

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

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

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

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

B Training

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

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

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

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

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

where $t \in (0, 1)$ denotes the timestep in the interpolant $\mathbf{x}_t \in \mathbb{R}^{N \times 3}$, $\mathbf{x}_1 \in \mathbb{R}^{N \times 3}$ 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 $\mathcal{U} = \{\vec{u}_1, \dots, \vec{u}_N \mid \vec{u}_i \in \mathbb{R}^3\}$ associated with a point cloud $\mathbf{x} \in \mathbb{R}^{N \times 3}$, we define a centering operation for the i -th row

$$\text{Center}(\mathbf{x})_i = \vec{u}_i - \frac{1}{N} \sum_{j=1}^N \vec{u}_j, \quad (\text{A15})$$

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

$$\text{RotationAlign}(\mathbf{x}_A, \mathbf{x}_B)_i = \mathbf{R}_{\text{opt}} \mathbf{x}_{iA}, \quad (\text{A16})$$

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

$$\mathbf{x}_A \leftarrow \text{Center}(\mathbf{x}_A) \quad (\text{A17})$$

$$\mathbf{x}_B \leftarrow \text{Center}(\mathbf{x}_B) \quad (\text{A18})$$

$$\mathbf{x}_A \leftarrow \text{RotationAlign}(\mathbf{x}_A, \mathbf{x}_B) \quad (\text{A19})$$

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

Algorithm 1 Conditional Flow Matching Training Loss

Require: Graph \mathcal{G} , target \mathbf{x}_1 , noise level σ , Model \mathbf{x}_1^θ

- 1: $\mathbf{x}_0 \sim p_0$, $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $t \sim \mathcal{U}(0, 1)$, $\mathbf{R} \sim \text{SO}(3)$
 - 2: $\mathbf{x}_0, \mathbf{x}_1 \leftarrow \text{GeometryAlign}(\mathbf{x}_0, \mathbf{x}_1)$ \triangleright This centers \mathbf{x}_0 and \mathbf{x}_1 and rotation-aligns \mathbf{x}_0 to \mathbf{x}_1 .
 - 3: $\epsilon \leftarrow \text{Center}(\epsilon)$
 - 4: $\mathbf{x}_t \leftarrow (1 - t)\mathbf{x}_0 + t\mathbf{x}_1 + \sigma\epsilon$
 - 5: $\mathbf{x}_t \leftarrow \text{ApplyRotation}(\mathbf{R}, \mathbf{x}_t)$
 - 6: $\mathbf{x}_1 \leftarrow \text{ApplyRotation}(\mathbf{R}, \mathbf{x}_1)$
 - 7: $\hat{\mathbf{x}}_1 \leftarrow \mathbf{x}_1^\theta(\mathbf{x}_t, t, \mathcal{G})$
 - 8: $\hat{\mathbf{x}}_1 \leftarrow \text{Center}(\hat{\mathbf{x}}_1)$
 - 9: **return** $\frac{1}{(1-t)^2} \|\hat{\mathbf{x}}_1 - \mathbf{x}_1\|^2$
-

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

$$\text{ApplyRotation}(\mathbf{R}, \mathbf{x})_i = \mathbf{R}\vec{r}_i, \quad (\text{A20})$$

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

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

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

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

C Sampling

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

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

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

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

Algorithm 2 ODE Sampling

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

```

1:  $t_n \leftarrow n/N$  for  $n \in \{0, \dots, N\}$ 
2:  $\mathbf{x}_0 \sim p_{\text{prior}}(\mathbf{x})$  ▷ Sample prior.
3:  $\mathbf{x}_0 \leftarrow \text{Center}(\mathbf{x}_0)$ 
4: for  $n \leftarrow 0$  to  $N - 1$  do
5:    $\Delta t \leftarrow t_{n+1} - t_n$  ▷ Compute step size.
6:    $\hat{\mathbf{x}}_1 \leftarrow \mathbf{x}_1^\theta(\mathbf{x}_{t_n}, t_n, \mathcal{G})$ 
7:    $\hat{\mathbf{x}}_1 \leftarrow \text{Center}(\hat{\mathbf{x}}_1)$ 
8:    $\mathbf{v} \leftarrow (\hat{\mathbf{x}}_1 - \mathbf{x}_{t_n}) / (1 - t_n)$ 
9:    $\mathbf{x}_{t_{n+1}} \leftarrow \mathbf{x}_{t_n} + \Delta t \cdot \mathbf{v}$  ▷ Euler step.
10: end for
11: return  $\mathbf{x}_1$ 

```

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

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

D Architectural Details

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

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

$$\mathbf{c}_i = \mathbf{c}^t + \mathbf{c}_i^{\mathcal{G}}. \quad (\text{A22})$$

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

D.1 Non-Equivariant DiTMC

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

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

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

D.1.1 Self-Attention Operation

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

$$\mathbf{q} = \mathbf{W}_q \mathbf{h}, \quad \mathbf{k} = \mathbf{W}_k \mathbf{h}, \quad \mathbf{v} = \mathbf{W}_v \mathbf{h}, \quad (\text{A23})$$

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

$$\text{sim}(\mathbf{q}, \mathbf{k}, \mathbf{u}) = \exp \left(\frac{\mathbf{q}^\top \cdot (\mathbf{k} \odot \mathbf{u})}{\sqrt{H}} \right), \quad (\text{A24})$$

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

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

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

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

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

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

D.1.2 Adaptive Layer Normalization and Adaptive Scale

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

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

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

Adaptive Scale is defined as

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

In each DiTMC block, we calculate

$$\alpha_{1i}, \beta_{1i}, \gamma_{1i}, \alpha_{2i}, \beta_{2i}, \gamma_{2i} = \mathbf{W}(\text{SiLU}(c_i)), \quad (\text{A29})$$

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

D.1.3 Readout

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

$$\alpha_i, \beta_i = \mathbf{W}(\text{SiLU}(c_i)), \quad (\text{A30})$$

with weight matrix $\mathbf{W} \in \mathbb{R}^{2H \times H}$ initialized to all zeros and $\alpha_i, \beta_i \in \mathbb{R}^H$ and do

$$\begin{aligned} \mathbf{h}_i &\leftarrow \text{AdaLN}(\mathbf{h}_i, \alpha_i, \beta_i), \\ \hat{\mathbf{x}}_i &\leftarrow \mathbf{W}_{\text{readout}} \mathbf{h}_i, \end{aligned}$$

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

D.2 SO(3) Equivariant DiTMC

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

D.2.1 Self-Attention Operation

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

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

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

$$\mathbf{u}_{ij} = \phi(r_{ij}) \odot \mathbf{c}_{ij}^G, \quad \hat{\mathbf{u}}_{ij} = \mathbf{p}_{ij}^{\text{PE}(3)} \odot \mathbf{c}_{ij}^G, \quad (\text{A32})$$

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

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

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

D.2.2 Adaptive Layer Normalization and Adaptive Scale

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

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

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

Equivariant adaptive scale is defined as

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

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

Within each SO(3)-equivariant DiTMC block, we calculate

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

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

D.2.3 Readout

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

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

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

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

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

D.3 Conditioning GNN

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

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

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

$$\mathbf{e}'_{ij} \leftarrow f_e(\mathbf{e}_{ij}, \mathbf{v}_i, \mathbf{v}_j) \quad (\text{A37})$$

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

$$\mathbf{v}'_i \leftarrow f_v \left(\mathbf{v}_i, \sum_j \mathbf{e}'_{ij} \right) \quad (\text{A38})$$

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

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

E Equivariance Proof for Euclidean Positional Embeddings

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

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

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

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

$$\mathbf{p}_{ij}^{\text{PE}(3),(\ell)}(\vec{r}_{ij}) = \phi_\ell(\|\vec{r}_{ij}\|) \odot \mathbf{Y}_\ell(\hat{r}_{ij}), \quad (\text{A40})$$

where $\phi_\ell : \mathbb{R} \mapsto \mathbb{R}^{1 \times H}$ is a radial filter function, $\hat{r} = \vec{r}/r$, and $\mathbf{Y}_\ell \in \mathbb{R}^{(2\ell+1) \times 1}$ 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 $(\phi_\ell \odot \mathbf{Y}_\ell) \in \mathbb{R}^{(2\ell+1) \times H}$. 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

$$\mathbf{p}_{ijc}^{\text{PE}(3),(\ell)}(\vec{r}_{ij}) = \phi_{\ell c}(\|\vec{r}_{ij}\|) \odot \mathbf{Y}_\ell(\hat{r}_{ij}). \quad (\text{A41})$$

Rotating the input positions in Eq. A41 leads to

$$\mathbf{p}_{ijc}^{\text{PE}(3),(\ell)}(\mathbf{R}\vec{r}_{ij}) = \phi_{\ell c}(\|\mathbf{R}\vec{r}_{ij}\|) \odot \mathbf{Y}_\ell(\mathbf{R}\hat{r}_{ij}) \quad (\text{A42})$$

$$= \phi_{\ell c}(\|\vec{r}_{ij}\|) \odot \mathbf{D}^{(\ell)}(\mathbf{R})\mathbf{Y}_\ell(\hat{r}_{ij}), \quad (\text{A43})$$

$$= \mathbf{D}^{(\ell)}(\mathbf{R})\mathbf{p}_{ijc}^{\text{PE}(3),(\ell)}(\vec{r}_{ij}) \quad (\text{A44})$$

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

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

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

transforms under rotation as

$$\mathbf{p}_{ij}^{\text{PE}(3)}(\mathbf{R}\vec{r}_{ij}) = \mathbf{D}(\mathbf{R}) \mathbf{p}_{ij}^{\text{PE}(3)}(\vec{r}_{ij}) \quad (\text{A46})$$

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

F Invariance Proof for the Dot-Product

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

$$\mathbf{h}(\mathbf{R}\vec{r}) = \mathbf{D}(\mathbf{R})\mathbf{h}(\vec{r}), \quad (\text{A47})$$

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

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

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

G Implementation details

G.1 Data Preprocessing

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

G.2 Input Featurization

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

G.3 Evaluation Metrics

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

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

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

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

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

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

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

G.4 Chirality Correction

Given the four 3D coordinates around a chirality center denoted as $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4 \in \mathbb{R}^3$ with $\mathbf{p}_i = (x_i, y_i, z_i)$ for $i = 1, 2, 3, 4$, we can compute the oriented volume OV of the tetrahedron via

$$\begin{aligned} OV(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4) &= \frac{1}{6} \cdot \det \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \end{bmatrix} \right) \\ &= \frac{1}{6} \cdot (\mathbf{p}_1 - \mathbf{p}_4) \cdot ((\mathbf{p}_2 - \mathbf{p}_4) \times (\mathbf{p}_3 - \mathbf{p}_4)). \end{aligned} \quad (\text{A53})$$

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

G.5 Pareto Front

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

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

| Method | COV-R [%]↑ | | AMR-R [Å]↓ | | COV-P [%]↑ | | AMR-P [Å]↓ | |
|-----------------------|---------------------|----------------------|------------------------|------------------------|---------------------|----------------------|------------------------|------------------------|
| | Mean | Median | Mean | Median | Mean | Median | Mean | Median |
| GeoMol (0.3M) | 91.5 | 100.0 | 0.225 | 0.193 | 86.7 | 100.0 | 0.270 | 0.241 |
| GeoDiff (1.6M) | 76.5 | 100.0 | 0.297 | 0.229 | 50.0 | <u>33.5</u> | 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 | <u>95.4</u> ±0.1 | 100.0 ±0.0 | <u>0.085</u> ±0.001 | 0.037 ±0.000 |
| *DiTMC+rPE-B (9.6M) | <u>96.3</u> ±0.0 | 100.0 ±0.0 | <u>0.070</u> ±0.001 | <u>0.027</u> ±0.000 | 95.7 ±0.1 | 100.0 ±0.0 | 0.080 ±0.000 | <u>0.035</u> ±0.000 |
| *DiTMC+PE(3)-B (8.6M) | 95.7 ±0.3 | 100.0 ±0.0 | 0.068 ±0.002 | 0.021 ±0.001 | 93.4 ±0.2 | 100.0 ±0.0 | 0.089 ±0.002 | 0.032 ±0.001 |

G.6 Ensemble Properties

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

H Additional Experimental Results

H.1 Results on GEOM-QM9

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

H.2 Results on GEOM-DRUGS

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

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

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

H.3 Results on GEOM-XL

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

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

| Method | COV-R [%] \uparrow | | AMR-R [\AA] \downarrow | | COV-P [%] \uparrow | | AMR-P [\AA] \downarrow | |
|------------------------|----------------------|-------------------|-------------------------------------|----------------------|--------------------------|--------------------------|-------------------------------------|-----------------------------|
| | Mean | Median | Mean | Median | Mean | Median | Mean | Median |
| GeoMol (0.3M) | 44.6 | 41.4 | 0.875 | 0.834 | 43.0 | 36.4 | 0.928 | 0.841 |
| GeoDiff (1.6M) | 42.1 | 37.8 | 0.835 | 0.809 | 24.9 | 14.5 | 1.136 | 1.090 |
| Tors. Diff. (1.6M) | 72.7 | 80.0 | 0.582 | 0.565 | 55.2 | 56.9 | 0.778 | 0.729 |
| MCF-S (13M) | 79.4 | 87.5 | 0.512 | 0.492 | 57.4 | 57.6 | 0.761 | 0.715 |
| MCF-B (64M) | 84.0 | 91.5 | 0.427 | 0.402 | 64.0 | 66.2 | 0.667 | 0.605 |
| MCF-L (242M) | <u>84.7</u> | <u>92.2</u> | <u>0.390</u> | 0.247 | 66.8 | 71.3 | 0.618 | 0.530 |
| DMT-L (150M) | 85.8 | 92.3 | 0.375 | <u>0.346</u> | 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 | <u>77.8</u> ± 0.1 | <u>85.7</u> ± 0.5 | <u>0.470</u> ± 0.001 | <u>0.387</u> ± 0.003 |
| *DiTMC+rPE-L (28.3M) | 78.7 ± 0.1 | 84.1 ± 0.4 | 0.438 ± 0.002 | 0.388 ± 0.005 | 78.1 ± 0.1 | 86.4 ± 0.3 | 0.466 ± 0.001 | 0.381 ± 0.003 |
| *DiTMC+PE(3)-L (31.1M) | 80.8 ± 0.3 | 85.6 ± 0.1 | 0.415 ± 0.003 | 0.376 ± 0.001 | 76.4 ± 0.2 | 82.6 ± 0.3 | 0.491 ± 0.002 | 0.414 ± 0.004 |

I Additional Ablations

I.1 Gaussian vs. Harmonic Prior

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

I.2 Conditioning Strategies on GEOM-QM9

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

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

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

Specifically, we ablate the following conditioning strategies:

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

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

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

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

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

I.3 Index Positional Encoding (iPE)

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

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

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

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

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

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

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

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

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

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

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

J Analysis of training loss as a function of latent time

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

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

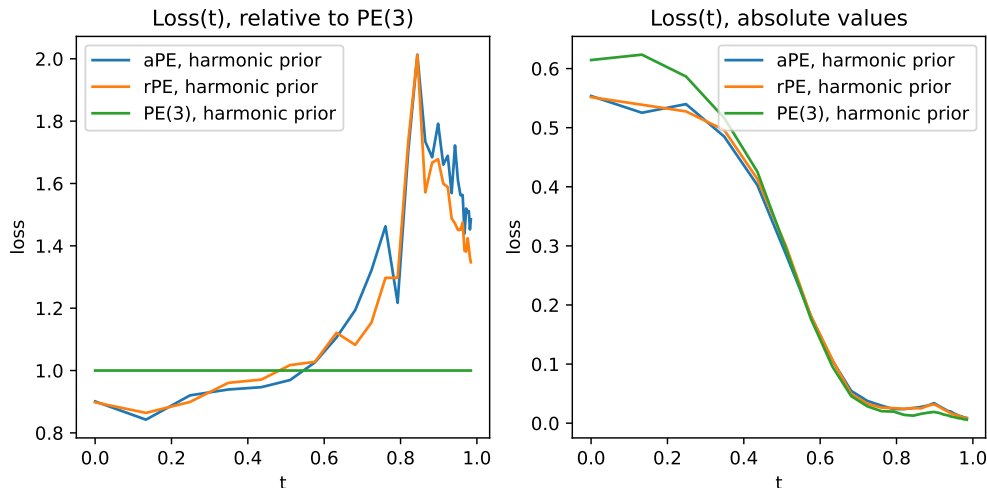


Figure A5: Loss as a function of time comparing different PE strategies. Results averaged over 1000 samples randomly drawn from the GEOM-QM9 validation set. **Left:** loss relative to PE(3) as a baseline. In the important regime close to the data distribution, the model PE(3) has lower loss, yielding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

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

K SMILES Classification Experiment via Conditioning GNN

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

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

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

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

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

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

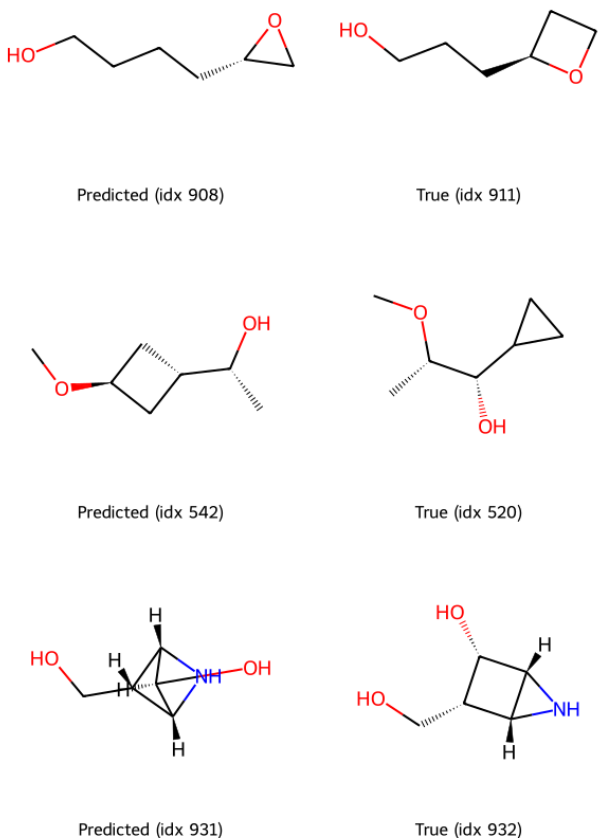


Figure A6: Misclassified examples for SMILES classification experiment. We randomly pick 3 examples, which are misclassified by a classification head without any GNN layers. We show that GNN layers are essential for correct classification of isomers.

L Analysis of Sampling Trajectories

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

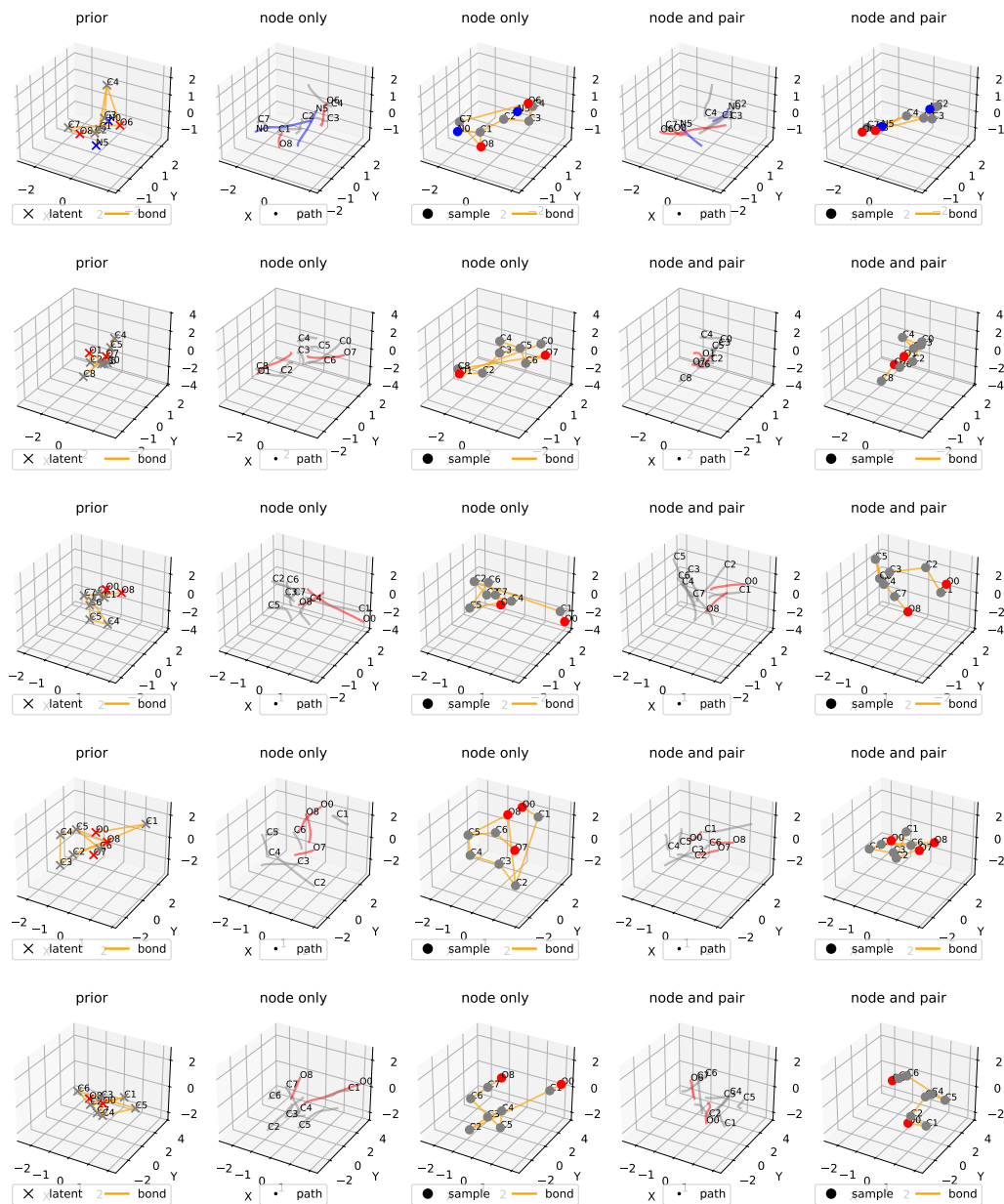


Figure A7: Comparison of molecular generation with node-only versus node- and pair-wise (all-pair) conditioning on GEOM-QM9. Each row shows a prior sample, sampling trajectories, and final generated structure for both models; pairwise conditioning preserves bonds from the conditioning graph \mathcal{G}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

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

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

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

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

M Visualization

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

N Code and Data Availability

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

DiTMC-aPE

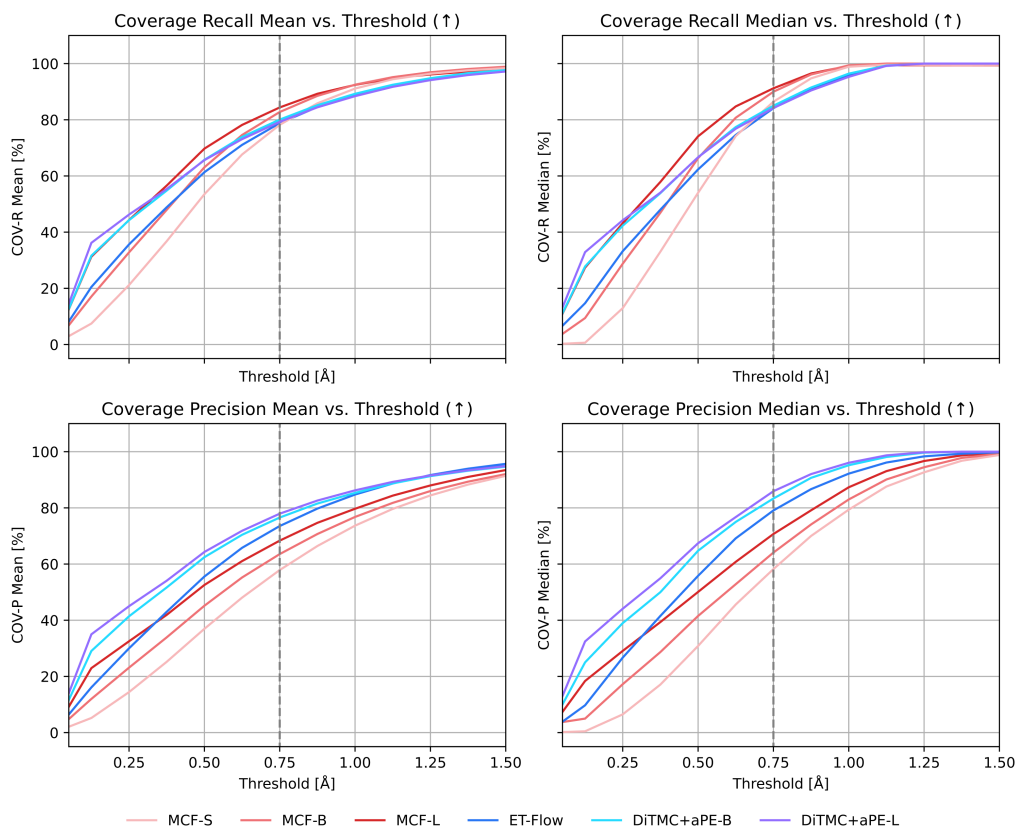


Figure A8: Coverage (COV) for precision (“-P”) and recall (“-R”) as a function of RMSD threshold δ for DiTMC+aPE and other state-of-the-art methods on GEOM-DRUGS. The vertical dashed line denotes the RMSD threshold $\delta = 0.75$ commonly used for evaluation on the GEOM-DRUGS dataset.

DiTMC-rPE

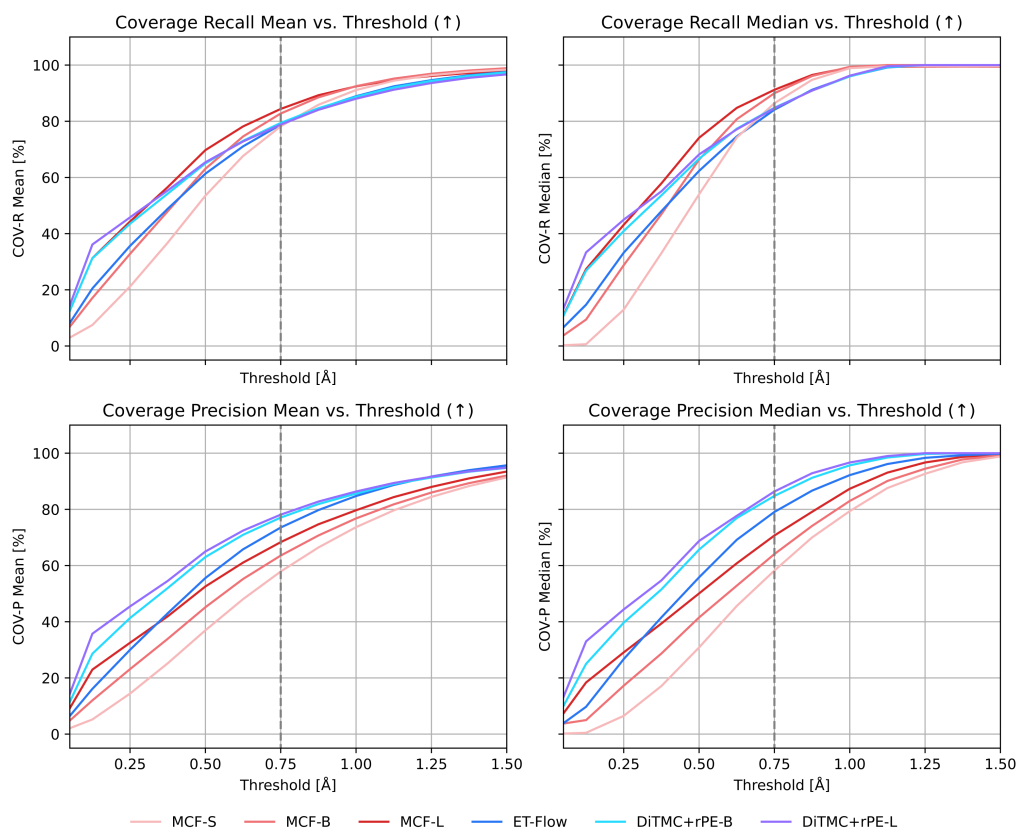


Figure A9: Coverage (COV) for precision (“-P”) and recall (“-R”) as a function of RMSD threshold δ for DiTMC+rPE and other state-of-the-art methods on GEOM-DRUGS. The vertical dashed line denotes the RMSD threshold $\delta = 0.75$ commonly used for evaluation on the GEOM-DRUGS dataset.

DiTMC-PE(3)

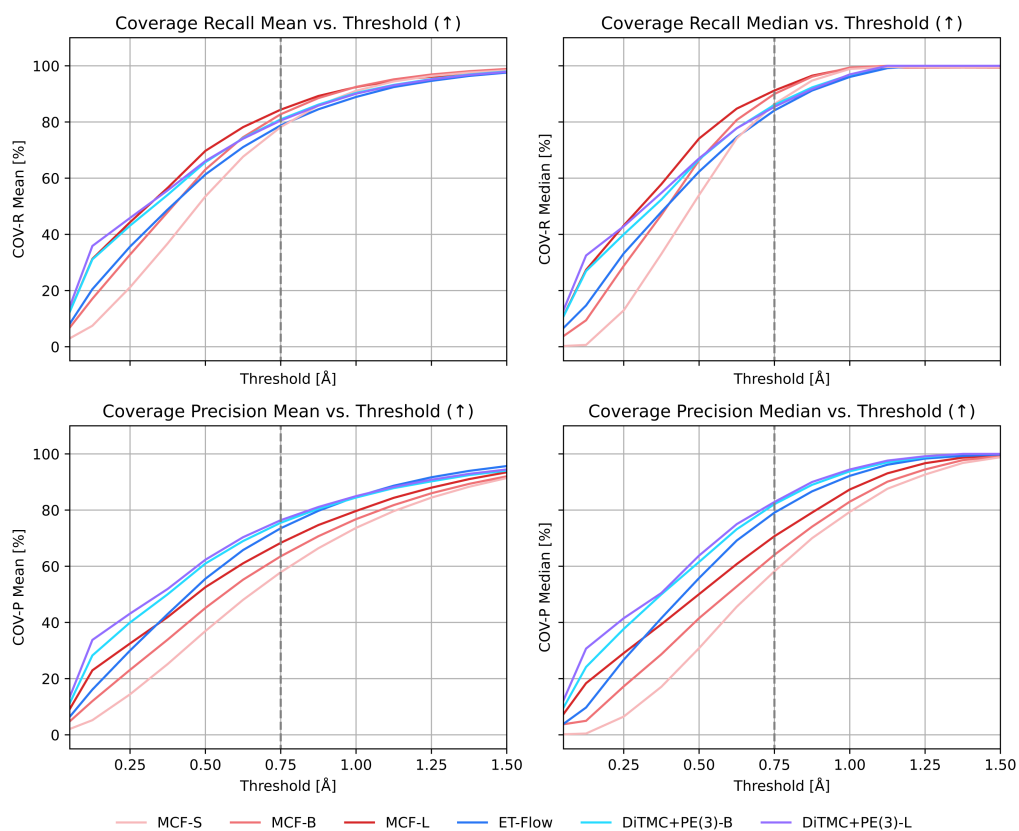


Figure A10: Coverage (COV) for precision (“-P”) and recall (“-R”) as a function of RMSD threshold δ for DiTMC+PE(3) and other state-of-the-art methods on GEOM-DRUGS. The vertical dashed line denotes the RMSD threshold $\delta = 0.75$ commonly used for evaluation on the GEOM-DRUGS dataset.

DiTMC-aPE

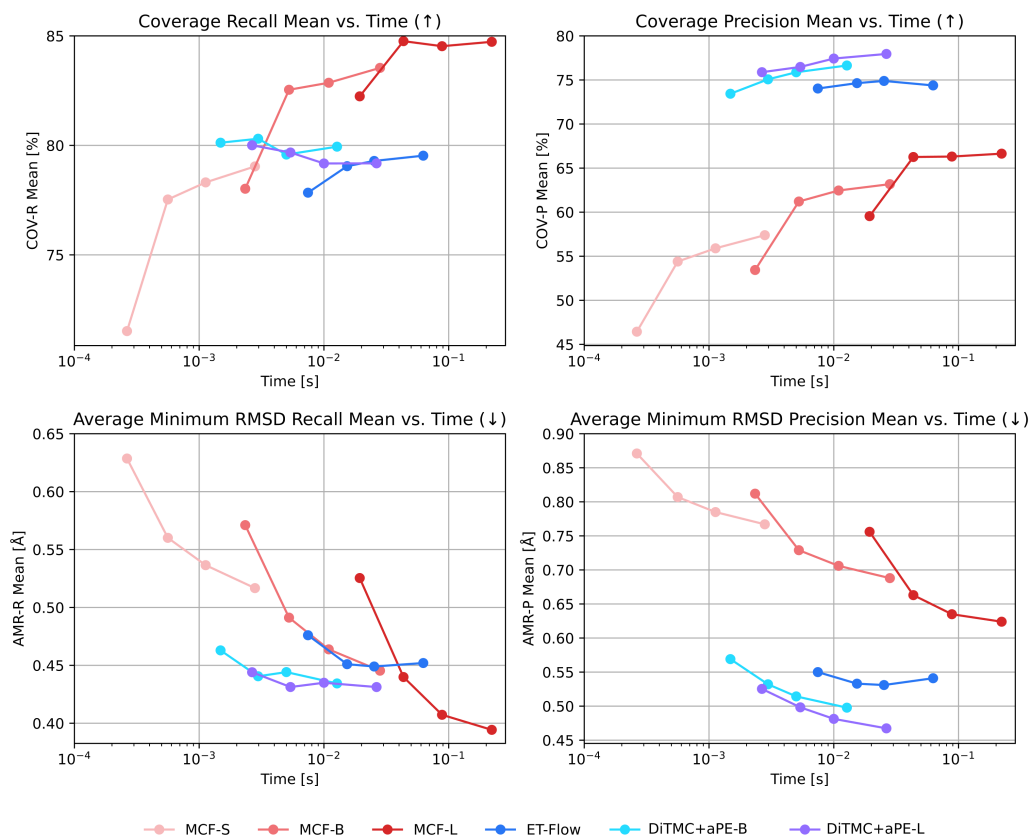


Figure A11: Average minimum RMSD (AMR) for precision (“-P”) and recall (“-R”) as a function of wall clock time per generated conformer. For each model, markers from left to right correspond to an increasing number of sampling steps during generation. Here, we follow Refs. [32, 47] and use 5, 10, 20,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

DiTMC-rPE

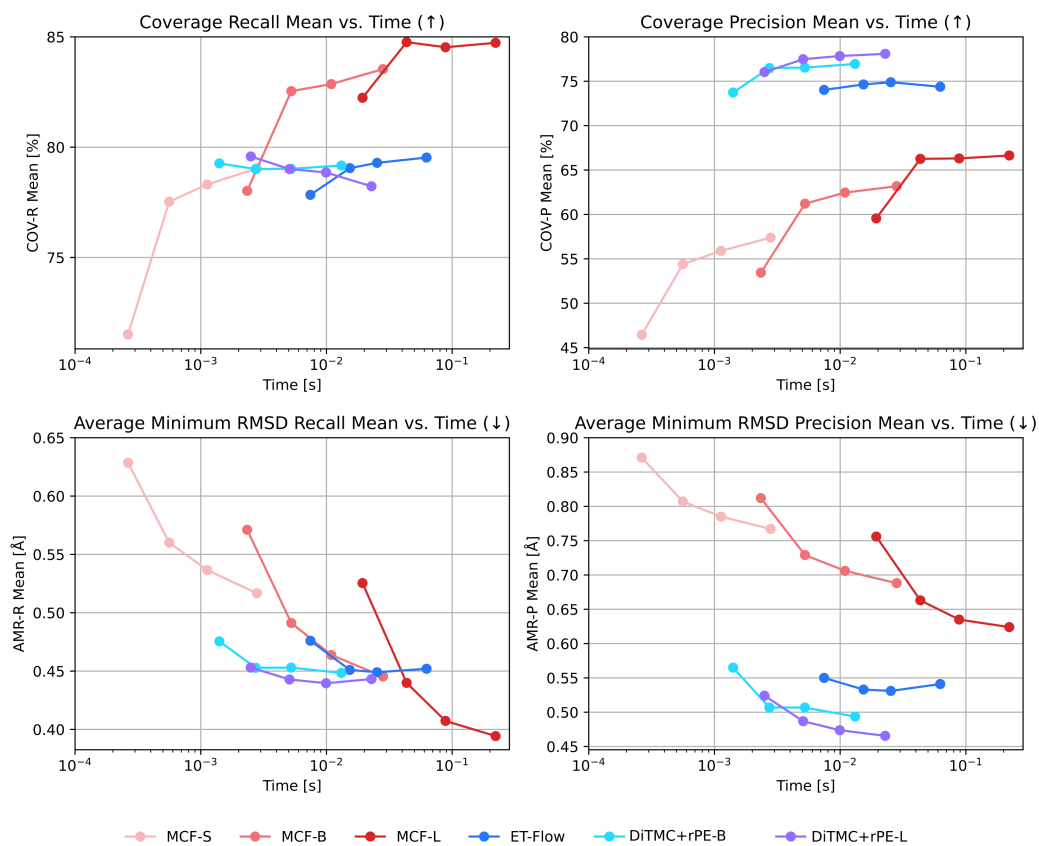


Figure A12: Average minimum RMSD (AMR) for precision (“-P”) and recall (“-R”) as a function of wall clock time per generated conformer. For each model, markers from left to right correspond to an increasing number of sampling steps during generation. Here, we follow Refs. [32, 47] and use 5, 10, 20,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

DiTMC-PE(3)

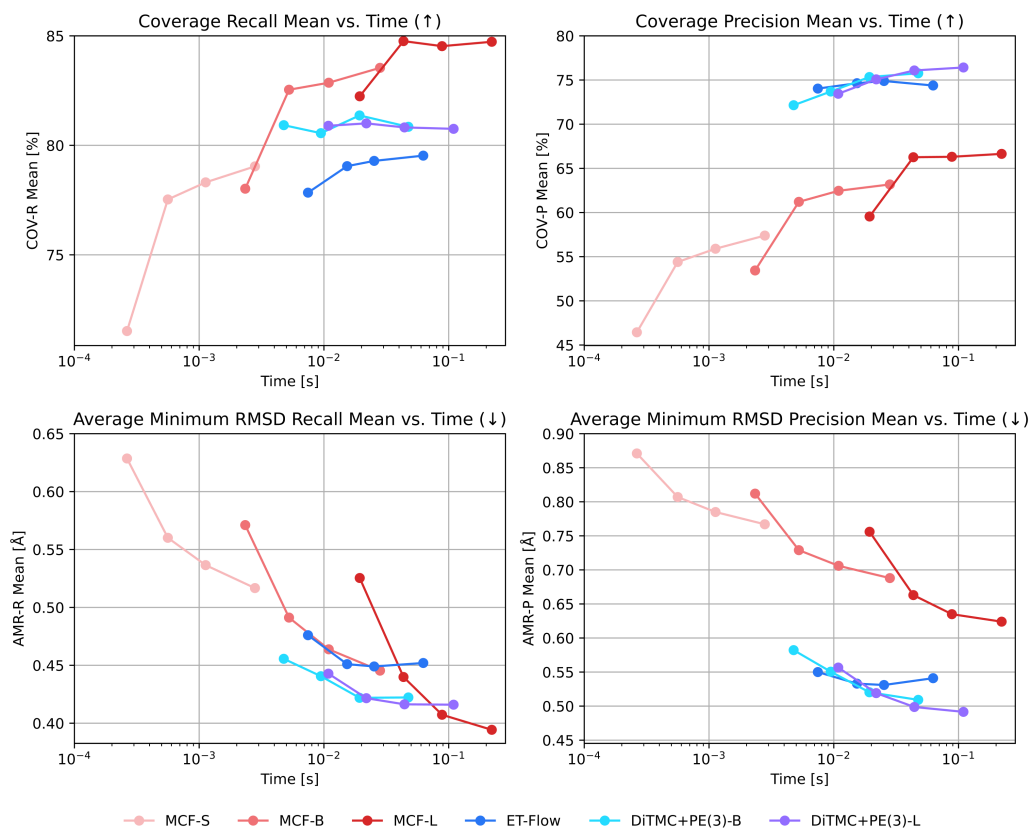


Figure A13: Average minimum RMSD (AMR) for precision (“-P”) and recall (“-R”) as a function of wall clock time per generated conformer. For each model, markers from left to right correspond to an increasing number of sampling steps during generation. Here, we follow Refs. [32, 47] and use 5, 10, 20,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

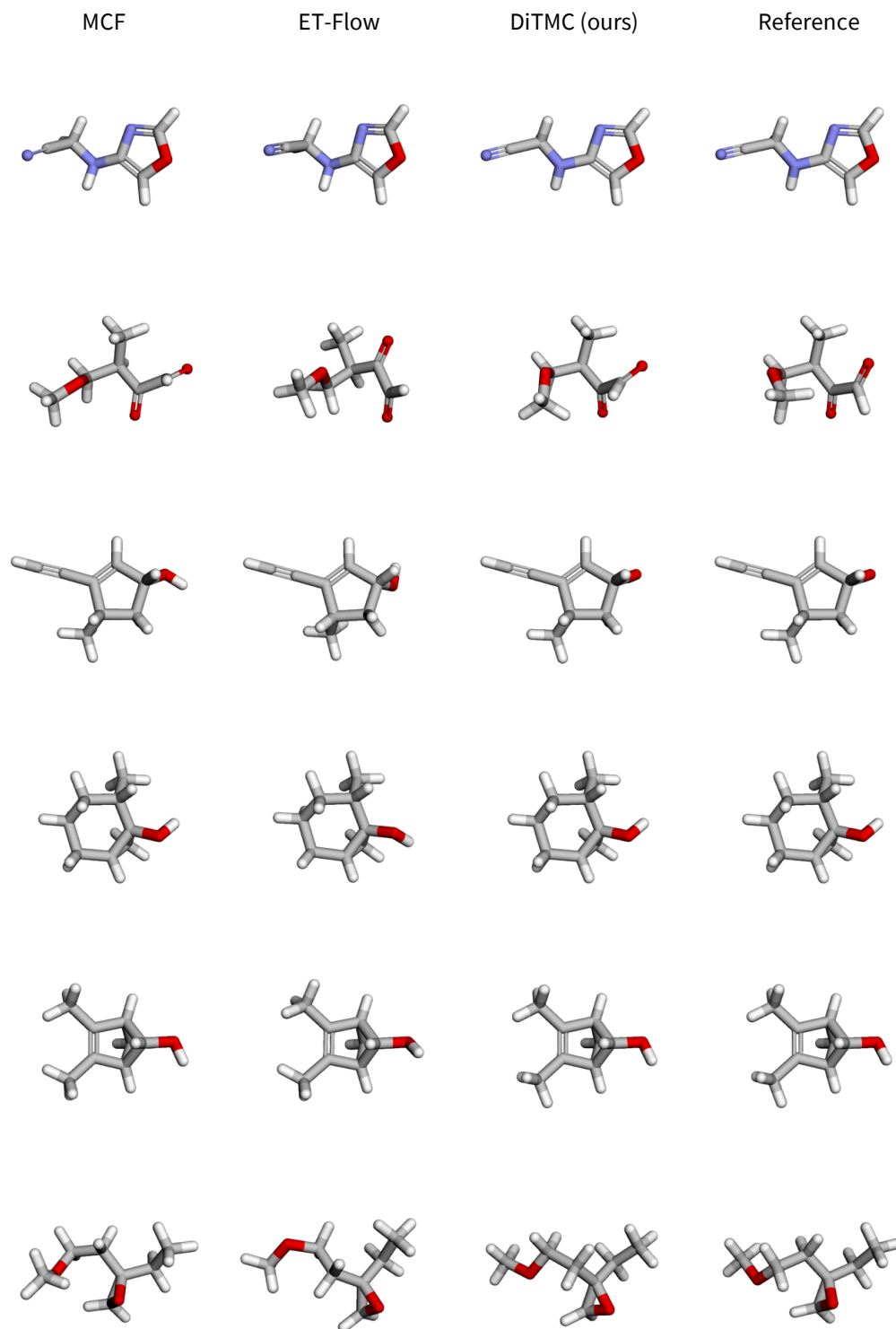


Figure A14: Comparison of conformers generated by MCF, ET-Flow, and DiTMC against ground-truth reference conformers from GEOM-QM9. The generated conformers are rotationally aligned with their corresponding reference conformer to facilitate comparison. **From left to right:** generated conformers from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

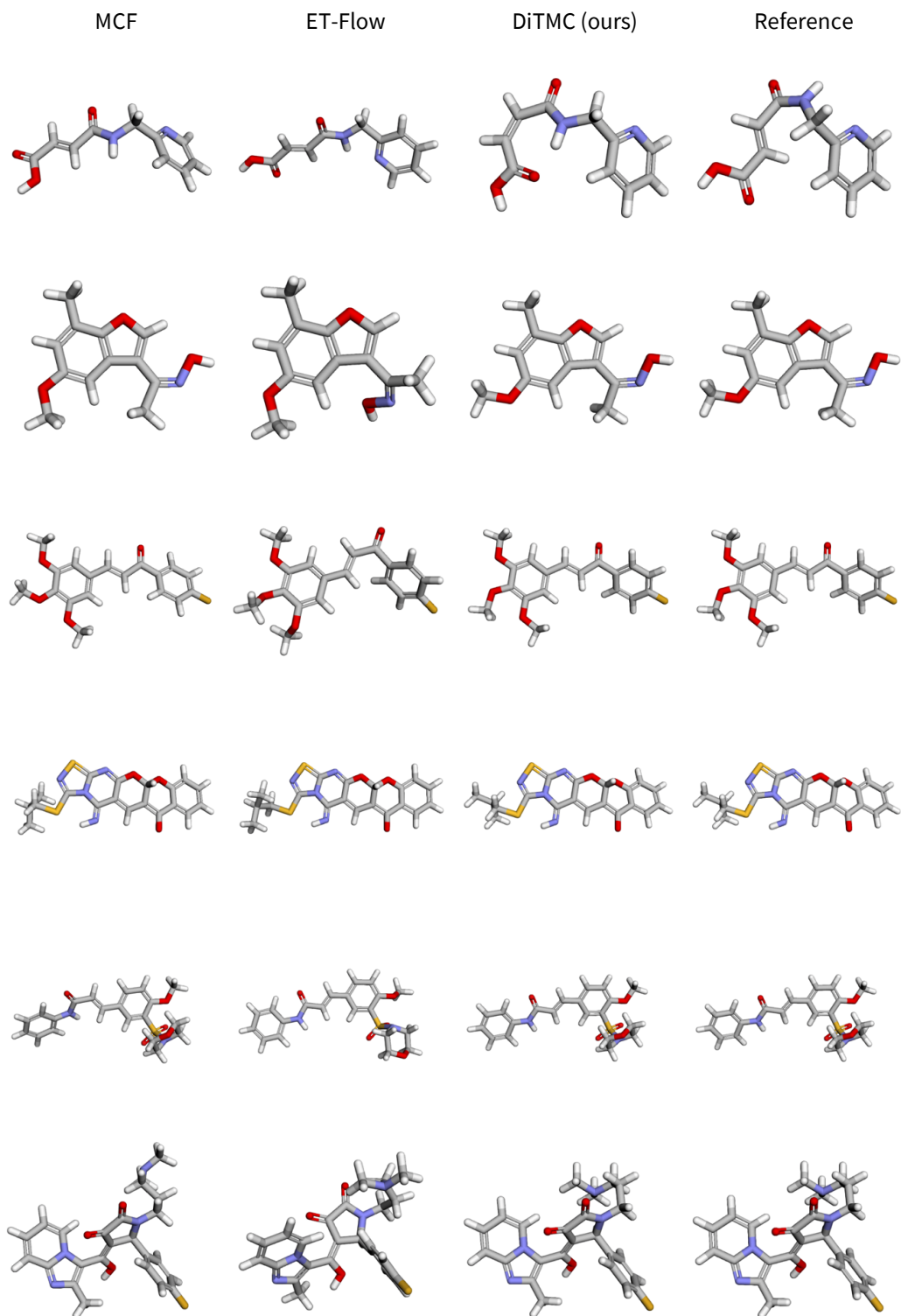


Figure A15: Comparison of conformers generated by MCF, ET-Flow, and DiTMC against ground-truth reference conformers from GEOM-DRUGS. The generated conformers are rotationally aligned with their corresponding reference conformer to facilitate comparison. **From left to right:** generated conformers from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

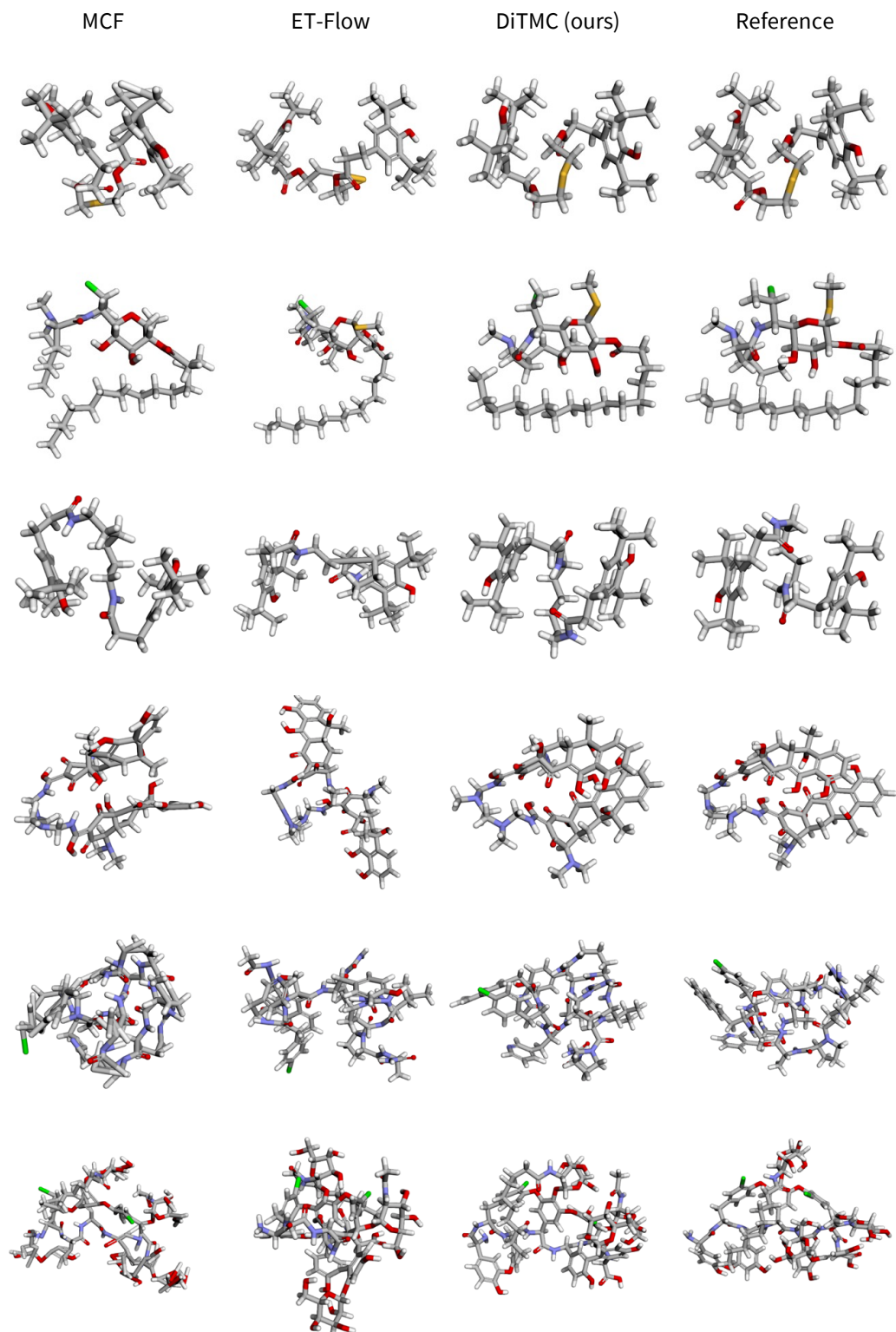


Figure A16: Comparison of conformers generated by MCF, ET-Flow, and DiTMC against ground-truth reference conformers from GEOM-XL. The generated conformers are rotationally aligned with their corresponding reference conformer to facilitate comparison. **From left to right:** generated conformers from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