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.

1 Introduction

A molecule's conformation (3D atomic arrangement) dictates its physical properties and biological activity, which is vital for drug discovery and material design. Traditional methods like Molecular Dynamics and Markov Chain Monte Carlo are computationally expensive because they require extensive simulation steps to explore the conformational space. In contrast, generative ML models are more efficient, allowing for direct sampling of high-quality 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 general diffusion transformer (DiT) [13] consistently delivers 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.

*Equal contribution.

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

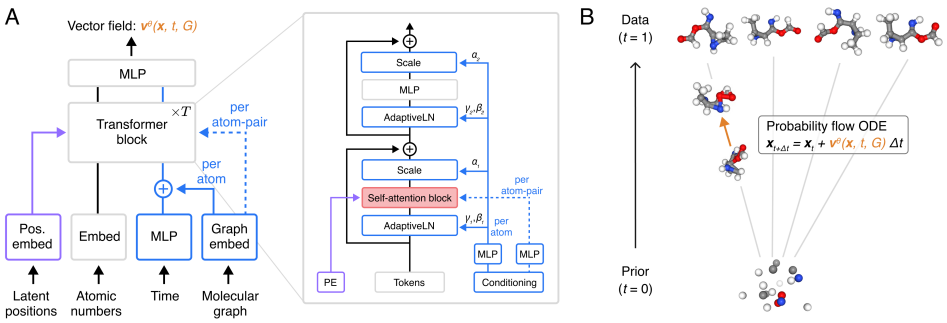

Figure 1: **(A)** Diffusion transformer for molecular conformer generation (DiTMC), with interchangeable self-attention blocks and positional embeddings (PEs); we evaluate various combinations as detailed in the main text. **(B)** DiTMC predicts a velocity per atom, used to model a probability flow ODE,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

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 DiT variants achieve state-of-the-art (SOTA) 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 (SOTA) 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 SOTA in domains such as 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].

3 Preliminaries

3.1 Molecular Conformers

A molecule can be associated with a 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. What is missing is the exact spatial arrangement of the $N = |\mathcal{V}|$ atoms 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 conditional probability distribution $p(\boldsymbol{x} \mid \mathcal{G})$, which will guide the design of our model architectures (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, 49, 50]. 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, 51], 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 [52]. 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, 53], 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).

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 $p(\boldsymbol{x}|\mathcal{G})$, 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 vector ("velocity") per atom, which is extracted from a readout layer. The complete architecture is summarized in Fig. 1A, with training details given in Appendix A.

4.1 Embeddings

We begin by initializing the node features using learnable embeddings $\boldsymbol{e}(z_i) \in \mathbb{R}^H$, where $z_i \in \mathbb{N}_+$ denotes the atomic number of atom i [35]. To encode the current atomic positions $\mathcal{R} = \{\vec{r}_1, \dots \vec{r}_N \,|\, \vec{r}_i \in \mathbb{R}^3\}$ of the latent state $\boldsymbol{x}_t$, we use positional embeddings (PEs). We examine a representative range of PEs that vary in the number of Euclidean symmetries they respect by construction, which affects how the latent representations transform under translations and rotations. Without loss of generality, we assume the positions are centered such that the center of mass vanishes (see section A.1).

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

$$\boldsymbol{p}_i^{\text{aPE}} = \text{MLP}(\vec{r}_i)\,, \tag{5}$$

where $\vec{r}_i \in \mathbb{R}^3$ is the Cartesian position of the i-th atom. This kind of PE is neither rotationally nor translationally invariant, and serves as a baseline without any symmetry constraints.

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

$$\boldsymbol{p}_{ij}^{\text{rPE}} = \text{MLP}(\vec{r}_{ij})\,. \tag{6}$$

This formulation ensures translational invariance but not rotational invariance.

Adapting ideas from equivariant message passing neural networks like PaiNN [54] or NequIP [55], we construct SO(3)-equivariant pairwise Euclidean Positional Embeddings (PE(3)) as a concatenation of $L+1$ components

$$\boldsymbol{p}_{ij}^{\text{PE(3)}} = \bigoplus_{\ell=0}^{L} \boldsymbol{\phi}_\ell(r_{ij}) \,\odot\, \boldsymbol{Y}_\ell(\hat{r}_{ij})\,, \tag{7}$$

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 $(\boldsymbol{\phi}_\ell \odot \boldsymbol{Y}_\ell) \in \mathbb{R}^{(2\ell+1)\times H}$ and (after concatenation) $\boldsymbol{p}_{ij}^{\text{PE(3)}} \in \mathbb{R}^{(L+1)^2\times H}$. Under rotation of the input positions, these PEs transform equivariantly (see Appendix B.2.3). 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.

4.2 Conditioning

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

$$\boldsymbol{c}^t = \text{MLP}(t)\,. \tag{8}$$

Atom-wise conditioning tokens are obtained from a GNN inspired by the processor module of the MeshGraphNet (MGN) framework [56]:

$$\textbf{atom-wise:} \quad \boldsymbol{c}_i^{\mathcal{G}} = \text{GNN}_{\text{node}}(\mathcal{V}, \mathcal{E})\,, \tag{9}$$

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

We define a pair-wise conditioning mechanism inspired by the Graphormer architecture [57]

$$\textbf{pair-wise:} \quad \boldsymbol{c}_{ij}^{\mathcal{G}} = \text{MLP}\left(s(i,j)\right), \tag{10}$$

where $s(i,j)$ denotes the graph geodesic (i.e., the shortest path between atoms i and j via edges in $\mathcal{G}$). This formulation allows conditioning on all atom pairs, even if they are not directly connected.

4.3 DiT Block

The embeddings and conditioning information are processed through multiple DiT blocks. Every pair-wise information is used during the self-attention update (see Eq. 11, whereas atom-wise information and the time conditioning signal are injected via adaptive layer norms (see Eq. 12).

Each DiT block transforms the current node features (input tokens) $\mathcal{H} = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \,|\, \boldsymbol{h}_i \in \mathbb{R}^H\}$ into a set of output tokens $\mathcal{H}'$, which serve as input for the next block. Additional inputs consist of positional embeddings $\mathcal{P}$ (representing the position of nodes in Euclidean space), as well as conditioning tokens $\mathcal{C}$, such as those encoding time ($\boldsymbol{c}^t$) and graph structure ($\boldsymbol{c}^{\mathcal{G}}$).

The central operation in each DiT block is a self-attention update,

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

followed by a node-wise Multi-Layer Perceptron (MLP). Here, $\mathcal{C}^{\text{Pair}}$ are special conditioning tokens acting on pairs of atoms (see Eq. 10). For PE(3) embeddings, we use an SO(3)-equivariant self-attention mechanism (subsubsection B.2.1) together with an SO(3)-equivariant MLP (Sec. B.4).

Similar to applications of DiTs in image synthesis, adaptive layer norm (AdaLN) and adaptive scale (AdaScale) [13] are used for conditioning (see Fig. 1A) based on per-atom bias and scaling parameters,

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

5 Experiments

Datasets and Metrics We conduct our experiments on the GEOM dataset [58], comprising QM9 (133,258 small molecules) and AICures (304,466 drug-like molecules). Reference conformers are generated using CREST [59]. Drug-like molecules exhibit greater structural diversity, including more rotatable bonds and multiple stereocenters. Data splits are taken from Ref. [60].

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 D.3 provides further details on the calculation of metrics. Following ET-Flow [32] and GeomMol [60] we also apply chirality correction (see Appendix D.4).

Conditioning Strategies To assess the impact of graph conditioning, we compare three different conditioning strategies against a baseline without conditioning. As discussed in Sec. E.1, we compare conditioning solely on atom-wise information (Eq. 9) with an extended scheme that also incorporates pairwise geodesic graph distances (Eq. 10). We also ablate conditioning on pairwise information extracted from our conditioning GNN for each edge corresponding to a chemical bond. We find all variants to be effective, but our proposed combination of geodesic distances and atom-wise information to perform best (see Appendix Tab. A8). 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.

Ablating Self-Attention and Positional Embedding Strategies The modular structure of DiTMC allows efficient exploration of the design space through variations in the PE strategy and associated attention blocks (see Sec. 4). All DiTMC models differ in the choice of PEs and self-attention formulation. For the architecture using PE(3), we reduce the number of heads to match the parameter count across models. Tab. 1 shows results on GEOM-QM9. All DiTMC models produce diverse,

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 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
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**

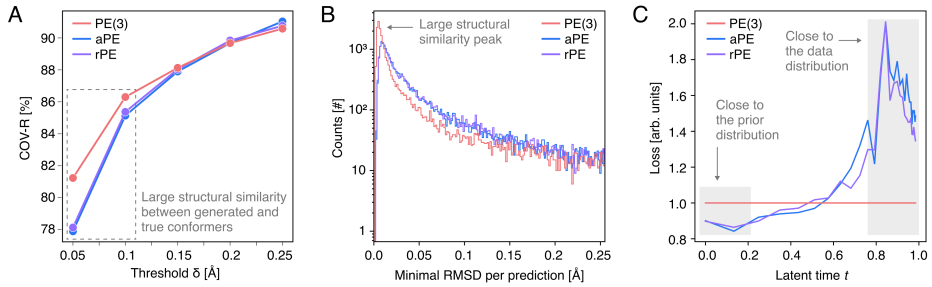

Figure 2: Analysis of equivariant (PE(3)) and non-equivariant (aPE, rPE) model formulations. **(A)** Mean coverage recall (COV-R) versus threshold δ (root mean square deviation (RMSD) to any reference conformer). **(B)** Histogram of the minimal RMSD per generated sample. **(C)** Loss as a function of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

high quality samples, outperforming the current SOTA in all AMR metrics and COV-P. We use the harmonic prior of Ref. [61] throughout, which yields improved results (see Appendix Tab. A15), we use it in all following experiments.

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 below.

Equivariance improves the fidelity of samples. We compute COV-R as a function of threshold δ for different PEs and self-attention blocks (Fig. 2A). SO(3)-equivariant attention with PE(3) outperforms the other variants at low δ, indicating that many of the generated geometries closely match the ground-truth structures. This appears as a leftward shift in the distribution of the minimal RMSD found per generated structure (Fig. 2B) and aligns with the observation that the PE(3) model gives better AMR-R and AMR-P values (see Tab. 1). To better understand this behavior, we analyze the loss over time t and find that models with non-equivariant PEs exhibit higher error near the data distribution ($t = 1$) (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 PE(3) model is approximately 3.5 times slower than models using aPE or rPE, while at inference time, the factor is ~ 3 (see Appendix Tab. A5). 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.

Drug-like Molecules Finally, we evaluate our architecture on the challenging GEOM-DRUGS dataset. For our experiments, we define a small base ("B") and a large ("L") model variant (see Appendix B). All DiTMC-L variants achieve SOTA results on GEOM-DRUGS for all precision metrics (see Tab. 2). Importantly, also the smaller ("B") models outperform the ETFlow-SS of similar size, underlining the effectiveness of our proposed approach (see Appendix Tab. A10).

Table 2: 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 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
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-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

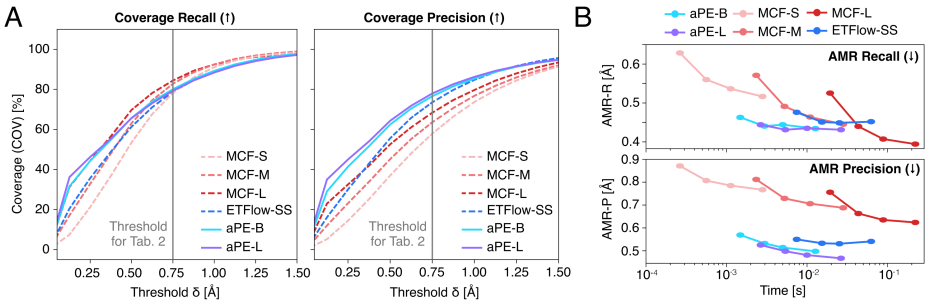

Figure 3: **(A)** Coverage as function of RMSD threshold δ and **(B)** average minimum RMSD (AMR) mean vs. time per conformer for aPE and other SOTA models. Per model markers from left to right correspond to 5, 10, 20, and 50 sampler steps following Refs. [32, 47]. Note, that the original MCF paper reports results with two different samplers. Benchmark results (Tab. 2) are obtained with DDPM sampler (1000 steps) and AMR vs. time results are reported for DDIM sampler (5–50 steps).

Accuracy vs. RMSD Threshold We analyse the coverage versus RMSD threshold δ for the aPE models (Fig. 3A). For small thresholds aPE-B and aPE-L outperform all other methods for coverage recall and precision. For large thresholds ($\rho \geq 0.4$Å) MCF-L starts to outperform aPE models in terms of coverage recall. For coverage precision, aPE-B and aPE-L perform better than all other methods for all thresholds. In particular for smaller thresholds, we see a strong benefit due to model scaling. We find similar results for rPE and PE(3) (see Appendix Fig. A10 and Fig. A11).

Pareto-Front We investigate the Pareto front of accuracy and computational efficiency, by plotting accuracy against wall-clock time per sample (see Fig. 3B and Appendix Sec. D.5). As measure for accuracy we consider the average minimum RMSD (AMR), since it is independent of the RMSD threshold. For precision, 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 the aPE model. For recall, aPE shifts the Pareto front for little compute times, but most accurate results at increased cost are obtained by MCF-L. Similar results are obtained for rPE (Fig. A13), but benefits for PE(3) are limited due to high computational cost of equivariant operations (Fig. A14).

Ensemble properties. To complement RMSD-based geometric evaluation with a chemically meaningful assessment, we report the median absolute error (MAE) of ensemble properties between generated and reference conformers. The RMSD metric can penalize conformers that differ due to rotations or atom reordering, but are otherwise chemically equivalent. For our analysis we follow the protocol of MCF [47] and ET-Flow [32] (see Appendix E.2) and compare energy E, dipole moment μ, HOMO-LUMO gap $\Delta\epsilon$, and minimum energy E_{min}. Our aPE-L model predicts ensemble properties more accurately than all baselines, highlighting the physical validity of our generated structures (see Tab. 3). In particular, MCF-L, which shows better performance for recall metrics, is outperformed by a large margin (up to a factor of 4 for energy).

Table 3: Median absolute error of ensemble properties between generated and reference conformers for GEOM-DRUGS. E, $\Delta\epsilon$, E_{min} are in kcal/mol, and μ is in debye. 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 Tab. A11 for results including standard deviations.

Method	$E\downarrow$	$\mu\downarrow$	$\Delta\epsilon\downarrow$	$E_{\text{min}}\downarrow$
OMEGA	0.68	0.66	0.68	0.69
Torsional Diff.	0.22	0.35	0.54	0.13
MCF-L	0.68	0.28	0.63	0.04
ET-Flow	0.18	0.18	0.35	0.02
$*$DiTMC+aPE-B	0.17	0.16	**0.27**	**0.01**
$*$DiTMC+aPE-L	**0.16**	**0.14**	**0.27**	**0.01**

Table 4: Out-of-distribution generalization results on GEOM-XL for models trained on GEOM-DRUGS reporting median RMSD in Å. Best results in **bold**, second best underlined; our models with an asterisk ($*$). Our results are averaged over three random seeds. See Tab. A13 for results including standard deviations.

Method	**AMR-R** $\downarrow$	**AMR-P** $\downarrow$
	102 molecules	
MCF - L	1.60	2.43
$*$DiTMC+aPE-B	1.60	2.49
$*$DiTMC+aPE-L	**1.51**	**2.30**
	77 molecules	
MCF - L	1.51	2.26
$*$DiTMC+aPE-B	1.47	2.24
$*$DiTMC+aPE-L	**1.28**	**2.14**

Generalization performance. We assess the generalization of our model to larger and unseen molecules using the GEOM-XL dataset [46]. It comprises 102 molecules of size $N > 100$ atoms, whereas the molecules in the training data contain only $N = 44$ atoms on average. Following MCF [47] we report the generalization performance for all 102 molecules and a subset of 77 molecules. Our models perform on par or better to the previously best-performing method MCF-L while using only a fraction of the parameters (see Tab. 4). Other baselines (like ET-Flow) are outperformed by a larger margin (see Appendix Tab. A13).

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 SOTA results on GEOM-QM9 and GEOM-DRUGS, excelling in both precision and physical validity. Through ablation studies, we assess the impact of incorporating Euclidean symmetries into DiTs. 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 DiT 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. 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] 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.

[50] 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.

[51] 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.

[52] 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.

[53] 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.

[54] 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.

[55] 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.

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

[57] 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.

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

[59] 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.

[60] 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.

[61] 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`.

[62] 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.

[63] 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.

[64] 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`.

[65] 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.

[66] 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.

[67] Tero Karras, Miika Aittala, Tuomas Kynkäänniemi, Jaakko Lehtinen, Timo Aila, and Samuli Laine. Guiding a diffusion model with a bad version of itself. *Advances in Neural Information Processing Systems*, 37:52996–53021, 2024.

[68] Tuomas Kynkäänniemi, Miika Aittala, Tero Karras, Samuli Laine, Timo Aila, and Jaakko Lehtinen. Applying guidance in a limited interval improves sample and distribution quality in diffusion models. *arXiv preprint arXiv:2404.07724*, 2024.

A Training

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

Instead of predicting the conditional vector field $\boldsymbol{u}_t(\boldsymbol{x}|\boldsymbol{x}_0, \boldsymbol{x}_1)$ directly, we choose to reparametrize the network such that it predicts the clean sample $\boldsymbol{x}_1$. Similar to Ref. [53], 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 raise 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 point of time 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. Here, we employ the Kabsch-Algorithm. For full geometry alignment "$\mathrm{GeometryAlign}(\boldsymbol{x}_A, \boldsymbol{x}_B)$", we do the following

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

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

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

In words, the operations from above first center both point clouds at the origin and then align them as much as possible via a rotation. This procedure also 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 $\boldsymbol{x}_1^\theta$

$\boldsymbol{x}_0 \sim p_0$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, $t \sim \mathcal{U}(0,1)$, $\boldsymbol{R} \sim \mathrm{SO}(3)$
$\boldsymbol{x}_0, \boldsymbol{x}_1 \leftarrow \mathrm{GeometryAlign}(\boldsymbol{x}_0, \boldsymbol{x}_1)$ ▷ 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 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.1). However, only DiTMC with 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 for both GEOM QM9 and drugs experiments, comparing a larger value of 0.5 and a smaller value of 0.05. We set the σ parameter 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 initial learning rate of $\mu_0 = 10^{-5}$. It is increased up to μ_{max} via a linear learning rate warmup up over the first 1% of training steps. Afterwards, it is decreased to μ_{min} via a cosine decay schedule. We use $\mu_{\text{max}} = 3 \times 10^{-4}$ and $\mu_{\text{min}} = 0$ for GEOM-QM9 and $\mu_{\text{max}} = 1 \times 10^{-4}$ and $\mu_{\text{min}} = 1 \times 10^{-5}$ for GEOM-DRUGS. We use a batch size of 128 for all data sets and models. All models on QM9 are trained for 250 epochs in total. For GEOM-DRUGS we determine the maximal number of epochs from the epochs the larger model "L" can perform within a fixed compute budget of 9 days. Thus, we train the PE(3) variants for 10 epochs and the aPE variants for 50 epochs. Fig. A4 shows the number of gradient steps per day that can be performed with the different DiTMC variants.

A.1 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. [62], 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. [63]). 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 sampling in section C.

A.2 Compute and Training Times

Tab. A5 compares training and inference time for our models (see also Fig. 3A for a visual comparison). Despite having a similar parameter count, the equivariant PE(3)-B model is approximately 3.5 times slower than models using aPE or rPE, while at inference time, the factor is ~ 3. The effect is even more pronounced for our large models, where the equivariant model is almost 5 times slower during training compared to its non-equivariant counterpart with a similar number of parameters. Even our large non-equivariant model is significantly faster to train and sample from than the equivariant base model with far fewer parameters.

All our 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). Due to compute constraints, our non-equivariant (DiTMC+aPE) models are trained for 50 epochs on GEOM drugs. The equivariant models are trained for 10 epochs on GEOM drugs resulting in a similar compute budget for training in terms of GPU hours. Our maximal compute budget per model is 9 days on a single Nvidia H100 GPU.

B Architecture Details

Tab. A5 shows the architectural details for our base and large model variants. Tab. A6 contains details about the MLPs we use throughout our architecture. We first describe the building blocks of our

non-equivariant DiTMC implementation and then its equivariant counterpart. Both architectures rely on conditioning tokens $\mathcal{C}$, for the time $\boldsymbol{c}^t \in \mathbb{R}^H$, per-node $\boldsymbol{c}_i^{\mathcal{G}} \in \mathbb{R}^H$, and per atom-pair $\boldsymbol{c}_{ij}^{\mathcal{G}} \in \mathbb{R}^H$. See main text section 4.2 for more details. Following Ref. [13], we use adaptive layer norm (AdaLN) and adaptive scale (AdaScale) to include per-node conditioning based on time t and molecular graph information. To that end, we construct conditioning tokens

$$\boldsymbol{c}_i = \boldsymbol{c}^t + \boldsymbol{c}_i^{\mathcal{G}}. \tag{A21}$$

B.1 Non Equivariant DiT

In the non-equivariant DiTMC formulations with aPE and rPE, we have the following set of tokens $\mathcal{H} = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \,|\, \boldsymbol{h}_i \in \mathbb{R}^H\}$.

B.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

$$\boldsymbol{q} = \boldsymbol{W}_q \tilde{\boldsymbol{h}}\,, \qquad \boldsymbol{k} = \boldsymbol{W}_k \tilde{\boldsymbol{h}}\,, \qquad \boldsymbol{v} = \boldsymbol{W}_v \tilde{\boldsymbol{h}}\,, \tag{A22}$$

where $\boldsymbol{W}_q, \boldsymbol{W}_k, \boldsymbol{W}_v \in \mathbb{R}^{H\times H}$ are trainable weight matrices and $\tilde{\boldsymbol{h}}$ is either identical to $\boldsymbol{h}$ or combines it with a PE (see below). We define a slightly modified similarity kernel

$$\text{sim}(\boldsymbol{q}, \boldsymbol{k}, \boldsymbol{u}) = \exp\left(\frac{\boldsymbol{q}^\intercal \cdot (\boldsymbol{k} \odot \boldsymbol{u})}{\sqrt{H}}\right), \tag{A23}$$

Table A5: Architectural details for different PE strategies on GEOM-QM9 and GEOM-DRUGS. Times are measured on GEOM-QM9 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-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

Table A6: Architecture 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. The use of gated GELU ensures equivariance (see section Sec. B.4).

Name	**Layers**	**Hidden Dim**	**Out Dim**	**Activation**	**Input**
DiT Block	2	$4H$	H	GELU	Tokens
SO(3) DiT 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. ositions $\vec{r}_i$
rPE embedding	2	H	H	SiLU	Rel. positions $\vec{r}_{ij}$
GNN embedding	2	H	H	SiLU	Node/edge features

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:

$$\mathrm{ATT}(\mathcal{H})_i = \frac{\sum_{j=1}^{N} \mathrm{sim}(\boldsymbol{q}_i, \boldsymbol{k}_j, \boldsymbol{u}_{ij}) \cdot (\boldsymbol{v}_j \odot \boldsymbol{u}_{ij})}{\sum_{j=1}^{N} \mathrm{sim}(\boldsymbol{q}_i, \boldsymbol{k}_j, \boldsymbol{u}_{ij})} \,. \tag{A24}$$

Queries, keys, and values are obtained with Eq. A22 from different (position-encoded) tokens $\tilde{\boldsymbol{h}}_i$ depending on the chosen PEs; further, the injected pair-wise information $\boldsymbol{u}_{ij}$ differs:

$$\tilde{\boldsymbol{h}}_i = \begin{cases} \boldsymbol{h}_i + \boldsymbol{p}_i^{\mathrm{aPE}} & \text{for absolute PEs}\,, \\ \boldsymbol{h}_i & \text{for relative PEs}\,, \end{cases} \qquad \text{and} \qquad \boldsymbol{u}_{ij} = \begin{cases} \boldsymbol{c}_{ij}^{\mathcal{G}} & \text{for absolute PEs}\,, \\ \boldsymbol{c}_{ij}^{\mathcal{G}} + \boldsymbol{p}_{ij}^{\mathrm{rPE}} & \text{for relative PEs}\,. \end{cases} \tag{A25}$$

Here $\boldsymbol{c}_{ij}^{\mathcal{G}} \in \mathbb{R}^H$ are pair-wise graph conditioning tokens (see Eq. 10) and $\boldsymbol{p}_i^{\mathrm{aPE}} \in \mathbb{R}^H$ and $\boldsymbol{p}_{ij}^{\mathrm{rPE}} \in \mathbb{R}^H$ are the absolute and relative PEs described above (see Eqs. 5 and 6).

B.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

$$\mathrm{AdaLN}(\boldsymbol{h}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \mathrm{LN}(\boldsymbol{h}) \odot (1 + \boldsymbol{\alpha}) + \boldsymbol{\beta}, \tag{A26}$$

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

Adaptive Scale is defined as

$$\mathrm{AdaScale}(\boldsymbol{h}, \boldsymbol{\gamma}) = \boldsymbol{h} \odot \boldsymbol{\gamma}. \tag{A27}$$

In each DiTMC block, we calculate

$$\boldsymbol{\alpha}_{1i}, \boldsymbol{\beta}_{1i}, \boldsymbol{\gamma}_{1i}, \boldsymbol{\alpha}_{2i}, \boldsymbol{\beta}_{2i}, \boldsymbol{\gamma}_{2i} = \boldsymbol{W}\big(\mathrm{SiLU}(\boldsymbol{c}_i)\big), \tag{A28}$$

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 DiT block behaves like the identity function at initialization.

B.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

$$\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i = \boldsymbol{W}\big(\mathrm{SiLU}(\boldsymbol{c}_i)\big), \tag{A29}$$

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

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

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.

B.2 SO(3) Equivariant DiT

Following the notation in Ref. [64], we denote $\mathrm{SO}(3)$-equivariant tokens as $\mathcal{H} = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \,|\, \mathbb{R}^{(L+1)^2 \times H}\}$, 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. [64] for an in-depth introduction into equivariant features. For all our experiments we use maximal degree of $L = 1$.

B.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 (subsubsection B.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 $\mathrm{SO}(3)$. 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 [54] or TorchMDNet [34]. An $\mathrm{SO}(3)$-equivariant formulation of self-attention is then given as

$$\mathrm{ATT}_{\mathrm{SO(3)}}(\mathcal{H})_i = \frac{\sum_{j=1}^{N} \mathrm{sim}(\boldsymbol{q}_i, \boldsymbol{k}_j, \boldsymbol{u}_{ij}) \cdot (\hat{\boldsymbol{u}}_{ij} \otimes \boldsymbol{v}_j)}{\sum_{j=1}^{N} \mathrm{sim}(\boldsymbol{q}_i, \boldsymbol{k}_j, \boldsymbol{u}_{ij})}\,, \tag{A30}$$

where equivariant queries, keys and values can be calculated similarly to Eq. A22 and '$\otimes$' denotes a Clebsch-Gordan (CG) tensor product contraction [64]. The dot-product in the similarity measure is taken along both feature and degree axes, such that the overall update preserves equivariance (see Appendix for details). Tokens and scaling vectors are calculated as

$$\tilde{\boldsymbol{h}}_i = \boldsymbol{h}_i\,, \qquad \boldsymbol{u}_{ij} = \boldsymbol{\phi}(r_{ij}) \odot \boldsymbol{c}^{\mathcal{G}}_{ij}\,, \qquad \hat{\boldsymbol{u}}_{ij} = \boldsymbol{p}^{\mathrm{PE(3)}}_{ij} \odot \boldsymbol{c}^{\mathcal{G}}_{ij}\,, \tag{A31}$$

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 $\boldsymbol{c}^{\mathcal{G}}_{ij} \in \mathbb{R}^{1\times H}$ 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. 7).

B.2.2 Adaptive Layer Normalization and Adaptive Scale

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

$$\mathrm{EquivAdaLN}(\boldsymbol{h}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \begin{cases} \mathrm{LN}(\boldsymbol{h}^{(\ell)}) \odot (1+\boldsymbol{\alpha}^{(0)}) + \boldsymbol{\beta} & \text{for } \ell = 0, \\ \mathrm{EquivLN}(\boldsymbol{h}^{(\ell)}) \odot (1+\boldsymbol{\alpha}^{(\ell)}) & \text{for } \ell > 0, \end{cases} \tag{A32}$$

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. A32 reduces to the standard adaptive layer normalization.

Equivariant adaptive scale is defined as

$$\mathrm{EquivAdaScale}(\boldsymbol{h}, \boldsymbol{\gamma}) = \boldsymbol{h}^{(\ell)} \odot \boldsymbol{\gamma}^{(\ell)}. \tag{A33}$$

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

$$\boldsymbol{\alpha}_{1i}, \boldsymbol{\beta}_{1i}, \boldsymbol{\gamma}_{1i}, \boldsymbol{\alpha}_{2i}, \boldsymbol{\beta}_{2i}, \boldsymbol{\gamma}_{2i} = \boldsymbol{W}\big(\mathrm{SiLU}(\boldsymbol{c}_i)\big), \tag{A34}$$

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.

B.2.3 Equivariant 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

$$f(S_g x) = T_g f(x)\,, \tag{A35}$$

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

$$\boldsymbol{p}_{ij}^{\text{PE(3)},(\ell)}(\vec{r}_{ij}) = \boldsymbol{\phi}_\ell(||\vec{r}_{ij}||) \odot \boldsymbol{Y}_\ell(\hat{r}_{ij})\,, \tag{A36}$$

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

$$\boldsymbol{p}_{ijc}^{\text{PE(3)},(\ell)}(\vec{r}_{ij}) = \boldsymbol{\phi}_{\ell c}(||\vec{r}_{ij}||) \odot \boldsymbol{Y}_\ell(\hat{r}_{ij})\,. \tag{A37}$$

Rotating the input positions in Eq. A37 leads to

$$\begin{aligned}
\boldsymbol{p}_{ijc}^{\text{PE(3)},(\ell)}(\boldsymbol{R}\vec{r}_{ij}) &= \boldsymbol{\phi}_{\ell c}(||\boldsymbol{R}\vec{r}_{ij}||) \odot \boldsymbol{Y}_\ell(\boldsymbol{R}\hat{r}_{ij}) && \text{(A38)}\\
&= \boldsymbol{\phi}_{\ell c}(||\vec{r}_{ij}||) \odot \boldsymbol{D}^{(\ell)}(\boldsymbol{R})\boldsymbol{Y}_\ell(\hat{r}_{ij}), && \text{(A39)}\\
&= \boldsymbol{D}^{(\ell)}(\boldsymbol{R})\boldsymbol{p}_{ijc}^{\text{PE(3)},(\ell)}(\vec{r}_{ij}) && \text{(A40)}
\end{aligned}$$

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. A35 and Eq. A40, each channel transforms equivariant and thus $\boldsymbol{p}_{ij}^{\text{PE(3)},(\ell)}(\vec{r}_{ij})$ is also equivariant.

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

$$\boldsymbol{p}_{ij}^{\text{PE(3)}}(\vec{r}_{ij}) = \bigoplus_{\ell=0}^{L} \boldsymbol{\phi}_\ell(r_{ij}) \odot \boldsymbol{Y}_\ell(\hat{r}_{ij})\,, \tag{A41}$$

transforms under rotation as

$$\boldsymbol{p}_{ij}^{\text{PE(3)}}(\boldsymbol{R}\vec{r}_{ij}) = \boldsymbol{D}(\boldsymbol{R})\, \boldsymbol{p}_{ij}^{\text{PE(3)}}(\vec{r}_{ij}) \tag{A42}$$

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. A35 the proposed positional embeddings $\boldsymbol{p}_{ij}^{\text{PE(3)}}$ are $\mathrm{SO}(3)$-equivariant.

B.3 Invariance of 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

$$\boldsymbol{h}(\boldsymbol{R}\vec{r}) = \boldsymbol{D}(\boldsymbol{R})\boldsymbol{h}(\vec{r}), \tag{A43}$$

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

$$\boldsymbol{g}(\boldsymbol{R}\vec{r})^T \cdot \boldsymbol{h}(\boldsymbol{R}\vec{r}) = \boldsymbol{g}(\vec{r})^T \underbrace{\boldsymbol{D}(\boldsymbol{R})^T \boldsymbol{D}(\boldsymbol{R})}_{=\text{Id}} \boldsymbol{h}(\vec{r}) = \boldsymbol{g}(\vec{r})^T \cdot \boldsymbol{h}(\vec{r}), \tag{A44}$$

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.

B.3.1 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

$$\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i = \boldsymbol{W}\big(\mathrm{SiLU}(\boldsymbol{c}_i)\big), \tag{A45}$$

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,

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

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 $\hat{\boldsymbol{y}}_i \in \mathbb{R}^3$.

B.4 Equivariant MLP

Standard MLPs do not preserve the equivariance of the tokens. However, it is possible to define an equivariant formulation for dense layers and so-called gated non-linearities which preserve equivariance. We use them to build equivariant MLPs for the node-wise refinement after the self-attention calculation. See e.g. Ref. [64] for more details.

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

$$\boldsymbol{v}_t^\theta(\boldsymbol{x}_t, t, \mathcal{G}) = \frac{\boldsymbol{x}_1^\theta(\boldsymbol{x}_t, t, \mathcal{G}) - \boldsymbol{x}_t}{1 - t}, \tag{A46}$$

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 $\boldsymbol{x}_0 \sim p_0(\boldsymbol{x})$ as well as 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.
$\boldsymbol{x}_0 \leftarrow \mathrm{Center}(\boldsymbol{x}_0)$
for $n \leftarrow 0$ to $N - 1$ **do**
  $\Delta t \leftarrow t_{n+1} - t_n$ ▷ Compute step size.
  $\hat{\boldsymbol{x}}_1 \leftarrow \boldsymbol{x}_1^\theta(\boldsymbol{x}_{t_n}, t_n, \mathcal{G})$
  $\hat{\boldsymbol{x}}_1 \leftarrow \mathrm{Center}(\hat{\boldsymbol{x}}_1)$
  $\boldsymbol{v} \leftarrow (\hat{\boldsymbol{x}}_1 - \boldsymbol{x}_{t_n})/(1 - t_n)$
  $\boldsymbol{x}_{t_{n+1}} \leftarrow \boldsymbol{x}_{t_n} + \Delta t \cdot \boldsymbol{v}$ ▷ Euler step.
end for

return $\boldsymbol{x}_1$

D Implementation details

D.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 [60], using the same 1000 molecules for testing.

D.2 Input Featurization

Tab. A7 defines the features we use for each atom or bond. Each feature is one-hot encoded before being passed to the network.

Table A7: 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

D.3 Evaluation Metrics

During evaluation, we follow the same procedure as described in Refs. [32, 46, 47, 60]. 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:

$$\text{COV-R}(C, \hat{C}, \delta) := \frac{1}{K}\left|\left\{k \in \{1,\dots,K\} \mid \exists_{l\in\{1,\dots,L\}}\text{RMSD}(\hat{C}_l, C_k) < \delta\right\}\right| \tag{A47}$$

$$\text{COV-P}(C, \hat{C}, \delta) := \frac{1}{L}\left|\left\{l \in \{1,\dots,L\} \mid \exists_{k\in\{1,\dots,K\}}\text{RMSD}(\hat{C}_l, C_k) < \delta\right\}\right| \tag{A48}$$

$$\text{AMR-R}(C, \hat{C}) := \frac{1}{K}\sum_{k\in\{1,\dots,K\}}\min_{l\in\{1,\dots,L\}}\text{RMSD}(\hat{C}_l, C_k) \tag{A49}$$

$$\text{AMR-P}(C, \hat{C}) := \frac{1}{L}\sum_{l\in\{1,\dots,L\}}\min_{k\in\{1,\dots,K\}}\text{RMSD}(\hat{C}_l, C_k) \tag{A50}$$

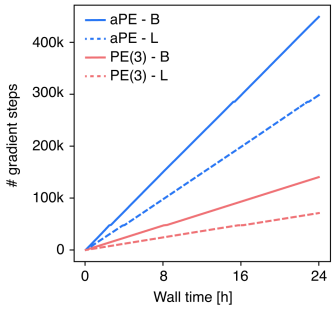

Figure A4: Number of gradient steps vs. wall time for PE(3) and aPE on the GEOM-DRUGS dataset measured on Nvidia H100 GPU with batch size 128.

D.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 volume V of the tetrahedron as follows

$$V(\mathbf{p_1}, \mathbf{p_2}, \mathbf{p_3}, \mathbf{p_4}) = \det\left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \end{bmatrix}\right) = (\mathbf{p_1} - \mathbf{p_4}) \cdot ((\mathbf{p_2} - \mathbf{p_4}) \times (\mathbf{p_3} - \mathbf{p_4})) \quad \text{(A51)}$$

Following GeomMol [60] and ET-Flow [32] we can then compare the orientation of the volume given by $\text{sign}(V)$ with the local chirality label produced by RDKit, which corresponds to a certain orientation as well (CW = +1 and CCW = -1) [65]. 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.

D.5 Time vs. Efficiency Analysis

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.

E Additional Experimental Results

E.1 Conditioning ablation

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. 4.2, we note that our conditioning GNN also produces edge-level representations, which can be used to define pair-wise conditioning tokens:

$$\textbf{bond-pair:} \quad \boldsymbol{c}_{ij}^{\mathcal{G}} = \begin{cases} \text{GNN}_{\text{edge}}(\mathcal{V}, \mathcal{E}) & \forall (i,j) \in \mathcal{E} \\ \bar{\boldsymbol{c}}^{\mathcal{G}} & \forall (i,j) \notin \mathcal{E}\,. \end{cases} \quad \text{(A52)}$$

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:

- **node only** conditioning using only atom-wise conditioning $\boldsymbol{c}_i^{\mathcal{G}}$ (Eq. 9).
- **node & all-pair** conditioning using both atom-wise conditioning $\boldsymbol{c}_i^{\mathcal{G}}$ (Eq. 9) and pair-wise conditioning on geodesic graph distances $\boldsymbol{c}_{ij}^{\mathcal{G}}$ (eq:pair-conditioning-tokens-all).

- **node & bond-pair** conditioning using both atom-wise conditioning $\boldsymbol{c}_i^{\mathcal{G}}$ (Eq. 9) and pair-wise conditioning $\boldsymbol{c}_{ij}^{\mathcal{G}}$ derived from edge-level representations of the conditioning GNN as discussed above.

As reported in Tab. A8, all conditioning variants significantly reduce the Average Minimum RMSD (AMR) for both recall (AMR-R) and precision (AMR-P) across the aPE and PE(3) settings, 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 K for a more in-depth analysis.

Table A8: Ablation of conditioning strategies in DiTMC (in brackets) with absolute PE and PE(3) 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+PE(3)-B (no conditioning)	71.5 ±0.5	99.0 ±1.4	0.358 ±0.003	0.266 ±0.005	40.4 ±0.2	42.2 ±0.4	0.683 ±0.003	0.653 ±0.005
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+PE(3)-B (node only)	95.5 ±0.4	**100.0** ±0.0	0.074 ±0.003	0.026 ±0.001	91.1 ±0.4	**100.0** ±0.0	0.114 ±0.003	0.041 ±0.001
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+PE(3)-B (node & bond-pair)	96.1 ±0.3	**100.0** ±0.0	**0.068** ±0.001	0.022 ±0.001	93.6 ±0.2	**100.0** ±0.0	0.091 ±0.003	0.035 ±0.001
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
DiTMC+PE(3)-B (node & all-pair)	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

E.2 Ensemble Properties

While RMSD-based metrics offer a geometric perspective on the quality of the generated conformers, they do not assess the chemical fidelity. To address this, we evaluate the median averaged errors of different ensemble properties between the generated and ground truth conformers. 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 [66] and compare the Boltzmann-weighted properties of the generated and ground truth ensembles. More specifically, we employ xTB [66] 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 in Table A11.

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

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-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

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.

Method	E **[kcal/mol]** $\downarrow$	μ **[D]** $\downarrow$	$\Delta\epsilon$ **[kcal/mol]** $\downarrow$	E_{min} **[kcal/mol]** $\downarrow$
OMEGA	0.68	0.66	0.68	0.69
GeoDiff	0.31	0.35	0.89	0.39
GeoMol	0.42	0.34	0.59	0.40
Torsional Diff.	0.22	0.35	0.54	0.13
MCF	0.68 $\pm$0.06	0.28 $\pm$0.05	0.63 $\pm$0.05	0.04 $\pm$0.00
ET-Flow	0.18 $\pm$0.01	0.18 $\pm$0.01	0.35 $\pm$0.06	<u>0.02</u> $\pm$0.00
$*$DiTMC+aPE-B	<u>0.17</u> $\pm$0.00	0.16 $\pm$0.01	<u>0.27</u> $\pm$0.01	**0.01** $\pm$0.00
$*$DiTMC+aPE-L	**0.16** $\pm$0.02	**0.14** $\pm$0.03	<u>0.27</u> $\pm$0.01	**0.01** $\pm$0.00
$*$DiTMC+rPE-B	**0.16** $\pm$0.03	0.16 $\pm$0.03	0.29 $\pm$0.02	<u>0.02</u> $\pm$0.00
$*$DiTMC+rPE-L	**0.16** $\pm$0.01	<u>0.15</u> $\pm$0.02	0.28 $\pm$0.06	**0.01** $\pm$0.00
$*$DiTMC+PE(3)-B	0.18 $\pm$0.01	0.18 $\pm$0.01	<u>0.27</u> $\pm$0.03	<u>0.02</u> $\pm$0.00
$*$DiTMC+PE(3)-L	<u>0.17</u> $\pm$0.01	**0.14** $\pm$0.01	**0.25** $\pm$0.01	**0.01** $\pm$0.00

E.3 Generalization Results on GEOM-QM9

Additionally, we want to assess how well our proposed model architecture generalizes to unseen molecules. Here we evaluate our models trained on GEOM-DRUGS on the GEOM-QM9 dataset. We report the generalization performance in Table A12.

E.4 Generalization Results on GEOM-XL

We also study how well our proposed model architecture generalizes to unseen molecules with a large number of atoms. For this we adopt the GEOM-XL dataset containing a total of 102 molecules with more than 100 atoms presented in [46]. We report the generalization performance in Table A13.

F Additional Ablations

F.1 Index Positional Encoding (iPE)

Tab. A14 compares a variant including index positional encoding (iPE) from classic transformer architectures with our base model using aPE on 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 graphs are generated from SMILES strings via 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. A14). 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

Table A12: Out-of-distribution generalization results on GEOM-QM9 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	COV-R [%]↑		AMR-R [Å]↓		COV-P [%]↑		AMR-P [Å]↓	
	Mean	Median	Mean	Median	Mean	Median	Mean	Median
ET-Flow	**86.7**	**100.0**	0.218	0.160	68.7	75.3	0.369	0.317
∗DiTMC+aPE-B	84.3 ±0.5	**100.0** ±0.0	0.209 ±0.004	0.134 ±0.003	69.4 ±0.7	84.4 ±3.2	0.329 ±0.004	0.250 ±0.008
∗DiTMC+aPE-L	80.6 ±0.5	**100.0** ±0.0	0.199 ±0.004	**0.115** ±0.004	68.8 ±0.2	**86.2** ±2.0	0.329 ±0.002	**0.224** ±0.007
∗DiTMC+rPE-B	82.3 ±0.2	**100.0** ±0.0	0.217 ±0.002	0.138 ±0.002	**69.6** ±0.4	85.7 ±1.3	0.327 ±0.004	0.238 ±0.008
∗DiTMC+rPE-L	80.5 ±0.4	**100.0** ±0.0	0.215 ±0.002	0.127 ±0.002	68.9 ±0.5	85.2 ±2.7	0.327 ±0.004	0.228 ±0.012
∗DiTMC+PE(3)-B	84.8 ±0.3	**100.0** ±0.0	0.205 ±0.003	0.130 ±0.005	69.1 ±0.1	81.41 ±2.9	0.329 ±0.004	0.254 ±0.015
∗DiTMC+PE(3)-L	85.1 ±0.2	**100.0** ±0.0	**0.195** ±0.004	**0.115** ±0.003	69.5 ±0.6	82.4 ±2.3	**0.319** ±0.05	0.245 ±0.009

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 model using iPE can effectively exploit the information contained in atom indices assigned by rdkit. A version of our base aPE model without atom-pair conditioning but iPE achieves comparable performance to our model 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.

F.2 Gaussian vs. Harmonic Prior

As shown in Tab. A15, 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. A15 verify, our method can also be used with a simple Gaussian prior effectively. For larger molecular graphs the expensive eigendecomposition required for the Harmonic prior could therefore be avoided, which helps scaling our approach more easily.

G 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 PEs 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 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. A6, 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.

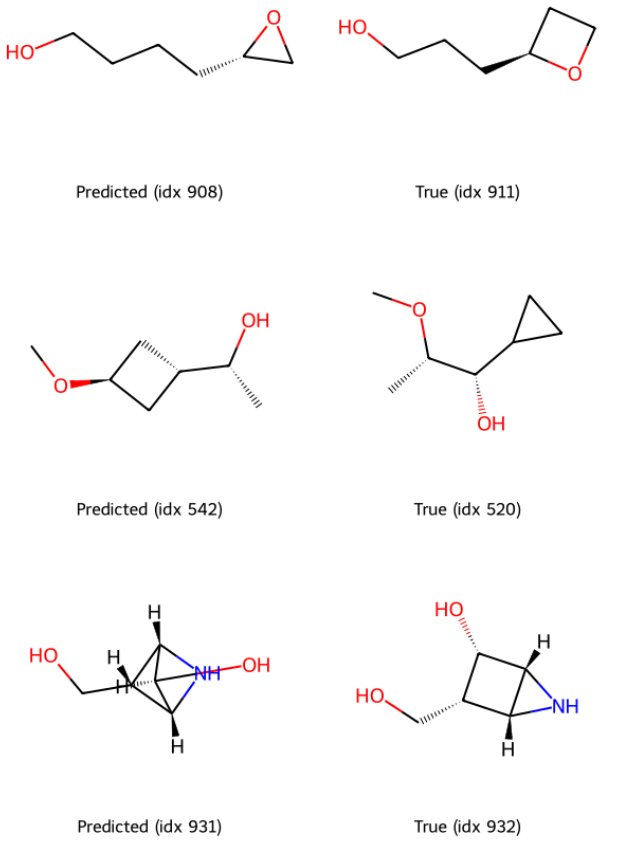

Figure A5: Misclassified exampled for SMILES classification experiment. We randomly pick 3 examples, which are misclassified by a classification head without any GNN layers. We show that GNN layers are essential for correct classification of isomers.

We further note, that absolute loss values for models trained with all our PE strategies decrease as latent time increases (see Fig. A6). 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 A) effectively penalizes errors close to the data distribution during training and helps with keeping the error low in this important regime.

H SMILES Classification Experiment via Conditioning GNN

A critical requirement of our DiTMC approach is the ability to disentangle distinct SMILES representations through conditioning. We investigate whether our proposed conditioning strategy 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

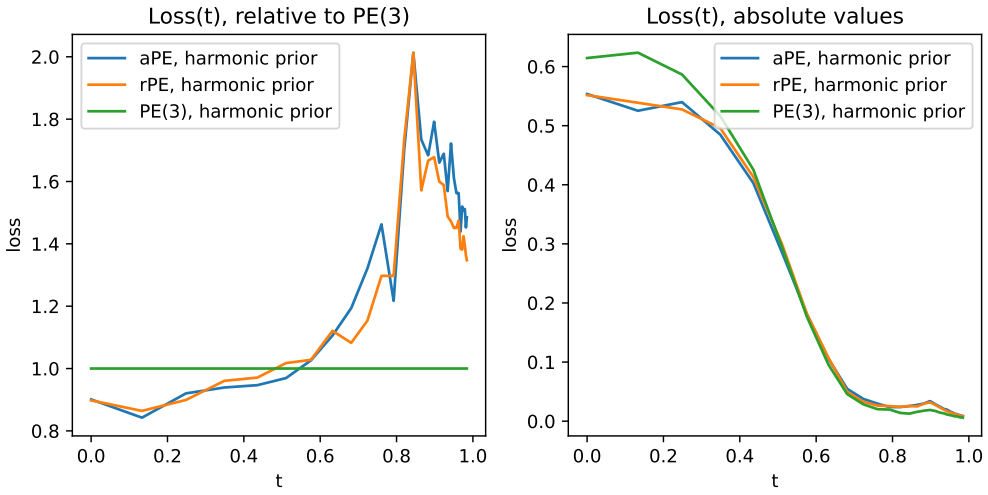

Figure A6: Loss as a function of time comparing different PE strategies. Results averaged over 1000 samples randomly drawn from the GEOM QM9 validation set. **Left:** loss relative to PE(3) as a baseline. In the important regime close to the data distribution, the model PE(3) has lower loss, yielding higher sample fidelity. **Right:** absolute loss values for all PEs. The loss decreases close to the data distribution for all models.

utilized in our conditioning graph neural network (Sec. 4.2, appendix D.2) plus a simple classification head.

We train models with a batch size of 3 for 5000 epochs (toy dataset) and batch size of 64 over 250 epochs (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. A5 shows that without GNN layers, isomers will be misclassified.

I Architecture Details for Conditioning GNN

To transform SMILES representations to per-token conditioning information, we use a GNN that directly operates on the bond graph induced by the SMILES string. Our model processes molecular graphs by first embedding node and edge features, initially represented as one-hot vectors (a full list of features is provided in Tab. A7). These features are projected into a shared latent space using two-layer multilayer perceptrons (MLPs). For message passing, we employ a graph neural network architecture inspired by the processor described in the MeshGraphNet (MGN) framework [56].

The 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:

$$\boldsymbol{e}'_{ij} \leftarrow f_e(\boldsymbol{e}_{ij}, \boldsymbol{v}_i, \boldsymbol{v}_j) \tag{A53}$$

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:

$$\boldsymbol{v}_i' \leftarrow f_v\left(\boldsymbol{v}_i, \sum_j \boldsymbol{e}_{ij}'\right) \tag{A54}$$

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 GNN is a set of node embeddings per atom, and a set of edge embeddings per bond. Those serve as atom-wise or bond-pair conditioning inputs to the DiT transformer, as detailed in Sec. 4.2 and Sec. E.1.

J Autoguidance

We apply *autoguidance*, a recent technique for enhancing generative models by leveraging the outputs of a weaker model to guide sampling from a stronger one [67]. Originally proposed in the context of image diffusion, autoguidance works by introducing a degraded model with parameters $\hat{\theta}$ trained on the same data and conditioning as the main model with parameters θ, but with additional constraints such as reduced capacity, increased noise or fewer gradient steps. During generation, the predicted vector field $\boldsymbol{v}^\theta(\boldsymbol{x}, t)$ is guided using the vector field predicted by $\boldsymbol{v}^{\hat{\theta}}(\boldsymbol{x}, t)$, where the discrepancy between the two models serves as a corrective signal. Intuitively, this nudges the generation process towards higher-quality samples by amplifying the difference between a less accurate and a more accurate model.

In our molecular conformer generation setup, we choose the parameters after training for 5 epochs without applying exponential moving average (EMA) for $\hat{\theta}$. We then use the difference in predicted vector fields from to bias the conformer generation trajectory. Following Ref. [68], we apply autoguidance only in the guidance interval $t \in [0, t_{max}]$ and additionally ablate t_{max} as the upper bound of the guidance interval. It becomes straightforward to adapt Algorithm 2 to reflect these changes during sampling as a drop-in repacement for the predicted vector field. The guided velocity (for $t \in [0, t_{max}]$) is computed as $\boldsymbol{v}_\eta(\boldsymbol{x}, t) = \eta \boldsymbol{v}^\theta(\boldsymbol{x}, t) + (1 - \eta)\boldsymbol{v}^{\hat{\theta}}(\boldsymbol{x}, t)$, where η is the guidance strength.

Fig. A7 shows that autoguidance yields enhanced fidelity of generated conformations, as indicated by improvements in precision metrics, but might negatively impact diversity (recall). Increasing guidance strength improves precision (COV-P, AMR-P) but may reduce recall (COV-R, AMR-R). A similar trend can be observed for the guidance interval, where increased t_{max} leads to stronger effects of autoguidance. This effectively establishes a trade-off between precision and recall that can be tuned post-hoc after training with minimal effort.

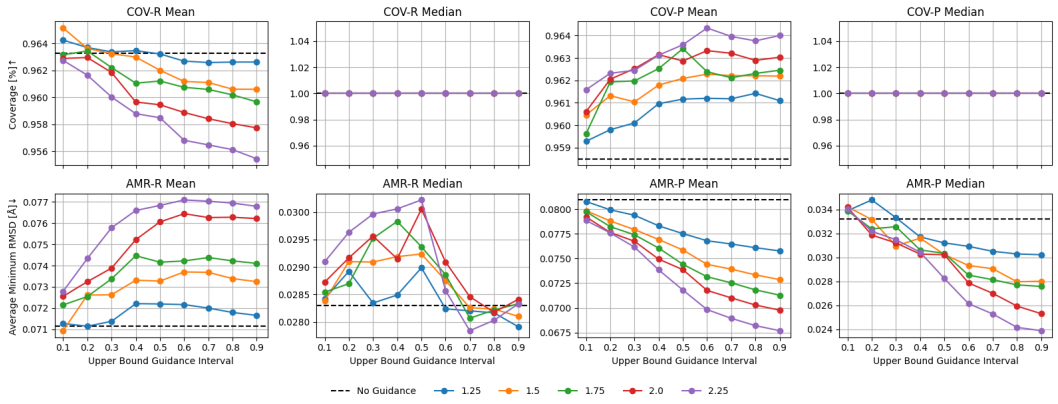

Figure A7: Autoguidance performance on GEOM-QM9 using our DiTMC+aPE-B model. We evaluate the effect of autoguidance on conformer generation quality across various guidance intervals (t_{max}) on the x-axes, and guidance strength (η) as different colors. Dashed black lines indicate performance without guidance.

K Analysis of Sampling Trajectories

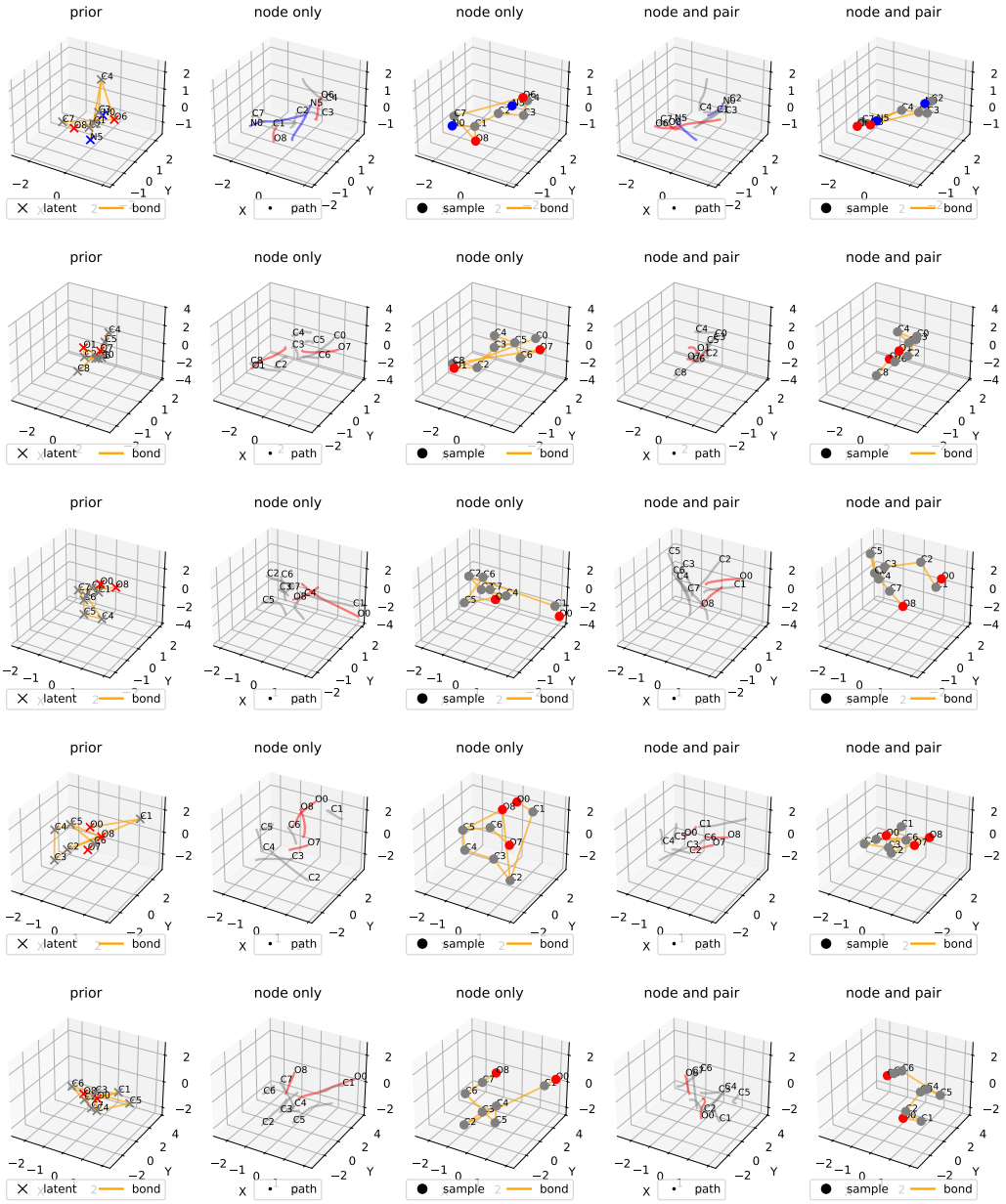

Figure A8: Comparison of molecular generation with node-only versus node- and pair-wise conditioning on GEOM-QM9. Each row shows a prior sample, sampling trajectories, and final generated structure for both models; pairwise conditioning preserves bonds from the conditioning graph $\mathcal{G}$.

Fig. A8 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. 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 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.

L Visualization

Fig. A15, Fig. A16, and Fig. A17 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.

M 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`.

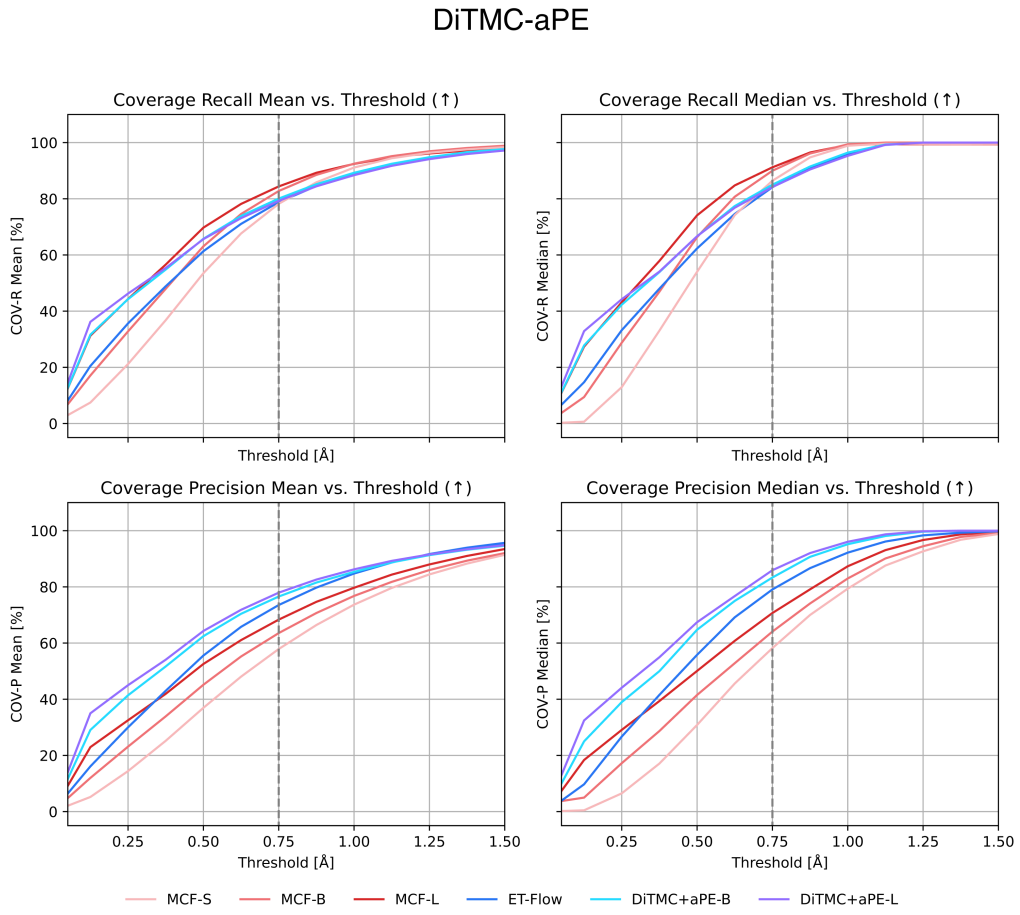

Figure A9: Coverage mean and median for recall and precision of DiTMC-aPE on GEOM-DRUGS. Vertical dashed line denotes the commonly employed $\rho = 0.75$ RMSD threshold.

Table A13: 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	**AMR-R** [Å]↓		**AMR-P** [Å]↓		# mols
	Mean	Median	Mean	Median	
GeoDiff	2.92	2.62	3.35	3.15	-
GeoMol	2.47	2.39	3.30	3.14	-
Tor. Diff.	2.05	1.86	2.94	2.78	-
MCF - S	2.22	1.97	3.17	2.81	102
MCF - B	2.01	1.70	3.03	2.64	102
MCF - L	1.97	1.60	2.94	2.43	102
ET-Flow	2.31	1.93	3.31	2.84	102
∗DiTMC+aPE-B	1.96 ±0.00	1.60 ±0.03	2.90 ±0.00	2.48 ±0.03	102
∗DiTMC+aPE-L	1.88 ±0.01	**1.51** ±0.02	**2.81** ±0.00	**2.30** ±0.02	102
∗DiTMC+rPE-B	1.97 ±0.01	1.61 ±0.01	2.86 ±0.00	2.33 ±0.01	102
∗DiTMC+rPE-L	1.96 ±0.02	1.61 ±0.02	2.82 ±0.00	2.42 ±0.02	102
∗DiTMC+PE(3)-B	1.98 ±0.01	1.67 ±0.02	3.03 ±0.00	2.60 ±0.01	102
∗DiTMC+PE(3)-L	**1.85** ±0.02	1.58 ±0.03	2.93 ±0.00	2.53 ±0.03	102
Tor. Diff.	1.93	1.86	2.84	2.71	77
MCF - S	2.02	1.87	2.9	2.69	77
MCF - B	1.71	1.61	2.69	2.44	77
MCF - L	1.64	1.51	2.57	2.26	77
ET-Flow	2.00	1.80	2.96	2.63	75
∗DiTMC+aPE-B	1.68 ±0.00	1.47 ±0.02	2.59 ±0.00	2.24 ±0.01	77
∗DiTMC+aPE-L	**1.56** ±0.01	**1.28** ±0.01	**2.47** ±0.00	2.14 ±0.01	77
∗DiTMC+rPE-B	1.69 ±0.01	1.41 ±0.03	2.52 ±0.00	**2.11** ±0.00	77
∗DiTMC+rPE-L	1.66 ±0.03	1.37 ±0.01	**2.47** ±0.00	2.18 ±0.02	77
∗DiTMC+PE(3)-B	1.73 ±0.01	1.55 ±0.01	2.71 ±0.00	2.35 ±0.01	77
∗DiTMC+PE(3)-L	1.57 ±0.01	1.46 ±0.01	2.60 ±0.00	2.27 ±0.02	77

Table A14: Ablating index positional encoding (iPE) on 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

Table A15: Ablation of PE strategies and Gaussian (G) and Harmonic (H) prior. 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.

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

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**

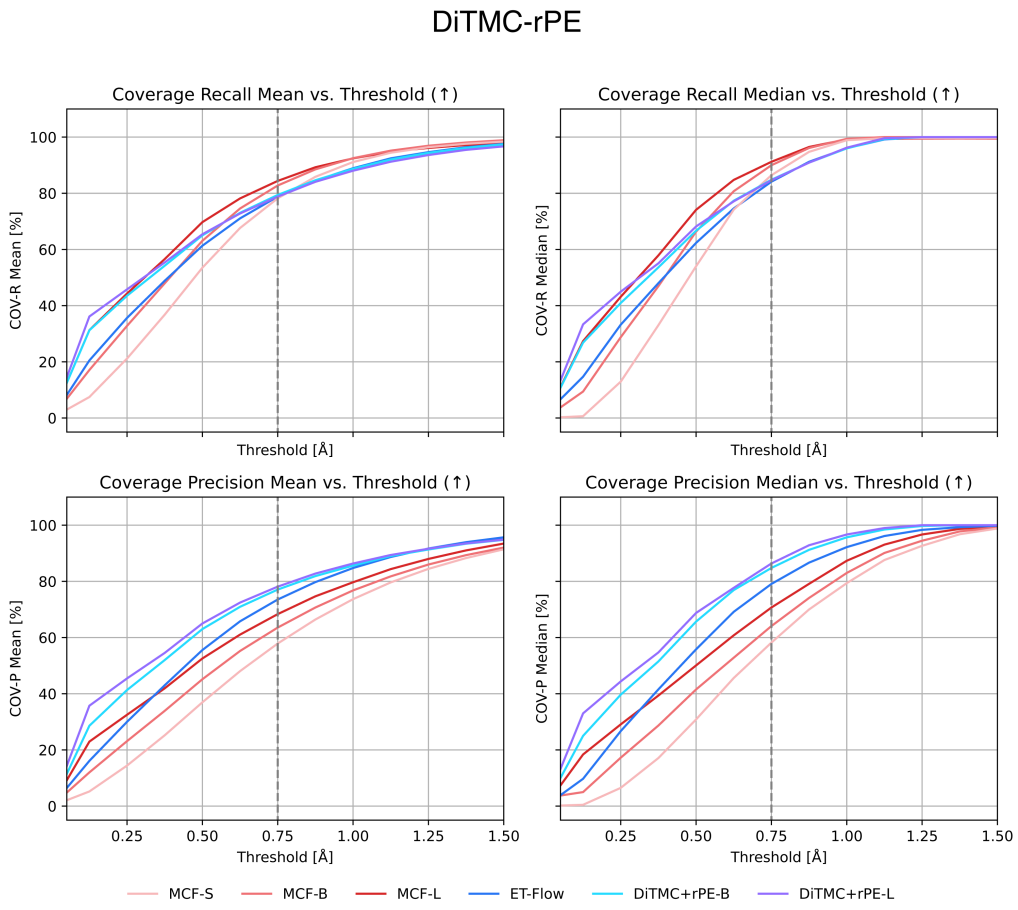

Figure A10: Coverage mean and median for recall and precision of DiTMC-rPE on GEOM-DRUGS. Vertical dashed line denotes the commonly employed $\rho = 0.75$ RMSD threshold.

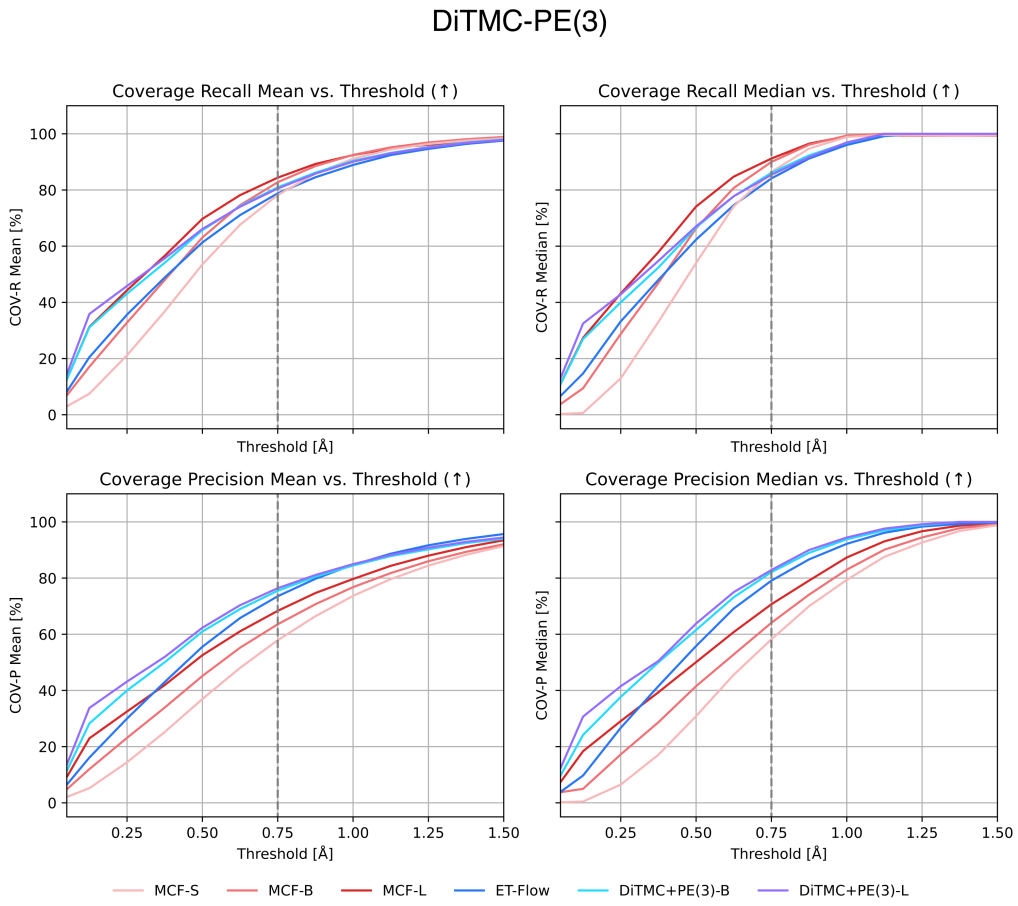

Figure A11: Coverage mean and median for recall and precision of DiTMC-PE(3) on GEOM-DRUGS. Vertical dashed line denotes the commonly employed $\rho = 0.75$ RMSD threshold.

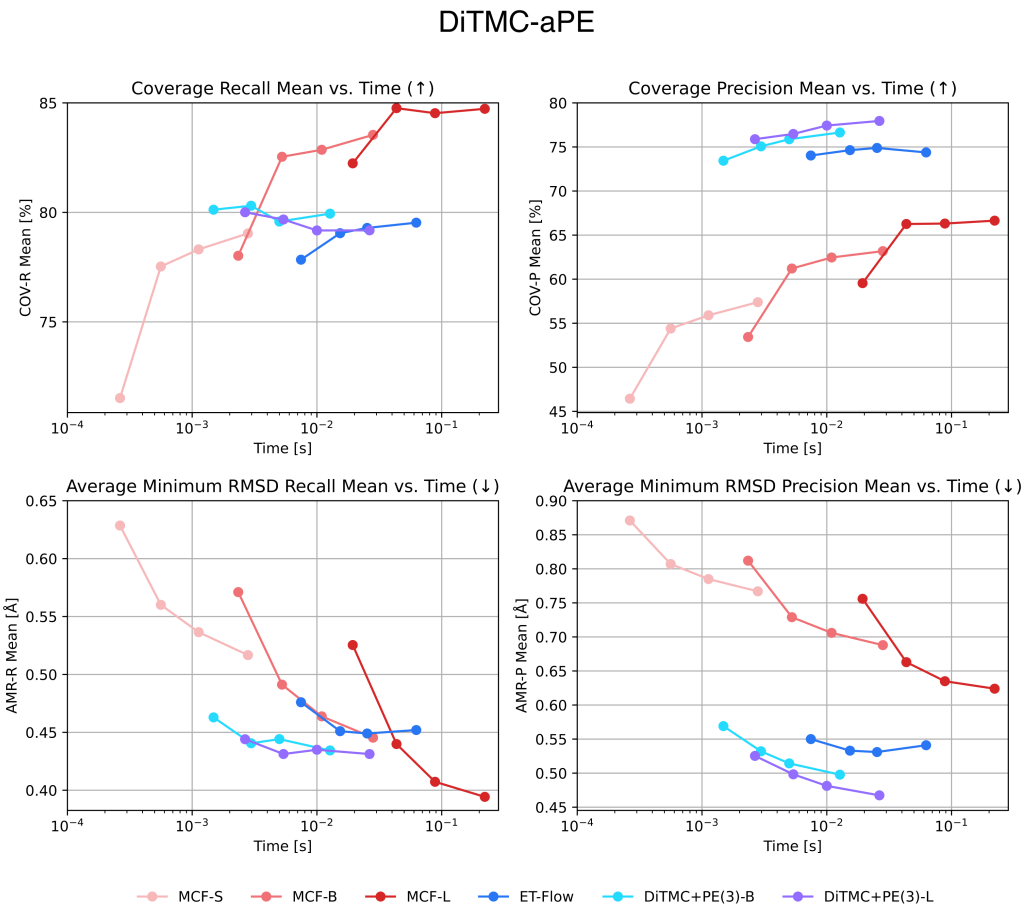

Figure A12: Average minimum RMSD (AMR) for precision ("-P") and recall ("-R") plotted against the time per generated conformer. For each model, markers from left to right correspond to an increasing number of sampling steps during generation. Here, we follow Refs. [32, 47] and use 5, 10, 20, and 50 sampler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

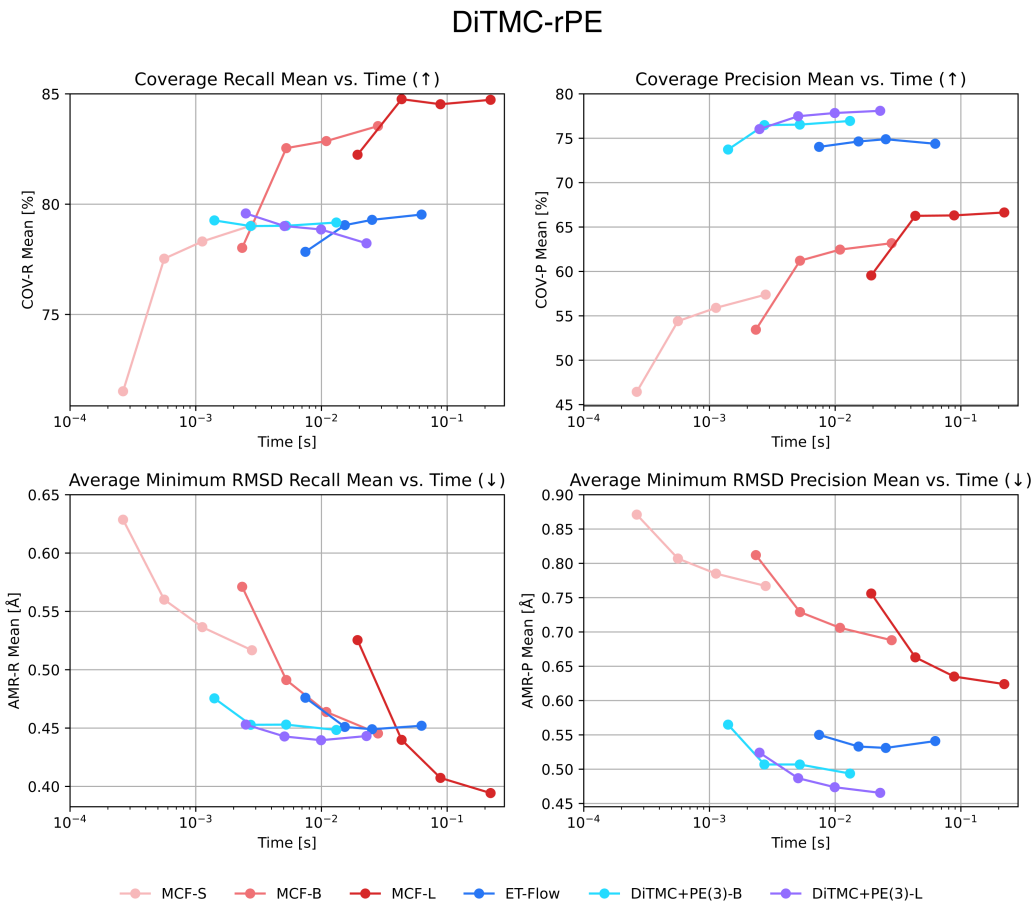

Figure A13: Average minimum RMSD (AMR) for precision ("-P") and recall ("-R") plotted against the time per generated conformer. For each model, markers from left to right correspond to an increasing number of sampling steps during generation. Here, we follow Refs. [32, 47] and use 5, 10, 20, and 50 sampler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

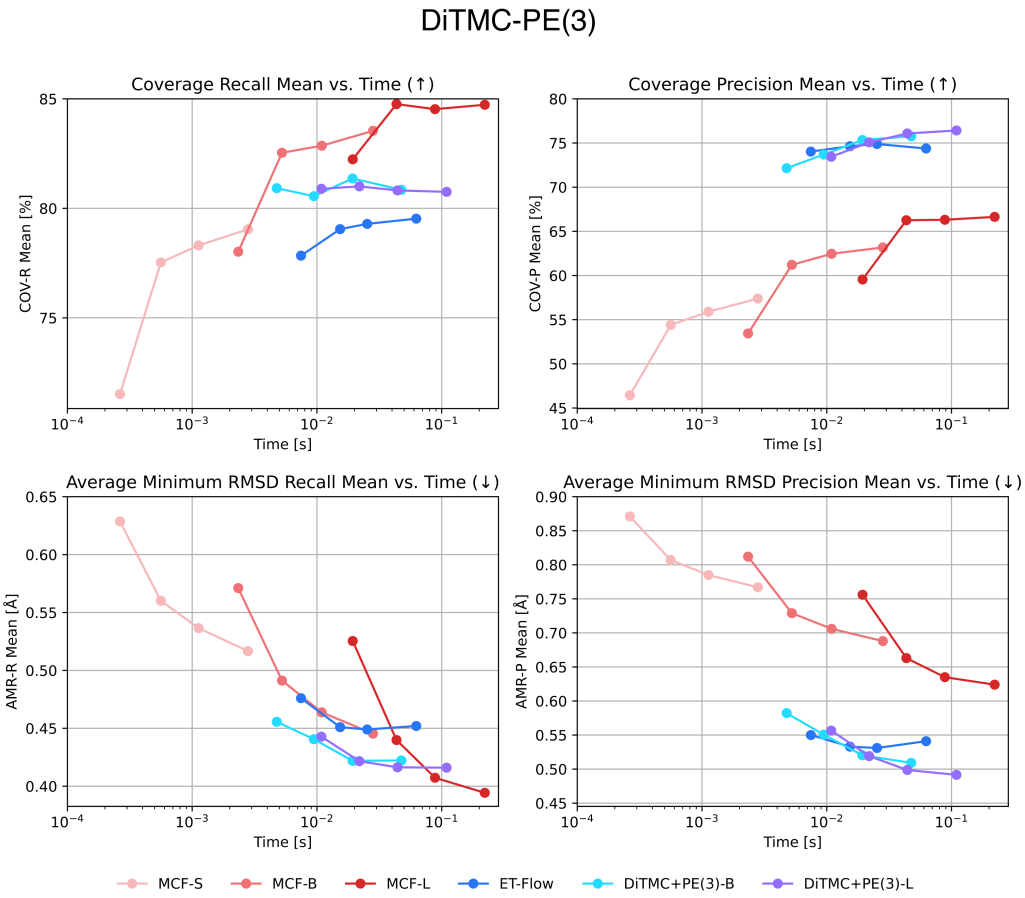

Figure A14: Average minimum RMSD (AMR) for precision ("-P") and recall ("-R") plotted against the time per generated conformer. For each model, markers from left to right correspond to an increasing number of sampling steps during generation. Here, we follow Refs. [32, 47] and use 5, 10, 20, and 50 sampler steps.

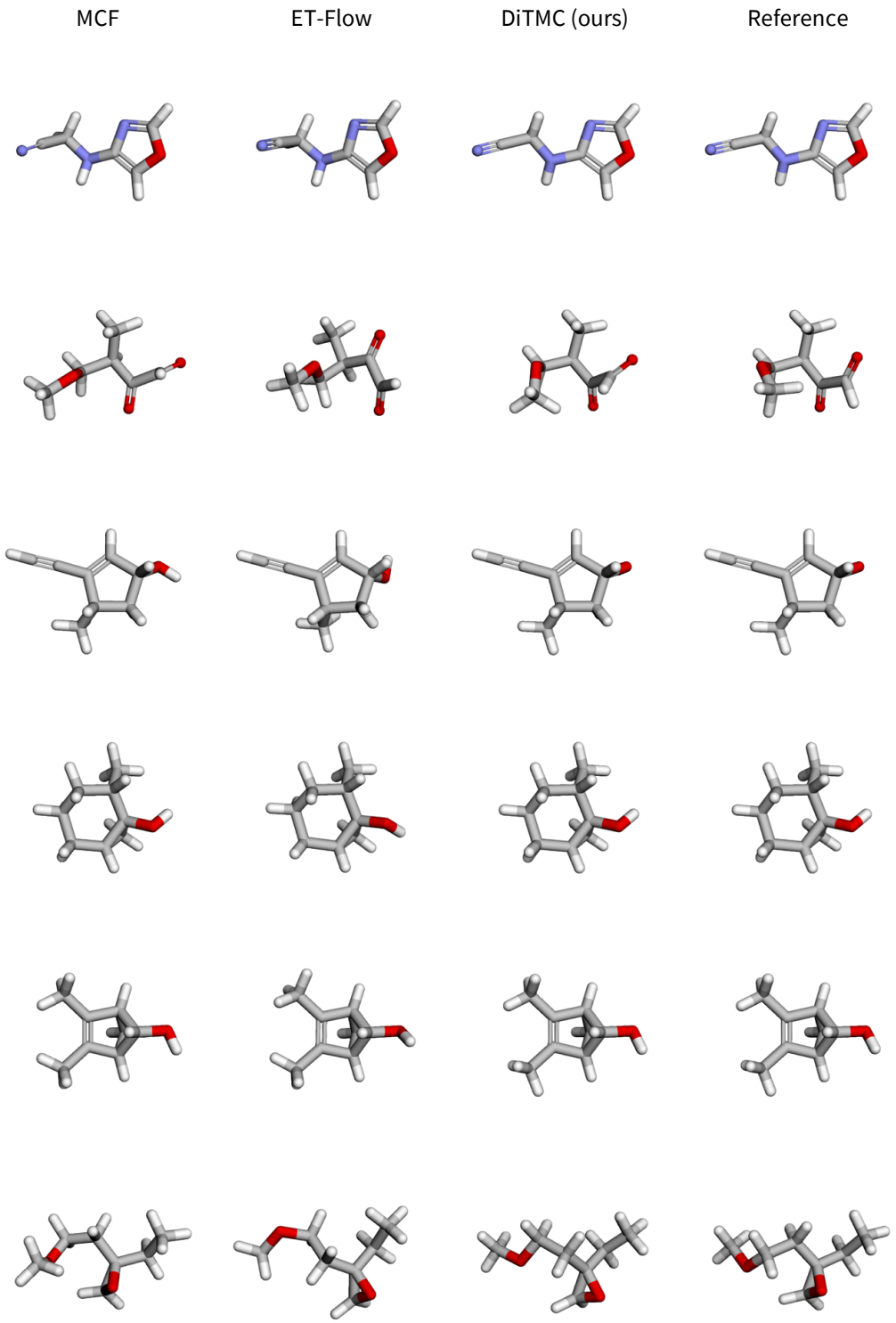

Figure A15: Comparison of conformers generated by MCF, ET-Flow, and DiTMC against ground-truth reference conformers from GEOM-QM9. The generated conformers are rotationally aligned with their corresponding reference conformer to facilitate comparison. **From left to right**: generated conformers from MCF, ET-Flow, DiTMC, and the corresponding ground-truth reference conformers.

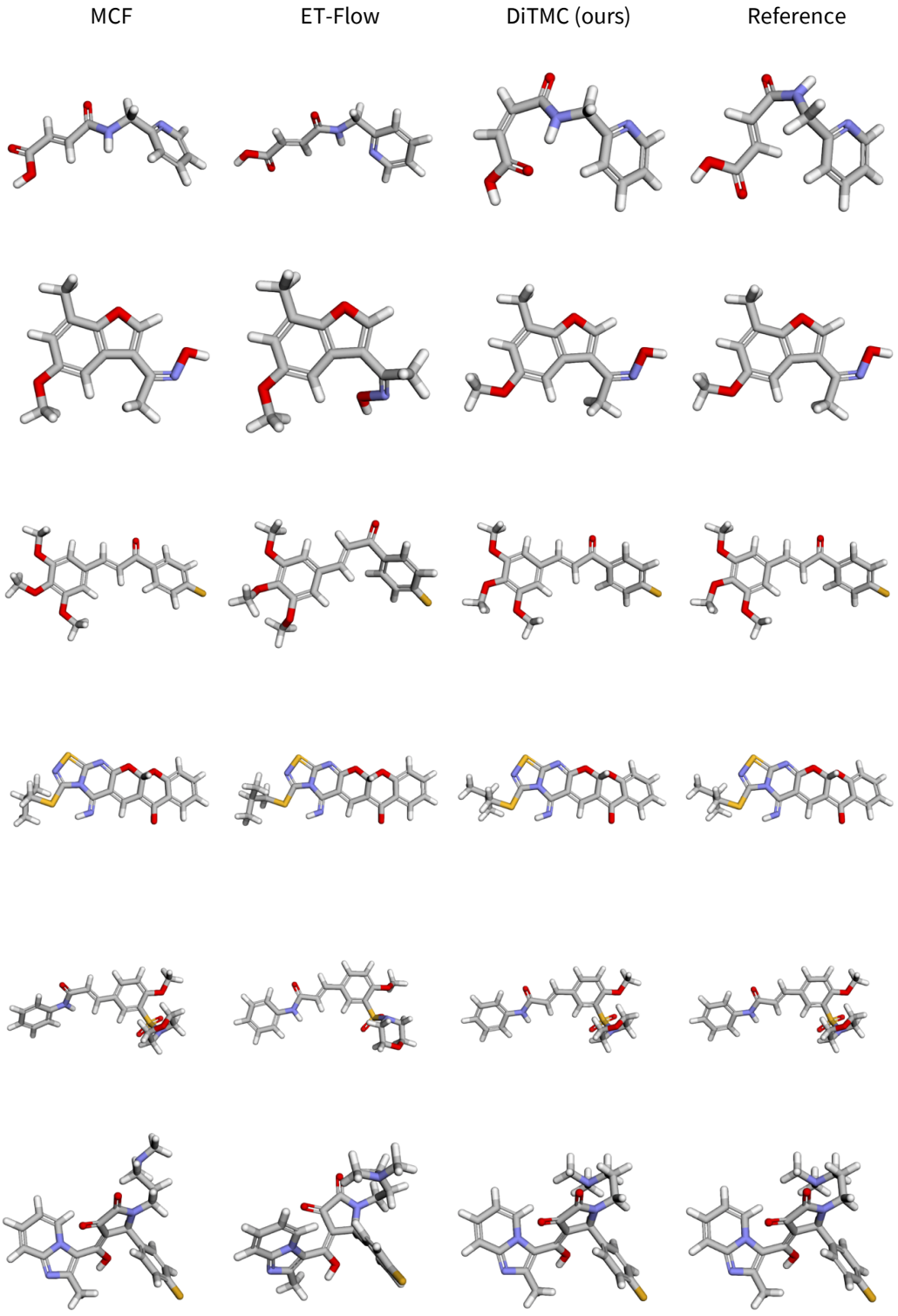

Figure A16: Comparison of conformers generated by MCF, ET-Flow, and DiTMC against ground-truth reference conformers from GEOM-DRUGS. The generated conformers are rotationally aligned with their corresponding reference conformer to facilitate comparison. **From left to right**: generated conformers from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

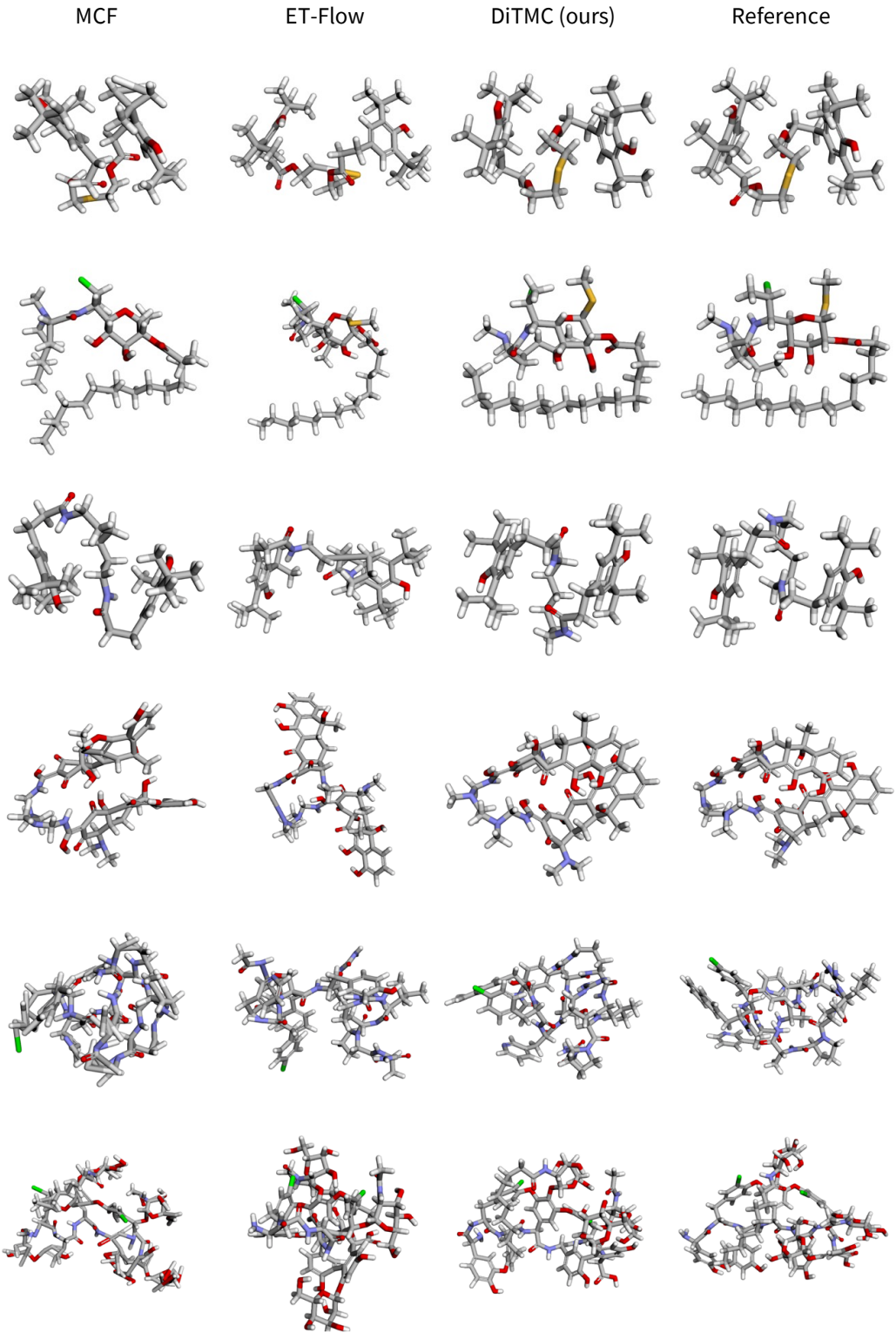

Figure A17: Comparison of conformers generated by MCF, ET-Flow, and DiTMC against ground-truth reference conformers from GEOM-XL. The generated conformers are rotationally aligned with their corresponding reference conformer to facilitate comparison. **From left to right**: generated conformers from MCF, ET-Flow, DiTMC, and the corresponding ground-truth reference conformers.