derivation. Compared with previous methods, MOLTD enables around 10× speed up while generate stable and valid molecules.

		QM9				DRUG	
# Metrics	NFE	Atom Sta (%)	Mol Sta (%)	Valid (%)	V&U&N (%)	Atom Sta (%)	Valid (%)
Data	-	99.0	95.2	97.7	-	86.5	99.9
ENF	-	85.0	4.9	40.2	-	-	-
G-Schnet	-	95.7	68.1	85.5	-	-	-
GDM-AUG	1000	97.6	71.6	90.4	66.8	77.1	91.8
EDM	1000	98.7	82.0	91.9	59.6	81.3	92.6
EDM-Bridge	1000	98.8	84.6	92.0	-	82.4	92.8
GeoLDM	1000	98.9	89.4	93.8	53.9	84.4	**99.3**
EquiFM	200	98.9	88.3	94.7	53.7	84.1	98.9
GeoBFN	100	98.6	87.2	93.0	64.4	78.9	93.1
GOAT	90	99.2	-	92.9	**72.3**	84.8	96.2
MOLTD	**12**	**99.40** $\pm$ 0.1	**92.53** $\pm$ 0.2	**96.04** $\pm$ 0.4	67.03 $\pm$ 0.6	**86.88**	95.33

*Note that, for DRUG dataset, molecule stability and uniqueness metric are omitted since they are nearly 0% and 100% respectively for all the methods, expect that MOLTD achieves molecule stability with 6.37 %.

valid and structurally diverse molecules, and their capacity to learn molecular distribution. We evaluate benchmarks over two widely adopted datasets, including **QM9** [12] and the **GEOM-DRUG** [42]. QM9 is a standard dataset that contains 130k 3D molecules with a maximum of 29 atoms, while GEOM-DRUG is a more challenging dataset containing around 450K molecules, each with an average of 44 atoms and up to 181 atoms. And the data configurations directly follow previous works [43, 41, 30, 31].

Evaluation Metrics The evaluation configuration follows the prior works [27, 41, 30]. After generating 10000 molecular geometries, the bond types are first predicted (single, double, triple, or none) based on pair-wise atomic distance and atom types [27]. To qualify the sampling efficiency, we report the number of function evaluations (**NFE**) utilized in sampling for each generative model. With the obtained molecular graph, we evaluate the quality by calculating both **atom stability** and **molecule stability** metrics. The **validity** (based on RDKit) is also reported, which is the percentage of valid molecules among all generated compounds. To comprehensively evaluate the capability of *de novo* molecule design [44], we report **validity**×**uniqueness**×**novelty** (V%U%N), to quantify the percentage of valid, unique, and novel molecules among the generated samples. On QM9, we additionally evaluate how well the model learns the molecular distribution. We report the total variation distance of atom types, and the Wasserstein distance of the bond angles and bond lengths between generated molecules and test set. We also report the strain energy to qualify the overall structure of the generated structure.

Baselines We compare MOLTD to several competitive baseline models. G-Schnet [24] and Equivariant Normalizing Flows (ENF) [40] are previous equivariant generative models for molecules, built on autoregressive and flow-based models respectively. Equivariant Graph Diffusion Models (EDM) with its non-equivariant variant (GDM) [27] are based on diffusion models for molecule generation. EDM-Bridge [41] further boosts the performance of EDM with well-designed informative prior bridges. Furthermore, MOLTD is compared with recent advancements for efficient molecular generation. EquiFM [9] and GOAT [45] are flow-matching models with equivariant optimal transport objective. GeoBFN [31] is the first work that leverages Bayesian flow networks for 3D molecular generation.

5.2 Main Results

The results of efficient molecular generation are presented in Table 1. As shown, MOLTD establishes a new state-of-the-art in generating high-quality molecules with only **12** sampling steps on both QM9 and GEOM-DRUG datasets. Notably, MOLTD achieves **7.5 $\times$ faster generation than flow matching models, 8.3 $\times$ faster than Bayesian Flow Network models, and nearly 100 $\times$ faster than diffusion-based models**, while maintaining superior generation quality. The actual runtime comparison in Table 6 shows that MOLTD's efficiency gain mirrors the improvements observed in the NFEs. Moreover, the results highlight MOLTD's ability to generate diverse and novel molecular

Table 2: Ablation results on QM9 and DRUG datasets. P stands for geometric-informed Prior and C stands for Consistency parameter objective. Metrics include atom stability, molecule stability, and validity. A higher number indicates a better generation quality. In all experiments, NFE is set to 12.

# Components	QM9 Atom Sta (%)	QM9 Mol Sta (%)	QM9 Valid (%)	QM9 V&U&N (%)	DRUG Atom Sta (%)	DRUG Mol Sta (%)	DRUG Valid (%)
Data	99.0	95.2	97.7	86.5	-	99.9	
w/o P and C	20.8	2.8	65.4	5.1	3.6	0.0	0.0
w/o P	87.54	48.51	64.10	54.06	69.57	5.42	94.20
w/o C	98.97	90.01	96.10	66.93	77.48	3.02	93.01
MOLTD	**99.40** $\pm$ 0.1	**92.53** $\pm$ 0.2	**96.04** $\pm$ 0.4	**67.03** $\pm$ 0.6	**86.88**	**6.37**	**95.33**

geometries, indicating that it is not over-fitted to a specific subset of the training data. This strong generalization capability underscores the potential of the MOLTD for broad applications in molecular design and drug discovery.

Results on molecular distribution learning, presented in Table 3, demonstrate that MOLTD achieves competitive or superior performance across all evaluated metrics. These results highlight MOLTD's capability for accurate distribution learning and fast sampling convergence.

5.3 Ablation Studies

First, we evaluate the **effectiveness of the two proposed acceleration techniques**. Results in Table 2 demonstrate that, while each technique independently enhances the quality of few-step generation, neither achieves state-of-the-art performance in isolation. This underscores the importance of designing tailored methods for each phase of the generative process and integrating them to achieve optimal results.

Furthermore, we demonstrate the **general applicability of proposed techniques** in two scenarios:

Table 3: Additional results on QM9, including NFE, atom type total variation, Wasserstein distance of bond length and bond angles, and strain energy. A lower number indicates a better generation quality.

# Metrics	NFE	Atom TV	Length W1	Angles W1	S-Energy
Data	-	0.0	0.0	0.0	19.4
EDM	1000	0.9	1.2	9.1	25.8
GeoLDM	1000	0.9	1.4	9.2	26.0
EquiFM	200	1.5	0.6	9.2	25.1
GeoBFN	100	1.3	1.4	8.7	25.3
MOLTD	12	1.2	1.0	8.1	24.9

*Results in the table are obtained by our own experiments.

First, we adapt both techniques to EDM, a diffusion-based molecular generative model. Specifically, we inject representative structural information at the final step of the forward process to construct an informed prior, and initialize sampling from this prior. Since diffusion models lack an explicit parameter space, we apply consistency distillation on a pre-trained EDM model to stabilize training, following [10]. As shown in Figure 3, the adapted techniques significantly accelerate EDM to just **30 NFEs** while maintaining state-of-the-art stability. However, the acceleration remains limited—MOLTD achieves better quality with only **12 NFEs**, owing to the advantage of the explicit parameter space in BFNs for modeling molecular geometry.

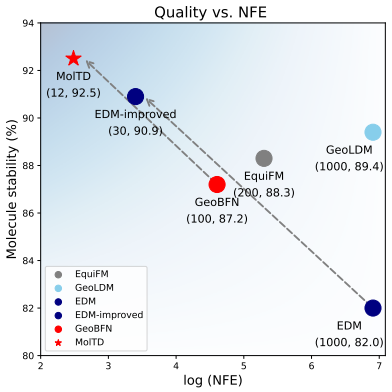

Figure 3: The acceleration effect of proposed method on BFN-based model and diffusion-based models. Note that the NFE is presented in log scale.

Secondly, we demonstrate the cross-data generalization ability of the Geometric-informed Prior. We constructed the geometric-prior using training data from GEOM-DRUG, and plugged it into the generative model trained on QM9—a dataset with distinct structural patterns (smaller molecules, different atom types). Using **12 NFEs** we generated 10000 molecules and evaluated on QM9: **Atom stability** 99.2% **, Molecule stability** 90.22%**, Validity** 94.58%. As the results show, using prior

Table 4: Comparison of structure-based drug design models. For Vina-based metrics, lower value indicate better performance. For others, larger value indicate better performance. MolTD achieves superior performance with significantly fewer NFEs.

Model	NFE	QED	SA	Vina Score		Vina Min	
				Mean	Median	Mean	Median
TargetDiff	1000	0.48	0.58	-5.47	-6.30	-6.64	-6.83
Decomp-R	1000	0.51	0.66	-5.19	-5.27	-6.03	-6.00
MolCRAFT	100	0.50	0.69	-6.59	**-7.04**	-7.27	**-7.26**
MolCRAFT*	25	0.51	0.65	-5.95	-6.70	-6.73	-6.89
MolTD	**25**	**0.54**	**0.72**	**-6.60**	-6.91	**-7.36**	-7.24

*Results in the table are obtained by our own experiments.

constructed from external dataset achieves the same accelerating effect as original MOLTD, demonstrating that the prior captures fundamental geometric structures shared across datasets.

Thirdly, we applied MOLTD for structure-based drug design, integrating it with MolCRAFT [3], a state-of-the-art method in this domain. The geometric-informed prior is constructed using Algorithm 1 from the set of target ligands within the training set. Given that MolCRAFT also utilizes BFNs as its generative backbone, we were able to seamlessly apply our consistency parameter objective to its parameter space. We adhered to previously established evaluation settings to report our results, which is detailed in the Appendix E. As the results show in Table 4, our method demonstrates comparable or superior performance with significantly fewer NFEs. This outcome underscores its considerable practical potential.

6 Conclusion

In this paper, we introduce MOLTD, a novel approach to accelerating 3D molecule generative models by addressing geometric generation challenges. Through theoretical and empirical analysis, we identify a two-phase generative pattern—permutation reordering and atomic feature adjustment—and propose two key techniques for accelerating each phase: a geometric-informed prior for faster reordering and a consistency parameter objective for accelerated adjustment. Extensive experiments demonstrate that MOLTD significantly improve sampling speed, achieving a speed-up of approximately **8**$\times$ compared to previous advancements, while maintaining state-of-the-art generation quality. Beyond improving efficiency, MOLTD offers new insights into geometric generative models, with applications in drug discovery, material design, and beyond.

Acknowledgments

The authors thank Keyue Qiu, Yanru Qu for the helpful discussions and proofreading of the paper, as well as the anonymous reviewers for reviewing the draft.

This work is supported by the Natural Science Foundation of China (Grant No. 62376133) and sponsored by Beijing Nova Program (20240484682) and the Wuxi Research Institute of Applied Technologies, Tsinghua University (20242001120).

References

[1] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J Ballard, Joshua Bambrick, et al. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, pages 1–3, 2024.

[2] Jiaqi Guan, Wesley Wei Qian, Xingang Peng, Yufeng Su, Jian Peng, and Jianzhu Ma. 3d equivariant diffusion for target-aware molecule generation and affinity prediction. *arXiv preprint arXiv:2303.03543*, 2023.

[3] Yanru Qu, Keyue Qiu, Yuxuan Song, Jingjing Gong, Jiawei Han, Mingyue Zheng, Hao Zhou, and Wei-Ying Ma. Molcraft: Structure-based drug design in continuous parameter space. *arXiv preprint arXiv:2404.12141*, 2024.

[4] Claudio Zeni, Robert Pinsler, Daniel Zügner, Andrew Fowler, Matthew Horton, Xiang Fu, Sasha Shysheya, Jonathan Crabbé, Lixin Sun, Jake Smith, et al. Mattergen: a generative model for inorganic materials design. *arXiv preprint arXiv:2312.03687*, 2023.

[5] xinshi chen, yuxuan zhang, chan lu, wenzhi ma, jiaqi guan, chengyue gong, jincai yang, hanyu zhang, ke zhang, shenghao wu, et al. Protenix-advancing structure prediction through a comprehensive alphafold3 reproduction. *bioRxiv*, pages 2025–01, 2025.

[6] Mihaly Varadi, Damian Bertoni, Paulyna Magana, Urmila Paramval, Ivanna Pidruchna, Malarvizhi Radhakrishnan, Maxim Tsenkov, Sreenath Nair, Milot Mirdita, Jingi Yeo, et al. Alphafold protein structure database in 2024: providing structure coverage for over 214 million protein sequences. *Nucleic acids research*, 52(D1):D368–D375, 2024.

[7] W Patrick Walters, Matthew T Stahl, and Mark A Murcko. Virtual screening—an overview. *Drug discovery today*, 3(4):160–178, 1998.

[8] Hengyuan Ma, Li Zhang, Xiatian Zhu, and Jianfeng Feng. Accelerating score-based generative models with preconditioned diffusion sampling. In *European Conference on Computer Vision*, pages 1–16. Springer, 2022.

[9] Yuxuan Song, Jingjing Gong, Minkai Xu, Ziyao Cao, Yanyan Lan, Stefano Ermon, Hao Zhou, and Wei-Ying Ma. Equivariant flow matching with hybrid probability transport for 3d molecule generation. *Advances in Neural Information Processing Systems*, 36, 2024.

[10] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

[11] Qiang Liu. Rectified flow: A marginal preserving approach to optimal transport. *arXiv preprint arXiv:2209.14577*, 2022.

[12] Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole Von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific data*, 1(1):1–7, 2014.

[13] Simon Axelrod and Rafael Gomez-Bombarelli. Geom: Energy-annotated molecular conformations for property prediction and molecular generation. *arXiv preprint arXiv:2006.05531*, 2020.

[14] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[15] Qinsheng Zhang and Yongxin Chen. Diffusion normalizing flow. *Advances in neural information processing systems*, 34:16280–16291, 2021.

[16] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022.

[17] Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, and Ricky TQ Chen. Multisample flow matching: Straightening flows with minibatch couplings. *arXiv preprint arXiv:2304.14772*, 2023.

[18] Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *arXiv preprint arXiv:2302.00482*, 2023.

[19] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

[20] Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. *arXiv preprint arXiv:2310.14189*, 2023.

[21] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International conference on machine learning*, pages 2323–2332. PMLR, 2018.

[22] Qi Liu, Miltiadis Allamanis, Marc Brockschmidt, and Alexander Gaunt. Constrained graph variational autoencoders for molecule design. *Advances in neural information processing systems*, 31, 2018.

[23] Chence Shi, Minkai Xu, Zhaocheng Zhu, Weinan Zhang, Ming Zhang, and Jian Tang. Graphaf: a flow-based autoregressive model for molecular graph generation. *arXiv preprint arXiv:2001.09382*, 2020.

[24] Niklas Gebauer, Michael Gastegger, and Kristof Schütt. Symmetry-adapted generation of 3d point sets for the targeted discovery of molecules. *Advances in neural information processing systems*, 32, 2019.

[25] Youzhi Luo and Shuiwang Ji. An autoregressive flow model for 3d molecular geometry generation from scratch. In *International conference on learning representations (ICLR)*, 2022.

[26] Tomohide Masuda, Matthew Ragoza, and David Ryan Koes. Generating 3d molecular structures conditional on a receptor binding site with deep generative models. *arXiv preprint arXiv:2010.14442*, 2020.

[27] Emiel Hoogeboom, Vıctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In *International Conference on Machine Learning*, pages 8867–8887. PMLR, 2022.

[28] Shitong Luo, Yufeng Su, Xingang Peng, Sheng Wang, Jian Peng, and Jianzhu Ma. Antigen-specific antibody design and optimization with diffusion-based generative models for protein structures. *Advances in Neural Information Processing Systems*, 35:9754–9767, 2022.

[29] Namrata Anand and Tudor Achim. Protein structure and sequence generation with equivariant denoising diffusion probabilistic models. *arXiv preprint arXiv:2205.15019*, 2022.

[30] Minkai Xu, Alexander S Powers, Ron O Dror, Stefano Ermon, and Jure Leskovec. Geometric latent diffusion models for 3d molecule generation. In *International Conference on Machine Learning*, pages 38592–38610. PMLR, 2023.

[31] Yuxuan Song, Jingjing Gong, Hao Zhou, Mingyue Zheng, Jingjing Liu, and Wei-Ying Ma. Unified generative modeling of 3d molecules with bayesian flow networks. In *The Twelfth International Conference on Learning Representations*, 2023.

[32] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys*, 56(4):1–39, 2023.

[33] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[34] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[35] Chence Shi, Shitong Luo, Minkai Xu, and Jian Tang. Learning gradient fields for molecular conformation generation. In *International Conference on Machine Learning*, pages 9558–9568. PMLR, 2021.

[36] Alex Graves, Rupesh Kumar Srivastava, Timothy Atkinson, and Faustino Gomez. Bayesian flow networks. *arXiv preprint arXiv:2308.07037*, 2023.

[37] Camille Georges Wermuth. *The practice of medicinal chemistry*. Academic Press, 2011.

[38] Xiao Qing Lewell, Duncan B Judd, Stephen P Watson, and Michael M Hann. Recap retrosynthetic combinatorial analysis procedure: a powerful new technique for identifying privileged molecular fragments with useful applications in combinatorial chemistry. *Journal of chemical information and computer sciences*, 38(3):511–522, 1998.

[39] Zian Li, Cai Zhou, Xiyuan Wang, Xingang Peng, and Muhan Zhang. Geometric representation condition improves equivariant molecule generation. *arXiv preprint arXiv:2410.03655*, 2024.

[40] Victor Garcia Satorras, Emiel Hoogeboom, Fabian Fuchs, Ingmar Posner, and Max Welling. E (n) equivariant normalizing flows. *Advances in Neural Information Processing Systems*, 34:4181–4192, 2021.

[41] Lemeng Wu, Chengyue Gong, Xingchao Liu, Mao Ye, and Qiang Liu. Diffusion-based molecule generation with informative prior bridges. *Advances in Neural Information Processing Systems*, 35:36533–36545, 2022.

[42] Simon Axelrod and Rafael Gomez-Bombarelli. Geom, energy-annotated molecular conformations for property prediction and molecular generation. *Scientific Data*, 9(1):185, 2022.

[43] Brandon Anderson, Truong Son Hy, and Risi Kondor. Cormorant: Covariant molecular neural networks. *Advances in neural information processing systems*, 32, 2019.

[44] W Patrick Walters and Mark Murcko. Assessing the impact of generative ai on medicinal chemistry. *Nature biotechnology*, 38(2):143–145, 2020.

[45] Haokai Hong, Wanyu Lin, and Kay Chen Tan. Fast 3d molecule generation via unified geometric optimal transport. *arXiv preprint arXiv:2405.15252*, 2024.

[46] Uri M Ascher and Linda R Petzold. *Computer methods for ordinary differential equations and differential-algebraic equations*. SIAM, 1998.

[47] Vıctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) equivariant graph neural networks. In *International conference on machine learning*, pages 9323–9332. PMLR, 2021.

[48] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[49] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3nd International Conference on Learning Representations*, 2014.

[50] Shitong Luo, Jiaqi Guan, Jianzhu Ma, and Jian Peng. A 3d generative model for structure-based drug design. *Advances in Neural Information Processing Systems*, 34:6229–6239, 2021.

[51] Jiaqi Guan, Xiangxin Zhou, Yuwei Yang, Yu Bao, Jian Peng, Jianzhu Ma, Qiang Liu, Liang Wang, and Quanquan Gu. Decompdiff: diffusion models with decomposed priors for structure-based drug design. *arXiv preprint arXiv:2403.07902*, 2024.

[52] Shikun Feng, Yuyan Ni, Yanyan Lan, Zhi-Ming Ma, and Wei-Ying Ma. Fractional denoising for 3d molecular pre-training. In *International Conference on Machine Learning*, pages 9938–9961. PMLR, 2023.

A Algorithms

For a better understanding of the whole procedure in construction the geometric-informed prior, we involve the detailed algorithms in Algorithm 1.

Algorithm 1 Construction of a Geometric-Informed Prior

Input: A set of M molecules $\{\boldsymbol{g}_n\}_{n=1}^{M}$ of the same size, and accuracy level α_p.
Output: The geometric-informed prior $\boldsymbol{\theta}_p$.
Initialize:
Set the reference molecule $\boldsymbol{g}_{\text{ref}} = \boldsymbol{g}_1$, and initialize the aligned molecules list Mol $= [\boldsymbol{g}_{\text{ref}}]$.
for $i = 2$ **to** M **do**
Compute the optimal permutation and rotation using the optimal transport (EOT) objective:

$$\pi_i^*, \mathbf{R}_i^* = \underset{\pi, \mathbf{R}}{\operatorname{argmin}} \|\pi(\mathbf{R}\boldsymbol{g}_i) - \boldsymbol{g}_{\text{ref}}\|_2$$

Append the aligned molecule to the list: Mol.append$(\pi^*(\mathbf{R}^*\boldsymbol{g}_i))$
end for
Extract Information: Compute the average of the aligned molecules:

$$\bar{\boldsymbol{g}} = \text{Mean}(\text{Mol})$$

Project to Parameter Space:

$$\boldsymbol{\theta}_p = \mathop{\mathbb{E}}_{p_S(\boldsymbol{y}|\bar{\boldsymbol{g}}, \alpha_p)} \delta(\boldsymbol{\theta}_{\alpha_p} - h(\boldsymbol{y}, 0, \alpha_p))$$

B Basic Introduction of Bayesian Flow Networks

In this section, we introduce the key components of Bayesian Flow Networks (BFNs) [36] from the perspective of Bayesian inference. As in standard Bayesian inference, a prior distribution $p_I(\boldsymbol{\theta}_0)$ is specified for each data modality. For example, for continuous data, we assume a Gaussian prior where $\boldsymbol{\theta}_0$ represents the mean and variance, whereas for discrete data, we use a categorical distribution where $\boldsymbol{\theta}_0$ corresponds to the probability of each category.

The *sender distribution* serves as the likelihood function in Bayesian inference, generating noisy observations $\boldsymbol{y}$ that iteratively update the prior parameters. These noisy signals are obtained by perturbing $\boldsymbol{g}$ according to the following distribution:

$$q(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n | \boldsymbol{g}) = \prod_{i=1}^{N} p_S(\boldsymbol{y}_i | \boldsymbol{g}, \alpha_i), \tag{11}$$

where p_S, referred to as the *sender distribution*, is typically modeled as a Gaussian. The parameters α_i control the noise level, ensuring that the sequence $\langle \boldsymbol{y}_1, \cdots, \boldsymbol{y}_n \rangle$ exhibits an increasing signal-to-noise ratio.

These noisy signals are then used to iteratively update the posterior distribution by modifying the parameter $\boldsymbol{\theta}$:

$$p_I(\boldsymbol{g} \mid \boldsymbol{y}, \boldsymbol{\theta}_i, \alpha) = p_I(\boldsymbol{g} \mid \boldsymbol{\theta}_{i+1}) = p_I(\boldsymbol{g} \mid h(\boldsymbol{y}, \boldsymbol{\theta}_i, \alpha)), \tag{12}$$

where the deterministic function h, known as the *Bayesian update function*, governs the update rule:

$$\boldsymbol{\theta}_i \leftarrow h(\boldsymbol{\theta}_{i-1}, \boldsymbol{y}_i, \alpha_i). \tag{13}$$

The explicit form of h depends on the data modality and the choice of p_S, as derived in [36]. With Equation (11) and Equation (13), the distribution of the parameter can be formulated as

$$q_U(\boldsymbol{\theta}_i \mid \boldsymbol{\theta}_{i-1}, \boldsymbol{g}, \alpha_i) = \mathop{\mathbb{E}}_{p_S(\boldsymbol{y}_i|\boldsymbol{\theta}_{i-1},\alpha_i)} \delta(\boldsymbol{\theta}_i - h(\boldsymbol{y}_i, \boldsymbol{\theta}_{i-1}, \alpha_i)) \tag{14}$$

As the sequence $\langle \boldsymbol{y}_1, \cdots, \boldsymbol{y}_n \rangle$ provides increasing information about $\boldsymbol{g}$, the posterior distribution is progressively refined, yielding a more accurate approximation of the target distribution.

However, during generation, the target molecule $\boldsymbol{g}$ is unknown, meaning the true noisy observations required for Bayesian updates are inaccessible. To address this, BFNs adopt a variational approach by training a neural network $\Phi(\boldsymbol{\theta}_t, t)$ to predict the noisy signal. Specifically, $\Phi(\boldsymbol{\theta})$ is trained to approximate the ground truth sample via the *output distribution*:

$$\hat{\boldsymbol{g}} \sim p_O(\boldsymbol{g}|\boldsymbol{\theta}; \Phi) = \prod_{d=1}^{D} p_O(\boldsymbol{g}^{(d)} \mid \Phi(\boldsymbol{\theta})^{(d)}), \tag{15}$$

where p_O is referred to as the *output distribution*, and the output of $\Phi(\boldsymbol{\theta})$ lies in its parameter space.

To generate predicted noisy signals $\hat{\boldsymbol{y}}$, we marginalize over the predicted $\hat{\boldsymbol{g}}$:

$$p_R(\boldsymbol{y}_i|\boldsymbol{\theta}_{i-1}, \alpha_i, \phi) = \mathop{\mathbb{E}}_{p_O(\boldsymbol{g}'|\boldsymbol{\theta}_{i-1};\phi)} p_S(\boldsymbol{y}_i|\boldsymbol{g}'; \alpha_i), \tag{16}$$

where p_R, known as the *receiver distribution*, incorporates both the output distribution and the sender distribution to approximate the true noisy signal.

BFNs is trained to approximate the distribution of $\langle \boldsymbol{y}_1, \cdots, \boldsymbol{y}_n \rangle$, in order to acquire accurate signal for Bayesian updates. The variational lower bound is optimized:

$$\begin{aligned} \log p_{\boldsymbol{\phi}}(\boldsymbol{g}) &\geq \mathbb{E}_q \left[\log \frac{p_{\boldsymbol{\phi}}(\boldsymbol{g} \mid \boldsymbol{y}_1, \ldots, \boldsymbol{y}_n) p_{\boldsymbol{\phi}}(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)}{q(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n \mid \boldsymbol{g})} \right] \\ &= -D_{\mathrm{KL}}(q \| p_{\boldsymbol{\phi}}(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)) \\ &\quad + \mathbb{E}_{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n \sim q} \left[\log p_\phi(\boldsymbol{g} \mid \boldsymbol{y}_1, \ldots, \boldsymbol{y}_n) \right] \end{aligned} \tag{17}$$

where $p_\phi(\boldsymbol{g} \mid \boldsymbol{y}_1, \ldots, \boldsymbol{y}_n) = p_O(\boldsymbol{g}|\boldsymbol{\theta}_n; \Phi)$. From this perspective, BFNs can be viewed as a latent variable model $\langle \boldsymbol{y}_1, \cdots, \boldsymbol{y}_n \rangle$ in the parameter space:

$$\begin{aligned} p_\phi(\boldsymbol{\theta}_0, \cdots, \boldsymbol{\theta}_n) &= \prod_{i=1}^{n} p_U(\boldsymbol{\theta}_i \mid \boldsymbol{\theta}_{i-1}, \alpha_i) \\ &= \prod_{i=1}^{n} \mathop{\mathbb{E}}_{p_R(\hat{\boldsymbol{y}}_i|\boldsymbol{\theta}_{i-1},\alpha_i,\phi)} \delta(\boldsymbol{\theta}_i - h(\hat{\boldsymbol{y}}_i, \boldsymbol{\theta}_{i-1}, \alpha_i)) \end{aligned} \tag{18}$$

It could also be view as a latent variable model in the sample space:

$$\begin{aligned} p_\phi\left(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n\right) &= p_\phi(\boldsymbol{y}_1) \prod_{i=2}^{n} p_\phi(\boldsymbol{y}_i \mid \boldsymbol{y}_{\{1:i-1\}}) \\ &= \prod_{i=1}^{n} p_\phi(\boldsymbol{y}_i \mid \boldsymbol{\theta}_{i-1}) \\ &= \prod_{i=1}^{n} \mathop{\mathbb{E}}_{p_O(\boldsymbol{g}'_i|\boldsymbol{\theta}_{i-1};\Phi)} \left[p_S(\boldsymbol{y}_i|\boldsymbol{g}'_i; \alpha_i) \right], \end{aligned}$$

C Details of Investigation of Generative Trajectory

C.1 Implementation Details

In this section, we provide more details on the investigation of the generative trajectory. In the investigation, we utilize official checkpoints from the open-source repositories of the geometric

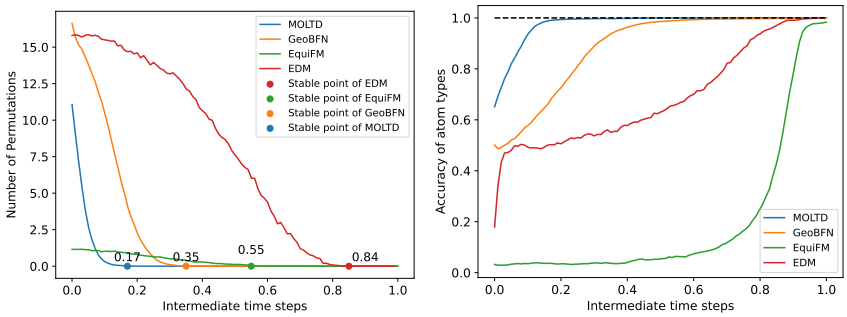

Figure 4: Investigation on trajectory of predicted molecules. **Left**: $\mathcal{D}_{\text{structure}}$ along the trajectory. **Right**: $\mathcal{D}_{\text{types}}$ along the trajectory. MOLTD achieve the earliest stable point among the baselines with accurate atom type prediction.

generative models, including EDM [27], EquiFM [9], and GeoBFN [31]. Each model requires different sampling steps to achieve comparable performance, and thus we use 1000 steps for EDM, 200 sampling for EquiFM with Euler discretization [46], and 100 steps for GeoBFN. We randomly sample 1000 trajectories for each model and calculate the mean of $\mathcal{D}_{\text{structre}}$ and $\mathcal{D}_{\text{types}}$ at each time step.

C.2 Investigation on the Generative Trajectory of Predicted Molecules

Furthermore, we analysis the generative trajectory of predicted molecules. GeoBFN directly generate of sequence of predicted molecules in the sampling process, while EDM and EquiFM could predict the molecules by denoising the noisy sample. For EDM, the molecules at time step t are predicted by:

$$\hat{\boldsymbol{g}}_0 = \frac{1}{\alpha_t}\boldsymbol{g}_t - \frac{\sigma_t}{\alpha_t}\hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{z}_t, t) \tag{19}$$

where α_t and σ_t are the noise schedules of EDM, $\hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{z}_t, t)$ is the predicted noise. For EquiFM, the noisy molecules are generated by $\boldsymbol{g}_t = (1-t)\boldsymbol{g}_0 + t\boldsymbol{\epsilon}$, and the estimated velocity $\boldsymbol{v}_\theta(\boldsymbol{g}_t, t)$ is trained to approximate $-\boldsymbol{g}_0 + \boldsymbol{\epsilon}$. As a result, the predicted molecules at time step t can be calculated by:

$$\hat{\boldsymbol{g}}_0 = \boldsymbol{g}_t - t\boldsymbol{v}_\theta \tag{20}$$

We conduct the same analysis as described in the main body of the paper, with the results presented in Figure 4. As shown, the trajectories of the predicted molecules exhibit a similar pattern to those of the noisy samples analyzed earlier. These trajectories display a distinct two-phase structure, with MOLTD achieving the earliest stable point among the baselines while maintaining accurate atom type predictions.

C.3 Analyzing the Distinct Generative Pattern of Different Generative Models

EDM typically requires around 1000 sampling steps to generate valid molecules, with the majority of these steps spent in the first phase to achieve structural stability, as shown in Figure 2. This inefficiency can be attributed to the stochastic nature of diffusion models, which necessitates numerous denoising steps to stabilize the structure.

EquiFM achieves minimal structural difference at the start of generation by employing the EOT objective for training. However, our analysis reveals that EquiFM also spends a large proportion of sampling steps in the first phase. This inefficiency can be attributed to the inaccurate prediction of atom types, as shown in the middle of Figure 2, which complicates the convergence to stable structures.

GeoBFN benefits from a smoother, less noisy trajectory compared to EDM, which leads to a faster convergence rate in the first phase. Additionally, its capability to handle multi-modal features in molecules enables accurate atom-type predictions. Consequently, GeoBFN reaches an earlier stable point compared with previous methods. However, GeoBFN experiences a large number of permutations at the start of the first phase, as its iterative sampling process begins with a fixed, uninformative prior. Furthermore, with a relatively long time span during the second phase, GeoBFN fails to apply acceleration techniques to further improve efficiency.

D Formal Proof of Theorems and Propositions

D.1 Proof of Theorem 4.3

Proof. Without loss of generality, assume that $i = 1$ and $j = 2$. Consider a permutation operator π_0 that swaps x_1^1 and x_1^2,

$$\pi_0 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

such that $\pi_0(x_1^1) = x_1^2$ and $\pi_0(x_1^2) = x_1^1$. From the definition of r, we know that $x_1^1 \notin V(x_0^2; r)$ and $x_1^2 \notin V(x_0^1; r)$. Thus,

$$\|\pi_0(x_1^1) - x_0^1\|_2 + \|\pi_0(x_1^2) - x_0^2\|_2 = \|x_1^2 - x_0^1\|_2 + \|x_1^1 - x_0^2\|_2 > 2r > \|x_1^1 - x_0^1\|_2 + \|x_1^2 - x_0^2\|_2.$$

This shows that after the permutation, $\|\pi_0(x_1) - x_0\|$ increases. Therefore,

$$\operatorname*{argmin}_{\pi} \|\pi(x_1) - x_0\|_2 = I_{N\times N}.$$

□

D.2 Proof of Theorem 4.4

Proof. Without loss of generality, assume $i = 1$ and $j = 2$. Given the known condition, $x_2^1 \notin V(x_0^1; r)$. First, consider the 2D case, where x_2^2 lies on the plane $\mathcal{C}_1$ formed by x_0^1, x_0^2, and x_2^1.

The condition for no permutation is

$$\|x_2^1 - x_0^1\|_2 + \|x_2^2 - x_0^2\|_2 < \|x_2^1 - x_0^2\|_2 + \|x_2^2 - x_0^1\|_2,$$

implies that the sum of the distances to the targets is smaller than the sum of the distances after the permutation. This is equivalent to

$$\|x_2^1 - x_0^1\|_2 - \|x_2^1 - x_0^2\|_2 < \|x_2^2 - x_0^1\|_2 - \|x_2^2 - x_0^2\|_2,$$

Further, this is equivalent to x_2^1 and x_2^2 lying outside one sheet of the hyperbola with foci at x_0^1 and x_0^2, passing through x_2^1. Thus, we have shown that the original statement holds for the 2D case.

When x_2^2 is not on the plane $\mathcal{C}$, we can rotate $\mathcal{C}_1$ around the line through x_0^1 and x_0^2 as the axis, so that x_2^1, x_2^2, x_0^1, and x_0^2 lie on a new plane $\mathcal{C}_2$. In this case, the distances from x_2^1 and x_2^2 to x_0^1 and x_0^2 remain unchanged, still satisfying the above equation. The hyperbola rotates along its imaginary axis, producing a two-sheeted hyperboloid. Therefore, the original statement holds in 3D as well. □

D.3 Proof of Proposition 4.5

Proof. Based on the introduction of BFNs in Appendix B, we could formulate the density function of generated molecules as:

$$\begin{aligned} p_\phi(\boldsymbol{x}|\boldsymbol{\theta}_p) &= \int p_\phi(\boldsymbol{x} \mid \boldsymbol{\theta}_p, \boldsymbol{\theta}_{p+1}^x, \cdots, \boldsymbol{\theta}_{p+n}^x) p_\phi(\boldsymbol{\theta}_{p+1}^x, \cdots, \boldsymbol{\theta}_{p+n}^x \mid \boldsymbol{\theta}_\alpha) d\boldsymbol{\theta}_{p+1:p+n}^x \\ &= \int p_\phi(\boldsymbol{x} \mid \boldsymbol{\theta}_{p+n}^x) \prod_{i=1}^{n} p_U\left(\boldsymbol{\theta}_{p+i} \mid \boldsymbol{\theta}_{p+i-1}; \alpha_i\right) d\boldsymbol{\theta}_{p+1:p+n}^x. \quad \text{(Markov property)} \end{aligned} \tag{21}$$

Note that $p_\phi(\boldsymbol{x} \mid \boldsymbol{\theta}_{p+n}^x) = p_\phi(\mathbf{R}\boldsymbol{x} \mid \mathbf{R}\boldsymbol{\theta}_{p+n}^x) = p_O\left(\mathbf{R}(\boldsymbol{x}) \mid \mathbf{R}(\boldsymbol{\theta}_{p+n}^x); \phi\right)$ due to the property of EGNN. Furthermore, based on the same argument used in Theorem 3.1 in [31], we could prove that $p_U\left(\boldsymbol{\theta}_{p+i} \mid \boldsymbol{\theta}_{p+i-1}; \alpha_{p+i}\right)$ satisfies the equivariant condition that $p_U\left(\boldsymbol{\theta}_{p+i} \mid \boldsymbol{\theta}_{p+i-1}; \alpha_i\right) =$

$p_U\left(\mathbf{R}\boldsymbol{\theta}_{p+i} \mid \mathbf{R}\boldsymbol{\theta}_{p+i-1}; \alpha_{p+i}\right)$. Thus, we could prove $p_\phi(\boldsymbol{x}|\boldsymbol{\theta}_p)$ is rotational-equivariant:

$$
\begin{aligned}
p_\phi(\boldsymbol{x}|\boldsymbol{\theta}_p) &= \int p_\phi(\boldsymbol{x} \mid \boldsymbol{\theta}_p, \boldsymbol{\theta}^x_{p+1}, \cdots, \boldsymbol{\theta}^x_{p+n}) p_\phi(\boldsymbol{\theta}^x_{p+1}, \cdots, \boldsymbol{\theta}^x_{p+n} \mid \boldsymbol{\theta}_\alpha) d\boldsymbol{\theta}^x_{p+1:p+n} \\
&= \int p_\phi(\boldsymbol{x} \mid \boldsymbol{\theta}^x_{p+n}) \prod_{i=1}^{n} p_U\left(\boldsymbol{\theta}_{p+i} \mid \boldsymbol{\theta}_{p+i-1}; \alpha_i\right) d\boldsymbol{\theta}^x_{p+1:p+n}. \qquad (22) \\
&= \int p_\phi(\mathbf{R}\boldsymbol{x} \mid \mathbf{R}\boldsymbol{\theta}^x_{p+n}) \prod_{i=1}^{n} p_U\left(\mathbf{R}\boldsymbol{\theta}_{p+i} \mid \mathbf{R}\boldsymbol{\theta}_{p+i-1}; \alpha_i\right) d\boldsymbol{\theta}^x_{p+1:p+n} \qquad (23) \\
&= \int p_\phi(\mathbf{R}\boldsymbol{x} \mid \mathbf{R}\boldsymbol{\theta}^x_{p+n}) \prod_{i=1}^{n} p_U\left(\mathbf{R}\boldsymbol{\theta}_{p+i} \mid \mathbf{R}\boldsymbol{\theta}_{p+i-1}; \alpha_i\right) |\det(\mathbf{R})|^n d\boldsymbol{\theta}^x_{p+1:p+n} \qquad (24) \\
&= p_\phi(\mathbf{R}\boldsymbol{x}|\mathbf{R}\boldsymbol{\theta}_p) \qquad (25)
\end{aligned}
$$

Here we used the change of variable formula and the fact that $|\det(\mathbf{R})| = 1$, as $\mathbf{R}$ is an rotation matrix.

For completeness, we include the derivation of $p_U\left(\boldsymbol{\theta}_i \mid \boldsymbol{\theta}_{i-1}; \alpha_i\right) = p_U\left(\mathbf{R}\boldsymbol{\theta}_i \mid \mathbf{R}\boldsymbol{\theta}_{i-1}; \alpha_i\right)$ from [31]: Recall that $p_U\left(\boldsymbol{\theta}_i \mid \boldsymbol{\theta}_{i-1}; \alpha_i\right) = \mathop{\mathbb{E}}_{p_O(\mathbf{y}_i|\boldsymbol{\theta}_{i-1};\alpha_i)} \delta\left(\boldsymbol{\theta}_i - h\left(\boldsymbol{\theta}_{i-1}, \mathbf{y}_i, \alpha_i\right)\right)$, then we have:

$$
\begin{aligned}
p_U\left(\mathbf{R}\boldsymbol{\theta}_i \mid \mathbf{R}\boldsymbol{\theta}_{i-1}; \alpha_i\right) &= \mathop{\mathbb{E}}_{p_O(\mathbf{y}_i|\mathbf{R}\boldsymbol{\theta}_{i-1};\alpha_i)} \delta\left(\mathbf{R}\boldsymbol{\theta}_i - h\left(\mathbf{R}\boldsymbol{\theta}_{i-1}, \mathbf{y}_i, \alpha_i\right)\right) \\
&= \int p_O\left(\mathbf{y}_i \mid \mathbf{R}\boldsymbol{\theta}_{i-1}; \alpha_i\right) \delta\left(\mathbf{R}\boldsymbol{\theta}_i - h\left(\mathbf{R}\boldsymbol{\theta}_{i-1}, \mathbf{y}_i, \alpha_i\right)\right) d\boldsymbol{y}_i \qquad (26)
\end{aligned}
$$

Then we apply integration-by-substitution and replace the variable $\boldsymbol{y}_i$ with a new variable $\boldsymbol{y}'_i$, *i.e.* $\boldsymbol{y}_i = \mathbf{R}\boldsymbol{y}'_i$, into the Eq. 26:

$$
\begin{aligned}
&\int p_O\left(\mathbf{y}_i \mid \mathbf{R}\boldsymbol{\theta}_{i-1}; \alpha_i\right) \delta\left(\mathbf{R}\boldsymbol{\theta}_i - h\left(\mathbf{R}\boldsymbol{\theta}_{i-1}, \mathbf{y}_i, \alpha_i\right)\right) d\boldsymbol{y}_i \\
&= \int p_O\left(\mathbf{R}\boldsymbol{y}'_i \mid \mathbf{R}\boldsymbol{\theta}_{i-1}; \alpha_i\right) \delta\left(\mathbf{R}\boldsymbol{\theta}_i - h\left(\mathbf{R}\boldsymbol{\theta}_{i-1}, \mathbf{R}\boldsymbol{y}'_i, \alpha_i\right)\right) d\mathbf{R}\boldsymbol{y}'_i \\
&= \int p_O\left(\mathbf{R}\boldsymbol{y}'_i \mid \mathbf{R}\boldsymbol{\theta}_{i-1}; \alpha_i\right) \delta\left(\mathbf{R}\boldsymbol{\theta}_i - h\left(\mathbf{R}\boldsymbol{\theta}_{i-1}, \mathbf{R}\boldsymbol{y}'_i, \alpha_i\right)\right) |\det(\mathbf{R})| d\boldsymbol{y}'_i \qquad (27)
\end{aligned}
$$

The rotation matrix $\mathbf{R}$ is a SO(3) matrix, thus the $|\det(\mathbf{R})| = 1$. And for the continuous coordinate variable, the update function h for continuous data [36] is also equivariant:

$$
h\left(\mathbf{R}\boldsymbol{\theta}_{i-1}, \mathbf{R}\boldsymbol{y}_i, \alpha_i\right) = \frac{\mathbf{R}\boldsymbol{\theta}_{i-1}\rho_{i-1} + \mathbf{R}\boldsymbol{y}_i\alpha_i}{\rho_i} = \mathbf{R}h\left(\boldsymbol{\theta}_{i-1}, \boldsymbol{y}_i, \alpha_i\right) \qquad (28)
$$

Putting these conditions back to the Eq. 27, we have that

$$
\begin{aligned}
p_U\left(\mathbf{R}\boldsymbol{\theta}_i \mid \mathbf{R}\boldsymbol{\theta}_{i-1}; \alpha_i\right) &= \int p_O\left(\mathbf{y}_i \mid \mathbf{R}\boldsymbol{\theta}_{i-1}; \alpha_i\right) \delta\left(\mathbf{R}\boldsymbol{\theta}_i - h\left(\mathbf{R}\boldsymbol{\theta}_{i-1}, \mathbf{y}_i, \alpha_i\right)\right) d\boldsymbol{y}_i \\
&= \int p_O\left(\mathbf{R}\boldsymbol{y}'_i \mid \mathbf{R}\boldsymbol{\theta}_{i-1}; \alpha_i\right) \delta\left(\mathbf{R}\boldsymbol{\theta}_i - \mathbf{R}h\left(\boldsymbol{\theta}_{i-1}, \boldsymbol{y}'_i, \alpha_i\right)\right) |\det(\mathbf{R})| d\boldsymbol{y}'_i \\
&= \int p_O\left(\boldsymbol{y}'_i \mid \boldsymbol{\theta}_{i-1}; \alpha_i\right) \delta\left(\boldsymbol{\theta}_i - h\left(\boldsymbol{\theta}_{i-1}, \boldsymbol{y}'_i, \alpha_i\right)\right) d\boldsymbol{y}'_i \\
&= p_U\left(\boldsymbol{\theta}_i \mid \boldsymbol{\theta}_{i-1}; \alpha_i\right) \qquad (29)
\end{aligned}
$$

□

D.4 Proof of Proposition 4.6

Proof. We only need to show the conditioned inequality:

$$
D_{KL}(p_S(\boldsymbol{y}|\boldsymbol{g}, t)||p_R(\boldsymbol{y}|\boldsymbol{\theta}, t)) \lesssim d(\boldsymbol{g}, \Phi(\boldsymbol{\theta}, t)) \qquad (30)
$$

Table 5: Ablation results of different sampling steps on QM9 and DRUG datasets. Metrics include NFE, atom stability, molecule stability, validity, and significance. A higher number indicates a better generation quality.

# Metrics	NFE	QM9 Atom Sta (%)	QM9 Mol Sta (%)	QM9 Valid (%)	QM9 V&U&N (%)	DRUG Atom Sta (%)	DRUG Mol Sta (%)	DRUG Valid (%)
Data	-	99.0	95.2	97.7	-	86.5	-	99.9
GeoBFN	100	98.6	87.2	93.0	64.1	78.9	-	93.1
	500	98.8	88.4	93.4	62.1	81.4	-	93.5
	1000	99.1	90.9	95.3	61.7	85.6	-	92.08
MolTD	9	98.88	86.90	93.24	66.82	80.16	2.50	92.28
	10	99.18	89.71	94.68	**67.57**	83.17	3.55	93.81
	11	99.29	91.20	95.02	67.33	85.40	4.98	94.77
	12	99.40	92.56	96.04	67.03	86.88	6.37	95.33
	13	**99.52**	**93.49**	**96.49**	64.77	**87.95**	**8.13**	**96.06**

For continuous modality, $p_S(\boldsymbol{y}|\boldsymbol{g},t) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{x},\alpha_t\boldsymbol{I})$ and $p_R(\boldsymbol{y}|\boldsymbol{\theta},t) = \mathcal{N}(\boldsymbol{y}|\Phi(\boldsymbol{\theta},t),\alpha_t\boldsymbol{I})$. Since the KL divergence between two Gaussains with the same variance is the square norm of their means, we have:

$$D_{KL}(p_S(\boldsymbol{y}|\boldsymbol{g},t)||p_R(\boldsymbol{y}|\boldsymbol{\theta},t)) \lesssim ||\boldsymbol{g} - \Phi(\boldsymbol{\theta},t)||^2 \tag{31}$$

For discrete data (atom types) or discretised data (number of charges), p_s and p_R are mixture of Gaussain distributions, based on the formulation of BFNs [36, 31]. Here we focus on the discrete data and the proof can be easily generalized to discretised data. For discrete data,

$$p_S = \mathcal{N}(\boldsymbol{y}|\alpha_t(K\boldsymbol{e_g} - 1), \alpha_t K\boldsymbol{I}), \tag{32}$$

where $\boldsymbol{e_g}$ is the one-hot encode of $\boldsymbol{g}$ and K is the total number of classes. And

$$p_R = \sum_{i=1}^{K} \Phi^i(\boldsymbol{\theta},t)\mathcal{N}(\boldsymbol{y}|\alpha_t(K\boldsymbol{e}^i - 1), \alpha_t K\boldsymbol{I}), \tag{33}$$

where the Φ^i is the i^{th} component of Φ, which determines the weights for each gaussian, and $\boldsymbol{e}^i$ is the one-hot encode of class i. We use p^i as a simplification of $\mathcal{N}(\boldsymbol{y}|\alpha_t(K\boldsymbol{e}^i - 1), \alpha_t K\boldsymbol{I})$. Now we have:

$$D_{KL}(p_S(\boldsymbol{y}|\boldsymbol{g},t)||p_R(\boldsymbol{y}|\boldsymbol{\theta},t)) = \int p_S(x)\log\frac{p_S(x)}{\sum_{i=1}^{K}\Phi^i p^i(x)}dx \tag{34}$$

$$= \int p_S(x)\log(p_S(x))dx - \int p_S(x)\log(\sum_{i=1}^{K}\Phi^i p^i(x))dx \tag{35}$$

$$\leq \int p_S(x)\log(p_S(x))dx - \sum_{i=1}^{K}\Phi^i\int p_S(x)\log(p^i(x))dx \quad (Jensen's\ Inequality) \tag{36}$$

$$= \sum_{i=1}^{K}\Phi^i\int p_S(x)\log\frac{p_S(x)}{p^i(x)}dx \tag{37}$$

$$=^{(1)} C_t * \sum_{i\neq g}\Phi^i||\boldsymbol{e_g} - \boldsymbol{e^i}||^2 \tag{38}$$

where in (1) we again used the fact that the KL divergence between two Gaussains with the same variance is the square norm of their means, C_t is a constant depend on t. As a result, the KL divergence is bounded by the probability of generating the wrong class $\sum_{i\neq g}\Phi^i$. □

E Implementation Details

We implement the Bayesian Flow Network using EGNNs [47] within the PyTorch framework [48]. The latent invariant feature dimension k is set to 1 for QM9 and 2 for DRUG, significantly reducing

the atomic feature dimensionality. Following the implementations of GeoBFN [31], we only take atom charges as atomic features. For training the parameter network Φ, we configure EGNNs with 9 layers and 256 hidden features for QM9, and 6 layers with 256 hidden features for DRUG, both trained with a batch size of 64. The model employs SiLU activations and is trained until convergence. Across all experiments, we adopt the Adam optimizer [49] with a fixed learning rate of 10^{-4} as the default training configuration. The training process requires approximately 2000 epochs for QM9 and 20 epochs for DRUG using RTX 3090.

We provide more details on the implementation of proposed techniques. Regarding the geometric-informed prior, we randomly sample 10 molecules for each molecular size to construct the prior. We use the accuracy level at 0.85 in all evaluation, which achieves the highest uniqueness of 97% and molecule stability of 90.0%. Regarding the consistency parameter objective, we use exponentially increasing curriculum. In QM9, the discretization number of the time span N starts with 100 and is doubled every 80 training epochs. In GEOM-DRUG, N starts with 100 and is doubled every 40000 training iterations. We enforce consistency of the predicted mean for atoms coordinates and charges, and the predicted probability for atom types. As the proposed consistency parameter objective could significantly accelerate the adjustment of atomic features, we perform early stop at the second phase of sampling. Empirically, we found that the early-stop point for both QM9 and GEOM-DRUG datasets could be chosen within $[0.6, 0.8]$ to achieve balance between sample quality and efficiency. To further improve the sample quality of few-step generation, we employed the noise reduced sampling method propose in [3].

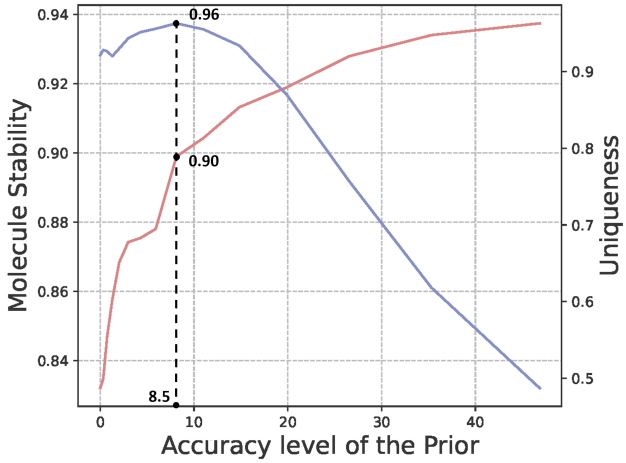

Figure 5: Ablation study on the accuracy level of geometric-inform prior.

For structure-based drug design, we follow previously established evaluation protocol.

- We use the CrossDocked dataset for training and testing, which originally contains 22.5 million protein-ligand pairs, and after the RMSD-based filtering and 30% sequence identity split by Luo et al [50], results in 100,000 training pairs and 100 test proteins. For each test protein, we sample 100 molecules for evaluation.
- For baselines, we consider TargetDiff [2], DecompDiff[51] and Molcraft[3], which are state-of-the-art methods in the fields.
- Evaluation metrics include Vina Score, a direct score of generated pose, Vina Min, which scores the optimized pose after a local minimization of energy, Drug-likeliness (QED) and synthetic accessibility (SA) ligand conformation. Sampling efficiency is evaluated by NFEs.

F Ablation Study

In this section, we present additional experimental results on QM9 and GEOM-DRUG to evaluate the impact of the number of sampling steps. As shown in Table 5, increasing the number of sampling steps enhances molecular stability and validity. Notably, with only 13 NFEs, MOLTD achieves

a record-breaking atom stability of 87.95% and a molecule stability of 8.13% on GEOM-DRUG. Furthermore, we did not observe a significant decline in the diversity or novelty of the generated molecules as the number of sampling steps increased.

Furthermore, we investigate the impact of varying accuracy levels in the geometric-informed prior, which determine the starting point of the generation. We focus on molecule stability and the uniqueness of generated molecules in the QM9 dataset, as presented in Figure 5. Since the prior is derived from ground truth molecules within the dataset, higher accuracy improves molecular stability. However, the uniqueness metric follows a concave trend with respect to accuracy level. At lower accuracy levels, the prior introduces representative structural information and a degree of randomness into the initial stages of generation, improving both stability and diversity compared to the fixed prior commonly used in GeoBFN [31]. Conversely, at higher accuracy levels, the prior may lead MOLTD to collapse onto a subset of molecules from which it was constructed. As shown in Figure 5, an accuracy level within $8 \sim 18$ strikes an optimal balance, achieving uniqueness above 90% and stability around 90%.

In Table 6, we evaluate the efficiency gain in terms of actual runtime. We calculate the actual runtime required to generate 1000 samples on the QM9 dataset, using a single RTX 3090 GPU with a batch size of 64:

Table 6: Comparison of models based on NFEs and Time.

Model	NFEs	Time (seconds)
MOLTD	12	4.86
GeoBFN	100	34.52
EquiFM	200	180
EDM	1000	760

As the results demonstrate, MOLTD achieves a sampling speed over 100x faster than diffusion-based models. Furthermore, the practical speed-up is even greater than what is reflected by the reduction in NFEs alone, as MolTD benefit from a more efficient implementation.

G Visualization

We provide a T-SNE visualizations of the representations produced by Frad [52] on QM9 in Figure 6. From the figure, it is evident that molecules with different sizes often have distinct modes in structures, which is reflected in their geometric representations learned by modern geometric encoders.

H Further discussion on the geometric-informed prior

In this section, we provide more discussion and analysis on the effect of geometric-informed prior.

We analysis the effect of distribution of molecule size. The geometric-informed priors are tailored to the number of atoms (N) in a molecule, and the number of molecules available for each size varies within different molecule datasets. And thus, with enlarged sample size, two factors contribute to improved generation quality:

- A Better Prior: A structural prior derived from more extensive data better captures the full spectrum of geometric features.
- A Better Generative Model: More training data yields a more robust generative model.

However, we highlight that our method could boost the generation efficiency, even with only 10 samples to construct the prior. We randomly sample 10 molecules for each N to create prior and generated 10000 molecules on QM9. As shown in Table 7, 10-sample prior provide similar acceleration effect as prior constructed form all training data. This result demonstrates the substantial practical value of the geometric-informed prior, even in data-scarce settings.

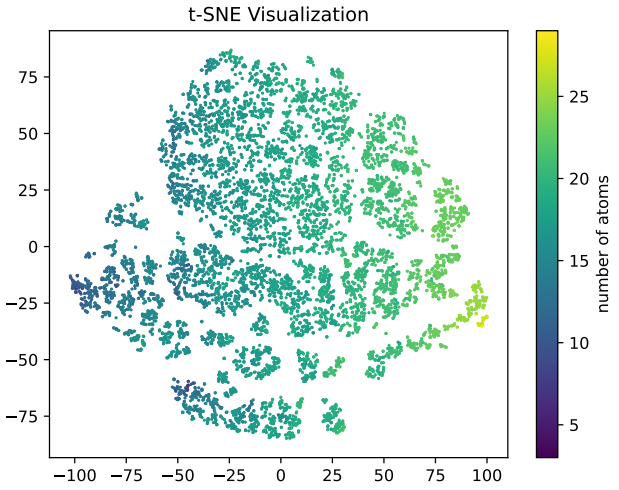

Figure 6: T-SNE visualizations of the representations produced by Frad. It is evident that molecules with different sizes often have distinct modes in structures, which is reflected in their geometric representations learned by modern geometric encoders.

Table 7: MOLTD using only 10 samples in the training set to construct geometric-informed prior

Method	NFE	Atom Stability	Mol Stability	Validity
GeoBFN	100	98.6	87.2	93.0
EquiFM	200	98.9	88.3	94.7
MolTD	12	99.4	92.53	96.04
MolTD (Prior using 10 samples)	12	99.1	90.1	95.3

I Limitation and Impact Statement

Although we demonstrate the effectiveness of our algorithm across datasets at different scale with comprehensive ablation studies, our focus is limited to the class of generative models based on iterative denoising processes. We do not consider auto-regressive or VAE-based models in this work. However, incorporating components from these architectures could potentially enhance the flexibility and performance of our approach, and we leave this as a promising direction for future research.

In this work, we propose MOLTD, a pioneering framework that achieves effective molecule generation in approximately 10 sampling steps, 7.5× faster than previous state-of-the-art methods. By reducing the sampling steps by an order of magnitude, our method paves a new way for scalable molecule generation, which may significantly enhance practical feasibility for real-world molecular design and drug discovery applications.

Furthermore, our systematic analysis of generative trajectories reveals fundamental limitations in conventional geometric diffusion paradigms, which drive our formulation of a phase-decoupled generation framework. We believe the insights gained from our exploration will inspire more researchers and bring more powerful methods to this field.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: In the experiments with QM9 and GEOM-DRUG datasets, we demonstrate our algorithm has superior efficiency and quality compared to other baselines.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitation of our work is discussed in Appendix I

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The assumptions of each theorem are sufficient, and the proofs are presented in Appendix D

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: In Appendix E, we provide implementation details for all of our experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: All data utilized in this study are open-access. The code will be publicly released upon paper acceptance.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: In Section 5.1 and Appendix E, we present the details of the experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We repeat the experiment on QM9 with multiple runs and reported the mean and variance of the considered metrics.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Discussion in Appendix E

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We follow Code of Ethics during the research.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Discussion in Appendix I

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The work does not present issues of high-risk misuse

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All data sources and baseline models are open-sourced. They have been properly credited and mentioned.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We do not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.