Sampling 3D Molecular Conformers with Diffusion Transformers

J. Thorben Frank[1,2*] **Winfried Ripken**[1,2*] **Gregor Lied**[1*]
Klaus-Robert Müller[1,2,3,4,5] **Oliver T. Unke**[3] **Stefan Chmiela**[1,2]

[1]Technical University Berlin [2]BIFOLD Berlin [3]Google DeepMind
[4]MPI for Informatics, Saarbrücken [5]Department of Artificial Intelligence, Korea University

thorbenjan.frank@gmail.com, oliverunke@google.com, stefan@chmiela.com

Abstract

Diffusion Transformers (DiTs) have demonstrated strong performance in generative modeling, particularly in image synthesis, making them a compelling choice for molecular conformer generation. However, applying DiTs to molecules introduces novel challenges, such as integrating discrete molecular graph information with continuous 3D geometry, handling Euclidean symmetries, and designing conditioning mechanisms that generalize across molecules of varying sizes and structures. We propose DiTMC, a framework that adapts DiTs to address these challenges through a modular architecture that separates the processing of 3D coordinates from conditioning on atomic connectivity. To this end, we introduce two complementary graph-based conditioning strategies that integrate seamlessly with the DiT architecture. These are combined with different attention mechanisms, including both standard non-equivariant and SO(3)-equivariant formulations, enabling flexible control over the trade-off between between accuracy and computational efficiency. Experiments on standard conformer generation benchmarks (GEOM-QM9, -DRUGS, -XL) demonstrate that DiTMC achieves state-of-the-art precision and physical validity. Our results highlight how architectural choices and symmetry priors affect sample quality and efficiency, suggesting promising directions for large-scale generative modeling of molecular structures. Code is available at `https://github.com/ML4MolSim/dit_mc`.

1 Introduction

The three-dimensional arrangement of atoms in a molecule, known as conformation, determines its biological activity and physical properties, making it fundamental to applications such as computational drug discovery and material design. Accurately predicting the most energetically favorable conformers (i.e., stable conformations) for large molecular systems is a highly non-trivial task. Traditional techniques, such as Molecular Dynamics and Markov Chain Monte Carlo, attempt to explore the conformational space by simulating physical movement or probabilistic sampling. However, these methods often require many simulation steps to move from one conformer to another, making them computationally expensive. Generative machine learning (ML) models offer a more targeted approach by allowing to directly sample from the space of promising conformations.

Recent years have seen significant progress, enabled by the development of specialized architectures for the generation of molecules [1–9] and materials [10–12]. This is in contrast to image and video synthesis, where the more generalized diffusion transformer (DiT) architecture [13] has

*Equal contribution.

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

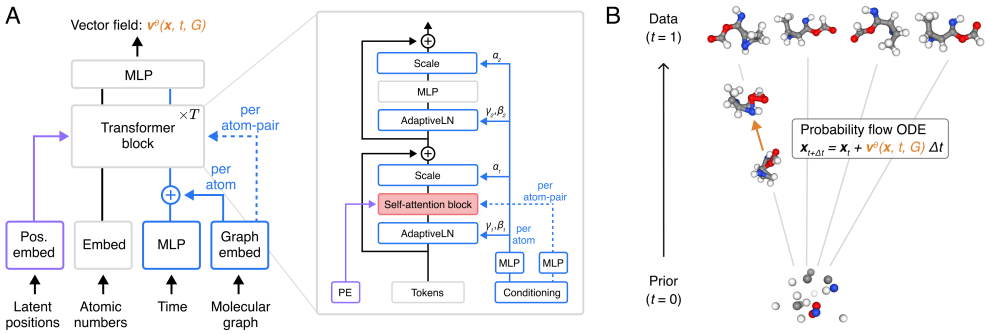

Figure 1: **(A)** Diffusion transformer for molecular conformer generation (DiTMC), with interchangeable self-attention blocks and positional embeddings (PEs); we evaluate various combinations as detailed in the main text. **(B)** DiTMC predicts a velocity per atom, used to model a probability flow ODE, which samples from the probability distribution $p(\boldsymbol{x}|\mathcal{G})$, where $\mathcal{G}$ is a molecular graph.

become a leading model, consistently delivering strong performance and efficiency across diverse applications [14–17]. Adapting DiTs, which were originally developed for grid-structured image data, to continuous, irregular molecular geometries poses unique challenges, which need to be addressed to unlock the potential of this powerful architecture for molecular conformer generation. Key design questions include how to encode molecular connectivity and incorporate Euclidean symmetries, such as translational and rotational invariance/equivariance.

In this work, we address these conceptual challenges and propose DiTMC, a new DiT-style architecture for molecular conformer generation. We introduce novel conditioning strategies based on molecular graphs, enabling the generation of 3D structures. Our modular architectural design allows us to systematically investigate the impact of different self-attention mechanisms within the DiT architecture on conformer generation quality and efficiency. We conduct a comparative study including standard (non-equivariant) self-attention with both absolute and relative positional embeddings and an explicitly $\mathrm{SO}(3)$-equivariant variant. While exact equivariance can positively impact performance, it also incurs significant computational costs. We find that simpler attention mechanisms are highly scalable and still perform competitively. Multiple of the tested DiTMC variants achieve state-of-the-art precision on established conformer generation benchmarks. Moreover, the molecular structure ensembles generated by our models align more closely with physical reality, as evidenced by the high accuracy of physical properties extracted from them. To summarize, our work contains the following main contributions:

- We propose two complementary conditioning strategies based on trainable conditioning tokens for (pairs of) atoms extracted from molecular graphs, which are designed to align with the architectural principles of DiTs. We propose to condition our self-attention formulation on geodesic graph distances extracted from molecular graphs and demonstrate that it significantly increases performance of our model.
- We investigate the impact of different self-attention mechanisms, including standard (non-equivariant) and SO(3)-equivariant formulations, on model accuracy and performance. We find that including symmetries can improve the fidelity of generated samples at the price of increased computational cost during training and inference.
- Based on our insights, we present a simple, non-equivariant, yet expressive DiT architecture that achieves state-of-the-art precision and physical validity on established benchmarks. Its performance improves with model scaling, making it a promising candidate for large-scale molecular conformer generation.

2 Related Work

Generative Modeling Generative models create diverse, high-quality samples from an unknown data distribution. They are widely used in image generation [18, 19], text synthesis [20] and the natural sciences [21–23]. Most recent approaches learn a probabilistic path from a simple prior to the data

distribution, enabling both efficient sampling and likelihood estimation [24–28]. This is achieved by different modeling paradigms, including flow matching [26] or denoising diffusion [24].

(Diffusion) Transformers Originally introduced for natural language processing [29], transformer architectures have become state-of-the-art in computer vision [18], and recently also found widespread adoption in quantum chemistry, e.g., for protein prediction [30, 31], 3D molecular generation [32, 33] or molecular dynamics simulations [34–36]. In the context of generative modeling, diffusion transformers [13] (DiTs) have emerged as powerful tools incorporating conditioning tokens, e.g., to prompt image generation [13] or to design molecules and materials with desirable properties [37]. This work applies prototypical DiTs to molecular conformer generation.

Molecular Conformer Generation Molecular conformer generation aims to find atomic arrangements (Cartesian coordinates) consistent with a given molecular graph. Numerous ML approaches have been proposed for sampling conformers, all aiming to improve upon conventional methods.

While early approaches were based on RDKit [38] or variational auto encoders [39–41], more recent advances employ diffusion and flow-based models [42, 43], including E-NFs [9], CGCF [44], GeoDiff [45], TorsionDiff [46], MCF [47], ET-Flow [32], and DMT [48].

De-Novo Molecular Generation Instead of conditioning on a given molecular graph, several existing approaches tackle the problem of finding novel and stable molecules given only a set of atoms [8, 49, 50]. Bridging the gap between unconditional and conformer generation, universal models have been proposed that can solve a multitude of tasks. In particular, models within the Uni-Mol model family [51, 52], make use of graph geodesic distances. However, their design requires tracking high-dimensional pair-representations throughout the architecture, which makes it less efficient.

3 Preliminaries

3.1 Molecular Conformers

A molecule can be represented as a molecular graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the nodes $\mathcal{V}$ correspond to atoms and the edges $\mathcal{E}$ represent chemical bonds between them. The nodes and edges contain information about their types, bond orders, and additional structural features such as branches, rings, and stereochemistry. The missing component is the exact spatial arrangement of the $N = |\mathcal{V}|$ atoms, represented as a 3D point cloud $\boldsymbol{x} \in \mathbb{R}^{N\times 3}$ in Euclidean space. Only the relative distances between atoms are relevant, as translating or rotating the entire point cloud $\boldsymbol{x}$ does not change the identity of the conformer. We frame conformer prediction as sampling from the $\mathrm{SE}(3)$-invariant conditional probability distribution $p(\boldsymbol{x} \,|\, \mathcal{G})$, which will guide the design of our model architectures in Sec. 4.

3.2 Conditional Flow Matching

Starting from an easy-to-sample base distribution $q_0 : \mathbb{R}^d \mapsto \mathbb{R}_{\geq 0}$, a generative process creates samples from a target distribution $q_1 : \mathbb{R}^d \mapsto \mathbb{R}_{\geq 0}$ [27]. Here, q_1 models the molecular conformer data with $d = N \times 3$. We aim to learn a *time-dependent vector field* $u_t(\boldsymbol{x}) : [0, 1] \times \mathbb{R}^{N\times 3} \mapsto \mathbb{R}^{N\times 3}$, which defines an ordinary differential equation (ODE) whose solution pushes samples $\boldsymbol{x}_0 \in \mathbb{R}^{N\times 3}$ from the prior to samples $\boldsymbol{x}_1 \in \mathbb{R}^{N\times 3}$ from the data distribution. We describe this transformation in terms of a *stochastic interpolant* $\boldsymbol{x}_t$ [26, 53, 54]. A noisy sample at time $t \in [0, 1]$ is defined as

$$\boldsymbol{x}_t = (1 - t) \cdot \boldsymbol{x}_0 + t \cdot \boldsymbol{x}_1 + \sigma \cdot \boldsymbol{\epsilon}, \tag{1}$$

where $\boldsymbol{\epsilon} \in \mathbb{R}^{N\times 3}$ is drawn from the standard normal distribution $\mathcal{N}(0, \boldsymbol{I})$ and scaled by a constant $\sigma \in \mathbb{R}_{\geq 0}$. We remark that t represents progress along this interpolation path, not physical time. Notably, stochastic interpolants enable transformations between arbitrary distributions and allow us to assess the performance of the generative process under varying prior distributions q_0. This contrasts with, e.g., score based diffusion methods [24, 55], which typically assume an isotropic Gaussian prior.

The stochastic interpolant induces a deterministic trajectory of densities $p_t(\boldsymbol{x})$, governed by an ODE known as the probability flow:

$$\mathrm{d}\boldsymbol{x} = u_t(\boldsymbol{x})\,\mathrm{d}t. \tag{2}$$

If the vector field $u_t(\boldsymbol{x})$ was tractable to sample, the weights of a neural network (NN) $\boldsymbol{v}^\theta(\boldsymbol{x}, t)$: $[0,1] \times \mathbb{R}^d \mapsto \mathbb{R}^d$ could be optimized directly by minimizing

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}(0,1),\boldsymbol{x}\sim p_t(\boldsymbol{x})} \left|\left| u_t(\boldsymbol{x}) - \boldsymbol{v}^\theta(\boldsymbol{x}, t) \right|\right|. \tag{3}$$

The learned vector field $\boldsymbol{v}^\theta$ could then be used to generate new samples from the target distribution by starting from $\boldsymbol{x}_0 \sim q_0$ and integrating the probability flow ODE (Eq. 2), for example, using a numerical scheme such as Euler's method, i.e., $\boldsymbol{x}_{t+\Delta t} = \boldsymbol{x}_t + \boldsymbol{v}^\theta(\boldsymbol{x}_t, t)\Delta t$ for time step Δt.

However, for arbitrary distributions q_0 and q_1, the objective in Eq. 3 is computationally intractable [56]. Instead, we consider the expectation over interpolated point pairs from the two distributions. Eq. 1 defines a *conditional probability distribution* $p_t(\boldsymbol{x}|\boldsymbol{x}_0, \boldsymbol{x}_1) = \mathcal{N}(\boldsymbol{x}|(1-t)\cdot\boldsymbol{x}_0 + t\cdot\boldsymbol{x}_1, \sigma^2)$, with *conditional vector field* $\boldsymbol{u}_t(\boldsymbol{x}|\boldsymbol{x}_0, \boldsymbol{x}_1) = \boldsymbol{x}_1 - \boldsymbol{x}_0$ [27]. The ability to directly sample from the conditional probability via Eq. 1 allows formulating the conditional flow matching (CFM) objective

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}(0,1),\boldsymbol{x}_0\sim q_0,\boldsymbol{x}_1\sim q_1,\boldsymbol{x}\sim p_t(\boldsymbol{x}|\boldsymbol{x}_0,\boldsymbol{x}_1)} \left|\left| \boldsymbol{u}_t(\boldsymbol{x}|\boldsymbol{x}_0, \boldsymbol{x}_1) - \boldsymbol{v}^\theta(\boldsymbol{x}, t) \right|\right|^2. \tag{4}$$

As shown in Ref. [26], the gradients of the two losses coincide, $\nabla_\theta\mathcal{L}_{\text{FM}} = \nabla_\theta\mathcal{L}_{\text{CFM}}$, thereby recovering the vector field that defines the probability flow ODE in Eq. 2. Following prior work [37, 57], we reparametrize the training objective to predict noise-free data $\boldsymbol{x}_0^\theta$ directly. During inference, we invert the reparametrization to obtain $\boldsymbol{v}^\theta$ for sampling (see Appendix C for details).

4 A New Diffusion Transformer for Molecular Conformer Sampling

We now describe the methodological advances of our work. We propose DiTMC, a new DiT-style architecture for conformer generation by learning a vector field using the loss in Eq. 4. As outlined in Sec. 3.1, this involves sampling from a conditional probability $p(\boldsymbol{x} \mid \mathcal{G})$, where $\mathcal{G}$ is the molecular graph representing atomic connectivity. Therefore, we choose our model to be a function $\boldsymbol{v}^\theta(\boldsymbol{x}, t, \mathcal{G})$, where the final output is a 3D velocity vector per atom, which is extracted from a readout layer.

Next, we outline the key components of the DiTMC architecture, with an overview shown in Fig. 1A. Training and architectural details are provided in Appendix B and Appendix D, respectively.

4.1 Conditioning Tokens

We begin by defining the conditioning tokens in DiTMC. Each DiTMC block receives a time conditioning token $\boldsymbol{c}^t \in \mathbb{R}^H$, as well as atom-wise conditioning tokens $\mathcal{C}^{\mathcal{G}}_{\text{atom}} = \{\boldsymbol{c}^{\mathcal{G}}_i \in \mathbb{R}^H \mid i \in [N]\}$, and pair-wise conditioning tokens $\mathcal{C}^{\mathcal{G}}_{\text{pair}} = \{\boldsymbol{c}^{\mathcal{G}}_{ij} \in \mathbb{R}^H \mid i, j \in [N]\}$ derived from $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.

Time conditioning The current time t of the latent state $\boldsymbol{x}_t$ is encoded via a two-layer MLP as

$$\boldsymbol{c}^t = \text{MLP}(t)\,. \tag{5}$$

Atom-wise conditioning Atom-wise graph conditioning tokens are obtained from a GNN inspired by the processor module of the MeshGraphNet (MGN) framework [58] as

$$\boldsymbol{c}^{\mathcal{G}}_i = \text{GNN}_{\text{node}}(\mathcal{V}, \mathcal{E})\,, \tag{6}$$

where $\boldsymbol{c}^{\mathcal{G}}_i$ denotes the final node representation for atom i. See Appendix D.3 for details on the GNN.

Pair-wise conditioning We define pair-wise graph conditioning tokens inspired by the Graphormer architecture [59] as

$$\boldsymbol{c}^{\mathcal{G}}_{ij} = \text{MLP}\left(s(i,j)\right), \tag{7}$$

where $s(i,j)$ denotes the graph geodesic, i.e., the shortest path between atoms i and j in $\mathcal{G}$. This formulation allows conditioning on all atom pairs, including those not directly connected by a bond.

4.2 Positional Embeddings

To encode the atomic positions $\mathcal{R} = \{\vec{r}_1, \ldots, \vec{r}_N \mid \vec{r}_i \in \mathbb{R}^3\}$ of the latent state $\boldsymbol{x}_t$, we use positional embeddings (PEs). We examine a representative range of positional embeddings that vary in the

number of Euclidean symmetries they respect by construction, which affects how the latent representations transform under translations and rotations. We denote the set of positional embeddings by $\mathcal{P}$, where $\mathcal{P} = \{\boldsymbol{p}_i \,|\, i \in [N]\}$ for the atom-wise (aPE) embeddings, and $\mathcal{P} = \{\boldsymbol{p}_{ij} \,|\, i, j \in [N],\, i \neq j\}$ for pair-wise (rPE or PE(3)) embeddings. Without loss of generality, we assume that the positions are centered such that the center of mass vanishes (see Appendix A).

Absolute Positional Embeddings Following Refs. [31, 37], atom-wise absolute Positional Embeddings (aPE) are calculated as

$$\boldsymbol{p}_i^{\text{aPE}} = \text{MLP}(\vec{r}_i)\,, \tag{8}$$

such that $\boldsymbol{p}_i^{\text{aPE}} \in \mathbb{R}^H$. This kind of positional embedding does not preserve rotationalal nor translationalal invariance, and serves as a baseline without any symmetry constraints.

Relative Positional Embeddings We use displacements vectors $\vec{r}_{ij} = \vec{r}_i - \vec{r}_j$ to build pairwise relative Positional Embeddings (rPE) as

$$\boldsymbol{p}_{ij}^{\text{rPE}} = \text{MLP}(\vec{r}_{ij})\,, \tag{9}$$

such that $\boldsymbol{p}_{ij}^{\text{rPE}} \in \mathbb{R}^H$. This formulation ensures translational invariance but not rotational invariance.

Euclidean Positional Embeddings Adapting ideas from equivariant message passing neural networks like PaiNN [60] or NequIP [61], we construct SO(3)-equivariant pairwise Euclidean Positional Embeddings (PE(3)) as a concatenation of $L+1$ components

$$\boldsymbol{p}_{ij}^{\text{PE(3)}} = \bigoplus_{\ell=0}^{L} \boldsymbol{\phi}_\ell(r_{ij}) \,\odot\, \boldsymbol{Y}_\ell(\hat{r}_{ij})\,, \tag{10}$$

where $\boldsymbol{\phi}_\ell : \mathbb{R} \mapsto \mathbb{R}^{1\times H}$ is a radial filter function, $\hat{r} = \vec{r}/r$, and $\boldsymbol{Y}_\ell \in \mathbb{R}^{(2\ell+1)\times 1}$ are spherical harmonics of degree $\ell = 0, \ldots, L$. The element-wise multiplication '$\odot$' between radial filters and spherical harmonics is understood to be "broadcasting" along axes with size 1, such that $(\boldsymbol{\phi}_\ell \,\odot\, \boldsymbol{Y}_\ell) \in \mathbb{R}^{(2\ell+1)\times H}$ and (after concatenation) $\boldsymbol{p}_{ij}^{\text{PE(3)}} \in \mathbb{R}^{(L+1)^2\times H}$. Under rotation of the input positions, these positional embeddings transform equivariantly (see Appendix E). Moreover, because displacement vectors are used as inputs, the embeddings are also invariant to translations. As a result, they respect the full set of Euclidean symmetries relevant to molecular geometry.

4.3 DiTMC Block

Based on the positional embedding strategies introduced in Sec. 4.2, we can define different DiTMC blocks that preserve the extent of Euclidean symmetries encoded in the embeddings throughout the model.

Each DiTMC block transforms a set of input tokens $\mathcal{H}$ into a set of output tokens $\mathcal{H}'$, which serve as input for the next block. For aPE and rPE, we have $\mathcal{H} = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \,|\, \boldsymbol{h}_i \in \mathbb{R}^H\}$, and for PE(3), we have $\mathcal{H} = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \,|\, \boldsymbol{h}_i \in \mathbb{R}^{(L+1)^2\times H}\}$.

In each DiTMC block, we inject time-based, as well as graph-based atom-wise and pair-wise conditioning information via the conditioning tokens introduced in Sec. 4.1. We use the pair-wise graph conditioning tokens $\mathcal{C}^{\mathcal{G}}_{\text{Pair}}$ during the self-attention update,

$$\boldsymbol{h}_i = \boldsymbol{h}_i + \text{ATT}(\mathcal{H}, \mathcal{P}, \mathcal{C}^{\mathcal{G}}_{\text{Pair}})_i\,, \tag{11}$$

where we employ a standard self-attention mechanism for aPE and rPE and an SO(3)-equivariant self-attention mechanism for PE(3). Additionally, we use the time conditioning token $\boldsymbol{c}^t$ and the atom-wise graph conditioning tokens $\mathcal{C}^{\mathcal{G}}_{\text{atom}} = \{\boldsymbol{c}^{\mathcal{G}}_i \in \mathbb{R}^H \,|\, i \in [N]\}$ to obtain per-atom bias and scaling parameters,

$$\alpha_{1i}, \beta_{1i}, \gamma_{1i}, \alpha_{2i}, \beta_{2i}, \gamma_{2i} = \text{MLP}(\boldsymbol{c}^t + \boldsymbol{c}^{\mathcal{G}}_i)\,, \tag{12}$$

and apply adaptive layer norm (AdaLN) and adaptive scale (AdaScale) [13] for conditioning (see Fig. 1A) similar to applications of DiTs in image synthesis.

We provide details on the non-equivariant DiTMC blocks based on aPE and rPE in Appendix D.1, and on the SO(3)-equivariant DiTMC block based on PE(3) in Appendix D.2.

Table 1: Results on GEOM-QM9 for different generative models (parameter counts in parentheses). -R indicates Recall, -P indicates Precision. Best results in **bold**, second best underlined; our models are marked with an asterisk (∗). Our results are averaged over three random seeds. See Appendix Tab. A9 for results including standard deviations.

Method	**COV-R** [%] ↑		**AMR-R** [Å] ↓		**COV-P** [%] ↑		**AMR-P** [Å] ↓	
	Mean	Median	Mean	Median	Mean	Median	Mean	Median
GeoMol (0.3M)	91.5	**100.0**	0.225	0.193	86.7	**100.0**	0.270	0.241
GeoDiff (1.6M)	76.5	**100.0**	0.297	0.229	50.0	33.5	0.524	0.510
Tors. Diff. (1.6M)	92.8	**100.0**	0.178	0.147	92.7	**100.0**	0.221	0.195
MCF-B (64M)	95.0	**100.0**	0.103	0.044	93.7	**100.0**	0.119	0.055
DMT-B (55M)	95.2	**100.0**	0.090	0.036	93.8	**100.0**	0.108	0.049
ET-Flow (8.3M)	**96.5**	**100.0**	0.073	0.030	94.1	**100.0**	0.098	0.039
∗DiTMC+aPE-B (9.5M)	96.1	**100.0**	0.073	0.030	95.4	**100.0**	0.085	0.037
∗DiTMC+rPE-B (9.6M)	96.3	**100.0**	0.070	0.027	**95.7**	**100.0**	**0.080**	0.035
∗DiTMC+PE(3)-B (8.6M)	95.7	**100.0**	**0.068**	**0.021**	93.4	**100.0**	0.089	**0.032**

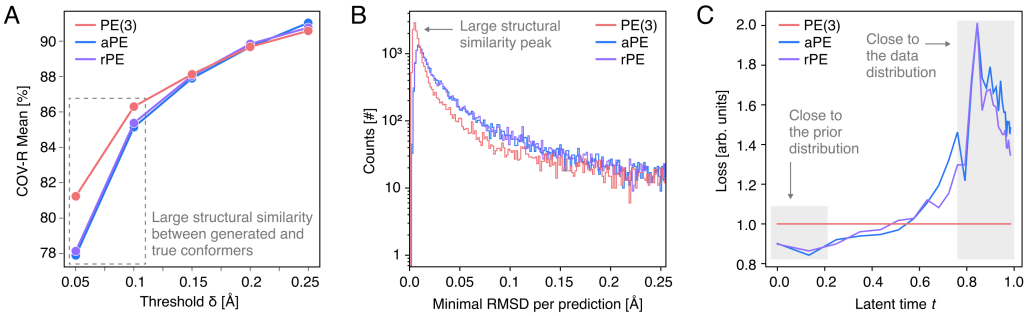

Figure 2: Analysis of SO(3)-equivariant (PE(3)) and non-equivariant (aPE, rPE) model formulations on GEOM-QM9. **(A)** Mean Coverage Recall (COV-R) versus root mean square deviation (RMSD) threshold δ to any reference conformer. **(B)** Histogram of the minimal RMSD per generated sample. **(C)** Loss as a function of latent time t relative to PE(3) loss (see Appendix J for details).

5 Experiments

Datasets and Metrics We conduct our experiments on the GEOM dataset [62], comprising QM9 (133,258 small molecules) and AICures (304,466 drug-like molecules). Reference conformers are generated using CREST [63]. Drug-like molecules exhibit greater structural diversity, including more rotatable bonds and multiple stereocenters. The data splits are taken from Ref. [64].

We evaluate our models' ability to generate accurate and diverse conformers using average minimum RMSD (AMR) and coverage (COV), measuring recall (ground-truth coverage) and precision (generation accuracy). A generated conformer is considered valid if it falls within a specified RMSD threshold of any reference conformer ($\delta = 0.5$Å for GEOM-QM9 and $\delta = 0.75$Å for GEOM-DRUGS). Following prior work, we generate $2K$ conformers per test molecule with K reference structures. Appendix G.3 provides further details on the calculation of metrics. Following ET-Flow [32] and GeomMol [64] we also apply chirality correction (see Appendix G.4).

Ablating Self-Attention and Positional Embedding Strategies The modular structure of DiTMC allows efficient exploration of the design space through variations in the positional embeddings and associated attention blocks (see Sec. 4). We define three model variants, DiTMC+aPE, DiTMC+rPE, and DiTMC+PE(3), which differ only in the choice of positional embedding and self-attention formulation. Architectural details can be found in Appendix D. On GEOM-QM9, our models produce diverse, high quality samples, outperforming the current state-of-the-art across all AMR-R, AMR-P, and COV-P metrics, demonstrating the broad applicability of our modular design and conditioning (see Tab. 1). We use the harmonic prior introduced in Ref. [65] throughout, which yields improved

Table 2: Ablation of conditioning strategies using DiTMC+aPE-B on GEOM-DRUGS. -R indicates Recall, -P indicates Precision. Best results in **bold**. Our results are averaged over three random seeds. See Appendix Tab. A14 for results on GEOM-QM9.

Method	**COV-R** [%]↑		**AMR-R** [Å]↓		**COV-P** [%]↑		**AMR-P** [Å]↓	
	Mean	Median	Mean	Median	Mean	Median	Mean	Median
No conditioning	16.6	4.7	1.132	1.110	5.15	1.2	1.849	1.845
Atom-wise	72.8	77.5	0.555	0.565	55.6	53.3	0.762	0.716
Atom-wise & Pair-wise	**79.9**	**85.4**	**0.434**	**0.389**	**76.5**	**83.6**	**0.500**	**0.423**

results compared to the Gaussian prior (see Appendix Tab. A13), and therefore adopt it in all subsequent experiments. Notably, our models maintain competitive performance even when using the Gaussian prior, contrary to the findings in Ref. [32].

Probing the effect of Euclidean symmetries Our PEs form a hierarchy based on the extent of Euclidean symmetry incorporated by construction. This enables a systematic evaluation of how incorporating symmetry affects model behavior. We summarize our findings on GEOM-QM9 below.

Equivariance improves the fidelity of samples. We analyze COV-R Mean as a function of RMSD threshold δ for all DiTMC variants (see Fig. 2A). The SO(3)-equivariant DiTMC+PE(3) outperforms the non-equivariant DiTMC+aPE and DiTMC+rPE at low δ, indicating that many of the generated conformers closely match the ground-truth structures. This appears as a leftward shift in the distribution of the minimal RMSD found per generated structure (as shown in Fig. 2B) and aligns with the observation that DiTMC+PE(3) achieves better AMR-R and AMR-P values (as reported in Tab. 1). To better understand this behavior, we examine the loss over time t and find that the non-equivariant models exhibit higher error near the data distribution ($t = 1$) (see Fig. 2C). The increase in error towards the end of the generation trajectory results in noisier structures and reduced fidelity.

Equivariance increases the computational cost for models of similar size. The benefit of higher fidelity comes at increased computational cost. During training, the equivariant DiTMC+PE(3) is approximately 3.5 times slower than the non-equivariant DiTMC+aPE and DiTMC+rPE, and about 3 times slower at inference (see Fig. 3B). All models use the same number of layers and differ only in the number of heads per layer in order to match the total parameter count, as discussed in Appendix D.

Conditioning Strategies To assess the impact of graph conditioning, we compare different conditioning strategies. As further discussed in Appendix I.2, we compare conditioning solely on atom-wise information (Eq. 6) with an extended scheme that also incorporates pair-wise geodesic graph distances (Eq. 7). In Tab. 2, we show ablation results on GEOM-DRUGS. We also ablate conditioning on pairwise information extracted from our conditioning GNN for each edge corresponding to a chemical bond (see Appendix I.2). We find all variants to be effective, but our proposed combination of geodesic distances and atom-wise information to perform best. This experiment underlines the importance of deriving conditioning tokens for *all* atom pairs, not just those connected by edges in the molecular graph (i.e. by chemical bonds), which lack global information about the graph structure.

Model Scaling To investigate model scaling, we define a small ("S"), base ("B"), and large ("L") DiTMC+aPE model variant (see Appendix D). Each model is trained with identical training hyperparameters on the GEOM-DRUGS dataset. We find strong relative improvements for all metrics (up to 54.1% for COV-P Mean) when scaling DiTMC+aPE-S to DiTMC+aPE-B (see Fig. 3A). Scaling DiTMC+aPE-B to DiTMC+aPE-L yields relative improvements, which however might be smaller than expected given the strong increase in performance from the small to the base model. Prior work has shown that to avoid diminishing returns, dataset and model sizes must be scaled together [66, 67]. Scaling only the model yields limited benefits as soon as model size saturates given the amount of data. Strong improvements of DiTMC+aPE-L over DiTMC+aPE-B are observed in terms of sampling fidelity (as indicated by higher COV-R and COV-P Mean values at small thresholds δ) but differences start to vanish at $\delta = 0.75$Å (see Fig. 4A). Moreover, we obtain relative improvement in terms of generalization to larger and previously unseen molecules from the GEOM-XL dataset [46] between 3.1% (for AMR-P Mean) and 7.3% (for AMR-P Median), i.e., DiTMC+aPE-L exhibits significantly stronger out-of-distribution performance compared to the smaller DiTMC+aPE-B (see Tab. 5).

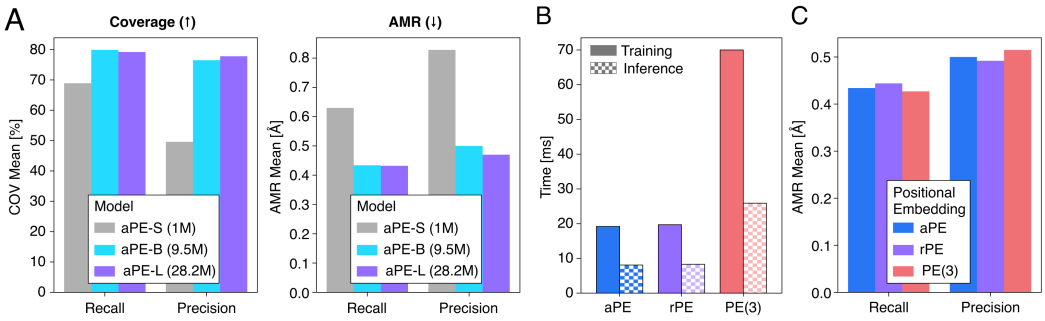

Figure 3: **(A)** Coverage (COV) mean and Absolute Minimum RMSD (AMR) mean for DiTMC+aPE models of increasing model capacity on GEOM-DRUGS. **(B)** Training and inference time for different positional embedding (PE) and associated self-attention strategies. **(C)** Recall and precision AMR mean for the different DiTMC models on GEOM-DRUGS. For panels (B) and (C) we use results from the base ("B") variant of each model.

Table 3: Results on GEOM-DRUGS for different generative models (parameter counts in parentheses). -R indicates Recall, -P indicates Precision. Best results in **bold**, second best underlined; our models are marked with an asterisk (∗). Our results are averaged over three random seeds. See Appendix Tab. A10 for results including standard deviations.

Method	**COV-R** [%]↑		**AMR-R** [Å]↓		**COV-P** [%]↑		**AMR-P** [Å]↓	
	Mean	Median	Mean	Median	Mean	Median	Mean	Median
GeoMol (0.3M)	44.6	41.4	0.875	0.834	43.0	36.4	0.928	0.841
GeoDiff (1.6M)	42.1	37.8	0.835	0.809	24.9	14.5	1.136	1.090
Tors. Diff. (1.6M)	72.7	80.0	0.582	0.565	55.2	56.9	0.778	0.729
MCF-L (242M)	84.7	92.2	0.390	**0.247**	66.8	71.3	0.618	0.530
DMT-L (150M)	**85.8**	**92.3**	**0.375**	0.346	67.9	72.5	0.598	0.527
ET-Flow - SS (8.3M)	79.6	84.6	0.439	0.406	75.2	81.7	0.517	0.442
∗DiTMC+aPE-B (9.5M)	79.9	85.4	0.434	0.389	76.5	83.6	0.500	0.423
∗DiTMC+rPE-B (9.6M)	79.3	84.6	0.444	0.400	77.2	84.6	0.492	0.414
∗DiTMC+PE(3)-B (8.6M)	80.8	85.6	0.427	0.396	75.3	82.0	0.515	0.437
∗DiTMC+aPE-L (28.2M)	79.2	84.4	0.432	0.386	77.8	85.7	0.470	0.387
∗DiTMC+rPE-L (28.3M)	78.7	84.1	0.438	0.388	**78.1**	**86.4**	**0.466**	**0.381**
∗DiTMC+PE(3)-L (31.1M)	80.8	85.6	0.415	0.376	76.4	82.6	0.491	0.414

Drug-like Molecules For the following experiments, we restrict our analysis to the base ("B") and large ("L") DiTMC variants. In contrast to the smaller GEOM-QM9 dataset, the size of GEOM-DRUGS allows us to explore the design space of DiTMC under a more realistic setting, where training is contstrained by a fixed compute budget [67]. Our budget of 9 GPU days allows training DiTMC+aPE-L and DiTMC+rPE-L for 50 epochs, and DiTMC+PE(3)-L for 10 epochs. For consistency, we train the base models for the same number of epochs as their large counterparts. All DiTMC variants achieve state-of-the-art performance on GEOM-DRUGS for COV-P and AMR-P (see Tab. 3). Importantly, even the smaller DiTMC+aPE-B and DiTMC+PE(3)-B models outperform ET-Flow-SS of similar model size across all metrics, underlining the effectiveness of our approach.

Coverage vs. RMSD Threshold. We further analyse the COV-R Mean and COV-P Mean as a function of RMSD threshold δ for DiTMC+aPE-B and DiTMC+aPE-L (see Fig. 4A). For small thresholds ($\delta < 0.4$Å) both DiTMC+aPE models outperform all other methods for coverage recall and precision. For larger thresholds ($\rho \geq 0.4$Å) MCF-L starts to outperform aPE models in terms of COV-R Mean. For COV-P Mean, DiTMC+aPE-B and DiTMC+aPE-L perform better than all other methods for all reasonably small thresholds ($\delta < 1.2$Å). In particular for the most relevant regime where thresholds are small, we see a strong benefit due to model scaling. We find similar results for DiTMC+rPE and DiTMC+PE(3) (see Appendix Fig. A9 and Fig. A10).

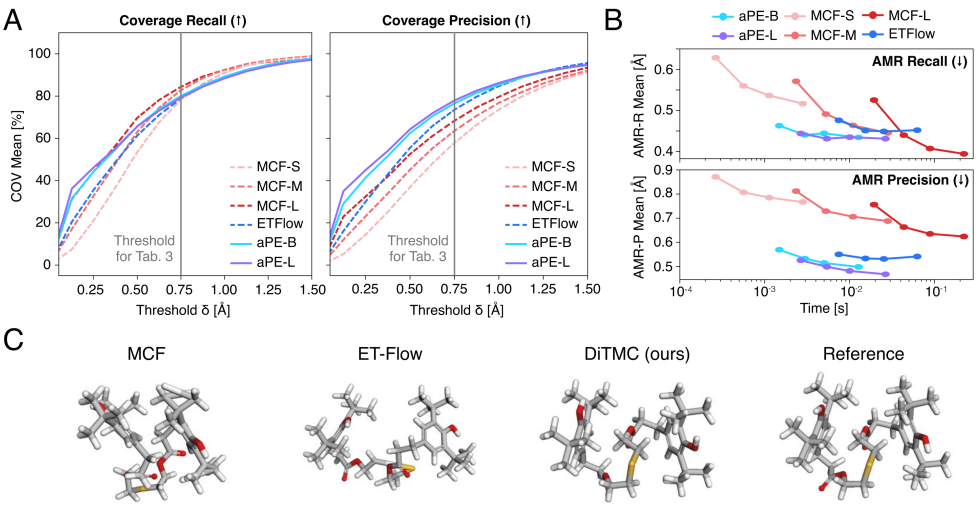

Figure 4: **(A)** Coverage (COV) mean as function of Root Mean Square Deviation (RMSD) threshold δ and **(B)** average minimum RMSD (AMR) mean vs. time per conformer for DiTMC+aPE and other state-of-the-art models. Per model markers from left to right correspond to 5, 10, 20, and 50 sampler steps following Refs. [32, 47]. Note, that the original MCF paper reports results with two different samplers. Benchmark results (Tab. 3) are obtained with DDPM sampler (1000 steps) and AMR vs. time results are reported for DDIM sampler (5–50 steps). **(C)** Comparison of conformers generated by MCF, ET-Flow, and DiTMC against ground-truth reference conformers from GEOM-XL. Generated conformers are rotationally aligned with their corresponding reference conformer.

Pareto Front. Additionally, we investigate the Pareto front of accuracy and computational efficiency, by plotting model accuracy as a function of wall clock time per generated conformer, following Refs. [32, 47] (see Appendix G.5 for details). As measure for accuracy we consider the average minimum RMSD (AMR), since it is independent of the RMSD threshold. We report results of DiTMC+aPE-B and DiTMC+aPE-L for AMR-R Mean and AMR-P Mean in Fig. 4B. For AMR-P Mean, both DiTMC+aPE models shift the whole Pareto front, yielding higher accuracy at lower computational cost. Even higher accuracies (at the cost of compute time) can be obtained by scaling DiTMC+aPE-B to DiTMC+aPE-L. For recall, both DiTMC+aPE models shift the Pareto front for little compute times, but most accurate results at increased cost are obtained by MCF-L. Similar results are obtained for DiTMC+rPE (see Appendix Fig. A12), but benefits for DiTMC+PE(3) are limited due to high computational cost of equivariant operations (see Appendix Fig. A13).

Ensemble Properties. To complement the RMSD-based geometric evaluation, we report the median absolute error (MAE) of ensemble properties between generated and reference conformers, following the protocol of MCF [47] and ET-Flow [32] (see Appendix G.6 for details). Our models predict ensemble properties more accurately than all baselines, highlighting the physical validity of our generated structures (see Tab. 4). In particular, MCF, which shows better performance for recall metrics, is outperformed by a large margin (up to a factor of four for energy). The strong performance of the aPE-B and aPE-L models underlines the potential of achieving high physical validity without any geometric priors, which have been hypothesized to be one of the reasons for ET-Flow outperforming MCF in the ensemble property task [32].

Generalization Performance Finally, we assess how well our models, trained on GEOM-DRUGS, generalize to larger and previously unseen molecules using the GEOM-XL dataset [46]. It comprises 102 molecules with more than 100 atoms, whereas the molecules in the training set contain only 44 atoms on average. Following MCF [47] we report results for all 102 molecules and a subset of 77 molecules. ET-Flow uses a slightly different subset of 75 molecules. Our aPE models perform on par with or better than the previously best-performing method MCF-L (see Tab. 5), while using approximately 8 times and 25 times fewer parameters for aPE-L and aPE-B, respectively. The other baselines, such as ET-Flow, are outperformed by a larger margin (see Appendix Tab. A12).

Table 4: Median absolute error of ensemble properties between generated and reference conformers for GEOM-DRUGS. Best results in **bold**, second best underlined; our models are marked with an asterisk (∗). Results for MCF, ET-Flow, and ours are averaged over three random seeds. See Appendix Tab. A11 for additional results including standard deviations.

Method	E **[kcal/mol]**↓	μ **[D]**↓	$\Delta\epsilon$ **[kcal/mol]**↓	E_{min} **[kcal/mol]**↓
MCF-L (242M)	0.68	0.28	0.63	0.04
ET-Flow (8.3M)	0.18	0.18	0.35	0.02
∗DiTMC+aPE-B (9.5M)	0.17	0.16	**0.27**	**0.01**
∗DiTMC+aPE-L (28.2M)	**0.16**	**0.14**	**0.27**	**0.01**

Table 5: Out-of-distribution generalization results on GEOM-XL for models trained on GEOM-DRUGS. -R indicates Recall, -P indicates Precision. Best results in **bold**, second best underlined; our models are marked with an asterisk (∗). Our results are averaged over three random seeds. See Appendix Tab. A12 for additional results including standard deviations.

Method	*75 / 77 mols*				*102 mols*			
	AMR-R [Å]↓		**AMR-P** [Å]↓		**AMR-R** [Å]↓		**AMR-P** [Å]↓	
	Mean	Median	Mean	Median	Mean	Median	Mean	Median
MCF-L (242M)	1.64	1.51	2.57	2.26	1.97	1.60	2.94	2.43
ET-Flow (8.3M)	2.00	1.80	2.96	2.63	2.31	1.93	3.31	2.84
∗DiTMC+aPE-B (9.5M)	1.68	1.47	2.59	2.24	1.96	1.60	2.90	2.48
∗DiTMC+aPE-L (28.2M)	**1.56**	**1.28**	**2.47**	**2.14**	**1.88**	**1.51**	**2.81**	**2.30**

6 Summary and Limitations

We propose a framework for molecular conformer generation that incorporates conditioning strategies tailored to the architectural design principles of DiTs. This modular framework enables a rigorous exploration of different positional embedding and self-attention strategies, allowing us to identify scalable generative architectures that perform competitively with prior methods on standard benchmarks. Our models achieve state-of-the-art results on GEOM-QM9 and GEOM-DRUGS, excelling in both precision and physical validity, and show strong generalization to larger, previously unseen molecules from the GEOM-XL dataset. Exemplary generations for GEOM-XL are shown in Fig. 4C, with additional samples provided in Appendix M.

Through ablation studies, we assess the impact of incorporating Euclidean symmetries into DiTMC. While such symmetries improve performance, they also increase computational cost. Notably, simpler non-equivariant variants remain highly effective. These findings allow us to develop an efficient, accurate, and scalable architecture suitable for large-scale conformer generation.

Nonetheless, some limitations persist. Our evaluation is currently restricted to small and medium-sized molecules, with larger, more flexible compounds left for future work. Moreover, the training process depends on high-quality ground-truth conformers, which may be unavailable in some cases. An interesting direction for future work, is extending the scope of our model beyond conformer generation, for example to de-novo molecular generation [8, 22, 50]. However, this would require a re-design of our proposed conditioning strategies, as those are specifically tailored to molecular graphs as inputs. In terms of computational cost, DiTMC scales quadratic in the number of atoms, resulting in large computational cost for bigger systems. Therefore, integrating recently developed Euclidean fast attention mechanisms [68], poses an interesting direction for further research. Finally, while our analysis advances understanding of equivariance within DiT-based generative models, drawing broader conclusions would require further study.

7 Acknowledgements

JTF, WR, KRM, and SC acknowledge support by the German Ministry of Education and Research (BMBF) for BIFOLD (01IS18037A). Further, this work was in part supported by the BMBF under Grants 01IS14013A-E, 01GQ1115, 01GQ0850, 01IS18025A, 031L0207D, and 01IS18037A. KRM was partly supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grants funded by the Korea government (MSIT) (No.2019-0-00079, Artificial Intelligence Graduate School Program, Korea University and No. 2022-0-00984, Development of Artificial Intelligence Technology for Personalized Plug-and-Play Explanation and Verification of Explanation). We further want to thank Romuald Elie, Michael Plainer, Adil Kabylda, Khaled Kahouli, Stefan Gugler, Martin Michajlow, Christoph Bornett, Johannes Maeß, Leon Werner and Maximilian Eißler for helpful discussion.

References

[1] 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.

[2] 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.

[3] Matthew Ragoza, Tomohide Masuda, and David Ryan Koes. Learning a continuous representation of 3d molecular structures with deep generative models. *arXiv preprint arXiv:2010.08687*, 2020.

[4] Niklas WA Gebauer, Michael Gastegger, Stefaan SP Hessmann, Klaus-Robert Müller, and Kristof T Schütt. Inverse design of 3d molecular structures with conditional generative neural networks. *Nature communications*, 13(1):973, 2022.

[5] 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.

[6] Martin Simonovsky and Nikos Komodakis. Graphvae: Towards generation of small graphs using variational autoencoders. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27*, pages 412–422. Springer, 2018.

[7] Nicola De Cao and Thomas Kipf. Molgan: An implicit generative model for small molecular graphs. *arXiv preprint arXiv:1805.11973*, 2018.

[8] 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.

[9] 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.

[10] Rui Jiao, Wenbing Huang, Peijia Lin, Jiaqi Han, Pin Chen, Yutong Lu, and Yang Liu. Crystal structure prediction by joint equivariant diffusion. *Advances in Neural Information Processing Systems*, 36:17464–17497, 2023.

[11] Giuseppe Vecchio, Renato Sortino, Simone Palazzo, and Concetto Spampinato. Matfuse: controllable material generation with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4429–4438, 2024.

[12] Claudio Zeni, Robert Pinsler, Daniel Zügner, Andrew Fowler, Matthew Horton, Xiang Fu, Zilong Wang, Aliaksandra Shysheya, Jonathan Crabbé, Shoko Ueda, et al. A generative model for inorganic materials design. *Nature*, pages 1–3, 2025.

[13] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4195–4205, October 2023.

[14] Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22669–22679, 2023.

[15] Shoufa Chen, Mengmeng Xu, Jiawei Ren, Yuren Cong, Sen He, Yanping Xie, Animesh Sinha, Ping Luo, Tao Xiang, and Juan-Manuel Perez-Rua. Gentron: Diffusion transformers for image and video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6441–6451, 2024.

[16] Yixin Liu, Kai Zhang, Yuan Li, Zhiling Yan, Chujie Gao, Ruoxi Chen, Zhengqing Yuan, Yue Huang, Hanchi Sun, Jianfeng Gao, Lifang He, and Lichao Sun. Sora: A review on background, technology, limitations, and opportunities of large vision models. *CoRR*, abs/2402.17177, 2024. URL `https://doi.org/10.48550/arXiv.2402.17177`.

[17] Xin Ma, Yaohui Wang, Xinyuan Chen, Gengyun Jia, Ziwei Liu, Yuan-Fang Li, Cunjian Chen, and Yu Qiao. Latte: Latent diffusion transformer for video generation. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856.

[18] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=YicbFdNTTy`.

[19] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[20] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10696–10706, 2022.

[21] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.

[22] Amira Alakhdar, Barnabas Poczos, and Newell Washburn. Diffusion models in de novo drug design. *Journal of Chemical Information and Modeling*, 64(19):7238–7256, 2024.

[23] Tim Hsu, Babak Sadigh, Vasily Bulatov, and Fei Zhou. Score dynamics: Scaling molecular dynamics with picoseconds time steps via conditional diffusion model. *Journal of Chemical Theory and Computation*, 20(6):2335–2348, 2024.

[24] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf`.

[25] Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 21696–21707. Curran Associates, Inc., 2021. URL `https://proceedings.neurips.cc/paper_files/paper/2021/file/b578f2a52a0229873fefc2a4b06377fa-Paper.pdf`.

[26] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=PqvMRDCJT9t`.

[27] Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

[28] Diederik P Kingma and Ruiqi Gao. Understanding diffusion objectives as the ELBO with simple data augmentation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=NnMEadcdyD.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[30] 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*, 630(8016):493–500, 2024.

[31] Tomas Geffner, Kieran Didi, Zuobai Zhang, Danny Reidenbach, Zhonglin Cao, Jason Yim, Mario Geiger, Christian Dallago, Emine Kucukbenli, Arash Vahdat, and Karsten Kreis. Proteina: Scaling flow-based protein structure generative models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=TVQLu34bdw.

[32] Majdi Hassan, Nikhil Shenoy, Jungyoon Lee, Hannes Stärk, Stephan Thaler, and Dominique Beaini. Et-flow: Equivariant flow-matching for molecular conformer generation. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 128798–128824. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/e8bd617e7dd0394ceadf37b4a7773179-Paper-Conference.pdf.

[33] Sarah Lewis, Tim Hempel, José Jiménez-Luna, Michael Gastegger, Yu Xie, Andrew YK Foong, Victor García Satorras, Osama Abdin, Bastiaan S Veeling, Iryna Zaporozhets, et al. Scalable emulation of protein equilibrium ensembles with generative deep learning. *bioRxiv*, pages 2024–12, 2024.

[34] Philipp Thölke and Gianni De Fabritiis. Equivariant transformers for neural network based molecular potentials. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=zNHzqZ9wrRB.

[35] J Thorben Frank, Oliver T Unke, Klaus-Robert Müller, and Stefan Chmiela. A euclidean transformer for fast and stable machine learned force fields. *Nature Communications*, 15(1): 6539, 2024.

[36] Yi-Lun Liao, Brandon M Wood, Abhishek Das, and Tess Smidt. Equiformerv2: Improved equivariant transformer for scaling to higher-degree representations. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=mCOBKZmrzD.

[37] Chaitanya K Joshi, Xiang Fu, Yi-Lun Liao, Vahe Gharakhanyan, Benjamin Kurt Miller, Anuroop Sriram, and Zachary W Ulissi. All-atom diffusion transformers: Unified generative modelling of molecules and materials. *arXiv preprint arXiv:2503.03965*, 2025.

[38] Sereina Riniker and Gregory A Landrum. Better informed distance geometry: using what we know to improve conformation generation. *Journal of chemical information and modeling*, 55 (12):2562–2574, 2015.

[39] Elman Mansimov, Omar Mahmood, Seokho Kang, and Kyunghyun Cho. Molecular geometry prediction using a deep generative graph neural network. *Scientific reports*, 9(1):20381, 2019.

[40] Gregor NC Simm and José Miguel Hernández-Lobato. A generative model for molecular distance geometry. *arXiv preprint arXiv:1909.11459*, 2019.

[41] Minkai Xu, Wujie Wang, Shitong Luo, Chence Shi, Yoshua Bengio, Rafael Gomez-Bombarelli, and Jian Tang. An end-to-end framework for molecular conformation generation via bilevel programming. In *International conference on machine learning*, pages 11537–11547. PMLR, 2021.

[42] Shitong Luo, Chence Shi, Minkai Xu, and Jian Tang. Predicting molecular conformation via dynamic graph score matching. *Advances in neural information processing systems*, 34: 19784–19795, 2021.

[43] 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.

[44] Minkai Xu, Shitong Luo, Yoshua Bengio, Jian Peng, and Jian Tang. Learning neural generative dynamics for molecular conformation generation. *arXiv preprint arXiv:2102.10240*, 2021.

[45] Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. Geodiff: A geometric diffusion model for molecular conformation generation. *arXiv preprint arXiv:2203.02923*, 2022.

[46] Bowen Jing, Gabriele Corso, Jeffrey Chang, Regina Barzilay, and Tommi Jaakkola. Torsional diffusion for molecular conformer generation. *Advances in neural information processing systems*, 35:24240–24253, 2022.

[47] Yuyang Wang, Ahmed A Elhag, Navdeep Jaitly, Joshua M Susskind, and Miguel Angel Bautista. Swallowing the bitter pill: Simplified scalable conformer generation. *arXiv preprint arXiv:2311.17932*, 2023.

[48] Zhiyuan Liu, Yanchen Luo, Han Huang, Enzhi Zhang, Sihang Li, Junfeng Fang, Yaorui Shi, Xiang Wang, Kenji Kawaguchi, and Tat-Seng Chua. NExt-mol: 3d diffusion meets 1d language modeling for 3d molecule generation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=p66a00KLWN.

[49] Lei Huang, Hengtong Zhang, Tingyang Xu, and Ka-Chun Wong. Mdm: Molecular diffusion model for 3d molecule generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 5105–5112, 2023.

[50] 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.

[51] Gengmo Zhou, Zhifeng Gao, Qiankun Ding, Hang Zheng, Hongteng Xu, Zhewei Wei, Linfeng Zhang, and Guolin Ke. Uni-mol: A universal 3d molecular representation learning framework. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=6K2RM6wVqKu.

[52] Xiaohong Ji, Zhen Wang, Zhifeng Gao, Hang Zheng, Linfeng Zhang, Guolin Ke, and Weinan E. Exploring molecular pretraining model at scale. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2025. Curran Associates Inc. ISBN 9798331314385.

[53] Michael Samuel Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=li7qeBbCR1t.

[54] Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

[55] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

[56] 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. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=CD9Snc73AW. Expert Certification.

[57] Jason Yim, Andrew Campbell, Andrew YK Foong, Michael Gastegger, José Jiménez-Luna, Sarah Lewis, Victor Garcia Satorras, Bastiaan S Veeling, Regina Barzilay, Tommi Jaakkola, et al. Fast protein backbone generation with se (3) flow matching. *arXiv preprint arXiv:2310.05297*, 2023.

[58] Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. Learning mesh-based simulation with graph networks. In *International conference on learning representations*, 2020.

[59] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 28877–28888. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/f1c1592588411002af340cbaedd6fc33-Paper.pdf.

[60] Kristof Schütt, Oliver Unke, and Michael Gastegger. Equivariant message passing for the prediction of tensorial properties and molecular spectra. In *International Conference on Machine Learning*, pages 9377–9388. PMLR, 2021.

[61] Simon Batzner, Albert Musaelian, Lixin Sun, Mario Geiger, Jonathan P Mailoa, Mordechai Kornbluth, Nicola Molinari, Tess E Smidt, and Boris Kozinsky. E (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nat. Commun.*, 13(1):2453, 2022.

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

[63] Philipp Pracht, Stefan Grimme, Christoph Bannwarth, Fabian Bohle, Sebastian Ehlert, Gereon Feldmann, Johannes Gorges, Marcel Müller, Tim Neudecker, Christoph Plett, et al. Crest—a program for the exploration of low-energy molecular chemical space. *The Journal of Chemical Physics*, 160(11), 2024.

[64] Octavian Ganea, Lagnajit Pattanaik, Connor Coley, Regina Barzilay, Klavs Jensen, William Green, and Tommi Jaakkola. Geomol: Torsional geometric generation of molecular 3d conformer ensembles. *Advances in Neural Information Processing Systems*, 34:13757–13769, 2021.

[65] Hannes Stark, Bowen Jing, Regina Barzilay, and Tommi Jaakkola. Harmonic prior self-conditioned flow matching for multi-ligand docking and binding site design. In *NeurIPS 2023 AI for Science Workshop*, 2023. URL https://openreview.net/forum?id=3WF88uMjGz.

[66] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[67] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[68] J Thorben Frank, Stefan Chmiela, Klaus-Robert Müller, and Oliver T Unke. Euclidean fast attention: Machine learning global atomic representations at linear cost. *arXiv preprint arXiv:2412.08541*, 2024.

[69] Jason Yim, Brian L. Trippe, Valentin De Bortoli, Emile Mathieu, Arnaud Doucet, Regina Barzilay, and Tommi Jaakkola. SE(3) diffusion model with application to protein backbone generation. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan

Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 40001–40039. PMLR, 2023.

[70] Jonas Köhler, Leon Klein, and Frank Noe. Equivariant flows: Exact likelihood generative learning for symmetric densities. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5361–5370. PMLR, 13–18 Jul 2020.

[71] Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Foundations of Crystallography*, 32(5):922–923, 1976.

[72] Oliver T. Unke and Hartmut Maennel. E3x: E(3)-equivariant deep learning made easy. *CoRR*, abs/2401.07595, 2024. URL `https://doi.org/10.48550/arXiv.2401.07595`.

[73] Greg Landrum et al. Rdkit: A software suite for cheminformatics, computational chemistry, and predictive modeling. *Greg Landrum*, 8(31.10):5281, 2013.

[74] Lagnajit Pattanaik, Octavian-Eugen Ganea, Ian Coley, Klavs F Jensen, William H Green, and Connor W Coley. Message passing networks for molecules with tetrahedral chirality. *arXiv preprint arXiv:2012.00094*, 2020.

[75] Christoph Bannwarth, Sebastian Ehlert, and Stefan Grimme. Gfn2-xtb—an accurate and broadly parametrized self-consistent tight-binding quantum chemical method with multipole electrostatics and density-dependent dispersion contributions. *Journal of Chemical Theory and Computation*, 15(3):1652–1671, 2019. doi: 10.1021/acs.jctc.8b01176. URL `https://doi.org/10.1021/acs.jctc.8b01176`. PMID: 30741547.

1. **Claims**

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

   Answer: [Yes]

   Justification: [NA]

   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: [NA]

   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: [NA]

   Justification: [NA]

   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: [NA]

   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: [NA]

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: [NA]

   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: [NA]

   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: [NA]

   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: [NA]

   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: [No]

    Justification: While it can be expected that the general field of AI4Science will have societal impact, our work is focused on the development of a new deep learning architecture in this field. To have a direct societal impact, further studies under realistic applications, i.e. wet-lab experiments would be required.

    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: [NA]

    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: [NA]

    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: [NA]

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: [NA]

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: [NA]

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: [NA]

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.

A From SE(3) to SO(3) Invariance

The target data distribution of molecular conformers $p_1(\boldsymbol{x})$ is $\mathrm{SE}(3)$-invariant, i.e. it does not change under translations and rotations of the input. Following Ref. [69], one can define an $\mathrm{SE}(3)$-invariant measure on $\mathrm{SE}(3)^N$ by keeping the center of mass fixed at zero, which can be achieved via the centering operation from Eq. A15. This defines a subgroup $\mathrm{SE}(3)_0^N$, called centered $\mathrm{SE}(3)$. It can then be shown, that one can define an $\mathrm{SE}(3)$-invariant measure on $\mathrm{SE}(3)_0^N$ by constructing an $\mathrm{SO}(3)$-invariant (rotationally invariant) measure on $\mathrm{SE}(3)_0^N$.

As a consequence, it is then sufficient to learn an $\mathrm{SO}(3)$-equivariant vector field on the space of centered input positions (see also Ref. [70]). This is achieved by centering $\boldsymbol{x}_0$, $\boldsymbol{x}_1$ and $\boldsymbol{z}$ for the calculation of the interpolant. Moreover, the neural network output (predicted velocities) and the clean target $\boldsymbol{x}_1$ must be centered to have zero center of mass.

We discuss the implications for training in Section B and for sampling in Section C.

B Training

Algorithm 1 describes the computation of the training loss for our flow matching objective. We start by sampling from the prior $\boldsymbol{x}_0 \sim p_0(\boldsymbol{x})$, the data distribution $\boldsymbol{x}_1 \sim p_1(\boldsymbol{x})$, and a Gaussian distribution $\boldsymbol{\epsilon} \sim N(\boldsymbol{x}; 0, \boldsymbol{I})$. The interpolant is then defined as

$$\boldsymbol{x}_t = (1-t)\cdot \boldsymbol{x}_0 + t \cdot \boldsymbol{x}_1 + \sigma \cdot \boldsymbol{\epsilon}\,, \tag{A13}$$

where $\sigma \in \mathbb{R}_{>0}$ is a non-zero noise scaling parameter.

Rather than directly predicting the conditional vector field $\boldsymbol{u}_t(\boldsymbol{x}|\boldsymbol{x}_0, \boldsymbol{x}_1)$, we choose to reparametrize the network such that it predicts the clean sample $\boldsymbol{x}_1$. Similar to Ref. [57], we add a weighting term $1/(1-t)^2$ to encourage the model to accurately capture fine details close to the data distribution. This gives rise to the following loss function

$$\mathcal{L} = \frac{1}{(1-t)^2}\|\boldsymbol{x}_1^\theta(\boldsymbol{x}_t, t, \mathcal{G}) - \boldsymbol{x}_1\|^2\,, \tag{A14}$$

where $t \in (0,1)$ denotes the timestep in the interpolant $\boldsymbol{x}_t \in \mathbb{R}^{N\times 3}$, $\boldsymbol{x}_1 \in \mathbb{R}^{N\times 3}$ is the clean geometry, and $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denotes the molecular graph. The full algorithm is summarized in Algorithm 1.

Geometry Alignment Given a set of vectors $\mathcal{U} = \{\vec{u}_1, \ldots, \vec{u}_N \mid \vec{u}_i \in \mathbb{R}^3\}$ associated with a point cloud $\boldsymbol{x} \in \mathbb{R}^{N\times 3}$, we define a centering operation for the i-th row

$$\mathrm{Center}(\boldsymbol{x})_i = \vec{u}_i - \frac{1}{N}\sum_{j=1}^{N}\vec{u}_j\,, \tag{A15}$$

which removes global drift in $\boldsymbol{x}$. Given two point clouds $\boldsymbol{x}_A \in \mathbb{R}^{N\times 3}$ and $\boldsymbol{x}_B \in \mathbb{R}^{N\times 3}$ with positions $\mathcal{R}_A = \{\vec{r}_{1A}\ \ldots, \vec{r}_{NA}\}$ and $\mathcal{R}_B = \{\vec{r}_{1B}\ \ldots, \vec{r}_{NB}\}$, we define a rotational alignment operation

$$\mathrm{RotationAlign}(\boldsymbol{x}_A, \boldsymbol{x}_B)_i = \boldsymbol{R}_{\text{opt}}\boldsymbol{x}_{iA}, \tag{A16}$$

where $\boldsymbol{R}_{\text{opt}} \in \mathbb{R}^{3\times 3}$ is the optimal rotation matrix, minimizing the root mean square deviation (RMSD) between the positions of point clouds A and B, which can be obtained via the Kabsch algorithm [71]. The full geometry alignment operation "$\mathrm{GeometryAlign}(\boldsymbol{x}_A, \boldsymbol{x}_B)$", is given by

$$\boldsymbol{x}_A \leftarrow \mathrm{Center}(\boldsymbol{x}_A) \tag{A17}$$

$$\boldsymbol{x}_B \leftarrow \mathrm{Center}(\boldsymbol{x}_B) \tag{A18}$$

$$\boldsymbol{x}_A \leftarrow \mathrm{RotationAlign}(\boldsymbol{x}_A, \boldsymbol{x}_B) \tag{A19}$$

In words, both point clouds are first centered at the origin and then optimally aligned by rotation. This procedure minimizes the path length of a linear interpolation between the point clouds A and B.

Algorithm 1 Conditional Flow Matching Training Loss

Require: Graph $\mathcal{G}$, target $\boldsymbol{x}_1$, noise level σ, Model $\boldsymbol{x}_1^\theta$

$\boldsymbol{x}_0 \sim p_0$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, $t \sim \mathcal{U}(0,1)$, $\boldsymbol{R} \sim \mathrm{SO}(3)$
$\boldsymbol{x}_0, \boldsymbol{x}_1 \leftarrow \mathrm{GeometryAlign}(\boldsymbol{x}_0, \boldsymbol{x}_1)$ ▷ This centers $\boldsymbol{x}_0$ and $\boldsymbol{x}_1$ and rotation-aligns $\boldsymbol{x}_0$ to $\boldsymbol{x}_1$.
$\boldsymbol{\epsilon} \leftarrow \mathrm{Center}(\boldsymbol{\epsilon})$
$\boldsymbol{x}_t \leftarrow (1-t)\boldsymbol{x}_0 + t\boldsymbol{x}_1 + \sigma\boldsymbol{\epsilon}$
$\boldsymbol{x}_t \leftarrow \mathrm{ApplyRotation}(\boldsymbol{R}, \boldsymbol{x}_t)$
$\boldsymbol{x}_1 \leftarrow \mathrm{ApplyRotation}(\boldsymbol{R}, \boldsymbol{x}_1)$
$\hat{\boldsymbol{x}}_1 \leftarrow \boldsymbol{x}_1^\theta(\boldsymbol{x}_t, t, \mathcal{G})$
$\hat{\boldsymbol{x}}_1 \leftarrow \mathrm{Center}(\hat{\boldsymbol{x}}_1)$
return $\frac{1}{(1-t)^2}\|\hat{\boldsymbol{x}}_1 - \boldsymbol{x}_1\|^2$

Data Augmentation The target data distribution of molecular conformers $p_1(\boldsymbol{x})$ is $\mathrm{SE}(3)$-invariant, i.e. it does not change under translations and rotations of the input. One can construct an $\mathrm{SE}(3)$-invariant density by learning an $\mathrm{SO}(3)$-equivariant vector field on centered $\mathrm{SE}(3)$ (see Section A). However, only DiTMC+PE(3) is $\mathrm{SO}(3)$-equivariant, whereas aPE and rPE violate $\mathrm{SO}(3)$-equivariance. Therefore, we learn equivariance approximately during training, using data augmentation. Specifically, we randomly sample rotation matrices $\boldsymbol{R}$ (orthogonal matrices with determinant +1) and apply them as

$$\mathrm{ApplyRotation}(\boldsymbol{R}, \boldsymbol{x})_i = \boldsymbol{R}\vec{r}_i, \tag{A20}$$

where $\vec{r}_i$ denotes the positions of the i-th atom, i.e., the i-th row in the point cloud $\boldsymbol{x} \in \mathbb{R}^{N\times 3}$.

Noise Scaling Parameter We ablated the noise scaling parameter σ on GEOM-QM9 and GEOM-DRUGS, comparing a larger value of 0.5 and a smaller value of 0.05. We set σ to the value that empirically worked best for each dataset: 0.05 for GEOM-QM9 and 0.5 for GEOM-DRUGS.

Optimizer and Hyperparameters We use the AdamW optimizer (weight decay 0.01) with batch size of 128 and learning rate of $\mu_{\text{max}} = 3 \times 10^{-4}$ for GEOM-QM9 and $\mu_{\text{max}} = 1 \times 10^{-4}$ for GEOM-DRUGS. First, we increase the initial learning rate of $\mu_0 = 10^{-5}$ up to μ_{max} via a linear learning rate warmup over the first 1% of training steps. Afterwards, the learning rate is decreased via a cosine decay schedule to $\mu_{\text{min}} = 0$ for GEOM-QM9 and $\mu_{\text{min}} = 1 \times 10^{-5}$ for GEOM-DRUGS.

Compute Budget and Training Times All models on GEOM-QM9 are trained for 250 epochs, which requires 2 days of training on Nvidia H100 GPU for aPE and rPE models and almost 4 days for PE(3). For GEOM-DRUGS, we fix the total compute budget per model to nine days on a single NVIDIA H100 GPU, due to computational constraints. Within this budget, we can train aPE-L and rPE-L for 50 epochs and PE(3)-L for 10 epochs. To ensure consistency within each PE strategy, the base ("B") model are trained for the same number of epochs as the corresponding large ("L") model.

C Sampling

For sampling, we use a simple Euler scheme with 50 steps to sample from the associated ordinary differential equation (ODE) as described in Algorithm 2. Since during training we predict the clean sample $\boldsymbol{x}_1$, we re-parametrize the velocity required for the integration as

$$\boldsymbol{v}_t^\theta(\boldsymbol{x}_t, t, \mathcal{G}) = \frac{\boldsymbol{x}_1^\theta(\boldsymbol{x}_t, t, \mathcal{G}) - \boldsymbol{x}_t}{1-t}, \tag{A21}$$

where $\boldsymbol{x}_1^\theta(\boldsymbol{x}_t, t, \mathcal{G})$ is the original output of DiTMC.

To ensure $\mathrm{SE}(3)$ invariance of the probability path from an (approximately) $\mathrm{SO}(3)$-equivariant velocity predictor, we center the prior $\boldsymbol{x}_0 \sim p_0(\boldsymbol{x})$ and the prediction of DiTMC in each ODE step.

Algorithm 2 ODE Sampling

Require: Model $\boldsymbol{x}_1^\theta$, Graph $\mathcal{G}$, steps $N > 0$

$t_n \leftarrow n/N$ for $n \in \{0, \ldots, N\}$
$\boldsymbol{x}_0 \sim p_{\text{prior}}(\boldsymbol{x})$ ▷ Sample prior.
$\boldsymbol{x}_0 \leftarrow \text{Center}(\boldsymbol{x}_0)$
for $n \leftarrow 0$ to $N-1$ **do**
  $\Delta t \leftarrow t_{n+1} - t_n$ ▷ Compute step size.
  $\hat{\boldsymbol{x}}_1 \leftarrow \boldsymbol{x}_1^\theta(\boldsymbol{x}_{t_n}, t_n, \mathcal{G})$
  $\hat{\boldsymbol{x}}_1 \leftarrow \text{Center}(\hat{\boldsymbol{x}}_1)$
  $\boldsymbol{v} \leftarrow (\hat{\boldsymbol{x}}_1 - \boldsymbol{x}_{t_n})/(1 - t_n)$
  $\boldsymbol{x}_{t_{n+1}} \leftarrow \boldsymbol{x}_{t_n} + \Delta t \cdot \boldsymbol{v}$ ▷ Euler step.
end for

return $\boldsymbol{x}_1$

Table A6: Architectural details for MLPs used in the model. The feature dimension is given as $H = n_{\text{heads}} \cdot d_{\text{head}}$ where n_{head} is the number of heads and d_{head} is the number of features per head.

Name	Layers	Hidden Dim	Out Dim	Activation	Input
DiTMC Block	2	$4H$	H	GELU	Tokens
SO(3) DiTMC Block	2	$4H$	H	gated GELU	Tokens
Time and Atom Cond.	1	–	$6H$	SiLU	$\boldsymbol{c}^t + \boldsymbol{c}_i^{\mathcal{G}}$
Bond pair	2	H	H	SiLU	$\boldsymbol{c}_{ij}^{\mathcal{G}}$
Time embedding	2	H	H	SiLU	$t \in [0, 1]$
Shortest-hop embedding	2	H	H	SiLU	Hop distance
aPE embedding	2	H	H	SiLU	Abs. positions $\vec{r}_i$
rPE embedding	2	H	H	SiLU	Rel. positions $\vec{r}_{ij}$
GNN embedding	2	H	H	SiLU	Node/edge features

D Architectural Details

In this section, we describe the architecture of DiTMC and the conditioning GNN in more detail.

All DiTMC models rely on conditioning tokens $\mathcal{C}$, for the time $\boldsymbol{c}^t \in \mathbb{R}^H$, per-atom $\boldsymbol{c}_i^{\mathcal{G}} \in \mathbb{R}^H$, and per atom-pair $\boldsymbol{c}_{ij}^{\mathcal{G}} \in \mathbb{R}^H$. See main text Section 4.1 for more details. Following Ref. [13], we use adaptive layer norm (AdaLN) and adaptive scale (AdaScale) to include per-atom conditioning tokens based on time t and molecular graph information. To that end, we construct conditioning tokens

$$\boldsymbol{c}_i = \boldsymbol{c}^t + \boldsymbol{c}_i^{\mathcal{G}}. \tag{A22}$$

Tab. A6 contains details on the MLPs used throughout our architecture, while Tab. A7 summarizes architectural details, as well as training and inference times for all DiTMC variants. Note that the number of attention heads in DiTMC+PE(3) is adjusted to match the total parameter count of the corresponding DiTMC+aPE and DiTMC+rPE variants, while all other hyperparameters are identical.

D.1 Non-Equivariant DiTMC

In the non-equivariant DiTMC formulations based on aPE and rPE, we have the following set of tokens $\mathcal{H} = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \,|\, \boldsymbol{h}_i \in \mathbb{R}^H\}$. Initial token representations are obtained via $\boldsymbol{h}_i = \mathbf{e}(z_i) + \boldsymbol{p}_i^{\text{aPE}}$ for aPE and $\boldsymbol{h}_i = \boldsymbol{e}(z_i)$ for rPE, where $\boldsymbol{e}(z_i) \in \mathbb{R}^H$ denotes a learnable embedding based on the atomic number $z_i \in \mathbb{N}_+$ of atom i [35].

Table A7: Architectural details as well as training and inference time (in ms) for different PE strategies on GEOM-QM9 and GEOM-DRUGS. Times are measured with batch size 128 on a single Nvidia H100 GPU. T means number of transformer layers, n_{heads} number of heads, d_{head} number of features per head in the attention update, and T_{MGN} number of layers for the conditioning mesh graph net. Thus, total feature dimension is given as $H = n_{\text{heads}} \cdot d_{\text{head}}$.

Model	T	n_{heads}	d_{head}	T_{MGN}	H	**Train** [ms]	**Infer** [ms]
DiTMC+aPE-S (1M)	2	4	32	1	128	5.8	2.0
DiTMC+aPE-B (9.5M)	6	8	32	2	256	19.2	8.1
DiTMC+rPE-B (9.6M)	6	8	32	2	256	19.7	8.3
DiTMC+PE(3)-B (8.6M)	6	6	32	2	192	70.0	25.9
DiTMC+aPE-L (28.2M)	8	12	32	3	384	32.7	9.5
DiTMC+rPE-L (28.3M)	8	12	32	3	384	33.5	10.1
DiTMC+PE(3)-L (31.1M)	8	10	32	3	320	151.6	41.8

D.1.1 Self-Attention Operation

For ease of notation, we only describe self-attention with a single head, but employ multi-head attention [29] with n_{heads} heads in our experiments. All self-attention blocks rely on query, key and value vectors, which are obtained from the input tokens $\mathcal{H} = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \mid \boldsymbol{h}_i \in \mathbb{R}^H\}$ as

$$\boldsymbol{q} = \boldsymbol{W}_q \boldsymbol{h}\,, \qquad \boldsymbol{k} = \boldsymbol{W}_k \boldsymbol{h}\,, \qquad \boldsymbol{v} = \boldsymbol{W}_v \boldsymbol{h}\,, \tag{A23}$$

where $\boldsymbol{W}_q, \boldsymbol{W}_k, \boldsymbol{W}_v \in \mathbb{R}^{H\times H}$ are trainable weight matrices. We define a slightly modified similarity kernel

$$\text{sim}(\boldsymbol{q}, \boldsymbol{k}, \boldsymbol{u}) = \exp\left(\frac{\boldsymbol{q}^\intercal \cdot (\boldsymbol{k} \odot \boldsymbol{u})}{\sqrt{H}}\right), \tag{A24}$$

where $\boldsymbol{u} \in \mathbb{R}^H$ is used to inject additional information, e.g., conditioning signals and/or positional embeddings, and '$\odot$' denotes element-wise multiplication.

For absolute and relative PEs, we slightly modify standard self-attention to allow injecting pair-wise information into the values in addition to using our modified similarity kernel

$$\text{ATT}(\mathcal{H}, \mathcal{P}, \mathcal{C}^{\mathcal{G}}_{\text{Pair}})_i = \frac{\sum_{j=1}^N \text{sim}(\boldsymbol{q}_i, \boldsymbol{k}_j, \boldsymbol{u}_{ij}) \cdot (\boldsymbol{v}_j \odot \boldsymbol{u}_{ij})}{\sum_{j=1}^N \text{sim}(\boldsymbol{q}_i, \boldsymbol{k}_j, \boldsymbol{u}_{ij})}\,, \tag{A25}$$

where queries, keys, and values are obtained with Eq. A23 and the injected pair-wise information $\boldsymbol{u}_{ij}$ depends on the positional embedding strategy with

$$\boldsymbol{u}_{ij} = \begin{cases} \boldsymbol{c}^{\mathcal{G}}_{ij} & \text{for absolute PEs}\,, \\ \boldsymbol{c}^{\mathcal{G}}_{ij} + \boldsymbol{p}^{\text{rPE}}_{ij} & \text{for relative PEs}\,. \end{cases} \tag{A26}$$

Here $\boldsymbol{c}^{\mathcal{G}}_{ij} \in \mathbb{R}^H$ are pair-wise graph conditioning tokens (see Eq. 7) and $\boldsymbol{p}^{\text{aPE}}_i \in \mathbb{R}^H$ and $\boldsymbol{p}^{\text{rPE}}_{ij} \in \mathbb{R}^H$ are the absolute and relative PEs described above (see Eqs. 8 and 9).

D.1.2 Adaptive Layer Normalization and Adaptive Scale

In the standard, non-equivariant setting, we can follow the standard approach of other DiT architectures. We define adaptive layer norm as

$$\text{AdaLN}(\boldsymbol{h}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \text{LN}(\boldsymbol{h}) \odot (1 + \boldsymbol{\alpha}) + \boldsymbol{\beta}, \tag{A27}$$

where LN is a standard layer normalization without trainable scale and bias, and "$\odot$" denotes entry-wise product.

Adaptive Scale is defined as

$$\text{AdaScale}(\boldsymbol{h}, \boldsymbol{\gamma}) = \boldsymbol{h} \odot \boldsymbol{\gamma}. \tag{A28}$$

In each DiTMC block, we calculate

$$\boldsymbol{\alpha}_{1i}, \boldsymbol{\beta}_{1i}, \boldsymbol{\gamma}_{1i}, \boldsymbol{\alpha}_{2i}, \boldsymbol{\beta}_{2i}, \boldsymbol{\gamma}_{2i} = \boldsymbol{W}\big(\text{SiLU}(\boldsymbol{c}_i)\big), \tag{A29}$$

where $\boldsymbol{W} \in \mathbb{R}^{6H \times H}$ and the output is split into six equally sized vectors $\boldsymbol{\alpha}_{1i}, \boldsymbol{\beta}_{1i}, \boldsymbol{\gamma}_{1i}, \boldsymbol{\alpha}_{2i}, \boldsymbol{\beta}_{2i}, \boldsymbol{\gamma}_{2i} \in \mathbb{R}^H$. The weight matrix $\boldsymbol{W}$ is initialized to all zeros, such that "AdaLN" behaves like identity at initialization. "AdaScale" damps all input tokens to zero at initialization such that the whole DiTMC block behaves like the identity function at initialization.

D.1.3 Readout

Given final tokens $\boldsymbol{h}$ after performing updates via T DiTMC blocks, we use a readout layer to predict the atomic positions of the clean data sample $\boldsymbol{x}_1$. As in the DiTMC blocks, we employ adaptive LN and therefore calculate

$$\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i = \boldsymbol{W}\big(\text{SiLU}(\boldsymbol{c}_i)\big), \tag{A30}$$

with weight matrix $\boldsymbol{W} \in \mathbb{R}^{2H \times H}$ initialized to all zeros and $\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i \in \mathbb{R}^H$ and do

$$\begin{aligned} \boldsymbol{h}_i &\leftarrow \text{AdaLN}(\boldsymbol{h}_i, \boldsymbol{\alpha}_i, \boldsymbol{\beta}_i)\,, \\ \hat{\boldsymbol{x}}_i &\leftarrow \boldsymbol{W}_{\text{readout}} \boldsymbol{h}_i\,, \end{aligned}$$

where $\boldsymbol{W}_{\text{readout}} \in \mathbb{R}^{3 \times H}$ is a trainable weight matrix. Thus, we predict a three-dimensional vector per-atom.

D.2 SO(3) Equivariant DiTMC

Following the notation in Ref. [72], we denote $\text{SO}(3)$-equivariant tokens as $\mathcal{H} = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \,|\, \boldsymbol{h}_i \in \mathbb{R}^{(L+1)^2 \times H}\}$, where L denotes the maximal degree of the spherical harmonics. We denote the features corresponding to the ℓ-th degree as $\boldsymbol{h}_i^{(\ell)} \in \mathbb{R}^{(2\ell+1) \times H}$, where the $(2\ell+1)$ entries corresponds to the orders $m = -\ell, \ldots, +\ell$ per degree ℓ. We refer the reader to Ref. [72] for an in-depth introduction into equivariant features. For the initial token representation $\boldsymbol{h}_i \in \mathbb{R}^{(L+1)^2 \times H}$ we set $\boldsymbol{h}_i^{(0)} = \boldsymbol{e}(z_i)$, where $\boldsymbol{e}(z_i) \in \mathbb{R}^{1 \times H}$ denotes a learnable embedding based on the atomic number $z_i \in \mathbb{N}_+$ for atom i [35]. All higher order features $\boldsymbol{h}_i^{(\ell)}$ with degree $\ell > 0$ are initialized with zero. For all our experiments we use maximal degree of $L = 1$.

D.2.1 Self-Attention Operation

Our equivariant version of self-attention uses the same transformations for queries, keys and values like the non-equivariant counterpart, as well as the modified similarity kernel (see Appendix subsubsection D.2.1). However, to preserve all Euclidean symmetries throughout the network, every token must transform equivariantly. One way to achieve this is by separating out the rotational degrees of freedom, encoding them with irreducible representations of the rotation group $\text{SO}(3)$. This introduces a "degree-axis" of size $(L+1)^2$, which encodes angular components of increasing order. The maximum degree L is chosen to ensure high fidelity at a reasonable computational cost. For example, setting $L = 1$ restricts the representation to scalars and vectors, as used in models like PaiNN [60] or TorchMDNet [34]. An $\text{SO}(3)$-equivariant formulation of self-attention is then given as

$$\text{ATT}_{\text{SO(3)}}(\mathcal{H})_i = \frac{\sum_{j=1}^N \text{sim}(\boldsymbol{q}_i, \boldsymbol{k}_j, \boldsymbol{u}_{ij}) \cdot (\hat{\boldsymbol{u}}_{ij} \otimes \boldsymbol{v}_j)}{\sum_{j=1}^N \text{sim}(\boldsymbol{q}_i, \boldsymbol{k}_j, \boldsymbol{u}_{ij})}\,, \tag{A31}$$

where equivariant queries, keys and values can be calculated similarly to Eq. A23 and '$\otimes$' denotes a Clebsch-Gordan (CG) tensor product contraction [72]. The dot-product in the similarity measure is taken along both feature and degree axes, such that the overall update preserves equivariance (see Appendix Appendix F for details). Tokens and scaling vectors are calculated as

$$\boldsymbol{u}_{ij} = \boldsymbol{\phi}(r_{ij}) \odot \boldsymbol{c}_{ij}^{\mathcal{G}}\,, \qquad \hat{\boldsymbol{u}}_{ij} = \boldsymbol{p}_{ij}^{\text{PE(3)}} \odot \boldsymbol{c}_{ij}^{\mathcal{G}}\,, \tag{A32}$$

where $\boldsymbol{\phi}(r_{ij}) \in \mathbb{R}^{(L+1)^2 \times H}$ is a radial filter, and the element-wise products with the pair-wise conditioning tokens $\boldsymbol{c}_{ij}^{\mathcal{G}} \in \mathbb{R}^{1 \times H}$ are broadcast along the degree axis. Importantly, the $2\ell + 1$

subcomponents of the radial filter for degree ℓ are obtained by repeating per-degree filter functions $\boldsymbol{\phi}_\ell(r_{ij}) \in \mathbb{R}^{1\times H}$ along the degree axis to preserve equivariance (see also Eq. 10).

Since, standard MLPs do not preserve SO(3)-equivariance, we use equivariant MLPs for the node-wise refinement after the self-attention calculation via an equivariant formulation for dense layers and so-called gated non-linearities (see e.g. Ref. [72] for more details).

D.2.2 Adaptive Layer Normalization and Adaptive Scale

In the $\mathrm{SO}(3)$-equivariant case, we define an adapted version of AdaLN and AdaScale, preserving equivariance. Our equivariant formulation of AdaLN is given as

$$\text{EquivAdaLN}(\boldsymbol{h}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \begin{cases} \text{LN}(\boldsymbol{h}^{(\ell)}) \odot (1 + \boldsymbol{\alpha}^{(0)}) + \boldsymbol{\beta} & \text{for } \ell = 0, \\ \text{EquivLN}(\boldsymbol{h}^{(\ell)}) \odot (1 + \boldsymbol{\alpha}^{(\ell)}) & \text{for } \ell > 0, \end{cases} \tag{A33}$$

where EquivLN is the equivariant formulation of layer normalization following Ref. [36] without trainable per-degree scales and LN is standard layer normalization without trainable scale and bias. Scaling vectors $\boldsymbol{\alpha}^{(\ell)} \in \mathbb{R}^{1\times H}$ are defined per degree ℓ, such that input scaling vectors are tensors $\boldsymbol{\alpha} \in \mathbb{R}^{(L+1)\times H}$. Bias vectors are only defined for the invariant ($\ell = 0$) component of the tokens, since adding a non-zero bias to components with $\ell > 0$ would lead to a non-equivariant operation (the bias does not transform under rotations). The element wise multiplication between $(1 + \boldsymbol{\alpha}^{(\ell)}) \in \mathbb{R}^{1\times H}$ and tokens $h^{(\ell)} \in \mathbb{R}^{(2\ell+1)\times H}$ is "broadcasted" along the degree-axis. For $L = 0$, Eq. A33 reduces to the standard adaptive layer normalization.

Equivariant adaptive scale is defined as

$$\text{EquivAdaScale}(\boldsymbol{h}, \boldsymbol{\gamma}) = \boldsymbol{h}^{(\ell)} \odot \boldsymbol{\gamma}^{(\ell)}. \tag{A34}$$

As for "EquivAdaLN", we define a separate $\boldsymbol{\gamma}^{(\ell)} \in \mathbb{R}^{1\times H}$ per degree ℓ, such that $\boldsymbol{\gamma} \in \mathbb{R}^{(L+1)\times H}$. Again, the element wise product is "broadcasted" along the degree-axis. Since no bias is involved, the invariant and equivariant parts in $\boldsymbol{h}$ can be treated equally.

Within each $\mathrm{SO}(3)$-equivariant DiTMC block, we calculate

$$\boldsymbol{\alpha}_{1i}, \boldsymbol{\beta}_{1i}, \boldsymbol{\gamma}_{1i}, \boldsymbol{\alpha}_{2i}, \boldsymbol{\beta}_{2i}, \boldsymbol{\gamma}_{2i} = \boldsymbol{W}\big(\text{SiLU}(\boldsymbol{c}_i)\big), \tag{A35}$$

where $\boldsymbol{\alpha}_{1i}, \boldsymbol{\alpha}_{2i}, \boldsymbol{\gamma}_{1i}, \boldsymbol{\gamma}_{2i} \in \mathbb{R}^{(L+1)\times H}$ and $\boldsymbol{\beta}_{1i}, \boldsymbol{\beta}_{2i} \in \mathbb{R}^{H}$. Thus, the weight matrix is given as $\boldsymbol{W} \in \mathbb{R}^{(4(L+1)+2)\times H}$ and initialized to all zeros, such that "EquivAdaLN" behaves like identity at initialization and "EquivAdaScale" returns zeros. Thus, also the $\mathrm{SO}(3)$-equivariant DiTMC block behaves like the identity function at initialization.

D.2.3 Readout

Given final equivariant features $\boldsymbol{h}_i \in \mathbb{R}^{(L+1)^2\times H}$ we use a readout layer to predict the atomic positions of the clean data sample $\boldsymbol{x}_1$. We employ our equivariant formulation of adaptive layer normalization and calculate

$$\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i = \boldsymbol{W}\big(\text{SiLU}(\boldsymbol{c}_i)\big), \tag{A36}$$

with weight matrix $\boldsymbol{W} \in \mathbb{R}^{2H(L+1)\times H}$ initialized to all zeros and $\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i \in \mathbb{R}^{H(L+1)}$. We then do,

$$\begin{aligned} \boldsymbol{h}_i &\leftarrow \text{EquivAdaLN}(\boldsymbol{h}_i, \boldsymbol{\alpha}_i, \boldsymbol{\beta}_i)\,, \\ \boldsymbol{y}_i &\leftarrow \boldsymbol{W}_{\text{readout}} \boldsymbol{h}_i^{(\ell=1)}\,, \end{aligned}$$

where $\boldsymbol{W}_{\text{readout}} \in \mathbb{R}^{1\times H}$ is a trainable weight vector that is applied along the feature axis in $\boldsymbol{h}$. Since $\boldsymbol{h}_i^{(\ell=1)} \in \mathbb{R}^{3\times H}$ this produces per-atom vectors $\hat{\boldsymbol{y}}_i \in \mathbb{R}^3$. As $\boldsymbol{h}_i$ are rotationally equivariant so is $\hat{\boldsymbol{y}}_i \in \mathbb{R}^3$.

D.3 Conditioning GNN

Our conditioning GNN directly operates on the molecular graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to obtain graph-based information for atom-wise conditioning. First, we processes the molecular graph $\mathcal{G}$ to derive node

and edge input features, initially represented as one-hot vectors (a full list of features is provided in Tab. A8). These features then are projected into a shared latent space using two-layer multilayer perceptrons (MLPs). Finally, we employ a GNN architecture inspired by the processor described in the MeshGraphNet (MGN) framework [58], which refines the features via message passing.

The conditioning GNN maintains and updates both node and edge representations across multiple layers. Each message passing block consists of two main steps: first, the edge representations are updated based on the current edge representations and the representations of the connected nodes:

$$\boldsymbol{e}'_{ij} \leftarrow f_e(\boldsymbol{e}_{ij}, \boldsymbol{v}_i, \boldsymbol{v}_j) \tag{A37}$$

where $\mathbf{e}_{ij}$ and $\mathbf{v}_i$ denote the input edge and node representations, and $\mathbf{e}'_{ij}$ are the updated edge representations. The learnable function f_e is implemented as a two-layer MLP. Next, node representations $\mathbf{v}_i$ are updated to $\mathbf{v}'_i$ using aggregated messages from neighboring edges:

$$\boldsymbol{v}'_i \leftarrow f_v\left(\boldsymbol{v}_i, \sum_j \boldsymbol{e}'_{ij}\right) \tag{A38}$$

where f_v is also a two-layer MLP, and the summation is over all edges ending at node i.

The final output of the described conditioning GNN is a set of node embeddings per atom, and a set of edge embeddings per bond. We use the final node embeddings as atom-wise graph conditioning tokens in DiTMC as detailed in Sec. 4.1. We want to highlight that one can also use the final edge embedding as pair-wise graph conditioning tokens in DiTMC. We discuss this further in Appendix I.2.

E Equivariance Proof for Euclidean Positional Embeddings

Given a set of transformations that act on a vector space $\mathbb{A}$ as $S_g : \mathbb{A} \mapsto \mathbb{A}$ to which we associate an abstract group G, a function $f : \mathbb{A} \mapsto \mathbb{B}$ is said to be equivariant w.r.t. G if

$$f(S_g x) = T_g f(x)\,, \tag{A39}$$

where $T_g : \mathbb{B} \mapsto \mathbb{B}$ is an equivalent transformation on the output space. Thus, in order to say that f is equivariant, it must hold that under transformation of the input, the output transforms "in the same way".

Let us now recall our definition for the equivariant positional embeddings for a single degree ℓ

$$\boldsymbol{p}_{ij}^{\mathrm{PE}(3),(\ell)}(\vec{r}_{ij}) = \boldsymbol{\phi}_\ell(||\vec{r}_{ij}||) \odot \boldsymbol{Y}_\ell(\hat{r}_{ij})\,, \tag{A40}$$

where $\boldsymbol{\phi}_\ell : \mathbb{R} \mapsto \mathbb{R}^{1\times H}$ is a radial filter function, $\hat{r} = \vec{r}/r$, and $\boldsymbol{Y}_\ell \in \mathbb{R}^{(2\ell+1)\times 1}$ are spherical harmonics of degree $\ell = 0 \dots L$. The element-wise multiplication '$\odot$' between radial filters and spherical harmonics is understood to be "broadcasting" along axes with size 1, such that $(\boldsymbol{\phi}_\ell \odot \boldsymbol{Y}_\ell) \in \mathbb{R}^{(2\ell+1)\times H}$. We have also made the dependence of PE(3) on the pairwise displacement vector $\vec{r}_{ij} = \vec{r}_i - \vec{r}_j$ explicit.

Lets not consider a single feature channel c in PE(3), which is given as

$$\boldsymbol{p}_{ijc}^{\mathrm{PE}(3),(\ell)}(\vec{r}_{ij}) = \boldsymbol{\phi}_{\ell c}(||\vec{r}_{ij}||) \odot \boldsymbol{Y}_\ell(\hat{r}_{ij})\,. \tag{A41}$$

Rotating the input positions in Eq. A41 leads to

$$\begin{aligned}
\boldsymbol{p}_{ijc}^{\mathrm{PE}(3),(\ell)}(\boldsymbol{R}\vec{r}_{ij}) &= \boldsymbol{\phi}_{\ell c}(||\boldsymbol{R}\vec{r}_{ij}||) \odot \boldsymbol{Y}_\ell(\boldsymbol{R}\hat{r}_{ij}) && \text{(A42)}\\
&= \boldsymbol{\phi}_{\ell c}(||\vec{r}_{ij}||) \odot \boldsymbol{D}^{(\ell)}(\boldsymbol{R})\boldsymbol{Y}_\ell(\hat{r}_{ij}), && \text{(A43)}\\
&= \boldsymbol{D}^{(\ell)}(\boldsymbol{R})\boldsymbol{p}_{ijc}^{\mathrm{PE}(3),(\ell)}(\vec{r}_{ij}) && \text{(A44)}
\end{aligned}$$

where $\boldsymbol{D}^{(\ell)} \in \mathbb{R}^{(2\ell+1)\times(2\ell+1)}$ are the Wigner-D matrices for degree ℓ and $\boldsymbol{R} \in \mathbb{R}^{3\times 3}$ is a rotation matrix. According to Eq. A39 and Eq. A44, each channel transforms equivariant and thus $\boldsymbol{p}_{ij}^{\mathrm{PE}(3),(\ell)}(\vec{r}_{ij})$ is also equivariant.

The concatenation of different degrees ℓ up to some maximal degree L as given in the main body of the text

$$\boldsymbol{p}_{ij}^{\text{PE(3)}}(\vec{r}_{ij}) = \bigoplus_{\ell=0}^{L} \boldsymbol{\phi}_\ell(r_{ij}) \odot \boldsymbol{Y}_\ell(\hat{r}_{ij}) \,, \tag{A45}$$

transforms under rotation as

$$\boldsymbol{p}_{ij}^{\text{PE(3)}}(\boldsymbol{R}\vec{r}_{ij}) = \boldsymbol{D}(\boldsymbol{R})\, \boldsymbol{p}_{ij}^{\text{PE(3)}}(\vec{r}_{ij}) \tag{A46}$$

with $\boldsymbol{D}(\boldsymbol{R}) = \bigoplus_{\ell=0}^{L} \boldsymbol{D}^{(\ell)}(\boldsymbol{R}) \in \mathbb{R}^{(L+1)^2 \times (L+1)^2}$ being a block-diagonal matrix with the Wigner-D matrices of degree $\boldsymbol{D}^{(\ell)}(\boldsymbol{R}) \in \mathbb{R}^{(2\ell+1)\times(2\ell+1)}$ along the diagonal. Therefore, according to Eq. A39 the proposed positional embeddings $\boldsymbol{p}_{ij}^{\text{PE(3)}}$ are $\mathrm{SO}(3)$-equivariant.

F Invariance Proof for the Dot-Product

In the self-attention update, the dot-product between query and key is computed as along the degree and the feature axis. Under rotation, the equivariant features behave as

$$\boldsymbol{h}(\boldsymbol{R}\vec{r}) = \boldsymbol{D}(\boldsymbol{R})\boldsymbol{h}(\vec{r}), \tag{A47}$$

where $\boldsymbol{D}(\boldsymbol{R})$ is the concatenation of Wigner-D matrices from above. The inner product along the degree axis for two features $\boldsymbol{h}$ and $\boldsymbol{g}$ behaves under rotation as

$$\boldsymbol{g}(\boldsymbol{R}\vec{r})^T \cdot \boldsymbol{h}(\boldsymbol{R}\vec{r}) = \boldsymbol{g}(\vec{r})^T \underbrace{\boldsymbol{D}(\boldsymbol{R})^T \boldsymbol{D}(\boldsymbol{R})}_{=\text{Id}} \boldsymbol{h}(\vec{r}) = \boldsymbol{g}(\vec{r})^T \cdot \boldsymbol{h}(\vec{r}), \tag{A48}$$

where we made use of the fact that the Wigner-D matrices are orthogonal matrices. Thus, the dot-product along the degree-axis is invariant and therefore taking the dot-product along the degree and then along the feature axis is also invariant.

G Implementation details

G.1 Data Preprocessing

For both GEOM-QM9 and GEOM-DRUGS, we use the first 30 conformers for each molecule with the lowest energies, i.e., highest Boltzmann weights. We use the train/test/val split from Geomol [64], using the same 1000 molecules for testing.

G.2 Input Featurization

Tab. A8 defines the features we use for each atom or bond. Each feature is computed using RDKit [73] and one-hot encoded before being passed to the network.

G.3 Evaluation Metrics

During evaluation, we follow the same procedure as described in Refs. [32, 46, 47, 64]. The root-mean-square deviation (RMSD) metric measures the average distance between atoms of a generated conformer with respect to its reference, while taking into account all possible symmetries. For $L = 2K$ let $\{\hat{C}_l\}_{l\in\{1,\dots,L\}}$ and $\{C_k\}_{k\in\{1,\dots,K\}}$ be the sets of generated conformers and reference conformers respectively. The average minimum RMSD (AMR) and coverage (COV) metrics for both recall (R) and precision (P) are defined as follows, where $\delta > 0$ is the coverage threshold:

Table A8: Atomic and bond features included in DiT-MC. All features are one-hot encoded.

Atom features	**Options**
Chirality	`TETRAHEDRAL_CW, TETRAHEDRAL_CCW, UNSPECIFIED, OTHER`
Number of hydrogens	0, 1, 2, 3, 4
Number of radical electrons	0, 1, 2, 3, 4
Atom type (QM9)	H, C, N, O, F
Atom type (DRUGS)	H, Li, B, C, N, O, F, Na, Mg, Al, Si, P, S, Cl, K, Ca, V, Cr, Mn, Cu, Zn, Ga, Ge, As, Se, Br, Ag, In, Sb, I, Gd, Pt, Au, Hg, Bi
Aromaticity	true, false
Degree	0, 1, 2, 3, 4, other
Hybridization	sp, sp^2, sp^3, sp^3d, sp^3d^2, other
Implicit valence	0, 1, 2, 3, 4, other
Formal charge	-5, -4, ..., 5, other
Presence in ring of size x	x = 3, 4, 5, 6, 7, 8, other
Number of rings atom is in	0, 1, 2, 3, other

Bond features	**Options**
Bond type	single, double, triple, aromatic

$$\text{COV-R}(C, \hat{C}, \delta) := \frac{1}{K} \left| \left\{ k \in \{1, \dots, K\} \mid \exists_{l \in \{1,\dots,L\}} \text{RMSD}(\hat{C}_l, C_k) < \delta \right\} \right| \tag{A49}$$

$$\text{COV-P}(C, \hat{C}, \delta) := \frac{1}{L} \left| \left\{ l \in \{1, \dots, L\} \mid \exists_{k \in \{1,\dots,K\}} \text{RMSD}(\hat{C}_l, C_k) < \delta \right\} \right| \tag{A50}$$

$$\text{AMR-R}(C, \hat{C}) := \frac{1}{K} \sum_{k \in \{1,\dots,K\}} \min_{l \in \{1,\dots,L\}} \text{RMSD}(\hat{C}_l, C_k) \tag{A51}$$

$$\text{AMR-P}(C, \hat{C}) := \frac{1}{L} \sum_{l \in \{1,\dots,L\}} \min_{k \in \{1,\dots,K\}} \text{RMSD}(\hat{C}_l, C_k) \tag{A52}$$

G.4 Chirality Correction

Given the four 3D coordinates around a chirality center denoted as $\mathbf{p_1}, \mathbf{p_2}, \mathbf{p_3}, \mathbf{p_4} \in \mathbb{R}^3$ with $\mathbf{p_i} = (x_i, y_i, z_i)$ for $i = 1, 2, 3, 4$, we can compute the oriented volume OV of the tetrahedron via

$$\begin{aligned} OV(\mathbf{p_1}, \mathbf{p_2}, \mathbf{p_3}, \mathbf{p_4}) &= \frac{1}{6} \cdot \det \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \end{bmatrix} \right) \\ &= \frac{1}{6} \cdot (\mathbf{p_1} - \mathbf{p_4}) \cdot ((\mathbf{p_2} - \mathbf{p_4}) \times (\mathbf{p_3} - \mathbf{p_4})) \,. \end{aligned} \tag{A53}$$

Following GeomMol [64] and ET-Flow [32], we can then compare the orientation of the volume given by $\text{sign}(OV)$ with the local chirality label produced by RDKit, which corresponds to a certain orientation as well (CW = +1 and CCW = -1) [74]. If the orientation of the volume differs from the RDKit label, we correct the chirality of the conformer by reflecting its positions against the z-axis.

G.5 Pareto Front

For all models, we generate conformers using 5, 10, 20 and 50 sampling steps on a single A100 GPU with a batch size of 128, following Refs. [32, 47]. The wall-clock time per generated sample is obtained by measuring the average time per batch and dividing by the batch size. As done in the original paper, we adopt DDIM sampling for MCF-S, MCF-B and MCF-L.

Table A9: Results on GEOM-QM9 for different generative models (number of parameters in parentheses). -R indicates Recall, -P indicates Precision. Best results in **bold**, second best underlined; our models are marked with an asterisk "$*$". Our results are averaged over three random seeds with standard deviation reported below. Other works do not report standard deviations.

Method	**COV-R [%]↑**		**AMR-R [Å]↓**		**COV-P [%]↑**		**AMR-P [Å]↓**	
	Mean	Median	Mean	Median	Mean	Median	Mean	Median
GeoMol (0.3M)	91.5	**100.0**	0.225	0.193	86.7	**100.0**	0.270	0.241
GeoDiff (1.6M)	76.5	**100.0**	0.297	0.229	50.0	33.5	0.524	0.510
Tors. Diff. (1.6M)	92.8	**100.0**	0.178	0.147	92.7	**100.0**	0.221	0.195
MCF-B (64M)	95.0	**100.0**	0.103	0.044	93.7	**100.0**	0.119	0.055
DMT-B (55M)	95.2	**100.0**	0.090	0.036	93.8	**100.0**	0.108	0.049
ET-Flow (8.3M)	**96.5**	**100.0**	0.073	0.030	94.1	**100.0**	0.098	0.039
$*$DiTMC+aPE-B (9.5M)	96.1 ±0.3	**100.0** ±0.0	0.074 ±0.001	0.030 ±0.001	95.4 ±0.1	**100.0** ±0.0	0.085 ±0.001	0.037 ±0.000
$*$DiTMC+rPE-B (9.6M)	96.3 ±0.0	**100.0** ±0.0	0.070 ±0.001	0.027 ±0.000	**95.7** ±0.1	**100.0** ±0.0	**0.080** ±0.000	0.035 ±0.000
$*$DiTMC+PE(3)-B (8.6M)	95.7 ±0.3	**100.0** ±0.0	**0.068** ±0.002	**0.021** ±0.001	93.4 ±0.2	**100.0** ±0.0	0.089 ±0.002	**0.032** ±0.001

G.6 Ensemble Properties

We adopt the property prediction task setup from MCF [47] and ET-Flow [32], where we draw a subset of 100 randomly sampled molecules from the test set of GEOM-DRUGS and generate $\min(2K, 32)$ conformers for a molecule with K ground truth conformers. Afterwards we relax the conformers using GFN2-xTB [75] and compare the Boltzmann-weighted properties of the generated and ground truth ensembles. More specifically, we employ xTB [75] to calculate the energy E, the dipole moment μ, the HOMO–LUMO gap $\Delta\epsilon$ and the minimum energy E_{min}. We repeat this procedure for three subsets each sampled with a different random seed and report the averaged median absolute error and standard deviation of the different ensemble properties.

H Additional Experimental Results

H.1 Results on GEOM-QM9

We report the results on GEOM-QM9 including standard deviations for all DiTMC models in Tab. A9.

H.2 Results on GEOM-DRUGS

We report the results on GEOM-DRUGS including standard deviations for all DiTMC models in Tab. A10. The median absolute error of ensemble properties on GEOM-DRUGS is shown in Tab. A11.

Additional results for the coverage vs. RMSD threshold analysis, including results for DiTMC+rPE and DiTMC+PE(3), are provided in Fig. A8, Fig. A9, and Fig. A10.

Additional results for the Pareto front analysis, including results for DiTMC+rPE and DiTMC+PE(3), are provided in Fig. A11, Fig. A12, and Fig. A13.

H.3 Results on GEOM-XL

We report the results on GEOM-XL including standard deviations for all DiTMC models in Tab. A12.

Table A10: Results on GEOM-DRUGS for different generative models (number of parameters in parentheses). -R indicates Recall, -P indicates Precision. Best results in **bold**, second best underlined; our models are marked with an asterisk "$*$". Our results are averaged over three random seeds with standard deviation reported below. Other works do not report standard deviations.

Method	**COV-R [%]↑**		**AMR-R [Å]↓**		**COV-P [%]↑**		**AMR-P [Å]↓**	
	Mean	Median	Mean	Median	Mean	Median	Mean	Median
GeoMol (0.3M)	44.6	41.4	0.875	0.834	43.0	36.4	0.928	0.841
GeoDiff (1.6M)	42.1	37.8	0.835	0.809	24.9	14.5	1.136	1.090
Tors. Diff. (1.6M)	72.7	80.0	0.582	0.565	55.2	56.9	0.778	0.729
MCF-S (13M)	79.4	87.5	0.512	0.492	57.4	57.6	0.761	0.715
MCF-B (64M)	84.0	91.5	0.427	0.402	64.0	66.2	0.667	0.605
MCF-L (242M)	84.7	92.2	0.390	**0.247**	66.8	71.3	0.618	0.530
DMT-L (150M)	**85.8**	**92.3**	**0.375**	0.346	67.9	72.5	0.598	0.527
ET-Flow - SS (8.3M)	79.6	84.6	0.439	0.406	75.2	81.7	0.517	0.442
$*$DiTMC+aPE-B (9.5M)	79.9 ±0.1	85.4 ±0.3	0.434 ±0.002	0.389 ±0.002	76.5 ±0.1	83.6 ±0.3	0.500 ±0.002	0.423 ±0.004
$*$DiTMC+rPE-B (9.6M)	79.3 ±0.1	84.6 ±0.2	0.444 ±0.002	0.400 ±0.002	77.2 ±0.1	84.6 ±0.2	0.492 ±0.001	0.414 ±0.002
$*$DiTMC+PE(3)-B (8.6M)	80.8 ±0.1	85.6 ±0.5	0.427 ±0.001	0.396 ±0.001	75.3 ±0.1	82.0 ±0.2	0.515 ±0.000	0.437 ±0.003
$*$DiTMC+aPE-L (28.2M)	79.2 ±0.1	84.4 ±0.2	0.432 ±0.003	0.386 ±0.003	77.8 ±0.1	85.7 ±0.5	0.470 ±0.001	0.387 ±0.003
$*$DiTMC+rPE-L (28.3M)	78.7 ±0.1	84.1 ±0.4	0.438 ±0.002	0.388 ±0.005	**78.1** ±0.1	**86.4** ±0.3	**0.466** ±0.001	**0.381** ±0.003
$*$DiTMC+PE(3)-L (31.1M)	80.8 ±0.3	85.6 ±0.1	0.415 ±0.003	0.376 ±0.001	76.4 ±0.2	82.6 ±0.3	0.491 ±0.002	0.414 ±0.004

I Additional Ablations

I.1 Gaussian vs. Harmonic Prior

As shown in Tab. A13, using the harmonic prior improves all metrics slightly for our models on GEOM-QM9. Using the harmonic prior however doesn't seem to be a crucial ingredient for the success of our method, as differences between Gaussian and Harmonic prior appear diminishing. As the results in Tab. A13 verify, our method can also be used with a simple Gaussian prior effectively. For larger molecular graphs the expensive eigendecomposition of the graph Laplacian required for the Harmonic prior could therefore be avoided, which helps scaling our approach more easily.

I.2 Conditioning Strategies on GEOM-QM9

To evaluate the effectiveness of various graph conditioning strategies in DiTMC , we compare the performance of different conditioning methods against a baseline model without any conditioning. In addition to conditioning strategies discussed in Sec. 4.1, we note that our conditioning GNN also produces edge-level representations, which can be used to define pair-wise graph conditioning tokens as

$$\textbf{bond-pair:} \quad \boldsymbol{c}_{ij}^{\mathcal{G}} = \begin{cases} \text{GNN}_{\text{edge}}(\mathcal{V}, \mathcal{E}) & \forall (i,j) \in \mathcal{E} \\ \bar{\boldsymbol{c}}^{\mathcal{G}} & \forall (i,j) \notin \mathcal{E}\,. \end{cases} \tag{A54}$$

These tokens only capture interactions between bonded atoms, i.e., when $(i,j) \in \mathcal{E}$. Conditioning tokens for non-bonded pairs are set to a learnable vector $\bar{\boldsymbol{c}}^{\mathcal{G}}$. Self-attention still operates on all atom pairs (i,j), even if they are not connected by a chemical bond.

Specifically, we ablate the following conditioning strategies:

Table A11: Median absolute error of ensemble properties between generated and reference conformers. Best results in **bold**, second best underlined; our models are marked with an asterisk "∗". Results for MCF, ET-Flow, and ours are averaged over three random seeds with standard deviation reported below. Other works do not report standard deviations.

Method	E **[kcal/mol]**↓	μ **[D]**↓	$\Delta\epsilon$ **[kcal/mol]**↓	E_{min} **[kcal/mol]**↓
GeoDiff (1.6M)	0.31	0.35	0.89	0.39
GeoMol (0.3M)	0.42	0.34	0.59	0.40
Torsional Diff. (1.6M)	0.22	0.35	0.54	0.13
MCF-L (242M)	0.68 ±0.06	0.28 ±0.05	0.63 ±0.05	0.04 ±0.00
ET-Flow (8.3M)	0.18 ±0.01	0.18 ±0.01	0.35 ±0.06	0.02 ±0.00
∗DiTMC+aPE-B (9.5M)	0.17 ±0.00	0.16 ±0.01	0.27 ±0.01	**0.01** ±0.00
∗DiTMC+aPE-L (28.2M)	**0.16** ±0.02	**0.14** ±0.03	0.27 ±0.01	**0.01** ±0.00
∗DiTMC+rPE-B (9.6M)	**0.16** ±0.03	0.16 ±0.03	0.29 ±0.02	0.02 ±0.00
∗DiTMC+rPE-L (28.3M)	**0.16** ±0.01	0.15 ±0.02	0.28 ±0.06	**0.01** ±0.00
∗DiTMC+PE(3)-B (8.6M)	0.18 ±0.01	0.18 ±0.01	0.27 ±0.03	0.02 ±0.00
∗DiTMC+PE(3)-L (31.1M)	0.17 ±0.01	**0.14** ±0.01	**0.25** ±0.01	**0.01** ±0.00

- **node only** conditioning using only atom-wise graph conditioning tokens $\boldsymbol{c}_i^{\mathcal{G}}$ (Eq. 6).
- **node & bond-pair** conditioning using both atom-wise graph conditioning tokens $\boldsymbol{c}_i^{\mathcal{G}}$ (Eq. 6) and pair-wise graph conditioning tokens $\boldsymbol{c}_{ij}^{\mathcal{G}}$ derived from edge-level representations of the conditioning GNN as discussed above (Eq. A54).
- **node & all-pair** conditioning using both atom-wise graph conditioning tokens $\boldsymbol{c}_i^{\mathcal{G}}$ (Eq. 6) and pair-wise graph conditioning tokens $\boldsymbol{c}_{ij}^{\mathcal{G}}$ based on geodesic graph distances (Eq. 7).

As reported in Tab. A14, all our proposed conditioning strategies significantly reduce the average minimum RMSD (AMR) for both recall (AMR-R) and precision (AMR-P) using DiTMC+aPE-B, compared to the unconditioned baseline.

Notably, the "node & all-pair" strategy achieves the best overall performance, with the lowest AMR values. These results highlight the strength of the all-pair conditioning strategy, which leverages graph geodesics to incorporate information from all atom pairs, rather than restricting conditioning to directly connected nodes or bonded pairs. This comprehensive approach captures more global structural information, thereby improving both precision and recall. See Appendix L for a more in-depth analysis.

I.3 Index Positional Encoding (iPE)

Tab. A15 compares a variant including index positional encoding (iPE) from classic transformer architectures with DiTMC+aPE-B on GEOM-QM9. Specifically, we use the node index and encode it via sinusoidal encodings into the tokens $\mathcal{H}$ *before* the first DiTMC block, similar to embedding the absolute positions via aPE. Since the molecular graphs are generated from SMILES strings via

Table A12: Out-of-distribution generalization results on GEOM-XL for models trained on GEOM-DRUGS. -R indicates Recall, -P indicates Precision. Best results in **bold**, second best underlined; our models are marked with an asterisk "∗". Our results are averaged over three random seeds with standard deviation reported below. Other works do not report standard deviations.

Method	*75 / 77 mols*				*102 mols*			
	AMR-R [Å]↓		**AMR-P [Å]↓**		**AMR-R [Å]↓**		**AMR-P [Å]↓**	
	Mean	Median	Mean	Median	Mean	Median	Mean	Median
Tor. Diff. (1.6M)	1.93	1.86	2.84	2.71	2.05	1.86	2.94	2.78
MCF-S (13M)	2.02	1.87	2.90	2.69	2.22	1.97	3.17	2.81
MCF-B (64M)	1.71	1.61	2.69	2.44	2.01	1.70	3.03	2.64
MCF-L (242M)	1.64	1.51	2.57	2.26	1.97	1.60	2.94	2.43
ET-Flow (8.3M)	2.00	1.80	2.96	2.63	2.31	1.93	3.31	2.84
∗DiTMC+aPE-B (9.5M)	1.68 ±0.00	1.47 ±0.02	2.59 ±0.00	2.24 ±0.01	1.96 ±0.00	1.60 ±0.03	2.90 ±0.00	2.48 ±0.03
∗DiTMC+aPE-L (28.2M)	**1.56** ±0.01	**1.28** ±0.01	**2.47** ±0.00	2.14 ±0.01	1.88 ±0.01	**1.51** ±0.02	**2.81** ±0.00	**2.30** ±0.02
∗DiTMC+rPE-B (9.6M)	1.69 ±0.01	1.41 ±0.03	2.52 ±0.00	**2.11** ±0.00	1.97 ±0.01	1.61 ±0.01	2.86 ±0.00	2.33 ±0.01
∗DiTMC+rPE-L (28.3M)	1.66 ±0.03	1.37 ±0.01	**2.47** ±0.00	2.18 ±0.02	1.96 ±0.02	1.61 ±0.02	2.82 ±0.00	2.42 ±0.02
∗DiTMC+PE(3)-B (8.6M)	1.73 ±0.01	1.55 ±0.01	2.71 ±0.00	2.35 ±0.01	1.98 ±0.01	1.67 ±0.02	3.03 ±0.00	2.60 ±0.01
∗DiTMC+PE(3)-L (31.1M)	1.57 ±0.01	1.46 ±0.01	2.60 ±0.00	2.27 ±0.02	**1.85** ±0.02	1.58 ±0.03	2.93 ±0.00	2.53 ±0.03

Table A13: Ablation of PE strategies and Gaussian (G) and Harmonic (H) prior on GEOM-QM9. We report mean coverage (COV) at a threshold of 0.5Å, and mean average minimum RMSD (AMR) for Recall -R and Precision -P. Best results in **bold**. All results are averaged over three random seeds.

Method	**COV-R [%]↑**		**AMR-R [Å]↓**		**COV-P [%]↑**		**AMR-P [Å]↓**	
	G	H	G	H	G	H	G	H
DiTMC+aPE-B	96.2	96.1	0.074	0.073	95.2	95.4	0.087	0.085
DiTMC+rPE-B	96.0	**96.3**	0.073	0.070	95.2	**95.7**	0.084	**0.080**
DiTMC+PE(3)-B	95.7	95.7	0.069	**0.068**	93.5	93.4	0.090	0.089

RDKit and RDKit has to some extend a canonical ordering, this information can be used by the transformer architecture. However, index positional encoding breaks permutation equivariance (as we show in Tab. A15). This might be undesirable as permutation equivariance is one of the fundamental symmetries when learning on graphs. Since the ordering of atoms in a SMILES string is not uniquely defined, the trained network depends on the used framework for parsing the SMILES string or even a particular software version. We use RDKit (version 2024.9.5) for parsing SMILES strings to graphs.

Nevertheless, our DiTMC+aPE-B using iPE can effectively exploit the information contained in atom indices assigned by RDKit. A version of DiTMC+aPE-B without atom-pair conditioning but iPE achieves comparable performance to DiTMC+aPE-B using geodesic distances as atom-pair conditioning (pairwise conditioning). As our pairwise conditioning strategy is similarly or more effective than iPE but additionally preserves permutation equivariance, it should be preferred over iPE and we don't use iPE in any of our other experiments.

Table A14: Ablation of conditioning strategies using DiTMC+aPE-B on GEOM-QM9. -R indicates Recall, -P indicates Precision. Best results in **bold**. Our results are averaged over three random seeds with standard deviation reported below.

Method	**COV-R [%]↑**		**AMR-R [Å]↓**		**COV-P [%]↑**		**AMR-P [Å]↓**	
	Mean	Median	Mean	Median	Mean	Median	Mean	Median
DiTMC+aPE-B (no conditioning)	68.6 ±1.0	91.7 ±2.1	0.405 ±0.005	0.325 ±0.004	36.8 ±0.5	36.4 ±2.2	0.729 ±0.006	0.703 ±0.007
DiTMC+aPE-B (node only)	96.3 ±0.0	**100.0** ±0.0	0.079 ±0.000	0.037 ±0.000	93.2 ±0.2	**100.0** ±0.0	0.112 ±0.001	0.051 ±0.000
DiTMC+aPE-B (node & bond-pair)	**96.5** ±0.1	**100.0** ±0.0	0.077 ±0.001	0.035 ±0.001	95.3 ±0.2	**100.0** ±0.0	0.092 ±0.001	0.046 ±0.002
DiTMC+aPE-B (node & all-pair)	96.1 ±0.3	**100.0** ±0.0	**0.074** ±0.001	**0.030** ±0.001	**95.4** ±0.1	**100.0** ±0.0	**0.085** ±0.001	**0.037** ±0.000

Table A15: Ablating index positional encoding (iPE) on GEOM-QM9 for different conditioning strategies (in brackets). To show the effect of atom permutations, we include results with randomly permuted atom indices (**perm.**). -R indicates Recall, -P indicates Precision. Best results in **bold**. Our results are averaged over three random seeds with standard deviation reported below.

Method	**COV-R [%]↑**		**AMR-R [Å]↓**		**COV-P [%]↑**		**AMR-P [Å]↓**	
	Mean	Median	Mean	Median	Mean	Median	Mean	Median
DiTMC+aPE-B (node only)	96.3 ±0.0	**100.0** ±0.0	0.079 ±0.000	0.037 ±0.000	93.2 ±0.2	**100.0** ±0.0	0.112 ±0.001	0.051 ±0.000
DiTMC+aPE-B (node only), **perm.**	96.3 ±0.0	**100.0** ±0.0	0.079 ±0.000	0.037 ±0.000	93.2 ±0.2	**100.0** ±0.0	0.112 ±0.001	0.051 ±0.000
DiTMC+aPE+iPE-B (node only)	**96.6** ±0.4	**100.0** ±0.0	0.079 ±0.002	0.037 ±0.001	**95.5** ±0.1	**100.0** ±0.0	0.093 ±0.001	0.046 ±0.001
DiTMC+aPE+iPE-B (node only), **perm.**	82.3 ±1.1	**100.0** ±0.0	0.229 ±0.008	0.108 ±0.005	60.0 ±0.9	61.8 ±1.4	0.493 ±0.008	0.416 ±0.011
DiTMC+aPE-B (node & pairwise)	96.1 ±0.3	**100.0** ±0.0	**0.074** ±0.001	**0.030** ±0.001	95.4 ±0.1	**100.0** ±0.0	**0.085** ±0.001	**0.037** ±0.000
DiTMC+aPE-B (node & pairwise), **perm.**	96.1 ±0.3	**100.0** ±0.0	**0.074** ±0.001	**0.030** ±0.001	95.4 ±0.1	**100.0** ±0.0	**0.085** ±0.001	**0.037** ±0.000

J Analysis of training loss as a function of latent time

In this section, we provide details for the analysis in Fig. 2C in the main part of the paper. We investigate the effect of the positional embeddings and self-attention formulations on the accuracy of the model. We therefore take pre-trained models on GEOM-QM9 and compute the training loss (as detailed in Algorithm 1) averaged over 1000 samples drawn randomly from the GEOM-QM9 validation set. We compute the loss for 30 logarithmically spaced values of $t_i = 1 - 10^{x_i}$, where $x_i \in [-1.8, 0]$ with uniform spacing. We skip the stochastic term in the loss as is done while sampling from the ODE.

As detailed in Fig. A5, we observe empirically that equivariance leads to a decreased loss close to the data distribution after training. This explains why our equivariant model more often succeeds to produce samples with increased fidelity, as depicted in figure Fig. 2B.

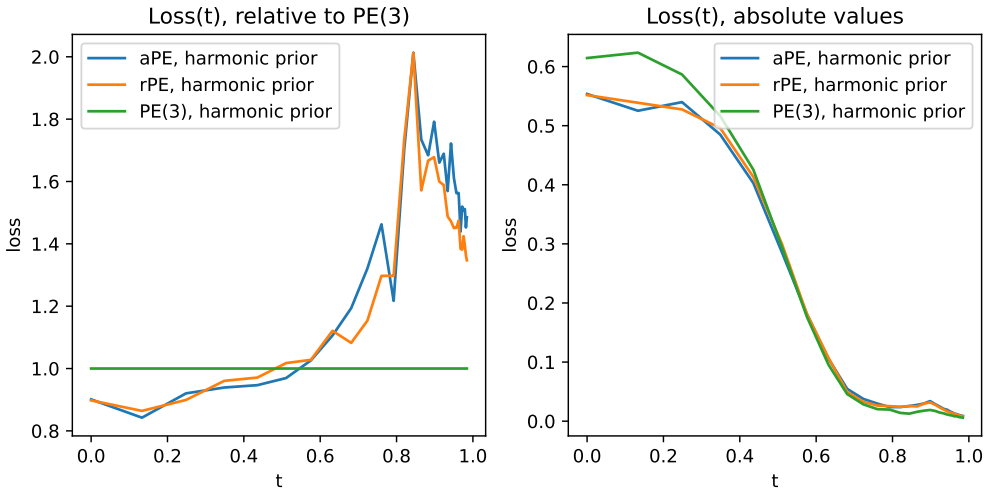

Figure A5: Loss as a function of time comparing different PE strategies. Results averaged over 1000 samples randomly drawn from the GEOM-QM9 validation set. **Left:** loss relative to PE(3) as a baseline. In the important regime close to the data distribution, the model PE(3) has lower loss, yielding higher sample fidelity. **Right:** absolute loss values for all PEs. The loss decreases close to the data distribution for all models.

We further note, that absolute loss values for models trained with all our PE strategies decrease as latent time increases (see Fig. A5). This is expected, as conditional vector fields for each data sample will start to interact more strongly moving away from the data distribution. Our weighted loss (see Appendix B) effectively penalizes errors close to the data distribution during training and helps with keeping the error low in this important regime.

K SMILES Classification Experiment via Conditioning GNN

The molecular graphs in our dataset are represented as SMILES strings. A critical requirement of DiTMC is the ability to distinguish distinct molecular graphs or SMILES representations through conditioning. In this section, we investigate whether our conditioning GNN is capable of learning the necessary information to distinguish between different SMILES strings for the generation of matching molecular conformers.

As a proxy evaluation task, we assess whether the conditioning network alone can function as a classifier of SMILES strings. To this end, we construct two training datasets: a toy dataset comprising three specific SMILES strings of a hydroxyl group moving along a carbon chain (`C(O)CCCCCCC`, `CC(O)CCCCCC`, `CCC(O)CCCCC`), and a larger set consisting of 1000 randomly sampled SMILES strings drawn from the GEOM-QM9 validation set. Each SMILES string becomes a seperate class, so for each class there is exactly one example in the training data. The classification task is performed on the graph representations of the molecules, employing the same feature set and GNN architecture utilized in our conditioning GNN (see Appendix D.3 and Appendix G.2) followed by a simple classification head.

We train different models with a batch size of 3 for 5000 epochs on the curated toy dataset and batch size of 64 over 250 epochs on the GEOM-QM9 subset. We report classification accuracy on the training sets directly to evaluate the model's discriminative capacity. Furthermore, we explore whether conditioning weights obtained from an end-to-end trained model retain discriminatory power by freezing them and attaching a linear classification head.

Our results, as shown in Tab. A16, reveal that a simple linear classifier lacking message-passing capabilities fails to distinguish certain SMILES strings. Overall, our results indicate that a simple two-layer GNN effectively captures the necessary conditioning information through end-to-end training. Fig. A6 shows that without GNN layers, isomers will be misclassified.

Table A16: We measure the discriminative power of our conditioning graph network on a training set of 1000 randomly sampled SMILES strings from the GEOM-QM9 validation set, as well as a toy dataset of 3 different SMILES strings. We investigate the required number of message passing layers, as well as using pre-trained weights from an end-to-end trained model.

GNN layers	Weight init	Trainable	Accuracy (GEOM-QM9)	Accuracy (Toy Data)
0	random	trainable	0.887	0.333
1	random	trainable	0.999	0.666
2	random	trainable	**1.000**	**1.000**
2	random	frozen	0.980	**1.000**
2	pre-trained	frozen	**1.000**	**1.000**

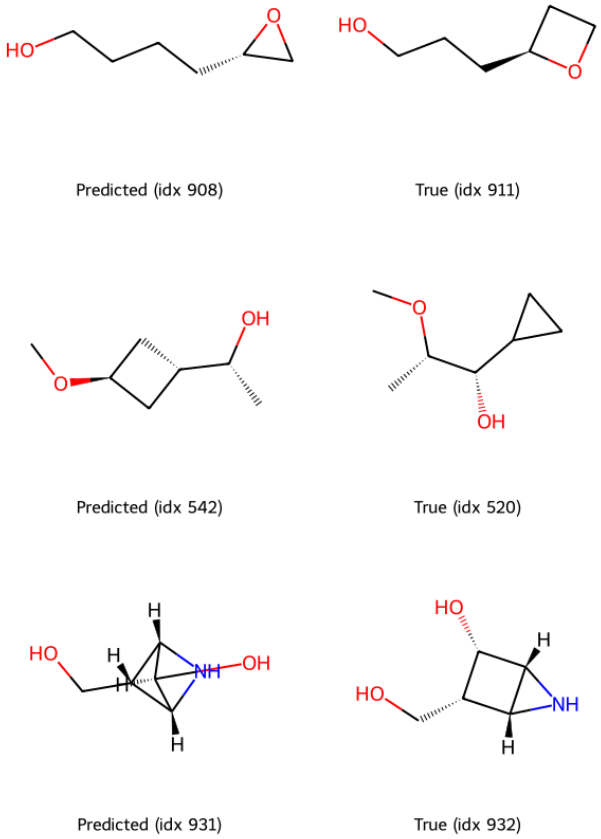

Figure A6: Misclassified exampled for SMILES classification experiment. We randomly pick 3 examples, which are misclassified by a classification head without any GNN layers. We show that GNN layers are essential for correct classification of isomers.

L Analysis of Sampling Trajectories

Fig. A7 compares the generative performance of DiTMC models trained on the GEOM-QM9 dataset under two different conditioning strategies: (1) node-only conditioning and (2) node plus pairwise conditioning, where we use the all-pair conditioning based on geodesic graph distances. Each row in the figure corresponds to one example molecule selected from the test set. The molecules are chosen to maximize the root mean squared deviation (RMSD) between the final generated structures of the

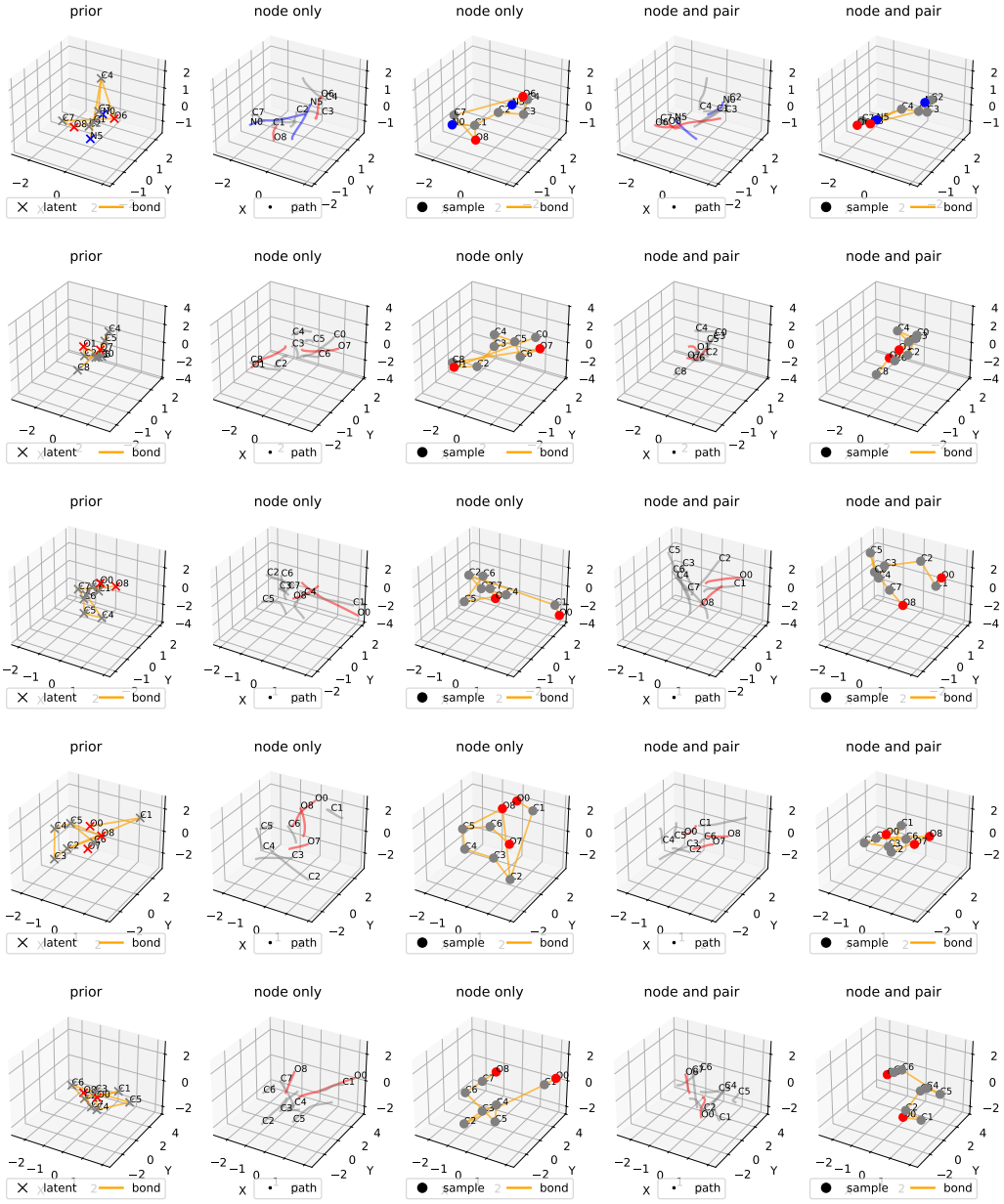

Figure A7: Comparison of molecular generation with node-only versus node- and pair-wise (all-pair) conditioning on GEOM-QM9. Each row shows a prior sample, sampling trajectories, and final generated structure for both models; pairwise conditioning preserves bonds from the conditioning graph $\mathcal{G}$.

two models. This selection highlights cases where the differences between the conditioning schemes are most pronounced.

Within each row, we display a sequence of images: the initial prior sample, followed by the intermediate trajectory of the ODE sampling process over 50 sampling steps for the node-only conditioned model, and the resulting final structure (with predicted bonds rendered in yellow). This sequence is repeated for the node- and pair-wise conditioned model, allowing a side-by-side visual comparison of the generation dynamics and final outputs.

The results reveal a consistent pattern: models trained with node-only conditioning fail to preserve bonding patterns from the conditioning graph $\mathcal{G}$. This manifests as bond stretching or atom permutation in the final structure. In contrast, the model, that is conditioned on pairwise geodesic distances, produces geometries that adhere more closely to expected chemical structure and the given bonds. We note that if atoms are simply permuted by the model using only node conditioning, the generated structure might still be valid in terms of the combination of generated 3D positions and atom types. The degraded performance of node conditioning versus node and pair-conditioning can therefore in part be explained by the used RMSD and Coverage metrics, which are not invariant to permutations of atoms.

Our findings still underscore the importance of incorporating both node-level and pairwise features in molecular generative models, in particular when agreement with the given conditioning on a bond graph is essential.

M Visualization

Fig. A14, Fig. A15, and Fig. A16 provide a visual comparison of conformers generated by MCF, ET-Flow, and DiTMC against the corresponding ground-truth reference conformers for the GEOM-QM9, GEOM-DRUGS and GEOM-XL datasets, respectively. For each dataset, we randomly select six reference conformers from the test split and generate conformers using each method. Finally, we apply rotation alignment of the generated conformers with their corresponding reference conformer.

N Code and Data Availability

The code and data to reproduce the main results of this paper, can be downloaded from here: `https://doi.org/10.5281/zenodo.15489212`, or via github: `https://github.com/ML4MolSim/dit_mc`.

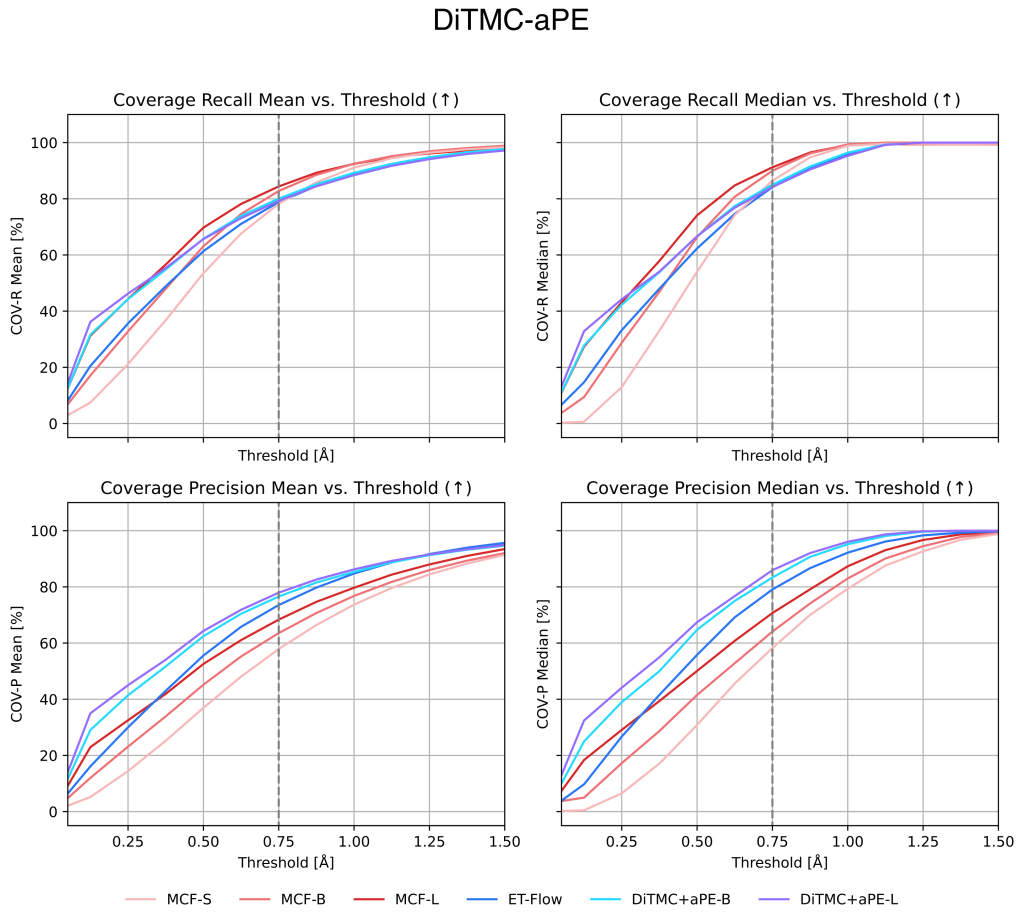

Figure A8: Coverage (COV) for precision ("-P") and recall ("-R") as a function of RMSD threshold δ for DiTMC+aPE and other state-of-the-art methods on GEOM-DRUGS. The vertical dashed line denotes the RMSD threshold $\delta = 0.75$ commonly used for evaluation on the GEOM-DRUGS dataset.

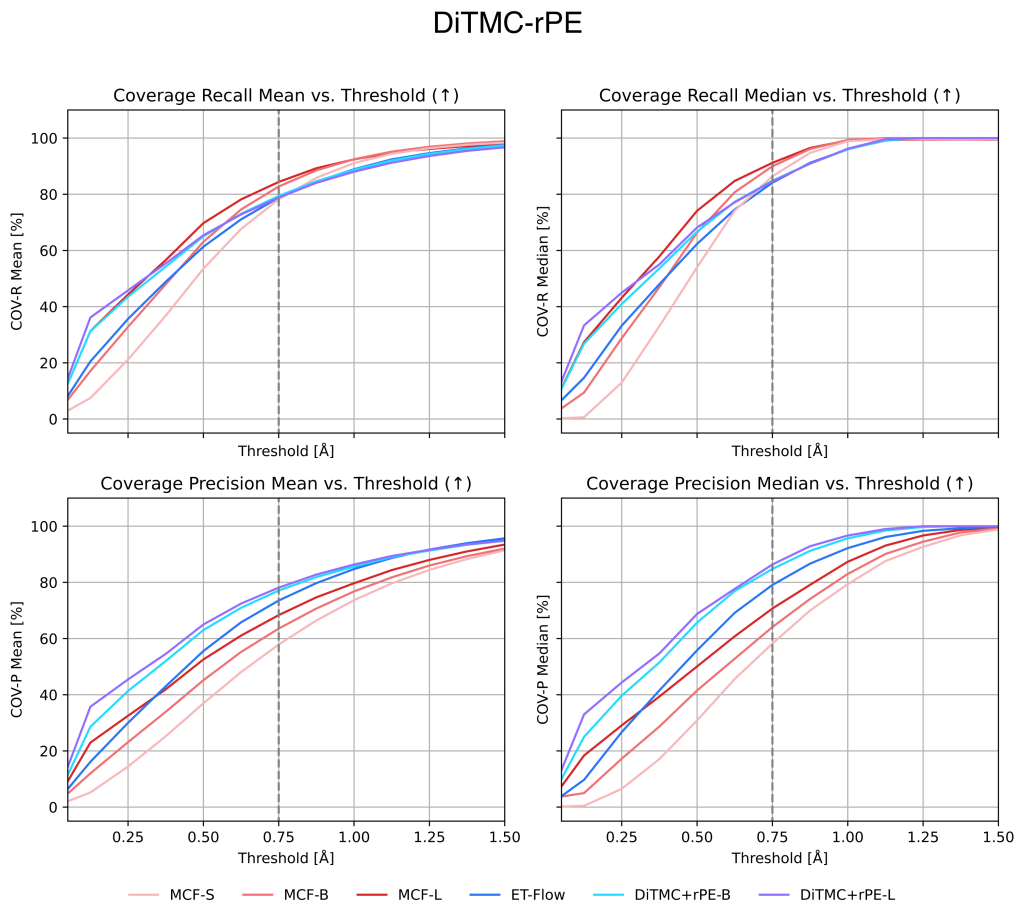

Figure A9: Coverage (COV) for precision ("-P") and recall ("-R") as a function of RMSD threshold δ for DiTMC+rPE and other state-of-the-art methods on GEOM-DRUGS. The vertical dashed line denotes the RMSD threshold $\delta = 0.75$ commonly used for evaluation on the GEOM-DRUGS dataset.

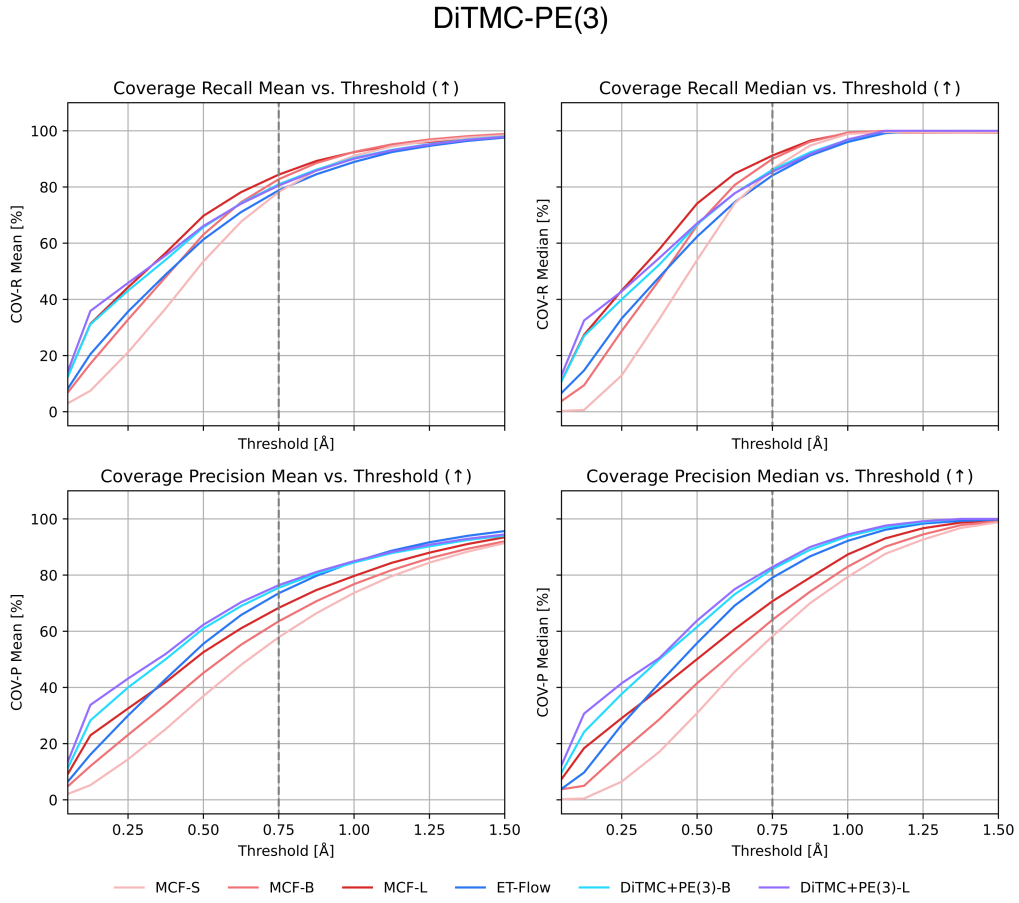

Figure A10: Coverage (COV) for precision ("-P") and recall ("-R") as a function of RMSD threshold δ for DiTMC+PE(3) and other state-of-the-art methods on GEOM-DRUGS. The vertical dashed line denotes the RMSD threshold $\delta = 0.75$ commonly used for evaluation on the GEOM-DRUGS dataset.

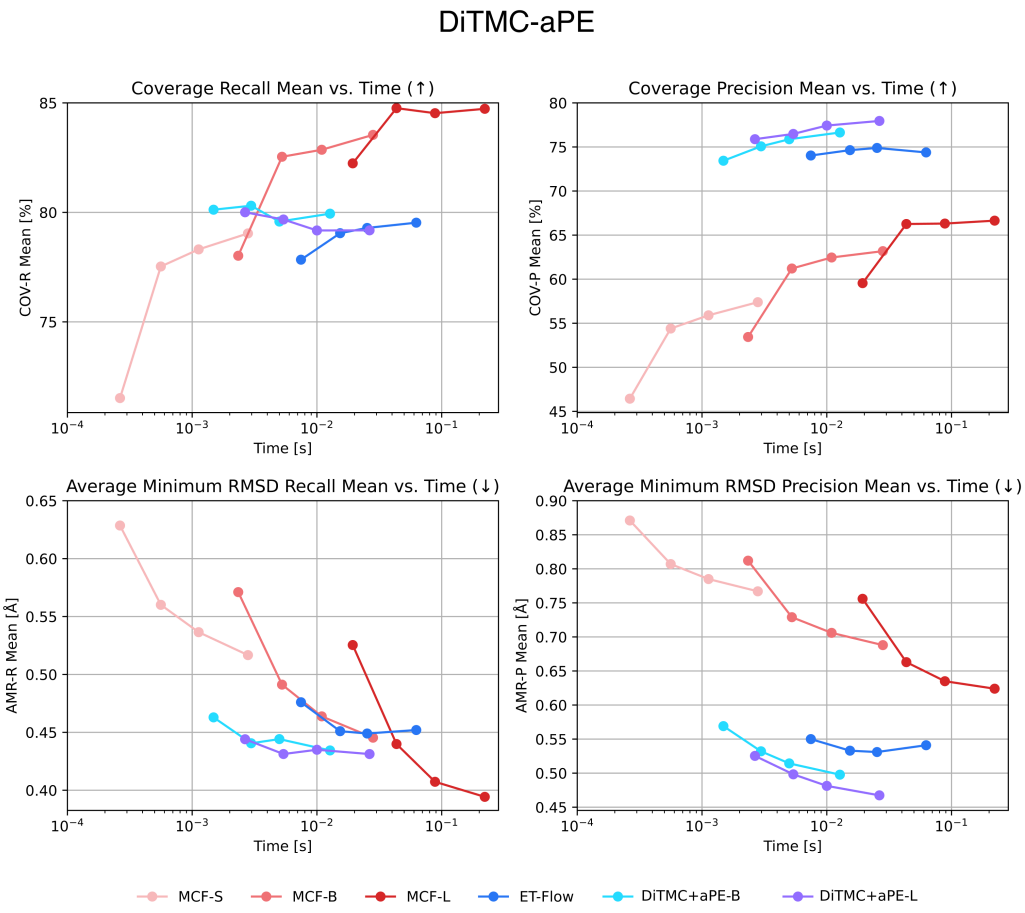

Figure A11: Average minimum RMSD (AMR) for precision ("-P") and recall ("-R") as a function of wall clock time per generated conformer. For each model, markers from left to right correspond to an increasing number of sampling steps during generation. Here, we follow Refs. [32, 47] and use 5, 10, 20, and 50 sampler steps.

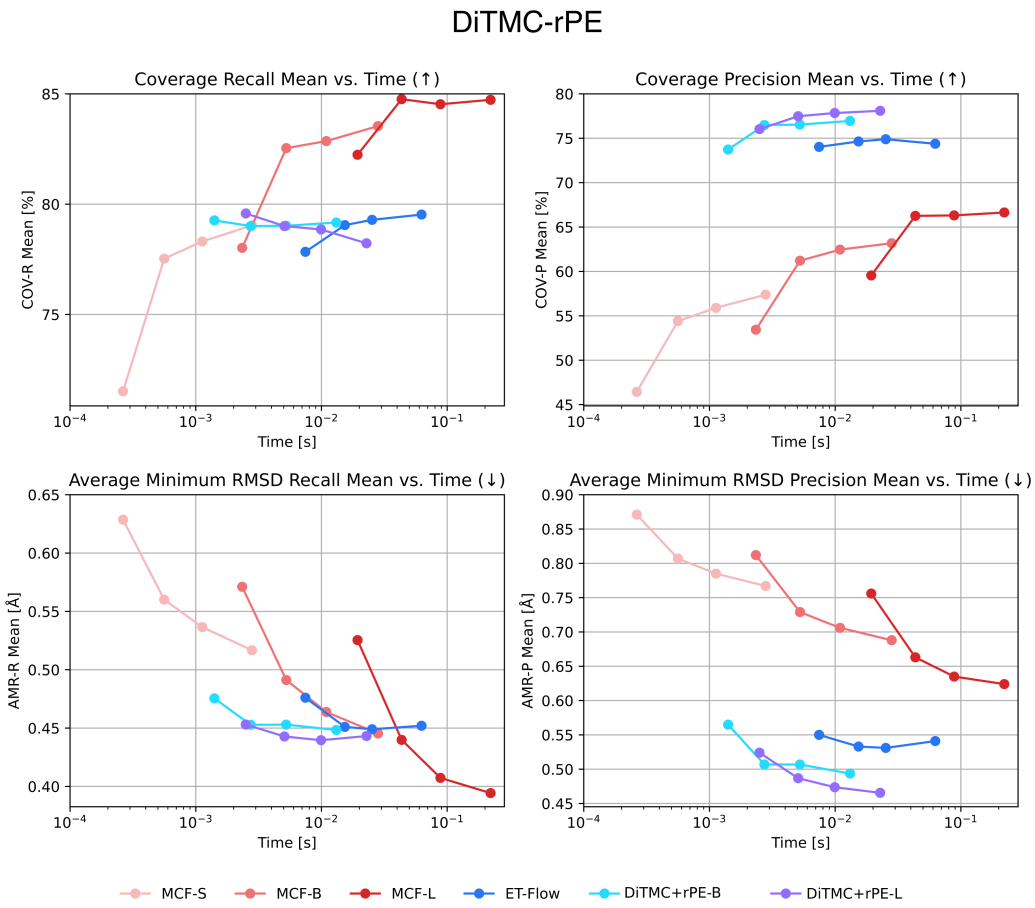

Figure A12: Average minimum RMSD (AMR) for precision ("-P") and recall ("-R") as a function of wall clock time per generated conformer. For each model, markers from left to right correspond to an increasing number of sampling steps during generation. Here, we follow Refs. [32, 47] and use 5, 10, 20, and 50 sampler steps.

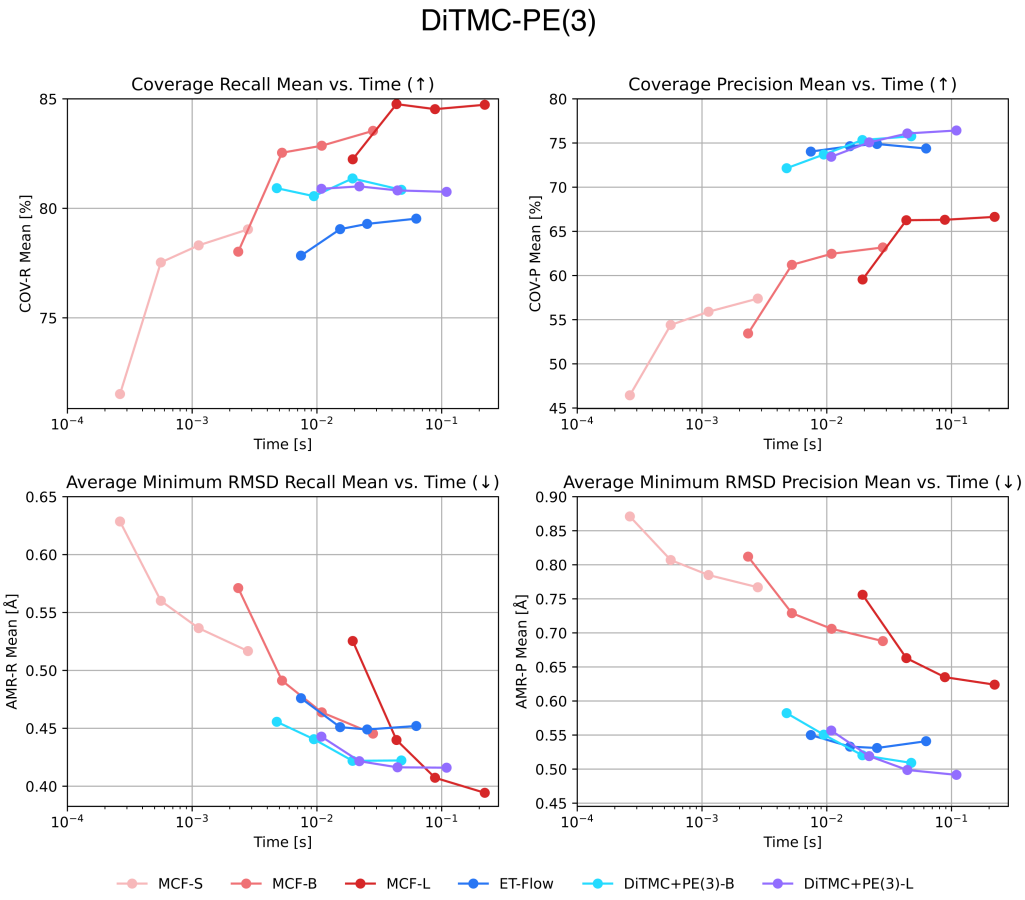

Figure A13: Average minimum RMSD (AMR) for precision ("-P") and recall ("-R") as a function of wall clock time per generated conformer. For each model, markers from left to right correspond to an increasing number of sampling steps during generation. Here, we follow Refs. [32, 47] and use 5, 10, 20, and 50 sampler steps.

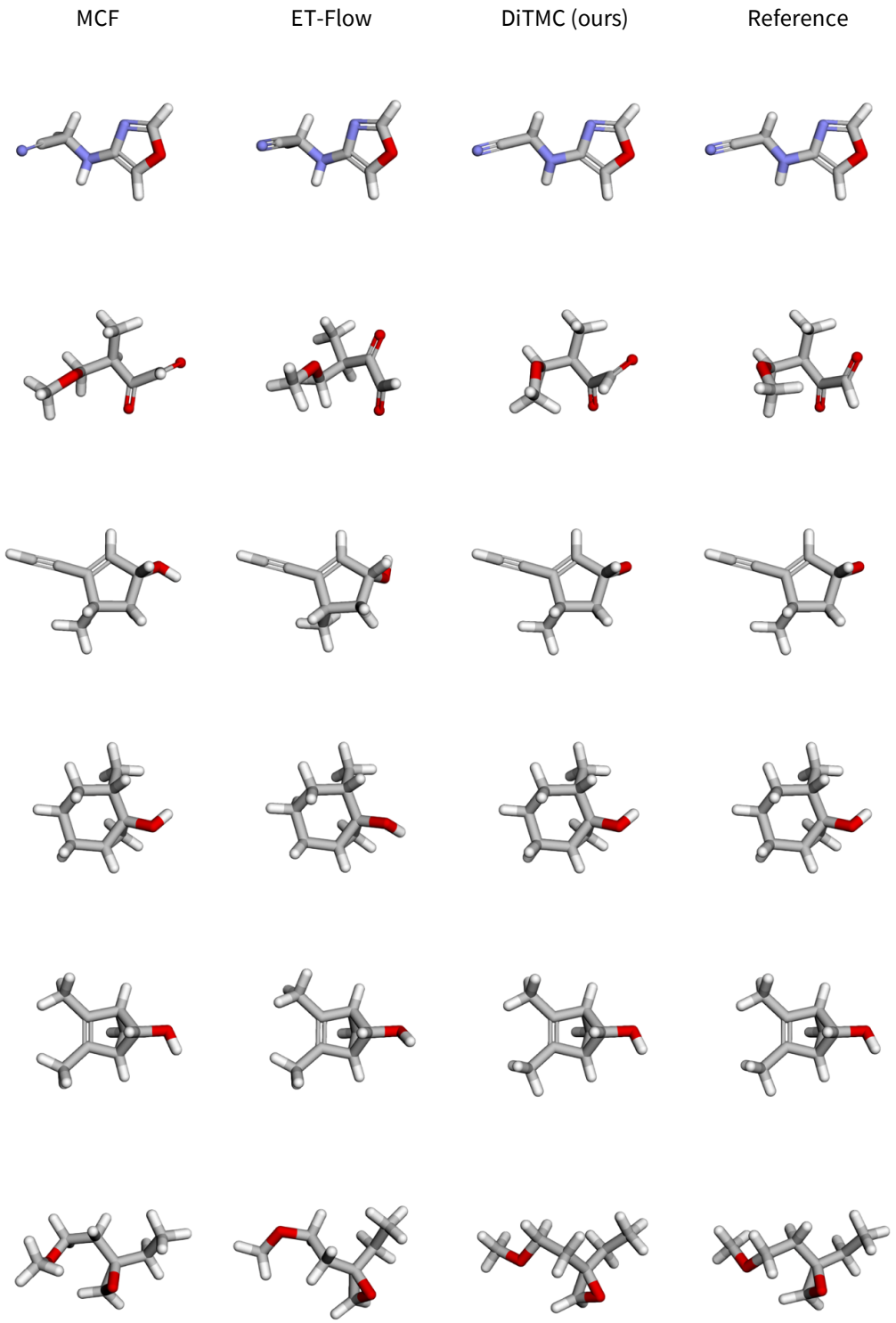

Figure A14: Comparison of conformers generated by MCF, ET-Flow, and DiTMC against ground-truth reference conformers from GEOM-QM9. The generated conformers are rotationally aligned with their corresponding reference conformer to facilitate comparison. **From left to right**: generated conformers from MCF, ET-Flow, DiTMC, and the corresponding ground-truth reference conformers.

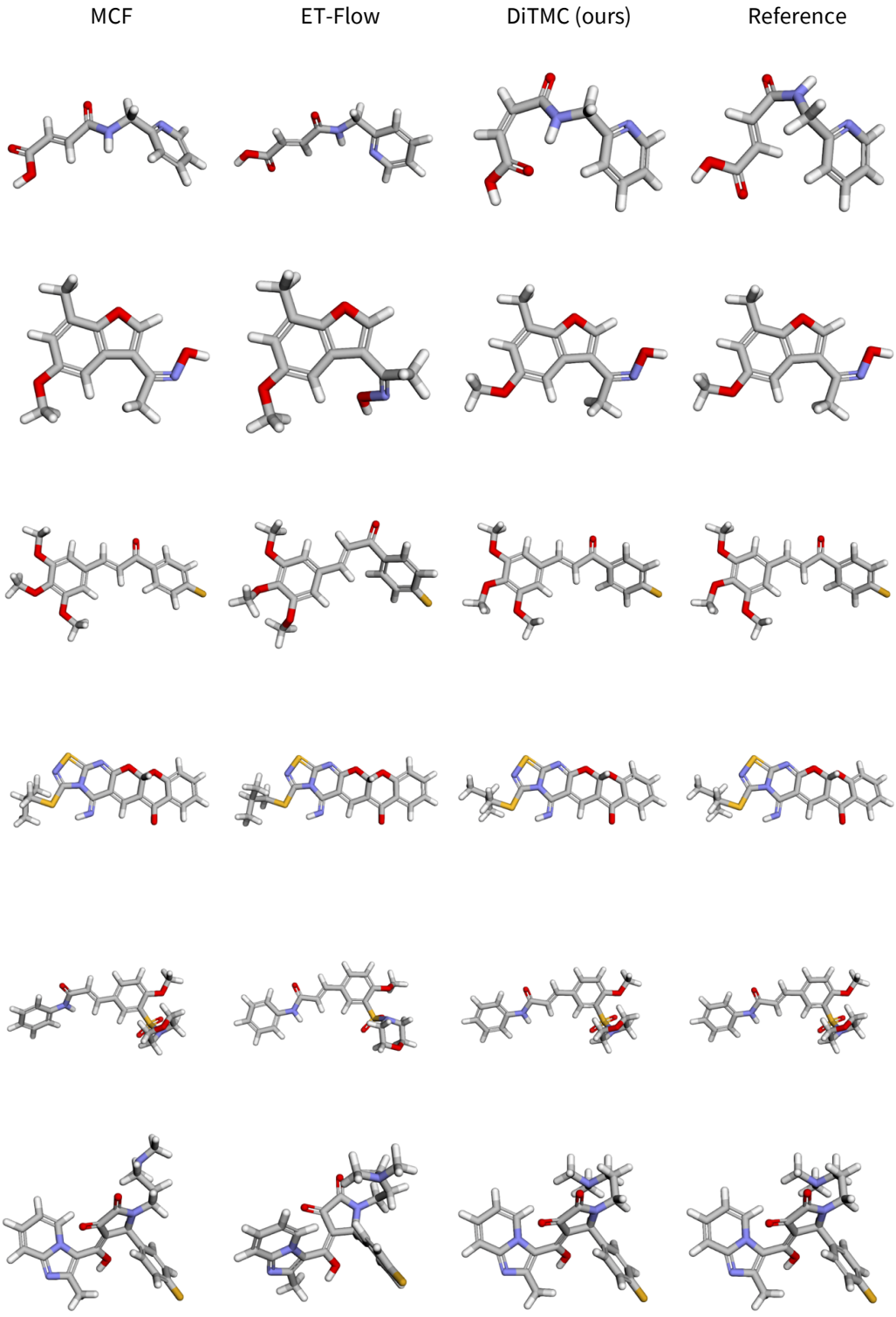

Figure A15: Comparison of conformers generated by MCF, ET-Flow, and DiTMC against ground-truth reference conformers from GEOM-DRUGS. The generated conformers are rotationally aligned with their corresponding reference conformer to facilitate comparison. **From left to right**: generated conformers from MCF, ET-Flow, DiTMC, and the corresponding ground-truth reference conformers.

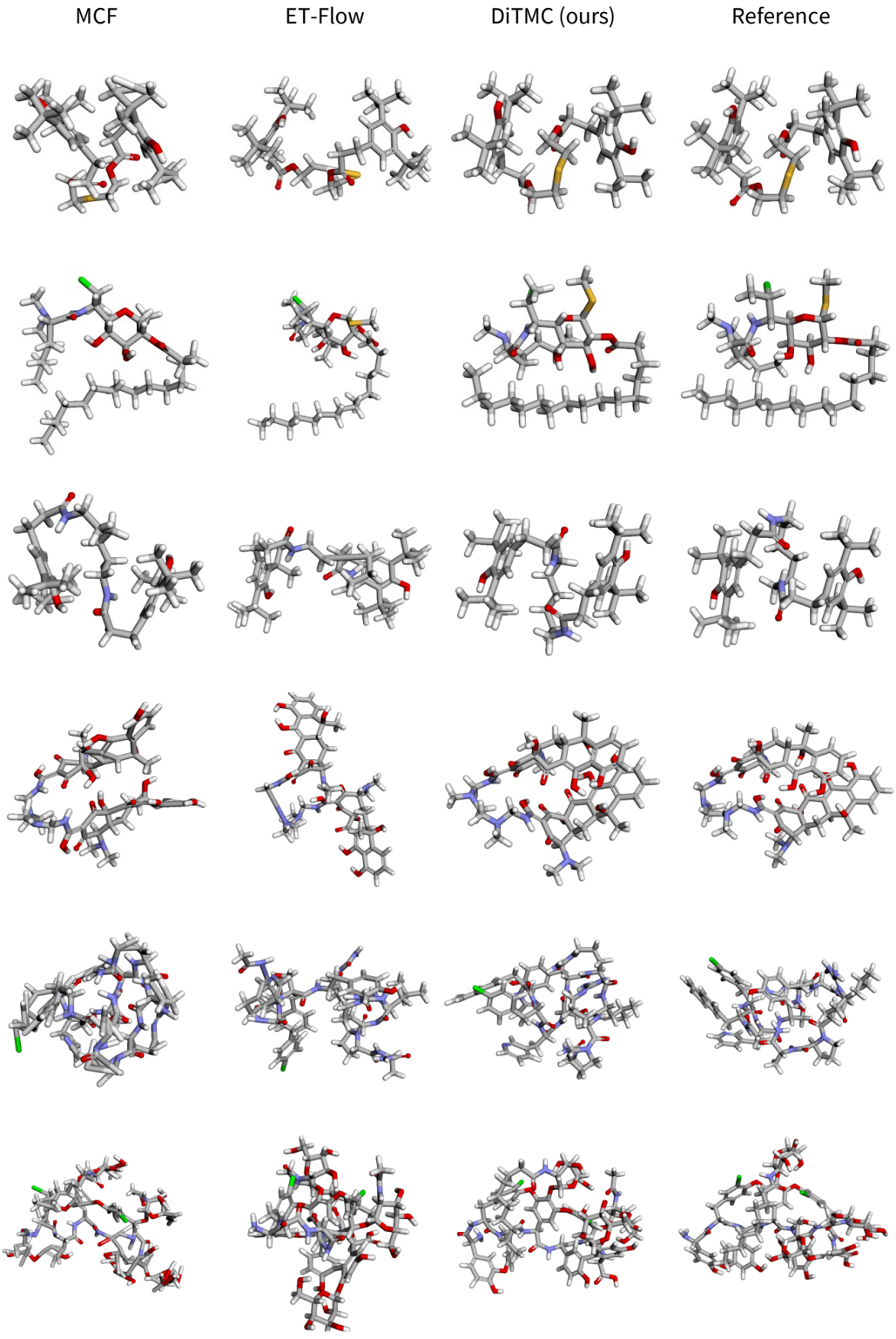

Figure A16: Comparison of conformers generated by MCF, ET-Flow, and DiTMC against ground-truth reference conformers from GEOM-XL. The generated conformers are rotationally aligned with their corresponding reference conformer to facilitate comparison. **From left to right**: generated conformers from MCF, ET-Flow, DiTMC, and the corresponding ground-truth reference conformers.