FLEXIFLOW: DECOMPOSABLE FLOW MATCHING FOR GENERATION OF FLEXIBLE MOLECULAR ENSEMBLE

Anonymous authors
Paper under double-blind review

ABSTRACT

Sampling useful three-dimensional molecular structures along with their most favorable conformations is a key challenge in drug discovery. Current state-of-the-art 3D de-novo design flow matching or diffusion-based models are limited to generating a single conformation. However, the conformational landscape of a molecule determines its observable properties and how tightly it is able to bind to a given protein target. By generating a representative set of low-energy conformers, we can more directly assess these properties and potentially improve the ability to generate molecules with desired thermodynamic observables. Towards this aim, we propose *FlexiFlow*, a novel architecture that extends flow-matching models, allowing for the joint sampling of molecules along with multiple conformations while preserving both equivariance and permutation invariance. We demonstrate the effectiveness of our approach on the QM9 and GEOM Drugs datasets, achieving state-of-the-art results in molecular generation tasks. Our results show that FlexiFlow can generate valid, unstrained, unique, and novel molecules with high fidelity to the training data distribution, while also capturing the conformational diversity of molecules. Moreover, we show that our model can generate conformational ensembles that provide similar coverage to state-of-the-art physics-based methods at a fraction of the inference time. Finally, FlexiFlow can be successfully transferred to the protein-conditioned ligand generation task, even when the dataset contains only static pockets without accompanying conformations.

1 INTRODUCTION

Flow matching Lipman et al. (2023) and diffusion models (Ho et al., 2020) now deliver state-of-the-art generation across images, audio, and 3D shapes (Yang et al., 2024), enabled by strong theory and flexible density modeling. Recent work sharpens flow matching for higher fidelity (Domingo-Enrich et al., 2025), compositional generation (Skreta et al., 2025), and faster, more stable, or guided sampling (Liu et al., 2025). Along these lines, we introduce a conditional flow decomposition framework that enables the joint generation of molecular graphs and multiple conformers. Recent diffusion and flow-matching models have advanced de novo molecular generation (Hoogeboom et al., 2022; Schneuing et al., 2024), protein design (Watson et al., 2023; Anand & Achim, 2022), and protein structure prediction (Jumper et al., 2021; Ingraham et al., 2019), leveraging E(3)/SE(3)-equivariant architectures to capture 3D symmetries. Their application to unconditional 3D small-molecule generation is highly promising (Vignac et al., 2023; Le et al., 2024; Irwin et al., 2025), reaching benchmark ceilings on QM9 (Ramakrishnan et al., 2014) and GEOM Drugs (Axelrod & Gómez-Bombarelli, 2022). However, current models typically produce only a single conformer per molecule, limiting conformational diversity critical for drug discovery, as different conformations can exhibit significantly different biological activities and properties. For a specific target protein, sampling multiple ligand conformations to

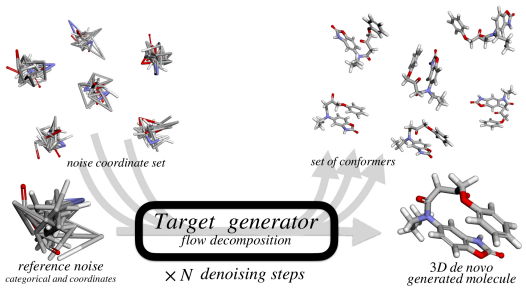

Figure 1: From noise samples, our model generates both molecular graphs and their conformational ensembles.

match distinct interaction patterns is crucial for optimizing binding affinity and selectivity of a drug candidate (Leach et al., 2009).

To address this limitation of current models, we propose a novel framework that extents flow matching to jointly generate molecular graphs and representative sets of conformations. To this end, we introduce FlexiFlow [1] , a novel architecture that uses this paradigm to handle two sets of coordinates. FlexiFlow preserves permutation invariance on categorical and equivariance on two sets of coordinates features, moreover, it leverages this decomposition to design novel molecular structures along with their 3D conformers (illustration in Figure 1). Our framework shows promising results in generating valid, unique, and novel molecular structures with realistic conformations. Unlike other approaches, like Bolzmann generators (Noé et al., 2019; Diez et al., 2025), FlexiFlow efficiently generates high-quality molecular structures and conformations while not requiring additional MD data for training and avoiding major computational and transferability limitations. The method can be successfully applied to other data modalities, where we present results on MNIST in Appendix E.

We summarize the key contributions as follows:

- We propose a novel framework leveraging conditional independence to decompose the flow matching objective, enabling simultaneous generation of a graph with a single reference conformation along with an arbitrary set of conformers.
- We introduce a new architecture, FlexiFlow, that handles equivariance on two coordinate sets for 3D molecular generation.
- We conduct extensive experiments on benchmark datasets, demonstrating that FlexiFlow achieves state-of-the-art performance in 3D molecular generation tasks.
- We compare FlexiFlow generated conformers with those generated by traditional physics-based methods (CREST (Pracht et al., 2020; 2024)), showing that FlexiFlow generates high-quality conformers at a fraction of the computational cost.
- We show that the FlexiFlow framework transfers effectively to tasks that lack conformation datasets. In particular, we investigate protein conditioning: given a specific static pocket, we generate molecular graphs, each with multiple conformations. This demonstrates the potential of our method for real drug-discovery applications.

2 Related Works

Flow matching unifies score-based diffusion and ODE-based generative modeling by learning vector fields that map simple priors to complex data distributions (Lipman et al., 2024). Subsequent work has improved its scalability (Wildberger et al., 2023) and sample quality (Gat et al., 2024). Recent 3D molecular design methods fall into two groups: (1) generate molecular graphs then infer bonds from coordinates (Hoogeboom et al., 2022), and (2) jointly generate the graph and its bonds (Irwin et al., 2025). EDM by (Hoogeboom et al., 2022) addresses this challenge using a diffusion model for 3D molecular structure generation. Other methods (GCDM (Morehead & Cheng, 2024), GFMDiff (Xu et al., 2024), EquiFM (Song et al., 2023), GeomLDM (Xu et al., 2023), GeomBFM (Song et al., 2024), MUDiff (Hua et al., 2023)) often produced unstable structures, but newer architectures and training strategies (FlowMol (Dunn & Koes, 2024), MiDi (Vignac et al., 2023) and EQGATdiff (Le et al., 2024)) have rapidly improved. Conditional flow matching (CFM) is now a leading approach (Song et al., 2023; Campbell et al., 2024), with recent work improving efficiency and generation quality Tabasco (Vonessen et al., 2025), FlowMol3 (Dunn & Koes, 2025), and SemlaFlow (Irwin et al., 2025). Conformers are critical when conditioning on a protein pocket (Peng et al., 2022; Dong et al., 2024), as they shape molecular bioactivity and properties. Conformers can be generated using physics-based tools such as CREST (Pracht et al., 2024) which searches for low-energy conformational minima are computationally expensive. More recent approaches, such as Adjoint Sampling (Havens et al., 2025), generate conformers via flow matching while keeping the molecular graph fixed. This substantially reduces the computational cost required to produce conformers that closely resemble minimum-energy states. We extend these advances with a novel conditional flow matching approach that decomposes the flow via a conditional independence assumption, enabling the joint generation of molecular graphs and their conformations.

[1] Per ICLR guidelines, we will provide an anonymous link for ACs and reviewers at the start of the discussion period; the code will be publicly released upon acceptance.

3 BACKGROUND

Flow matching offers a versatile and efficient framework to learn a generative process that maps an easy-to-sample distribution p_{noise} to a complex one, often denoted as $p_{data} = q$. To transport p_{noise} to q, we can define a marginal probability path $p_t(x)$, parameterized by t, that connects the two. For the sake of simplicity, we assume $x \in \mathbb{R}^n$ but the flow matching framework can be extended to more complex structured data. At the beginning of the path, when $t = 0$, $p_0(x) = p_{noise}(x)$, whereas at the end, when $t = 1$, $p_1(x) = q(x)$. The vector field is learned by a neural network $v_t(x; \theta)$ that acts as a mediator between the two distributions and learns to approximate the true vector field $u_t(x)$. The training objective is to minimize the following flow matching loss: $\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{t,\, x\sim p_t(x)} \left\| v_t(x;\theta) - u_t(x) \right\|^2$. In practice, evaluating $p_t(x)$ and $u_t(x)$ is computationally intractable, as they require integrating over the entire data distribution, which cannot be done analytically. The solution for this problem was proposed by Conditional Flow Matching (CFM), by Lipman et al. (2023):

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

where $q(x_1)$ represents the target data distribution, $p_t(x \mid x_1)$ the conditional probability path connecting the prior at $t = 0$ to a distribution concentrated around x_1 at $t = 1$, and $u_t\big(x \mid x_1\big)$ the target conditional vector field.

4 METHOD

We aim to extend the flow matching paradigm to handle simultaneous flow integration on a set of vectors $\mathcal{S} = \{y_i\}_{i=1}^m$ with a representative vector $x \in \mathcal{S}$.

4.1 FLOW DECOMPOSITION

We consider a time-dependent probability density function $p_t : \mathbb{R}^n \times \mathbb{R}^{n\times m} \to \mathbb{R}$ and v_t a time-dependent vector field $v_t : \mathbb{R}^n \times \mathbb{R}^{n\times m} \to \mathbb{R}^n \times \mathbb{R}^{n\times m}$, where $t \in [0, 1]$. Under the conditional independence assumption on $\mathcal{S} = \{y_i\}_{i=1}^m$, and p_t can be defined as:

$$p_t(x, \mathcal{S}) := \prod_{y\in\mathcal{S}} p_t(x, y). \tag{2}$$

The vector field v_t generates a flow $\psi_t : \mathbb{R}^n \times \mathbb{R}^{n\times m} \to \mathbb{R}^n \times \mathbb{R}^{n\times m}$, which is obtained by solving the corresponding ordinary differential equation (ODE):

$$\frac{\partial \psi_t(x, \mathcal{S})}{\partial t} = v_t(\psi_t(x, \mathcal{S})), \quad \psi_0(x, \mathcal{S}) = (x, \mathcal{S}). \tag{3}$$

We define the flow ψ_t on $(x, \mathcal{S})$ as the concatenation of m independent flows, each of them evaluated on the pair (x, y), $y \in \mathcal{S}$

$$\psi_t(x, \mathcal{S}) = \Big\|_{y\in\mathcal{S}} \psi_t(x, y). \tag{4}$$

The push-forward equation allows us to transform a known distribution (e.g., Gaussian or uniform) p_0 into a complex data distribution p_1.

$$p_0(x, \mathcal{S}) = p_t(\psi_t(x, \mathcal{S})) \cdot \left[\det\left(\nabla_{(x,\mathcal{S})}\psi_t(x, \mathcal{S})\right)\right]. \tag{5}$$

Using Equations 2 and 4, we can decompose push-forward as:

$$\prod_{y\in\mathcal{S}} p_0(x, y) = \prod_{y\in\mathcal{S}} p_t(\psi_t(x, y)) \cdot \left[\det\left(\nabla_{(x,y)}\psi_t(x, y)\right)\right]. \tag{6}$$

As the flow decomposition is applied to the concatenated set of independent flows, all the findings of Lipman et al. (2023) remain valid. For the sake of completeness, we provide in Appendix C.1 a proof that the determinant of a block diagonal matrix is the product of the determinants of the

blocks. Since, the flow $\psi_t(x, \mathcal{S})$ is defined as the concatenation of independent flows $\psi_t(x, y_i)$, we decompose ψ_t into:

$$\frac{\partial \psi_t(x, \mathcal{S})}{\partial t} = \left(\frac{\partial \psi_t(x, y_1)}{\partial t}, \ldots, \frac{\partial \psi_t(x, y_m)}{\partial t} \right) = (v_t(\psi_t(x, y_1)), \ldots, v_t(\psi_t(x, y_m))) \tag{7}$$

where each $\psi_t(x, y_i)$ is independent of the other y_j for $j \neq i$. Finally, following Lipman et al. (2023), we can define the flow matching objective for pairs $(x, \mathcal{S})$ as:

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{t, p_t(x,\mathcal{S})} \left\| v_t(x, \mathcal{S}; \theta) - u_t(x, \mathcal{S}) \right\|^2 = \sum_{y \in \mathcal{S}} \mathbb{E}_{t, p_t(x,y)} \left\| v_t(x, y; \theta) - u_t(x, y) \right\|^2, \tag{8}$$

where v_t is modeled with a neural network parametrized by θ. Since, $p_t(x, \mathcal{S})$ and $u_t(x, \mathcal{S})$ do not have a closed form, we cannot optimize directly over those terms. However, we can leverage Conditional Flow Matching (CFM) (Lipman et al., 2023) that we extend to pairs $(x, \mathcal{S})$:

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t \sim \mathcal{U}[0,1],\, (x_1, \mathcal{S}_1) \sim q(x_1, \mathcal{S}_1),\, (x, \mathcal{S}) \sim p_t(x, \mathcal{S} | x_1, \mathcal{S}_1)} \left\| v_t(x, \mathcal{S}; \theta) - u_t\big(x, \mathcal{S} \mid x_1, \mathcal{S}_1\big) \right\|^2, \tag{9}$$

where $q(x_1, \mathcal{S}_1)$ represents the target data distribution, $p_t(x, \mathcal{S} \mid x_1, \mathcal{S}_1)$ the conditional probability path connecting the prior at $t = 0$ to a distribution concentrated around $(x_1, \mathcal{S}_1)$ at $t = 1$, and $u_t\big(x, \mathcal{S} \mid x_1, \mathcal{S}_1\big)$ the target conditional vector field.

4.2 Flow decomposition on molecular graphs

Our objective is to define a model that generates novel molecular structures along with their low-energy conformations. We denote by $\mathcal{X}$ the molecular space, whose its elements are molecular graphs. A molecular graph with n atoms is defined as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{S}\}$ where $\mathcal{V} \in \mathbb{N}^{n \times 2}$ are the vertices (atom and charge types), $\mathcal{E} \in \mathbb{N}^{n \times n}$ represent the edges (bonds) and $\mathcal{S} = \{y_1, \ldots, y_m \mid y_i \in \mathbb{R}^{n \times 3}\}$ a set of m conformations. There is a one-to-one correspondence between atoms in each conformation and nodes in the graph. We decomposed each molecular graph in a set of 4-tuples

$$\mathcal{D}_{\mathcal{G}} = \{(\mathcal{V}, \mathcal{E}, x, y) \,|\, y \in \mathcal{S}\}, \tag{10}$$

where $x \in \mathcal{S}$ is the representative conformation. We denote with $\mathcal{D}$ the full dataset, that is the union over all the set of 4-tuples $\mathcal{D}_{\mathcal{G}}$. There are multiple ways to choose x. In our case, we select the conformation that is closest to the average conformation, i.e., $x = \arg\min_{y \in \mathcal{S}} \|y - \frac{1}{N} \sum_{y \in \mathcal{S}} y\|^2$.

Equivariance. To support the generation of each data point $(\mathcal{V}, \mathcal{E}, x, y) \in \mathcal{D}$, our architecture supports two separate inputs for x and y, and maintains equivariance to rotation not only when the same rotation R is applied to both x and y, but also when the two different rotations $R_x \neq R_y$ are applied independently. This allows the model to focus on learning the symmetries while retaining equivariance. In the following section, we will show that by using the scalar product on the feature coordinates (see Equation 13), we can construct messages conditioned on both x and y, while preserving equivariance.

4.2.1 FlexiFlow model

We introduce the FlexiFlow architecture which supports the generation of $(\mathcal{V}, \mathcal{E}, x, y)$, where x and y belong to the set $\mathcal{S}$, and x is the representative conformation. FlexiFlow is inspired by SemlaFlow, originally proposed by Irwin et al. (2025). In this section, we highlight the differences and novelties introduced by FlexiFlow. The architecture is depicted in Figure 2, defined as the composition of one featurization layer, L repeated FlexiFlow layers, and a final feature refinement layer. The most relevant architectural differences compared to Irwin et al. (2025) are: (1) FlexiFlow requires an extra input y, (2) invariant features are used to condition both x and y, (3) FlexiFlow exchanges information between x and y, while preserving equivariance on both x and y. Full details of the entire architecture are deferred to the Appendix B.1. With minor architectural modification (all detailed in Appendix B.3), we additionally extended FlexiFlow to support protein pocket conditioning ligand generation.

Following the SemlaFlow notation, the atom and charge types in $\mathcal{V}$ and the bond types in $\mathcal{E}$ are represented as one hot vectors, denoted by h and e, respectively. Note that h is formed by concatenating

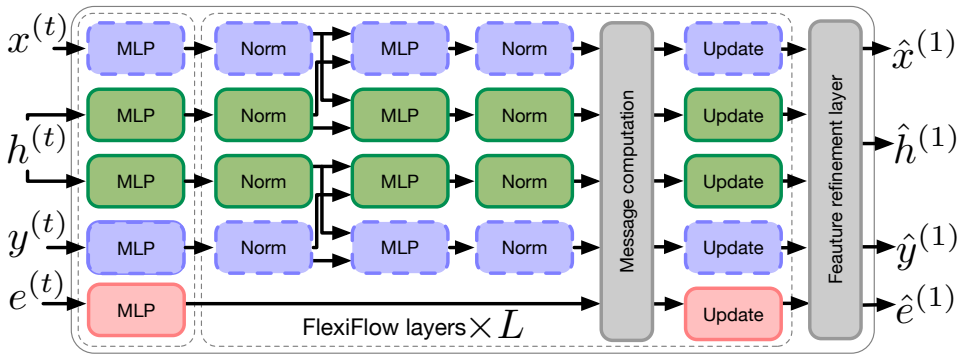

Figure 2: The FlexiFlow architecture takes equivariant and invariant features as input at time $t \in [0, 1]$ and produces predictions at $t = 1$. The left macro dashed block is the featurization layer. Blocks in the same column with the same color share weights. Solid blocks represent invariant features, while dashed blocks represent equivariant features. Message computation and feature refinement layer blocks produces both invariant and equivariant features.

two one-hot vectors: one encoding the atom type and the other encoding the charge type. Furthermore, h is concatenated with temporal information $t \in [0, 1]$. For ease of exposition, we adopt a slight abuse of notation, overwriting the symbols h, e, x, y to denote their transformed representations through the *featurization layer*. The featurization layer produces invariant features $h \in \mathbb{R}^{n \times d}$ and edge features $e \in \mathbb{R}^{n \times n \times d}$ with two multilayer perceptrons (MLPs). A shared linear layer maps x and y into coordinate sets $x \in \mathbb{R}^{n \times d \times 3}$ and $y \in \mathbb{R}^{n \times d \times 3}$ (see Appendix B.1 for details). The feature tensors h, e, x, and y are then fed into L stacked *FlexiFlow* layers. Each FlexiFlow layer consists of a feed forward layer followed by a graph attention layer. The feed forward consists of two MLPs Φ_θ and Ψ_θ for invariant and equivariant features respectively. Each invariant features h_i^x and h_i^y, where $h_i = h_i^x = h_i^y$ in the first FlexiFlow layer, are updated considering also the equivariant features as follows:

$$h_i^{x\,\mathrm{ff}} = h_i^x + \Phi_\theta([\,\tilde{h}_i^x\,,\,\|\tilde{x}_i\|\,]) \quad h_i^{y\,\mathrm{ff}} = h_i^y + \Phi_\theta([\,\tilde{h}_i^y\,,\,\|\tilde{y}_i\|\,]), \tag{11}$$

where $[\cdot, \cdot]$ denotes concatenation and $\tilde{x}_i$, and $\tilde{y}_i$ are normalized invariant features obtained through normalization layers (see Appendix B.1). Note that the norm of a coordinate set is defined component-wise, i.e., $\|x_i\| = [\|x_{i,1}\|, \ldots, \|x_{i,d}\|]$. This design choice allows the invariant features to propagate information to x and y independently. The equivariant feature update is also performed independently on x_i and y_i as follows:

$$x_i^{\mathrm{ff}} = x_i + W_g\Big(\sum_{j=1}^{d}(W_f\,\tilde{x}_j) \otimes \Psi_\theta(\tilde{h}_i^x)\Big) \quad y_i^{\mathrm{ff}} = y_i + W_g\Big(\sum_{j=1}^{d}(W_f\,\tilde{y}_j) \otimes \Psi_\theta(\tilde{h}_i^y)\Big), \tag{12}$$

where W_f and W_g are two linear projections (see Appendix B.1).

These features x^{ff}, y^{ff}, $h^{x\,\mathrm{ff}}$ and $h^{y\,\mathrm{ff}}$ are used as input to the graph attention layer in combination with the edge features e^x and e^y, where in the first FlexiFlow layer $e = e^x = e^y$. Similar to the feed-forward layer, the graph attention layer shares the same normalization layers and MLPs for both invariant and equivariant features. The key difference from SemlaFlow, however, is that we now combine the features corresponding to x and y. The messages are computed as follows:

$$x_p = \tilde{x}_i^{\mathrm{ff}} \cdot \tilde{x}_j^{\mathrm{ff}\,T}, \quad y_p = \tilde{y}_i^{\mathrm{ff}} \cdot \tilde{y}_j^{\mathrm{ff}\,T}, \quad h_p^x = [W_h\tilde{h}_i^{x\,\mathrm{ff}} \parallel W_j\tilde{h}_j^{x\,\mathrm{ff}}], \quad h_p^y = [W_h\tilde{h}_i^{y\,\mathrm{ff}} \parallel W_h\tilde{h}_j^{y\,\mathrm{ff}}] \tag{13}$$
$$\omega_p^x = [h_p^x, x_p, e^x], \quad \omega_p^y = [h_p^x \cdot h_p^y, x_p, y_p, e^x \cdot e^y]$$

where W_h represents a linear projection and the final messages are computed using two separate MLPs on ω_p^x and ω_p^y, as they have different input dimensions. These messages are then used to update x^{ff}, y^{ff}, $h^{x\,\mathrm{ff}}$, $h^{y\,\mathrm{ff}}$, e^x and e^y (see Appendix B.1 for complete details). Note that the scalar product, e.g. when calculating x_p, is to be understood component-wise, i.e., $x_i \cdot x_j^T = [x_{i,1}x_{j,1}^T, \ldots x_{i,d}x_{j,d}^T]$. Sharing the information between x and y as described in Equation 13 allows the FlexiModel to retain the equivariance on the coordinates.

Theorem 4.1 (Equivariance). *The FlexiFlow model is equivariant with respect to the coordinates x and y. Let $\tilde{x}$ and $\tilde{y}$ be the normalized coordinate sets as defined in Equation 13, and let $R_x \in SO(3)$ and $R_y \in SO(3)$ be rotation matrices. The only exchange of information between x and y occurs in Equation 13. Applying any rotations R_x and R_y to $\tilde{x}$ and $\tilde{y}$ does not affect the scalar product, since*

$$R_x \tilde{x}_i \tilde{x}_j^T R_x^T = \tilde{x}_i R_x R_x^T \tilde{x}_j^T = \tilde{x}_i \tilde{x}_j^T .$$

The same argument applies to $\tilde{y}$. Since SemlaFlow (Irwin et al., 2025) is equivariant with respect to the coordinates, the remainder of the proof follows directly.

The *features refinement layer* applies a feed forward layer on x^{ff}, y^{ff}, $h^{x\,\text{ff}}$, $h^{y\,\text{ff}}$ followed by an edge features aggregator on e^x, e^y. Lastly, three shared MLPs are used to predict the logits for atoms, charges types from $h^{x\,\text{ff}}$ and bonds types from e^x. In Appendix B.1 we provide a detailed mathematical formulation for each component of the architecture.

Loss. Following Equation 9, the model is trained to minimize the coupled conditional flow-matching loss, which now additionally incorporates the categorical loss component. We reformulate the loss over molecular graphs as a composition of different terms: $\mathcal{L}_{x,y}$ coordinates loss for x and y computed as the mean squared error between the predicted and target, $\mathcal{L}_a$, $\mathcal{L}_c$, $\mathcal{L}_e$ computed as the negative log likelihood between the predicted and target atoms, charges and bonds types respectively, $\mathcal{L}_{\text{reg}}$ as regularization loss which aims to enforce the bonds lengths consistency on the predicted molecular graphs and align the categorical features of y on x. The full loss is thus defined as: $\mathcal{L}_{FlexiFlow} = \mathcal{L}_{x,y} + \mathcal{L}_a + \mathcal{L}_c + \mathcal{L}_e + \mathcal{L}_{\text{reg}}$. We provide an extensive description of each loss component as part of the Appendix B.4.

Inference. Algorithm 1 describes the inference scheme. Let $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ denote the sets of atom types, bond types, and charge types, respectively. We denote with $(a, b, c) \sim \text{Cat}(1/|\mathcal{A}|) \cdot \text{Cat}(1/|\mathcal{B}|) \cdot \text{Cat}(1/|\mathcal{C}|)$, the sampling from three independent categorical distributions with uniform probability over each set. We denote with f_θ a trained FlexiFlow model parametrized by θ. Δt represents the time step. The function CatUpdate is used to perform the update features at inference time, and applies the strategy developed by Campbell et al. (2024). Since x_t and y_t are sampled independently, our flow decomposition permits two modes of sampling: (1) drawing both x_t and y_t from $\mathcal{N}(0, \text{I})$, or (2) fixing x_t to a specific noise configuration while sampling $y_t \sim \mathcal{N}(0, \text{I})$. This enables us to generate either different molecular graphs with their conformations or the same molecular graph with multiple conformations.

Algorithm 1 Inference scheme

Input: Δt
$x_t \sim \mathcal{N}(0, \text{I})$, $y_t \sim \mathcal{N}(0, \text{I})$, $t \leftarrow 0$
$a_t, b_t, c_t \sim \text{Cat}(1/|\mathcal{A}|) \cdot \text{Cat}(1/|\mathcal{B}|) \cdot \text{Cat}(1/|\mathcal{C}|)$
while t < 1 **do:**
  $(\hat{x}_1, \hat{y}_1, \hat{a}_1, \hat{b}_1, \hat{c}_1) \leftarrow f_\theta(x_t, y_t, a_t, b_t, c_t)$
  $x_t \leftarrow x_t + \Delta t\,(\hat{x}_1 - x_t)/(1-t)$
  $y_t \leftarrow y_t + \Delta t\,(\hat{y}_1 - y_t)/(1-t)$
  $a_t \leftarrow \text{CatUpdate}(\hat{a}_1, a_t, t, \Delta t)$
  $b_t \leftarrow \text{CatUpdate}(\hat{b}_1, b_t, t, \Delta t)$
  $c_t \leftarrow \text{CatUpdate}(\hat{c}_1, c_t, t, \Delta t)$
  $t \leftarrow t + \Delta t$
end while

5 Experiments

We compare FlexiFlow against the following state-of-the-art 3D generative models EDM (Hoogeboom et al., 2022), GCDM (Morehead & Cheng, 2024), GFMDiff (Xu et al., 2024), EquiFM (Song et al., 2023), GeomLDM (Xu et al., 2023), GeomBFM (Song et al., 2024), MUDiff (Hua et al., 2023), FlowMol (Dunn & Koes, 2024), MiDi (Vignac et al., 2023), EQGAT-diff (Le et al., 2024), Tabasco (Vonessen et al., 2025), FlowMol3 (Dunn & Koes, 2025) and SemlaFlow (Irwin et al., 2025). On the QM9 and GEOM Drugs benchmarks, FlexiFlow achieves equal or improved scores for atomic and molecular stability (i.e., stable electron configurations), validity (compliance with basic chemical rules), novelty (fraction of generated molecules absent from the training set), and uniqueness (fraction of distinct molecular graphs). Unlike all other competitors, FlexiFlow is able to generate a set of conformations together with the molecule. To assess the diversity of these conformers, we compute

$$D(S) = \frac{1}{|S|} \sum_{x \in S} \min_{\substack{y \in S \\ y \neq x}} \text{RMSD}(x, y), \tag{14}$$

that take for each conformer x its minimal RMSD (after optimal alignment) to any other conformer in S and average those nearest-neighbor RMSDs over the whole set. Then we perform the same operation $D(\mathcal{S}^*)$ on the energy minimized set $\mathcal{S}^*$ (with MMFF94). $D(\mathcal{S})$ aims to captures conformer diversity and whether minimization collapses them into shared minima. To further assess the extent to which the generated conformers cover the low-energy space, we employ the computationally expensive state-of-the-art physics-based method CREST (Pracht et al., 2024). In this setting, we report Absolute Mean RMSD (AMR) and Coverage (Cov) with respect to low-energy references (see Appendix B.8). Both metrics are defined in terms of precision (P) and recall (R): AMR-R computes the average RMSD from each CREST-generated conformers to its closest generated conformer, while AMR-P computes the average RMSD from each generated conformer to its closest CREST-generated conformers. To compute the Cov metrics, we use a threshold $\delta = \{0.0, \ldots, 2.5\}$ Å with step 0.125 Å and report Cov-R and Cov-P. Cov-R(δ) is the fraction of CREST-generated conformers that have at least one generated conformer within RMSD δ, Cov-P(δ) is the fraction of generated

Table 1: The table shows the results on QM9, where the methods are grouped into those which infer bonds from coordinates (top) and those which generate bonds directly (bottom). Methods marked with * publish only results over molecules that are both unique and valid. Tabasco authors reported 34% Novelty (no Uniqueness reported); we achieve 90%. See Table 3. NFE refers to the number of inference steps.

Model	Atom Stab ↑	Mol Stab ↑	Valid ↑	Unique ↑	NFE
EDM	98.7	82.0	91.9	98.9*	1000
GCDM	98.7	85.7	94.8	98.4*	1000
GFMDiff	98.9	87.7	96.3	98.8*	500
EquiFM	98.9	88.3	94.7	98.7*	210
GeoLDM	98.9	89.4	93.8	98.8	1000
MUDiff	98.8	89.9	95.3	99.1	1000
GeoBFN	99.3	93.3	96.9	95.4	2000
FlowMol	99.7	96.2	97.3	–	100
MiDi	99.8	97.5	97.9	97.6	500
Tabasco	–	–	100.0	–	100
EQGAT-diff	**99.9**$_{\pm 0.0}$	98.7$_{\pm 0.18}$	99.0$_{\pm 0.16}$	**100.0**$_{\pm 0.0}$	500
SemlaFlow	**99.9**$_{\pm 0.0}$	**99.7**$_{\pm 0.03}$	**99.4**$_{\pm 0.03}$	95.4$_{\pm 0.12}$	100
FLEXIFLOW	**100.0**$_{\pm 0.0}$	**99.9**$_{\pm 0.01}$	**99.9**$_{\pm 0.01}$	**100.0**$_{\pm 0.00}$	100

Table 2: The table shows the results on GEOM Drugs, where the methods are grouped into those which infer bonds from coordinates (top) and those which generate bonds directly (bottom). Methods marked with * uses the estimates for the molecule stability provided by Irwin et al. (2025) as the papers do not report this metric. Tabasco results show the best model with guidance. NFE refers to the number of inference steps.

Model	Atom Stab ↑	Mol Stab ↑	Valid ↑	Unique ↑	Novel ↑	NFE
EDM	81.3	0.0*	–	–	–	1000
GCDM	89.0	5.2	–	–	–	1000
MUDiff	84.0	60.9	98.9	–	–	1000
GFMDiff	86.5	3.9	–	–	–	500
EquiFM	84.1	0.0*	98.9	–	–	–
GeoBFN	86.2	0.0*	91.7	–	–	2000
GeoLDM	98.9	61.5*	**99.3**	–	–	1000
FlowMol	99.0	67.5	51.2	–	–	100
MiDi	**99.8**	91.6	77.8	**100.0**	**100.0**	500
EQGAT-diff	**99.8**$_{\pm 0.0}$	93.4$_{\pm 0.21}$	94.6$_{\pm 0.24}$	**100.0**$_{\pm 0.0}$	99.9$_{\pm 0.07}$	500
Flowmol3	–	–	**99.9**$_{\pm 0.10}$	–	–	250
Tabasco	–	–	**97.0**$_{\pm 0.10}$	–	92.0	100
SemlaFlow	**99.8**$_{\pm 0.0}$	**97.3**$_{\pm 0.08}$	93.9$_{\pm 0.19}$	**100.0**$_{\pm 0.0}$	99.6$_{\pm 0.03}$	100
FLEXIFLOW	**99.9**$_{\pm 0.0}$	**99.9**$_{\pm 0.01}$	92.0$_{\pm 0.10}$	**100.0**$_{\pm 0.0}$	99.9$_{\pm 0.01}$	100

conformers that have at least one CREST-generated conformers within δ. Finally, to illustrate its potential, we demonstrate ligand generation conditioned on a PDBBind protein pocket (Wang et al., 2005) and present qualitative MNIST results in Appendix E.

5.1 GENERATION QM9 & GEOM DRUGS

Training set-up. We use QM9 (Ramakrishnan et al., 2014) and GEOM Drugs (Axelrod & Gómez-Bombarelli, 2022) datasets to train our model using the training split for both from (Vignac et al., 2023; Le et al., 2024; Irwin et al., 2025) training the model with the hydrogens. Since, QM9 lacks multiple conformers, 20 RDKit conformers are generated per molecule, while GEOM Drugs already provides on average 23 ± 10 semi-empirical DFT conformers per molecule. See Appendix B.6 for further details on the data processing for both datasets. We trained the model on QM9 for 40 epochs and on GEOM Drugs for 4 epochs, refer to Appendix B.5, for further details on the training.

We compare FlexiFlow with the most recent state-of-the-art models for molecular generation on QM9 and GEOM Drugs. To compare our model we sample $30k$ molecules from trained models. By reporting results on both datasets in Tables 1 and 2, it is noticeable that FlexiFlow achieves state-of-the-art results on both datasets, performing comparably in terms of uniqueness and validity with strong scores on novelty, atom and molecular stability on GEOM Drugs and QM9. Again, we emphasize that FlexiFlow can generate an arbitrary number of conformations for a given molecule, unlike all other methods.

5.2 GENERATED CONFORMERS QM9 & GEOM DRUGS

To evaluate the extent to which the generated conformers explore different energy minima, we generate 20 molecular structures for molecules containing 25, 30, 35, 40, 45, and 50 atoms (Figure 3). For each set S, we compute the metric defined in Equation 14, selecting before energy minimization the closest conformers in the set in terms of RMSD for each generated 3D molecular graph. We observe that the minimum pairwise distance between conformers in S consistently increases with the number of atoms also after minimization. This indicates that, even in the worst case scenario when selecting the closest conformers on the generated set, the states tend to fall into different energy minima not collapsing in the same state (see Appendix B.9 for details).

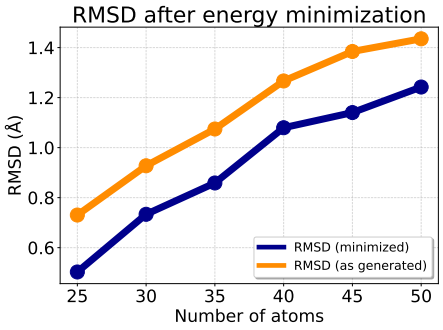

Figure 3: Figure show RMSD previous and after energy minimization.

We benchmark FlexiFlow against RDKit ETKDG, Adjoint Sampling (AS) (Havens et al., 2025), and CREST: for 100 molecules we sample 300 conformers each with FlexiFlow (100 NFE), use

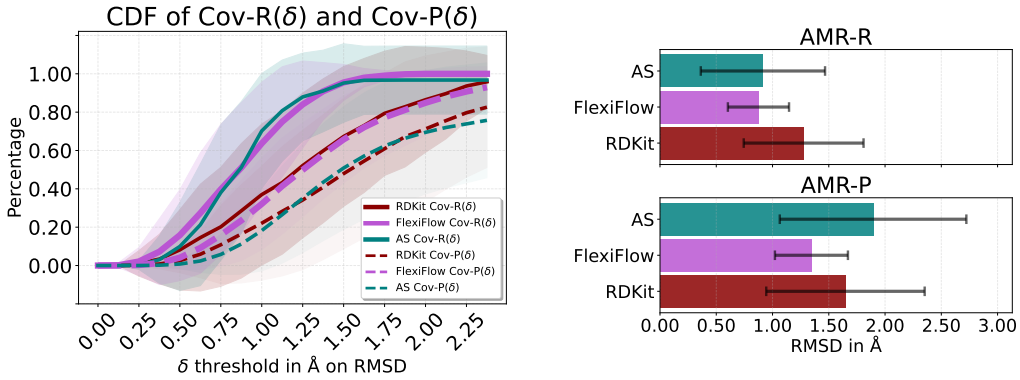

Figure 4: Figures show Cov (left) and AMR (right) precision–recall for GEOM Drugs comparing CREST-generated conformer with RDKit, FlexiFlow and Adjoint Sampling (AS); see Appendix B.8 for metric details.

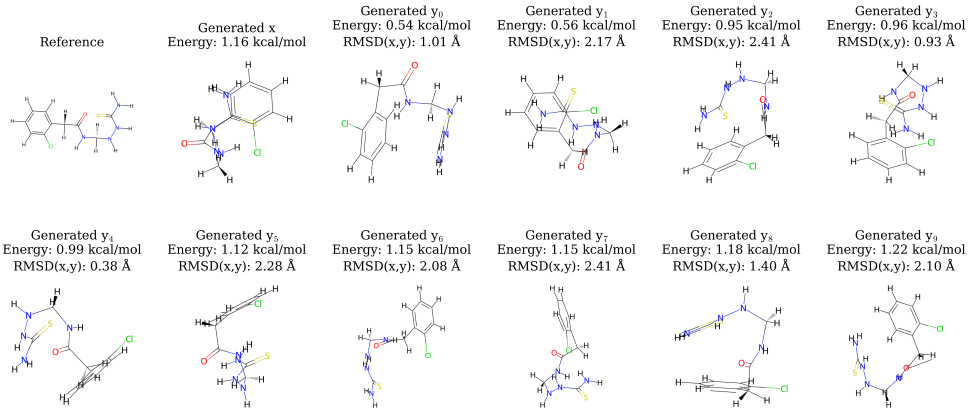

Figure 5: Figure shows in the top right the graph of the generated molecule in 2D for illustration purposes. The rest of the grid shows the reference graph with its conformer (x), followed by a set of generated conformers (y) for the same generated molecular graph. The y conformers are aligned to the x conformer for better visualization and we report energy and the RMSD value between each generated conformer (y) and the reference one (x).

the lowest-energy FlexiFlow conformer to seed CREST (Pracht et al., 2024) and ETKDG (Riniker & Landrum, 2015) reference ensembles, and run AS with the same NFE to generate its conformers. In Figure 4, we report Coverage (Cov) and Average Minimum RMSD (AMR) between the generated conformers and the optimal energy minima conformers obtained with CREST. These are reported in terms of recall (R) and precision (P) (see Appendix B.8 for details). Figure 4 on the left shows the cumulative distributions for Cov-R(δ) (solid lines) and Cov-P(δ) (dashed lines) for RDKit, Adjoint Sampling (AS), and FlexiFlow. For both Cov-R(δ) and Cov-P(δ), the earliest the curves reach 1 the better. As shown, FlexiFlow performs at least as well as AS on Cov-R(δ), and it outperforms both AS and RDKit on Cov-P(δ). Those results suggest that AS tends to produce conformers clustered around fewer energy minima, whereas FlexiFlow explores the conformational space more broadly, covering more distinct minima, as jointly indicated by Cov-P(δ) and Cov-R(δ). Figure 4 on the right shows AMR-R and AMR-P, where lower values indicate better performance. Here, FlexiFlow outperforms both AS and RDKit on average. Finally, Figure 5 illustrates an example of a generated molecule where its y conformers are in a different state compared to the x state. We report the energy of the conformers along with the RMSD with respect to the reference state x. More detailed results are reported in Appendix D.1, D.2, and D.3.

Ablation comparing different model settings: Results in Table 3 show the performances of FlexiFlow with different model sizes, epochs and number of inference steps (NFE) on QM9, by generating 100 molecules with 300 conformers each (30k molecules per run). We can see that increasing the model size from small (S) to large (L) leads to improved performances across all metrics, as well as increasing the number of training epochs.

Ms-Ep-NFE	Valid ↑	Novel ↑	Strain x ↓	Strain y ↓
S-10-50	$96.9_{\pm 0.16}$	$99.9_{\pm 0.01}$	$6.13_{\pm 0.02}$	$1.51_{\pm 0.02}$
S-10-100	$95.8_{\pm 0.15}$	$96.4_{\pm 0.07}$	$5.63_{\pm 0.01}$	$1.43_{\pm 0.02}$
S-10-500	$94.9_{\pm 0.17}$	$95.7_{\pm 0.08}$	$5.11_{\pm 0.07}$	$1.53_{\pm 0.05}$
S-15-50	$97.8_{\pm 0.10}$	$99.9_{\pm 0.01}$	$5.95_{\pm 0.04}$	$2.41_{\pm 0.02}$
S-15-100	$95.9_{\pm 0.13}$	$100.0_{\pm 0.00}$	$5.40_{\pm 0.01}$	$2.05_{\pm 0.03}$
S-15-500	$95.8_{\pm 0.13}$	$100.0_{\pm 0.00}$	$4.98_{\pm 0.02}$	$2.39_{\pm 0.06}$
M-10-50	$97.0_{\pm 0.12}$	$93.9_{\pm 0.06}$	$4.03_{\pm 0.01}$	$2.80_{\pm 0.06}$
M-10-100	$97.2_{\pm 0.15}$	$99.1_{\pm 0.04}$	$3.54_{\pm 0.04}$	$1.93_{\pm 0.02}$
M-10-500	$98.3_{\pm 0.13}$	$97.1_{\pm 0.06}$	$3.34_{\pm 0.06}$	$1.03_{\pm 0.02}$
M-15-50	$98.0_{\pm 0.13}$	$91.9_{\pm 0.03}$	$3.54_{\pm 0.04}$	$1.75_{\pm 0.02}$
M-15-100	$100.0_{\pm 0.00}$	$95.8_{\pm 0.07}$	$3.18_{\pm 0.03}$	$1.31_{\pm 0.04}$
M-15-500	$100.0_{\pm 0.00}$	$96.7_{\pm 0.07}$	$3.02_{\pm 0.01}$	$1.82_{\pm 0.03}$
L-10-100	$99.9_{\pm 0.01}$	$91.8_{\pm 0.03}$	$2.12_{\pm 0.03}$	$0.88_{\pm 0.04}$
L-20-100	$99.9_{\pm 0.01}$	$89.9_{\pm 0.01}$	$1.11_{\pm 0.05}$	$0.61_{\pm 0.02}$
L-30-100	$99.9_{\pm 0.01}$	$90.7_{\pm 0.01}$	$0.69_{\pm 0.01}$	$0.48_{\pm 0.02}$
L-40-100	$99.9_{\pm 0.01}$	$90.0_{\pm 0.01}$	$0.51_{\pm 0.01}$	$0.24_{\pm 0.01}$

Table 3: The table shows additional metrics on QM9. Model size (Ms), S (17.2M), M (24.7M) and L (37.7M) params (see Appendix B.5.1 for details), Epochs (Ep) and number of inference steps (NFE). For all runs reported the Uniqueness is $100.0_{\pm 0.0}$. The strain of the top-10 y conformers for each x generated molecular graph is reported.

Increasing the NFE does not lead to improved performances, this is observable in the strain energies of the x and y conformers, which do not improve significantly when increasing the NFE from 100 to 500. This suggests that the model is already able to generate high-quality samples with a relatively low number of inference steps, which is beneficial for efficiency.

5.3 Protein Conditioning

We provide some additional experiments on targeted generation with protein conditioning, training FlexiFlow on a subset of 8k protein-ligand complexes training samples from PDBBind (Wang et al., 2004) with ligand QED $>$ 0.5 and tested using the Corso et al. (2023) splits. During training, for 300 epochs we interleave GEOM Drugs batches with PDBBind protein–ligand complexes batches to support multiple conformers; since PDBBind lacks multiple ligands per protein, we reuse the same ligand conformation x for each ligand conformation y. By sampling 120 molecules for each testset protein, our method finds better Vina scores (in kcal/mol) on y conformers in 2,124 cases out of 13,440. Figure 7 illustrates a testset example on protein 6jb0, where one of the y conformers achieves a better Vina scores than x. Figure 6 shows the Vina score (Eberhardt et al., 2021) distribution of top-1 and top-10 ligands sampled per test set protein across the 13,440 unique x–y ligand pairs. FlexiFlow achieves an average Vina score of –7.4 kcal/mol for the top-1 prediction comparable with the target test set data distribution. In a similar experimental setting, we trained a model variant that excluded the molecular flexibility information, by omitting the GEOM Drugs dataset during training. As reported in Appendix D.6, the model trained with GEOM Drugs, exhibited improved strained-energy profiles for the generated top-k y ligand conformations. This suggests that incorporating molecular flexibility, can enhance conformation generation performance on other downstream task, even when specific conformation datasets are missing.

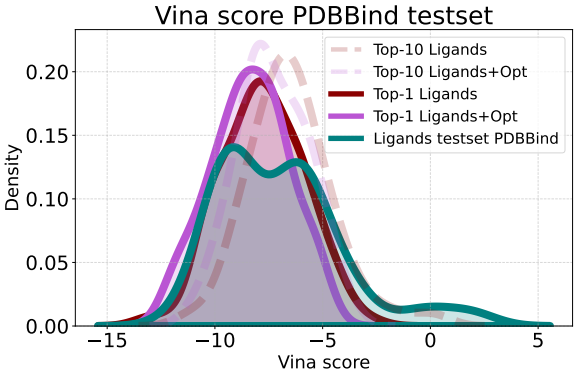

Figure 6: Vina score distribution for the generated ligands on PDBBind testset proteins. Opt reports results after slight ligand adjustments using MMFF94 conditioned to the protein.

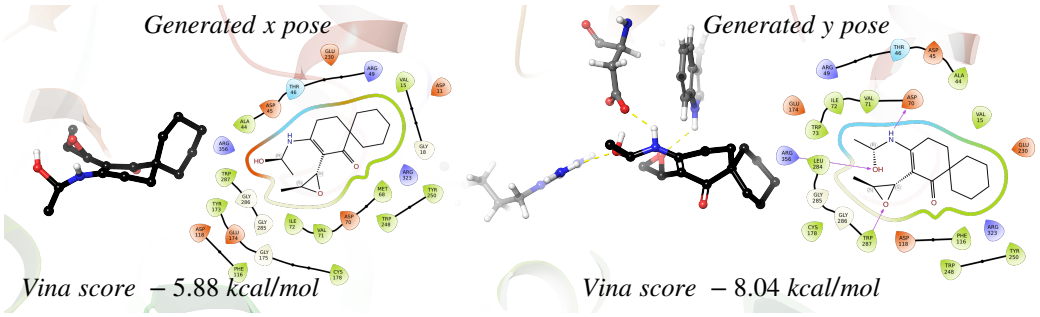

Figure 7: Figure reports x and y generated poses for a generated ligand on protein 6jb0 (QED=0.78).

6 Conclusion

We introduced FlexiFlow, a novel approach that leverages conditional independence to decompose the flow for the simultaneous generation of 3D molecular graphs and conformer sets. The FlexiFlow architecture preserves equivariant properties for coordinates and invariant properties for atom types. We demonstrated its effectiveness in both de novo molecular generation and conformer generation, achieving results comparable to or better than existing methods while producing diverse sets of high-quality conformations. In addition, we extended the approach to protein-conditioned ligand generation. Immediate future work will focus on extending the model to support protein dynamics with minimal modifications, though this will require careful data preparation and extensive training.

REFERENCES

Namrata Anand and Tudor Achim. Protein structure and sequence generation with equivariant denoising diffusion probabilistic models. *CoRR*, abs/2205.15019, 2022. doi: 10.48550/ARXIV.2205.15019. URL https://doi.org/10.48550/arXiv.2205.15019.

Simon Axelrod and Rafael Gómez-Bombarelli. Geom, energy-annotated molecular conformations for property prediction and molecular generation. *Scientific Data*, 9(1), April 2022. ISSN 2052-4463. doi: 10.1038/s41597-022-01288-4. URL http://dx.doi.org/10.1038/s41597-022-01288-4.

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016. URL http://arxiv.org/abs/1607.06450.

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, February 2019. ISSN 1549-9626. doi: 10.1021/acs.jctc.8b01176. URL http://dx.doi.org/10.1021/acs.jctc.8b01176.

Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi S. Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=kQwSbv0BR4.

Gabriele Corso, Hannes Stärk, Bowen Jing, Regina Barzilay, and Tommi S. Jaakkola. Diffdock: Diffusion steps, twists, and turns for molecular docking. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=kKF8_K-mBbS.

Juan Viguera Diez, Mathias Jacob Schreiner, Ola Engkvist, and Simon Olsson. Boltzmann priors for implicit transfer operators. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=pRCOZllZdT.

Carles Domingo-Enrich, Michal Drozdzal, Brian Karrer, and Ricky T. Q. Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=xQBRrtQM8u.

Zibin Dong, Yifu Yuan, Jianye Hao, Fei Ni, Yao Mu, Yan Zheng, Yujing Hu, Tangjie Lv, Changjie Fan, and Zhipeng Hu. Aligndiff: Aligning diverse human preferences via behavior-customisable diffusion model. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=bxfKIYfHyx.

Ian Dunn and David Ryan Koes. Mixed continuous and categorical flow matching for 3d de novo molecule generation. *CoRR*, abs/2404.19739, 2024. doi: 10.48550/ARXIV.2404.19739. URL https://doi.org/10.48550/arXiv.2404.19739.

Ian Dunn and David Ryan Koes. Flowmol3: Flow matching for 3d de novo small-molecule generation. *CoRR*, abs/2508.12629, 2025. doi: 10.48550/ARXIV.2508.12629. URL https://doi.org/10.48550/arXiv.2508.12629.

Jerome Eberhardt, Diogo Santos-Martins, Andreas F. Tillack, and Stefano Forli. Autodock vina 1.2.0: New docking methods, expanded force field, and python bindings. *Journal of Chemical Information and Modeling*, 61(8):3891–3898, July 2021. ISSN 1549-960X. doi: 10.1021/acs.jcim.1c00203. URL http://dx.doi.org/10.1021/acs.jcim.1c00203.

Octavian Ganea, Lagnajit Pattanaik, Connor W. Coley, Regina Barzilay, Klavs F. Jensen, William H. Green Jr., and Tommi S. Jaakkola. Geomol: Torsional geometric generation of molecular 3d conformer ensembles. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 13757–13769, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/725215ed82ab6306919b485b81ff9615-Abstract.html.

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T. Q. Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/f0d629a734b56a642701bba7bc8bb3ed-Abstract-Conference.html.

Thomas A. Halgren. Merck molecular force field. i. basis, form, scope, parameterization, and performance of mmff94. *Journal of Computational Chemistry*, 17(5–6):490–519, April 1996. ISSN 0192-8651. doi: 10.1002/(sici)1096-987x(199604)17:5/6⟨490::aid-jcc1⟩3.0.co;2-p. URL http://dx.doi.org/10.1002/(SICI)1096-987X(199604)17:5/6<490::AID-JCC1>3.0.CO;2-P.

Aaron J. Havens, Benjamin Kurt Miller, Bing Yan, Carles Domingo-Enrich, Anuroop Sriram, Brandon M. Wood, Daniel Levine, Bin Hu, Brandon Amos, Brian Karrer, Xiang Fu, Guan-Horng Liu, and Ricky T. Q. Chen. Adjoint sampling: Highly scalable diffusion samplers via adjoint matching. *CoRR*, abs/2504.11713, 2025. doi: 10.48550/ARXIV.2504.11713. URL https://doi.org/10.48550/arXiv.2504.11713.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

Emiel Hoogeboom, Victor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 8867–8887. PMLR, 2022. URL https://proceedings.mlr.press/v162/hoogeboom22a.html.

Chenqing Hua, Sitao Luan, Minkai Xu, Zhitao Ying, Jie Fu, Stefano Ermon, and Doina Precup. Mudiff: Unified diffusion for complete molecule generation. In Soledad Villar and Benjamin Chamberlain (eds.), *Learning on Graphs Conference, 27-30 November 2023, Virtual Event*, volume 231 of *Proceedings of Machine Learning Research*, pp. 33. PMLR, 2023. URL https://proceedings.mlr.press/v231/hua24a.html.

John Ingraham, Vikas K. Garg, Regina Barzilay, and Tommi S. Jaakkola. Generative models for graph-based protein design. In *Deep Generative Models for Highly Structured Data, ICLR 2019 Workshop, New Orleans, Louisiana, United States, May 6, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=SJgxrLLKOE.

Ross Irwin, Alessandro Tibo, Jon Paul Janet, and Simon Olsson. Semlaflow - efficient 3d molecular generation with latent attention and equivariant flow matching. In Yingzhen Li, Stephan Mandt, Shipra Agrawal, and Mohammad Emtiyaz Khan (eds.), *International Conference on Artificial Intelligence and Statistics, AISTATS 2025, Mai Khao, Thailand, 3-5 May 2025*, volume 258 of *Proceedings of Machine Learning Research*, pp. 3772–3780. PMLR, 2025. URL https://proceedings.mlr.press/v258/irwin25a.html.

Bowen Jing, Gabriele Corso, Jeffrey Chang, Regina Barzilay, and Tommi S. Jaakkola. Torsional diffusion for molecular conformer generation. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/994545b2308bbbbc97e3e687ea9e464f-Abstract-Conference.html.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, July 2021. ISSN 1476-4687. doi: 10.1038/s41586-021-03819-2. URL http://dx.doi.org/10.1038/s41586-021-03819-2.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

Tuan Le, Julian Cremer, Frank Noé, Djork-Arné Clevert, and Kristof T. Schütt. Navigating the design space of equivariant diffusion-based generative models for de novo 3d molecule generation. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=kzGuiRXZrQ.

Andrew R. Leach, Valerie J. Gillet, Richard A. Lewis, and Robin Taylor. Three-dimensional pharmacophore methods in drug discovery. *Journal of Medicinal Chemistry*, 53(2):539–558, October 2009. ISSN 1520-4804. doi: 10.1021/jm900817u. URL http://dx.doi.org/10.1021/jm900817u.

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, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky T. Q. Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *CoRR*, abs/2412.06264, 2024. doi: 10.48550/ARXIV.2412.06264. URL https://doi.org/10.48550/arXiv.2412.06264.

Guan-Horng Liu, Jaemoo Choi, Yongxin Chen, Benjamin Kurt Miller, and Ricky T. Q. Chen. Adjoint schrödinger bridge sampler. *CoRR*, abs/2506.22565, 2025. doi: 10.48550/ARXIV.2506.22565. URL https://doi.org/10.48550/arXiv.2506.22565.

Alex Morehead and Jianlin Cheng. Geometry-complete diffusion for 3d molecule generation and optimization. *Communications Chemistry*, 7(1), July 2024. ISSN 2399-3669. doi: 10.1038/s42004-024-01233-z. URL http://dx.doi.org/10.1038/s42004-024-01233-z.

Frank Noé, Simon Olsson, Jonas Köhler, and Hao Wu. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365(6457):eaaw1147, 2019. doi: 10.1126/science.aaw1147. URL https://www.science.org/doi/abs/10.1126/science.aaw1147.

Xingang Peng, Shitong Luo, Jiaqi Guan, Qi Xie, Jian Peng, and Jianzhu Ma. Pocket2mol: Efficient molecular sampling based on 3d protein pockets. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 17644–17655. PMLR, 2022. URL https://proceedings.mlr.press/v162/peng22b.html.

Philipp Pracht, Fabian Bohle, and Stefan Grimme. Automated exploration of the low-energy chemical space with fast quantum chemical methods. *Physical Chemistry Chemical Physics*, 22(14): 7169–7192, 2020. ISSN 1463-9084. doi: 10.1039/c9cp06869d. URL http://dx.doi.org/10.1039/C9CP06869D.

Philipp Pracht, Stefan Grimme, Christoph Bannwarth, Fabian Bohle, Sebastian Ehlert, Gereon Feldmann, Johannes Gorges, Marcel Müller, Tim Neudecker, Christoph Plett, Sebastian Spicher, Pit Steinbach, Patryk A. Wesołowski, and Felix Zeller. Crest—a program for the exploration of low-energy molecular chemical space. *The Journal of Chemical Physics*, 160(11), March 2024. ISSN 1089-7690. doi: 10.1063/5.0197592. URL http://dx.doi.org/10.1063/5.0197592.

Raghunathan Ramakrishnan, Pavlo O. Dral, Matthias Rupp, and O. Anatole von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1(1), August 2014. ISSN 2052-4463. doi: 10.1038/sdata.2014.22. URL http://dx.doi.org/10.1038/sdata.2014.22.

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, November 2015. ISSN 1549-960X. doi: 10.1021/acs.jcim.5b00654. URL http://dx.doi.org/10.1021/acs.jcim.5b00654.

Arne Schneuing, Charles Harris, Yuanqi Du, Kieran Didi, Arian Jamasb, Ilia Igashov, Weitao Du, Carla Gomes, Tom L. Blundell, Pietro Lio, Max Welling, Michael Bronstein, and Bruno Correia. Structure-based drug design with equivariant diffusion models. *Nature Computational Science*, 4(12):899–909, December 2024. ISSN 2662-8457. doi: 10.1038/s43588-024-00737-x. URL http://dx.doi.org/10.1038/s43588-024-00737-x.

Marta Skreta, Lazar Atanackovic, Joey Bose, Alexander Tong, and Kirill Neklyudov. The superposition of diffusion models using the itô density estimator. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=2o58Mbqkd2.

Yuxuan Song, Jingjing Gong, Minkai Xu, Ziyao Cao, Yanyan Lan, Stefano Ermon, Hao Zhou, and Wei-Ying Ma. Equivariant flow matching with hybrid probability transport for 3d molecule generation. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/01d64478381c33e29ed611f1719f5a37-Abstract-Conference.html.

Yuxuan Song, Jingjing Gong, Hao Zhou, Mingyue Zheng, Jingjing Liu, and Wei-Ying Ma. Unified generative modeling of 3d molecules with bayesian flow networks. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=NSVtmmzeRB.

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. *Trans. Mach. Learn. Res.*, 2024, 2024. URL https://openreview.net/forum?id=CD9Snc73AW.

Clément Vignac, Nagham Osman, Laura Toni, and Pascal Frossard. Midi: Mixed graph and 3d denoising diffusion for molecule generation. In Danai Koutra, Claudia Plant, Manuel Gomez Rodriguez, Elena Baralis, and Francesco Bonchi (eds.), *Machine Learning and Knowledge Discovery in Databases: Research Track - European Conference, ECML PKDD 2023, Turin, Italy, September 18-22, 2023, Proceedings, Part II*, volume 14170 of *Lecture Notes in Computer Science*, pp. 560–576. Springer, 2023. doi: 10.1007/978-3-031-43415-0_33. URL https://doi.org/10.1007/978-3-031-43415-0_33.

Carlos Vonessen, Charles Harris, Miruna T. Cretu, and Pietro Liò. TABASCO: A fast, simplified model for molecular generation with improved physical quality. *CoRR*, abs/2507.00899, 2025. doi: 10.48550/ARXIV.2507.00899. URL https://doi.org/10.48550/arXiv.2507.00899.

Renxiao Wang, Xueliang Fang, Yipin Lu, and Shaomeng Wang. The pdbbind database: Collection of binding affinities for protein–ligand complexes with known three-dimensional structures. *Journal of Medicinal Chemistry*, 47(12):2977–2980, May 2004. ISSN 0022-2623. doi: 10.1021/jm030580l. URL https://doi.org/10.1021/jm030580l.

Renxiao Wang, Xueliang Fang, Yipin Lu, Chao-Yie Yang, and Shaomeng Wang. The pdbbind database:methodologies and updates. *Journal of Medicinal Chemistry*, 48(12):4111–4119, 2005. doi: 10.1021/jm048957q. URL https://doi.org/10.1021/jm048957q. PMID: 15943484.

Joseph L. Watson, David Juergens, Nathaniel R. Bennett, Brian L. Trippe, Jason Yim, Helen E. Eisenach, Woody Ahern, Andrew J. Borst, Robert J. Ragotte, Lukas F. Milles, Basile I. M. Wicky, Nikita Hanikel, Samuel J. Pellock, Alexis Courbet, William Sheffler, Jue Wang, Preetham Venkatesh, Isaac Sappington, Susana Vázquez Torres, Anna Lauko, Valentin De Bortoli, Emile Mathieu, Sergey Ovchinnikov, Regina Barzilay, Tommi S. Jaakkola, Frank DiMaio, Minkyung Baek, and David Baker. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976):1089–1100, July 2023. ISSN 1476-4687. doi: 10.1038/s41586-023-06415-8. URL http://dx.doi.org/10.1038/s41586-023-06415-8.

Jonas Wildberger, Maximilian Dax, Simon Buchholz, Stephen R. Green, Jakob H. Macke, and Bernhard Schölkopf. Flow matching for scalable simulation-based inference. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/3663ae53ec078860bb0b9c6606e092a0-Abstract-Conference.html.

Can Xu, Haosen Wang, Weigang Wang, Pengfei Zheng, and Hongyang Chen. Geometric-facilitated denoising diffusion model for 3d molecule generation. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (eds.), *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pp. 338–346. AAAI Press, 2024. doi: 10.1609/AAAI.V38I1.27787. URL https://doi.org/10.1609/aaai.v38i1.27787.

Minkai Xu, Alexander S. Powers, Ron O. Dror, Stefano Ermon, and Jure Leskovec. Geometric latent diffusion models for 3d molecule generation. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 38592–38610. PMLR, 2023. URL https://proceedings.mlr.press/v202/xu23n.html.

Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM Comput. Surv.*, 56(4):105:1–105:39, 2024. doi: 10.1145/3626235. URL https://doi.org/10.1145/3626235.

APPENDIX

A LLM USAGE

We used LLM to polish part of the text.

B FLEXIFLOW

This section provides additional details about the architecture and training setup of FlexiFlow. Moreover, we provide additional information about metrics, datasets used to train the models, processing of the data and models hyperparameters. Because it is not possible to adopt the exact same notation, we instead use a notation closely aligned with that of SemlaFlow (Irwin et al., 2025), with the aim of helping the reader more clearly discern the key architectural differences.

B.1 FLEXIFLOW MODEL IN DETAILS

We provide in the following sections a description of each module of the FlexiFlow architecture. The full architecture can be summarized as the composition of featurization, L repeated FlexiFlow layers, and a feature refinement layer:

$$\text{FlexiFlow} \;=\; \underbrace{\text{Refinement}}_{L+2} \circ \underbrace{\text{FlexiFlowLayer} \circ \cdots \circ \text{FlexiFlowLayer}}_{L \text{ times}} \circ \underbrace{\text{Featurization}}_{1}. \tag{15}$$

Featurization layer. Similarly to Irwin et al. (2025), we process the input $h \in \mathbb{R}^{n\times|\mathcal{A}|\times|\mathcal{C}|}$, $e \in \mathbb{R}^{n\times n\times"\mathcal{B}|}$, $x \in \mathbb{R}^{n\times 3}$ and $y \in \mathbb{R}^{n\times 3}$ using a featurization layer that maps the input features into a higher dimensional space $\mathbb{R}^d$. $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ represent the sets of atom, bond, and charge types, respectively. To this end, we use two different MLPs for the invariant features h and e, while we use a shared linear layer for the coordinates x and y. Specifically, the coordinates are first reshaped into $\mathbb{R}^{n\times 1\times 3}$ and then projected into a higher dimensional space $\mathbb{R}^{n\times d\times 3}$. According to Irwin et al. (2025) this operation helps to increase the expressivity of the model. For notation consistency h and e are cloned into h^x, h^y and e^x, e^y respectively, where h^x and e^x are used to update the features of x and h^y and e^y for y.

FlexiFlow layer. Each FlexiFlow layer consists of a feed-forward block followed by a graph attention block:

$$\text{FlexiFlowLayer} = \mathcal{G}_{\text{attn}} \circ \mathcal{F}. \tag{16}$$

FEED-FORWARD: The features h^x, h^y, e^x, e^x, x and y are now now normalized as follows:

$$\tilde{x} = \phi_{\text{equi}}(x) \quad \tilde{y} = \phi_{\text{equi}}(y) \quad \tilde{h}^x = \phi_{\text{inv}}(h^x) \quad \tilde{h^y} = \phi_{\text{inv}}(h^y) \tag{17}$$

where $\phi_{\text{equi}}(\cdot)$ and $\phi_{\text{inv}}(\cdot)$ are the equivariant and invariant normalization layers, from Vignac et al. (2023) and Ba et al. (2016), respectively. Subsequently, we use the normalized features to update the invariant features h^x and h^y. These are updated as follows:

$$h_i^{x\,\text{ff}} \;=\; h_i^x + \Phi_\theta\big(\big[\,\tilde{h}_i^x\,,\,\|\tilde{x}_i\|\,\big]\big) \qquad h_i^{y\,\text{ff}} \;=\; h_i^y + \Phi_\theta\big(\big[\,\tilde{h}_i^y\,,\,\|\tilde{y}_i\|\,\big]\big), \tag{18}$$

where $[\cdot,\cdot]$ denotes concatenation and the coordinate norm is taken component-wise:

$$\|\tilde{x}_i\| = [\|\tilde{x}_{i,1}\|, \ldots, \|\tilde{x}_{i,d}\|], \tag{19}$$

and Φ_θ is an MLP with this structure $\Phi_\theta(z) = W_2 \cdot \text{SiLU}(W_1 z + b_1) + b_2$ that maps back to the original dimensionality the features $\Phi_\theta : \mathbb{R}^{d+s} \to \mathbb{R}^d$. Since $h^{x\,\text{ff}}$ and $h^{y\,\text{ff}}$ are updated based on $\tilde{x}$ and $\tilde{y}$, respectively, the update of $h^{x,\text{ff}}$ is influenced by x features, while the update of $h^{y,\text{ff}}$ is influenced by y features.

To update the equivariant features x and y,

$$x_i^{\text{ff}} = x_i + W_g\Big(\sum_{j=1}^{d}(W_f\,\tilde{x}_j) \otimes \Psi_\theta(\tilde{h}_i^x)\Big) \quad y_i^{\text{ff}} = y_i + W_g\Big(\sum_{j=1}^{d}(W_f\,\tilde{y}_j) \otimes \Psi_\theta(\tilde{h}_i^y)\Big), \tag{20}$$

where $\otimes$ denotes the outer product, $\Psi_\theta : \mathbb{R}^d \to \mathbb{R}^p$ is and MLP defined as: $\Psi_\theta(z) = V_2 \cdot \text{SiLU}(V_1 \cdot z + c_1) + c_2$, W_g and W_f are two linear projections and p is the projection dimension.

We summarize the feed-forward block with the compact for as $\mathcal{F}$.

GRAPH ATTENTION: The graph attention layer aims to combine invariant and equivariant features with the attention mechanism to update the nodes features representations. This module is important to let the y features be influenced by x features. In this paragraph we discuss about the two components which is made of, the message computation and the attention mechanism, where the last is subdivided in two parts, the invariant attention and the equivariant attention.

Message computation: To compute the messages we first normalize the coordinates and invariant features using the same normalization scheme as in the feed forward layer:

$$\tilde{x} = \phi_{\text{equi}}(x) \quad \tilde{y} = \phi_{\text{equi}}(y) \quad \tilde{h}_x = \phi_{\text{inv}}(h^x) \quad \tilde{h}_y = \phi_{\text{inv}}(h^y) \tag{21}$$

Then, we perform the $\cdot$ product between the normalized coordinates to obtain x_{pairs} and y_{pairs}. We also project the invariant features using a linear transformation obtaining $\hat{h}_x$ and $\hat{h}_y$. The messages are thus computed as follows:

$$x_p = \tilde{x}_i^{\text{ff}} \cdot \tilde{x}_j^{\text{ff}\,T}, \;\; y_p = \tilde{y}_i^{\text{ff}} \cdot \tilde{y}_j^{\text{ff}\,T}, \;\; h_p^x = [W_g \tilde{h}_i^{x\,\text{ff}} \parallel W_g \tilde{h}_j^{x\,\text{ff}}], \;\; h_p^y = [W_g \tilde{h}_i^{y\,\text{ff}} \parallel W_g \tilde{h}_j^{y\,\text{ff}}] \tag{22}$$

Finally, we concatenate the features as reported below to obtain the final messages for x and y:

$$\omega_p^x = [h_p^x, x_p, e^x], \quad \omega_p^y = [h_p^x \cdot h_p^y, x_p, y_p, e^x \cdot e^y] \tag{23}$$

where W_g represents a linear projection. The final messages are computed using two separate MLPs for MLP_x and MLP_y from ω_p^x and ω_p^y as they have different input shapes, obtaining the messages that are used to update the x, y, h^x, h^y, e^x and e^y features in the equivariant and invariant attention blocks. These are now denoted as $\omega_{p,h}^x$, $\omega_{p,h}^y$, $\omega_{p,x}^x$, $\omega_{p,y}^y$, $\omega_{p,e}^x$ and $\omega_{p,e}^y$.

Invariant Attention: We proceed updating the h^x and h^y features separately as outlined below, where σ denote the following operation $\sigma(\mathbf{z})_i = e^{z_i} / \sum_{j=1}^K e^{z_j}$ applied on the penultimate feature dimension:

$$\alpha_x^k = \sigma(\omega_{p,h}^x) \quad \alpha_y^k = \sigma(\omega_{p,h}^y) \tag{24}$$

$$\tilde{h}^x = W_v \phi_{\text{inv}}(h^x) \quad \tilde{h}^y = W_v \phi_{\text{inv}}(h^y) \quad \tilde{h}^k \in \mathbb{R}^{N \times d\text{head}} \text{ for } k = 1, \ldots, n_{\text{heads}} \tag{25}$$

Thus, we apply the same process to aggregate the features for each head as reported below:

$$h_{\text{aggr}}^k = \sum_j \alpha_{ij}^k \cdot \tilde{h}_j^k \quad w_i^k = \sqrt{\sum_j (\alpha_{ij}^k)^2} \quad h^{\text{message}} = W_z [h_{\text{aggr}}^k \cdot w_i^k] \tag{26}$$

where $W_v : \mathbb{R}^d \to \mathbb{R}^m$ and $W_z : \mathbb{R}^m \to \mathbb{R}^d$ are linear projections and m is the latent message dimensionality.

Equivariant Attention: We now update the coordinates features x^{ff} and y^{ff} using and equivariant attention mechanism following Irwin et al. (2025). Similarly to the invariant attention, we compute the attention weights as follows:

$$\alpha_x^k = \sigma(\omega_{p,x}^x) \quad \alpha_y^k = \sigma(\omega_{p,y}^y) \tag{27}$$

$$\tilde{\alpha}^k = W_r \alpha^k \quad \tilde{x}_k = W_r \phi_{\text{equi}}(x_k) \quad \tilde{y}_k = W_{e^1} \phi_{\text{equi}}(y_k) \tag{28}$$

The same attention strategy is applied to $\tilde{x}$ and $\tilde{y}$ to aggregate the contribution of each node to the final message update:

$$x_{\text{aggr}}^k = \sum_j^N \tilde{\alpha}_{ij}^k \cdot \left(\frac{\tilde{x}_i - \tilde{x}_j}{(\tilde{x}_i - \tilde{x}_j + \epsilon)} \right) \quad w_i^k = \sqrt{\sum_j (\tilde{\alpha}_{ij}^k)^2} \quad x^{\text{message}} = W_s [x_{\text{aggr}}^k \cdot w_i^k] \tag{29}$$

where ϵ is equal to 10^{-12} and $W_r : \mathbb{R}^d \to \mathbb{R}^m$ and $W_s : \mathbb{R}^m \to \mathbb{R}^d$ are two linear projections and m is the latent message dimensionality. Thus, we finally update the features as follows using this scheme:

$$x = x + x^{\text{message}} \;\; y = y + y^{\text{message}} \tag{30}$$

$$h^x = h^x + h^{x,\text{message}} \;\; h^y = h^y + h^{y,\text{message}} \tag{31}$$

$$e^x = e^x + \omega_{p,e}^x \;\; e^y = e^y + \omega_{p,e}^y \tag{32}$$

We now refer to one graph attention block forward pass with this notation $\mathcal{G}_{attn}$.

Features refinement. We apply a final feed forward layer x, y, h^x, h^y and use the output to shift the representations at the last FlexiFlow layer. On x and y we use the same equivariant norm, and a linear transformation to shrink the coordinate sets into one. On the edge features we apply the Edge Update Layer before the final MLP projection, see the dedicated Appendix section B.2. Subsequently we apply two different invariant normalization layers to h^x (atoms, charges) and e^x (bonds), and use three separate MLP to obtain the logits for the atoms, charges and bonds types. These are finally projected into $\mathbb{R}^{n\times|\mathcal{A}|}$, $\mathbb{R}^{n\times|\mathcal{C}|}$ and $\mathbb{R}^{n\times n\times|\mathcal{B}|}$, where $|\mathcal{A}|$, $|\mathcal{C}|$ and $|\mathcal{B}|$ are the cardinality of the sets $\mathcal{A}$, $\mathcal{C}$ and $\mathcal{B}$ for atom, charge and bond types.

Protein conditioning. We model a protein as a set of invariant and equivariant features for each protein atom, from one-hot encoded $\rho^{inv} \in \mathbb{R}^{\ell\times|\mathcal{H}|}$ and coordinates $\rho^{equi} \in \mathbb{R}^{\ell\times 3}$ where ℓ is the number of protein atoms and $\mathcal{H}$ is the set of protein atom types. To support the conditioning on the protein features, we added a protein feed forward layer before the graph attention layer in each FlexiFlow layer. Each protein atom invariant and equivariant features set of ρ^{inv} and ρ^{equi} are than fed into the graph attention layer, where we separately compute the messages between ligand-ligand and ligand-protein and concatenate them before the attention mechanism. Further details are provided below are provided in Appendix B.3.

B.2 Edge feature refinement layer FlexiFlow

Similarly to Irwin et al. (2025), we postprocess the features of the bonds (edges) using a dedicated layer, which we call Edge Update Layer (EUL). The EUL takes as input the 3D coordinat sets of the atoms (nodes) $x \in \mathbb{R}^{n\times d\times 3}$, their features $h \in \mathbb{R}^{n\times d}$ and the bond features $e \in \mathbb{R}^{n\times n\times d}$. It outputs the updated bond features e'.

We normalize x, h and e using the equivariant and invariant normalizations ϕ_{equi} and ϕ_{inv} respectively:

$$\tilde{x} = \phi_{\text{equi}}(x) \;\; \tilde{h} = \phi_{\text{inv}}(h^x) \;\; \tilde{e} = \phi_{\text{inv}}(e^x) \tag{33}$$

Over the coordinate sets, we compute the geometric distances and inner products, and then concatenate them:

$$\Delta_{ij} = \tilde{x}_i - \tilde{x}_j, \qquad d_{bij} = \|\Delta_{ij}\|_2^2, \qquad p_{ij} = \langle \tilde{x}_i, \tilde{x}_j \rangle, \tag{34}$$

$$\tag{35}$$

Subsequently, we apply a linear layer to project the node features to the message dimension d:

$$\tilde{h}_i = W_h \tilde{h}_i + b_h, \quad W_h \in \mathbb{R}^{d\times d}. \tag{36}$$

and we form the pair features by concatenation:

$$h_{ij}^{\text{pair}} = [\tilde{h}_i \parallel \tilde{h}_j] \in \mathbb{R}^{2d}. \tag{37}$$

Finally, we concatenate all the features to form the input to the message MLP:

$$F_{ij} = [\, h_{ij}^{\text{pair}} \parallel \; d_{ij} \parallel \; p_{ij} \parallel \; \tilde{e}_{ij} \,] \in \mathbb{R}^{2d+2+d} \tag{38}$$

$$m_{ij} = \sigma(W_l F_{ij} + b_1), \tag{39}$$

$$e'_{ij} = W_q m_{ij} + b_2, \tag{40}$$

where σ is the SiLU activation function, $W_l \in \mathbb{R}^{d\times(2d+2+d)}$, $W_q \in \mathbb{R}^{d\times d}$ and $e' \in \mathbb{R}^{n\times n\times d}$ are the updated bond features.

B.3 Protein conditioning

This section provides additional about how the ligand-protein interaction conditioning is computed. After Equation 13, we use the normalized features coordinates $\tilde{x}$ of the ligand, and normalize the protein coordinates features ρ_{ff}^{equi}, protein atoms features ρ_{ff}^{inv} and ligand atom features h^x as follows:

$$\tilde{\rho}^{equi} = \phi_{\text{equi}}\big(\rho_{ff}^{equi}\big), \qquad \tilde{\rho}^{inv} = \phi_{\text{inv}}\big(\rho_{ff}^{inv}\big), \qquad \tilde{h} = \phi_{\text{inv}}(h^x). \tag{41}$$

We concatenate the protein and ligand atoms features pairwise:

$$h_{ij}^{\rho^{inv}} = [\,\tilde{h}_i \;\|\; \tilde{\rho}_j^{inv}\,] \in \mathbb{R}^{2d}. \tag{42}$$

Next, we compute the distances between the ligand and protein atoms:

$$d_{ijs} = \sqrt{\|\tilde{x}_{is} - \tilde{\rho}_{js}^{equi} + \varepsilon\|_2^2}. \tag{43}$$

where ε is equal to $1e-12$. Lastly, we concatenate the distances to the pair features and apply an MLP to project them to the desired dimension of the message:

$$m_{ij}^{\rho} = [\,h_{ij}^{\rho^{inv}} \;\|\; D_{ij}\,] \in \mathbb{R}^{(2d+s)}, \tag{44}$$

$$m_{ij}^{\rho} = W_o\,\sigma\big(W_a m_{ij}^{\rho} + b_a\big) + b_o \quad \in \mathbb{R}^d, \tag{45}$$

with $W_a \in \mathbb{R}^{d\times(2d+s)}$, $W_o \in \mathbb{R}^{d\times d}$, σ the SiLU activation, while b_a and b_o are the respective bias terms.

The message is computed separately for x and y coordinate set, using the same weights and protein features. Finally, we concatenate the x and y coordinate, atoms and message features with the protein coordinate, atoms and message features, respectively, and follow the same steps to perform the features update according with Equation 26 and 29.

B.4 Loss setting FlexiFlow

To support the generation of the graphs x and $\mathcal{S}$, our loss follow this scheme:

$$\mathcal{L} = \mathcal{L}_{x,y} + \mathcal{L}_a + \mathcal{L}_c + \mathcal{L}_e + \mathcal{L}_{\text{reg}} \tag{46}$$

where (1) $\mathcal{L}_{x,y}$ coordinates loss for x and y is defined as the mean squared error between the predicted and target coordinates:

$$\mathcal{L}_{x,y} = \frac{1}{N}\sum_{i=1}^{N} ||\hat{x}_i^{(1)} - x_i^{(1)}||^2 + \frac{1}{N}\sum_{i=1}^{N} ||\hat{y}_i^{(1)} - y_i^{(1)}||^2 \tag{47}$$

where $\hat{x}_i^{(1)}$ and $\hat{y}_i^{(1)}$ are the predicted coordinates for the i-th atom in x and y respectively, while $x_i^{(1)}$ and $y_i^{(1)}$ are the ground truth coordinates and N the number of atoms. (2) $\mathcal{L}_a$, $\mathcal{L}_c$, $\mathcal{L}_e$ are the negative log-likelihood losses for the categorical atoms, charges and bonds types respectively:

$$\mathcal{L}_a = -\frac{1}{n}\sum_{i=1}^{n} \log p(\hat{a}_i^{(1)} = a_i^{(1)}), \tag{48}$$

$$\mathcal{L}_c = -\frac{1}{n}\sum_{i=1}^{n} \log p(\hat{c}_i^{(1)} = c_i^{(1)}), \tag{49}$$

$$\mathcal{L}_e = -\frac{1}{n^2}\sum_{(i,j)\in\mathcal{E}} \log p(\hat{e}_{ij}^{(1)} = e_{ij}^{(1)}) \tag{50}$$

where $\hat{a^{(1)}}_i$, $\hat{c}_i^{(1)}$ and $\hat{e}_{ij}^{(1)}$ are the predicted atom type, charge and bond type for the i-th atom and (i,j)-th bond respectively, while $a_i^{(1)}$, $c_i^{(1)}$ and $e_{ij}^{(1)}$ are the ground truth values and $\mathcal{E}$ is the set of edges in the molecular graph. (3) $\mathcal{L}_{\text{reg}}$ a regularization loss is made by different components: (3.1) enforce bonds lengths consistency on both x and y:

$$D_{ij}^{x,1} = \|x_i^{(1)} - x_j^{(1)}\|_2\,, \qquad D_{ij}^{x,p} = \|\hat{x}_i - \hat{x}_j\|_2\,, \tag{51}$$

$$D_{ij}^{y,1} = \|y_i^{(1)} - y_j^{(1)}\|_2\,P_{ij}, \qquad D_{ij}^{y,p} = \|\hat{y}_i - \hat{y}_j\|_2\,P_{ij}. \tag{52}$$

where $x_i^{(1)}, y_i^{(1)}$ be target (ground-truth) coordinates and $\hat{x}_i, \hat{y}_i$ the predicted coordinates and D the respective pairwise distances.

Let $e_{ijk}^{(1)}$ be the (target) bond-type logits and define the bond presence mask

$$B_{bij} = \mathbf{1}\left[\arg\max_k e_{ijk}^{(1)} > 0\right], \tag{53}$$

so only pairs with a non-zero bond type contribute. The adjacency (distance) constraint losses implemented in the code are the mean absolute deviations over all pairs:

$$\mathcal{L}_{\text{adj}}^{x} = \frac{1}{N^2}\sum_{i,j=1}^{N} B_{ij}\left|D_{ij}^{x,p} - D_{ij}^{x,1}\right|, \qquad \mathcal{L}_{\text{adj}}^{y} = \frac{1}{N^2}\sum_{i,j=1}^{N} B_{ij}\left|D_{ij}^{y,p} - D_{ij}^{y,1}\right|. \tag{54}$$

Additionally, (3.2) we align the categorical for y on x:

$$\mathcal{L}_{\text{bond-align}} = \left(\frac{\sum_i^n \sum_j^n \|\hat{E}_{ij}^x - \hat{E}_{ij}^y\|_2^2}{\sqrt{n^2+\varepsilon}}\right) - \frac{1}{n^2}\sum_{(i,j)\in\mathcal{E}} \log p(\hat{e}_{ij}^y = e_{ij}), \tag{55}$$

$$\mathcal{L}_{\text{type-align}} = \left(\frac{\sum_i \|\hat{H}_i^x - \hat{H}_i^y\|_2^2}{\sqrt{n}+\varepsilon}\right) - \frac{1}{n}\sum_{i=1}^{n} \log p(\hat{a}_i^y = a_i), \tag{56}$$

$$\mathcal{L}_{\text{charge-align}} = \left(\frac{\sum_i \|\hat{C}_i^x - \hat{C}_i^y\|_2^2}{\sqrt{n}+\varepsilon}\right) - \frac{1}{n}\sum_{i=1}^{n} \log p(\hat{c}_i^y = c_i). \tag{57}$$

where ε is a small constant to avoid division by zero, $\hat{E}_{ij}^x, \hat{E}_{ij}^y$ are the predicted bond type logits for the (i,j)-th bond in x and y respectively, while $\hat{H}_i^x, \hat{H}_i^y$ and $\hat{C}_i^x, \hat{C}_i^y$ are the predicted atom type and charge logits for the i-th atom in x and y respectively. The full regularization loss is thus defined as:

$$\mathcal{L}_{\text{reg}} = \mathcal{L}_{\text{adj}}^x + \mathcal{L}_{\text{adj}}^y + \mathcal{L}_{\text{bond-align}} + \mathcal{L}_{\text{type-align}} + \mathcal{L}_{\text{charge-align}}. \tag{58}$$

B.5 Training scheme & Interpolants setting

Let $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ denote the sets of atom types, bond types, and charge types, respectively. We denote by $(a,b,c) \sim \text{Cat}(1/|\mathcal{A}|) \cdot \text{Cat}(1/|\mathcal{B}|) \cdot \text{Cat}(1/|\mathcal{C}|)$ the sampling of a triplet from three independent categorical distributions, each uniform over its corresponding set. We draw samples from our time-dependent categorical distribution following the approach of Campbell et al. (2024). Specifically, the time variable t is sampled from a Beta distribution with parameters $\alpha = 2.0$ and $\beta = 1.0$. During training, given samples from the noised data distribution, the objective is to predict the corresponding target data distribution. The coordinate interpolants x_t and y_t are obtained following Tong et al. (2024).

Algorithm 2 Training scheme

$(x_1, y_1, a_1, b_1, c_1) \sim p_{data}$
$x_0 \sim \mathcal{N}(0, \text{I})$, $y_0 \sim \mathcal{N}(0, \text{I})$, $t \sim \text{Beta}(\alpha, \beta)$
$x_t \sim \mathcal{N}(tx_1 + (1-t)x_0, \sigma^2)$
$y_t \sim \mathcal{N}(ty_1 + (1-t)y_0, \sigma^2)$
$a_0, b_0, c_0 \sim \text{Cat}(1/|\mathcal{A}|) \cdot \text{Cat}(1/|\mathcal{B}|) \cdot \text{Cat}(1/|\mathcal{C}|)$,
$a_t, b_t, c_t \sim \text{CatInterp}(t, a_0, a_1) \cdot \text{CatInterp}(t, b_0, b_1) \cdot \text{CatInterp}(t, c_0, c_1)$,
while Training **do:**
    $(\hat{x}_1, \hat{y}_1, \hat{a}_1^x, \hat{b}_1^x, \hat{c}_1^x, \hat{a}_1^y, \hat{b}_1^y, \hat{c}_1^y) \leftarrow f_\theta(x_t, y_t, a_t, b_t, c_t)$
    $\mathcal{L}(\theta) = \mathcal{L}_{x,y}(\hat{x}_1, x_1, \hat{y}_1, y_1) + \mathcal{L}_a(\hat{a}_1^x, \hat{a}_1^y, a_1) + L_c(\hat{c}_1^x, \hat{c}_1^y, c_1) +$
        $\mathcal{L}_b(\hat{b}_1^x, \hat{b}_1^y, b_1) + L_{reg}(\hat{x}_1, x_1, \hat{y}_1, y_1)$
end while

B.5.1 MODEL & TRAINING HYPERPARAMETERS

Here we provide the list of hyperparameters that are kept fixed across the models configurations: dimension edge features = 128 and invariant positional embedding size = 64. In Table 4 are reported the parameters that vary in the model configuration. Refer to Irwin et al. (2025) for further details. All the result in the paper that do not specifically mention the model size use the Large configuration of it.

Model type	n_layers	d_model	d_message & d_message_hidden	n_attn_heads	Parameters
Small (S)	6	384	64	12	17.2M
Medium (M)	8	384	128	32	24.7M
Large (L)	12	384	128	32	37.7M

Table 4: The table reports the parameters that vary across model configurations, while all other fixed parameters are listed below. d stands for model features dimensionality.

Key training configuration details:

- training seed = 42
- coordinate noise $\sigma = 0.2$ on the interpolated coordinates
- Adam (Kingma & Ba, 2015) with learning rate lr=$1e-3$ and weight decay 0.0
- LinearLR is used as learning rate scheduler with start_factor=$1e-2$ and total_iters=10000
- all the models are trained using exponential moving average (EMA)

B.6 DATA PRE-PROCESSING MOLECULAR STRUCTURES

QM9. The QM9 dataset (Ramakrishnan et al., 2014) consists of ∼134k small organic molecules with up to 9 heavy atoms (C, O, N, F) and their corresponding 3D conformations. We follow the standard split used in previous works (Hoogeboom et al., 2022; Vignac et al., 2023), using $\sim 100k$ molecules for training. We preprocess the data following the steps outlined in (Irwin et al., 2025), which include centering the molecules at the origin, normalizing the coordinates, checking validity of the molecular graphs and graph fragmentation with RdKit. Since the hydrogen atoms are kept, the resulting model vocabulary is composed by (H, C, N, O, F).

Since QM9 provides only one conformer per molecule, we augment the dataset by generating 20 conformers per molecule using the ETKDG method (Riniker & Landrum, 2015) implemented in RdKit (obtaining ∼1.8M total samples). We then optimize these conformers using the MMFF94 force field (Halgren, 1996) to ensure physically plausible structures. The target x conformer is selected as the closest conformation to the mean, while the remaining conformers form the set $\mathcal{S}$.

GEOM Drugs. The GEOM Drugs dataset (Axelrod & Gómez-Bombarelli, 2022) contains ∼400k unique drug-like molecules with up to 181 atoms (H, B, C, N, O, F, Si, P, S, Cl, Br, I) and their corresponding 3D conformations. Conversely to QM9, GEOM Drugs provides multiple conformers per molecule, with an average of 21 conformers per molecule. To achieve these conformers, the authors used GFN2-xTB calculations (Bannwarth et al., 2019), a physics-based method that provides accurate low-energy conformers.

Similarly to QM9, we selected the target x conformer as the closest conformation to the mean, while the remaining conformers form the set $\mathcal{S}$. We use the same splits as in previous works (Vignac et al., 2023; Le et al., 2024), with $\sim 5.8M$ molecules for training. The data are processed similarly to QM9, ensuring that the molecules are centered, normalized, and valid. The model vocabulary comprises (H, B, C, N, O, F, Si, P, S, Cl, Br, I), with hydrogens retained during training. Our final training set contains $243{,}718$ unique molecular target graphs x and $5{,}491{,}198$ distinct molecular conformers y.

B.7 METRICS FOR DE-NOVO GENERATION

We provide a description of the metrics used for 3D generation, specifically, validity, uniqueness, novelty, atom stability, and molecule stability.

* **Validity:** Validity measures the percentage of generated molecules that are chemically valid according with the sanitization check done by RDKit. A molecule is considered valid if it passes the sanitization check, which includes checks for valence, aromaticity, and other.
* **Uniqueness:** Uniqueness measures the percentage of unique molecular graphs converted into canonical SMILES strings.
* **Novelty:** Novelty measures the percentage of generated molecules that are not present in the training set.
* **Atom Stability:** Atom stability measures the percentage of atoms in the generated molecules that have a valid valence according to their element type.
* **Molecule Stability:** Molecule stability measures the percentage of generated molecules where all atoms are stable.
* **Energy:** According to the Boltzmann distribution, the probability of a conformation x is determined by its energy $U(x)$ as $P(x_i) = Z^{-1}(e^{-U(x_i)/k_B T})$, where $U(x_i)$ is the energy of state i parametrized by the MMFF96 force field, k_B is the Boltzmann constant, T is the absolute temperature, Z is the partition function.
* **Strain:** The strain is computed as $U(x) - U(x^*)$, where x is a conformation and x^* is the energy minimized conformer.

B.8 METRICS FOR CONFORMER GENERATION

Similarly to (Ganea et al., 2021), (Jing et al., 2022) and (Havens et al., 2025), we compute Average Minimum RMSD (AMR) and Coverage (Cov) for Precision (P) and Recall (R).

We generate with FlexiFlow a number N of conformers for each molecule generated from scratch, then we use the conformer with the lowest energy as the input to CREST, which generates a set of M reference conformers, where $M < N$. Equivalently, to generate the conformers with RDKit, we use the conformer with the lowest energy generated by FlexiFlow as the input to RDKit to produce a set of N conformers.

Lastly, we compare against the reference CREST conformers to the generated ones. Finally, we use the RMSD metric to compare generated conformers to reference conformers, which aims to capture both the quality and diversity of the generated conformers. The RMSD is computed as the minimum distance between two conformers, taking into account the molecular structure.

R stands for recall, P for precision, Cov for coverage, and AMR for average minimum RMSD. In this context, recall measures the coverage of the generated conformers against the reference conformers, while precision measures how closely at least one of the generated conformers approximates a reference conformer.

$$\text{Cov-R}(\delta) := \frac{1}{M} \left|\{m \in \{1, \dots, M\} : \exists n \in \{1, \dots, N\}, \quad \text{RMSD}(C_n, C_m) < \delta\}\right| \tag{59}$$

$$\text{AMR-R} := \frac{1}{M} \sum_{m \in \{1,\dots,M\}} \min_{n \in \{1,\dots,N\}} \text{RMSD}(C_n, C_m) \tag{60}$$

$$\text{Cov-P}(\delta) := \frac{1}{N} \left|\{n \in \{1, \dots, N\} : \exists m \in \{1, \dots, M\}, \quad \text{RMSD}(C_n, C_m) < \delta\}\right| \tag{61}$$

$$\text{AMR-P} := \frac{1}{N} \sum_{n \in \{1,\dots,N\}} \min_{m \in \{1,\dots,M\}} \text{RMSD}(C_n, C_m) \tag{62}$$

where $\delta > 0$ is the coverage threshold.

B.9 RMSD AFTER ENERGY MINIMIZATION METRIC

We perform additional minimization until convergence (within a fixed number of minimization steps) using a physico-chemical force field (in this case MMFF94). Thus, an RMSD > 0 indicates that we have two conformers $\mathbf{x}_1$ and $\mathbf{x}_2$ such that $\mathbf{x}_1 \neq \mathbf{x}_2$ and

$$\nabla E(\mathbf{x}_1) \equiv \nabla E(\mathbf{x}_2) \equiv 0. \tag{63}$$

For both $\mathbf{x}_1$ and $\mathbf{x}_2$, there exist neighborhoods with radii d_1 and d_2 such that

$$f(\mathbf{x}_1) < f(\mathbf{x}) \quad \text{for all } \mathbf{x} \text{ satisfying } \|\mathbf{x} - \mathbf{x}_1\| < d_1, \tag{64}$$

and similarly for $\mathbf{x}_2$. This means they correspond to distinct local minima.

C FLOW DECOMPOSITION

As part of this section of the Appendix, we provide the proof of the determinant decomposition of the Jacobian of the flow $\psi_t(x, \mathcal{S})$ with respect to $(x, \mathcal{S})$ under local dependence. Moreover, we formulate the target velocity field $u_t(x, \mathcal{S} \mid x_1, \mathcal{S}_1)$ in closed form under linear interpolation on sets.

C.1 PROOF DETERMINANT DECOMPOSITION

Let be $\mathcal{S} = \{y_1, \ldots, y_m\}$ and define $\psi_t(x, \mathcal{S})$ as the concatenation of the flows among $\mathcal{S}$

$$\psi_t(x, \mathcal{S}) := \Big\|_{y \in \mathcal{S}} \psi_t(x, y) = \big(\psi_t^{(1)}(x, y_1), \ldots, \psi_t^{(m)}(x, y_m)\big), \tag{65}$$

where each block $\psi_t^{(i)}(x, y_i) \in \mathbb{R}^{n \times m}$.

Lemma 1: We consider the Jacobian of the flow ψ_t as:

$$J(x, \mathcal{S}) := \nabla_{(x,\mathcal{S})} \psi_t(x, \mathcal{S}). \tag{66}$$

where J satisfies the local dependence property, $\frac{\partial \psi_t^i}{\partial y_j} = 0 \quad \text{for } j \neq i$, such that each block depends only on (x, y_i).

Under the local dependence assumption, we now reconduct the determinant of J to a block diagonal matrix to allow the flow decomposition. For simplicity, we provide a proof for $m = 2$ and $n = 2$, however, the proof can be easily extended to $m > 2$. We start with three 2-dimensional vectors:

$$x = [x_1 \quad x_2], \quad y^{(1)} = [y_{1,1} \quad y_{1,2}], \quad y^{(2)} = [y_{2,1} \quad y_{2,2}].$$

We concatenate the vectors as follows:

$$v_1 = [x_1 \quad x_2 \quad y_{1,1} \quad y_{1,2}], \qquad v_2 = [x_1 \quad x_2 \quad y_{2,1} \quad y_{2,2}]$$
$$z = [v_1 \quad v_2] = [x_1 \quad x_2 \quad y_{1,1} \quad y_{1,2} \quad x_1 \quad x_2 \quad y_{2,1} \quad y_{2,2}].$$

Thus $z \in \mathbb{R}^8$ in this special case. Hence we can define the full flow as the concatenation of two independent flows:

$$\psi_t(z) = \begin{bmatrix} \psi_t^{(1)}(z) \\ \psi_t^{(2)}(z) \end{bmatrix}$$

where each $\psi_t^{(i)}(z) \in \mathbb{R}^4$ for $i = 1, 2$. Therefore, we can compute the Jacobian of the full flow $\psi_t(z)$ with respect to z as:

$$J(z) = \frac{\partial \psi_t(z)}{\partial z} \in \mathbb{R}^{8 \times 8}.$$

The Jacobian has a block-diagonal structure:

$$J(z) = \begin{bmatrix} J^{(1)} & 0 \\ 0 & J^{(2)} \end{bmatrix},$$

where each block $J^{(i)} = \frac{\partial \psi_t^{(i)}(z)}{\partial z} \in \mathbb{R}^{4 \times 8}$ for $i = 1, 2$. This, for $J^{(1)}$, where $i = 1, \ldots, 4$,

$$\psi_t^{(1)}(z) = z_i^2 + \prod_{k=1}^{4} z_k. \tag{67}$$

Thus

$$\frac{\partial \psi_t^{(1)}}{\partial z_j} = \begin{cases} 2z_i + \prod_{\substack{k=1 \\ k \neq i}}^{4} z_k, & j = i, \\ \prod_{\substack{k=1 \\ k \neq j}}^{4} z_k, & j \neq i,\ j \in \{1,2,3,4\}, \\ 0, & j \in \{5,6,7,8\}. \end{cases} \tag{68}$$

For $J^{(2)}$, where $i = 5, \ldots, 8$,

$$\psi_t^{(2)}(z) = z_i^2 + \prod_{k=5}^{8} z_k. \tag{69}$$

Thus

$$\frac{\partial \psi_t^{(2)}}{\partial z_j} = \begin{cases} 2z_i + \prod_{\substack{k=5 \\ k \neq i}}^{8} z_k, & j = i, \\ \prod_{\substack{k=5 \\ k \neq j}}^{8} z_k, & j \neq i,\ j \in \{5,6,7,8\}, \\ 0, & j \in \{1,2,3,4\}. \end{cases} \tag{70}$$

As the Jacobian J results in block diagonal matrix, it can be factorized as the product of block determinants. Therefore,

$$\det\big(\nabla_{(x,\mathcal{S})}\psi_t(x,\mathcal{S})\big) = \prod_{i=1}^{m} \det\big(\nabla_{(x,y_i)}\psi_t^i(x,y_i)\big). \tag{71}$$

C.2 Target vector field u_t for the special case of linear interpolants on sets

We define the target velocity field as the conditional expectation of the trajectory velocity:

$$u_t(z \mid z_1) = \mathbb{E}[\dot{z}_t \,|\, z_t = z,\, z_1]\,. \tag{72}$$

where $\dot{z}_t$ correspond to the z_t derivative. In our case, the path definition reduces to linear interpolation which can be defined as:

$$z_t = (1-t)z_0 + tz_1, \tag{73}$$

Thus,

$$u_t(z \mid z_1) = \mathbb{E}[z_1 - z_0 \mid z_t = z,\, z_1], \tag{74}$$

and the path is a straight line that connects z_0 to z_1, hence the velocity is constant along the path. The extension to $(x, \mathcal{S})$ can be seen by reformulating $z = (x, \mathcal{S})$ and defining the path elementwise:

$$(x_t, y_{t,i}) = (1-t)(x_0, y_{0,i}) + t(x_1, y_{1,i}), \quad i = 1, \ldots, m. \tag{75}$$

where $(x_0, \mathcal{S}_{0,i})$ is sampled from the prior distribution and $(x_1, \mathcal{S}_{1,i})$ from the data distribution and i indicates the i-th element in the set $\mathcal{S}$. So $(x_t, \mathcal{S}_t) = \{(1-t)(x_0, y_{0,i}) + t(x_1, y_{1,i})\}_{i=1}^m$. Now we can condition on $(x_t, \mathcal{S}_t) = (x, \mathcal{S})$ with endpoint $(x_1, \mathcal{S}_1)$:

$$u_t(x, \mathcal{S} \mid x_1, \mathcal{S}_1) = \mathbb{E}\Big[(x_1 - x_0,\, \{y_{1,i} - y_{0,i}\}_{i=1}^m) \,\Big|\, (x_t, \mathcal{S}_t) = (x, \mathcal{S}), (x_1, \mathcal{S}_1)\Big]. \tag{76}$$

D ADDITIONAL RESULTS

D.1 ADDITIONAL RESULTS TOP-K FRACTION GENERATED CONFORMERS QM9 AND GEOM DRUGS

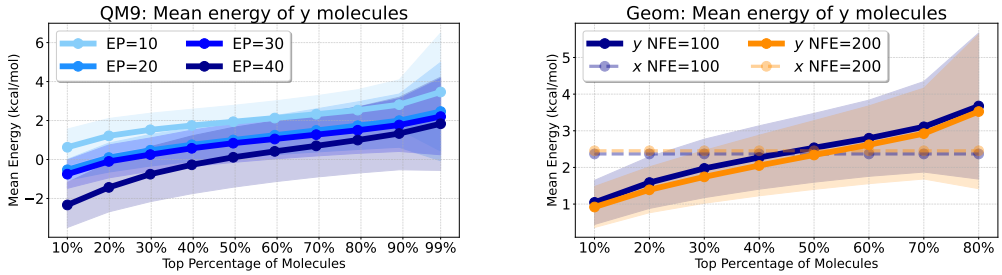

Figure 8: Figures show the energy per atom computed with MMFF94 force field on the top-k fraction of y conformers per x reference molecule generated. The trend is shown on QM9 varying the epochs (left) and GEOM Drugs varying number of inference steps (NFE) reporting trends on x and y.

Normalized energies of FlexiFlow generated conformers. In this setting, we use MMFF94 (Halgren, 1996) as the energy function for conformers, normalized by the number of atoms (kcal/mol/atom). We sampled 100 molecules with 300 conformations per molecule from both GEOM Drugs and QM9. Figures 8 show the mean conformer energy for each molecule: QM9 on the left and GEOM Drugs on the right. The x-axis in both plots reports the percentage of top-k molecules ranked by energy. For QM9, we observe that low-energy conformers are concentrated within the top 30%, with only marginal improvements beyond this range. Energies also decrease as the number of training epochs increases. For GEOM Drugs, the energies of both the representative conformer and the remaining conformers remain stable across inference steps. Since the training reference conformer x is chosen as the one closest to the average conformation within the set, its energy is expected to be near the mean. This is confirmed in the figure: the energy of x is around 2.2 kcal/mol/atom, while the energies along y range from 1 to 3 kcal/mol/atom.

D.2 ADDITIONAL ENERGIES QM9

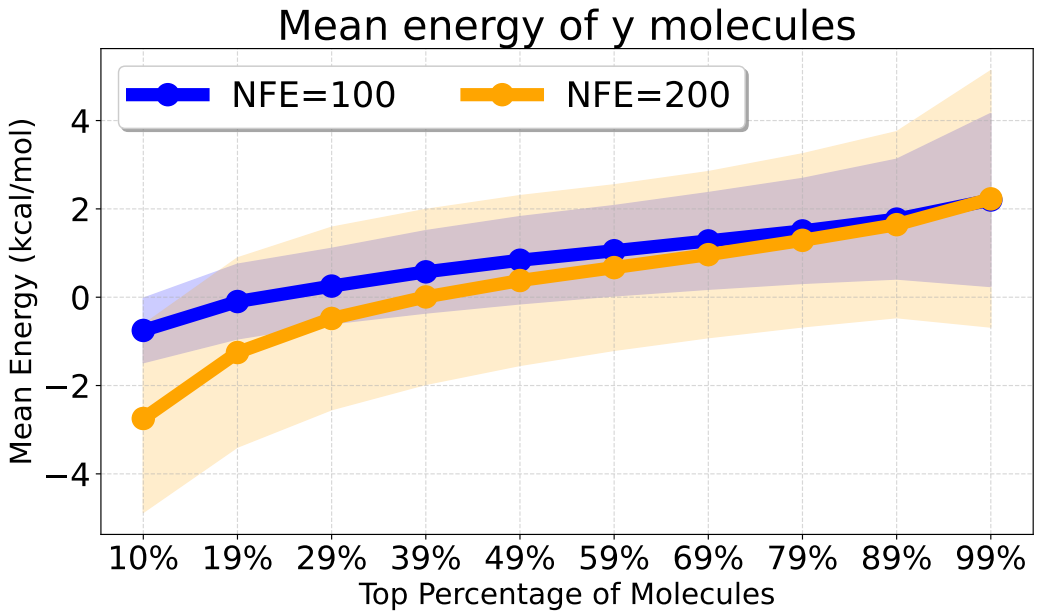

Figure 9: The conformers are sorted based on their energy values, and we plot mean and standard deviation for the top-k fraction of conformers generated. The results are reported by sampling 300 conformers per molecule on 100 molecules using the base model trained on QM9 for 40 epochs. In figure we can observe that using more denoising steps (NFE), from 100 to 200 steps, during inference leads to lower energy values. However, the cost of sampling with more steps is linearly higher.

D.3 ADDITIONAL RESULTS CONFORMERS QM9

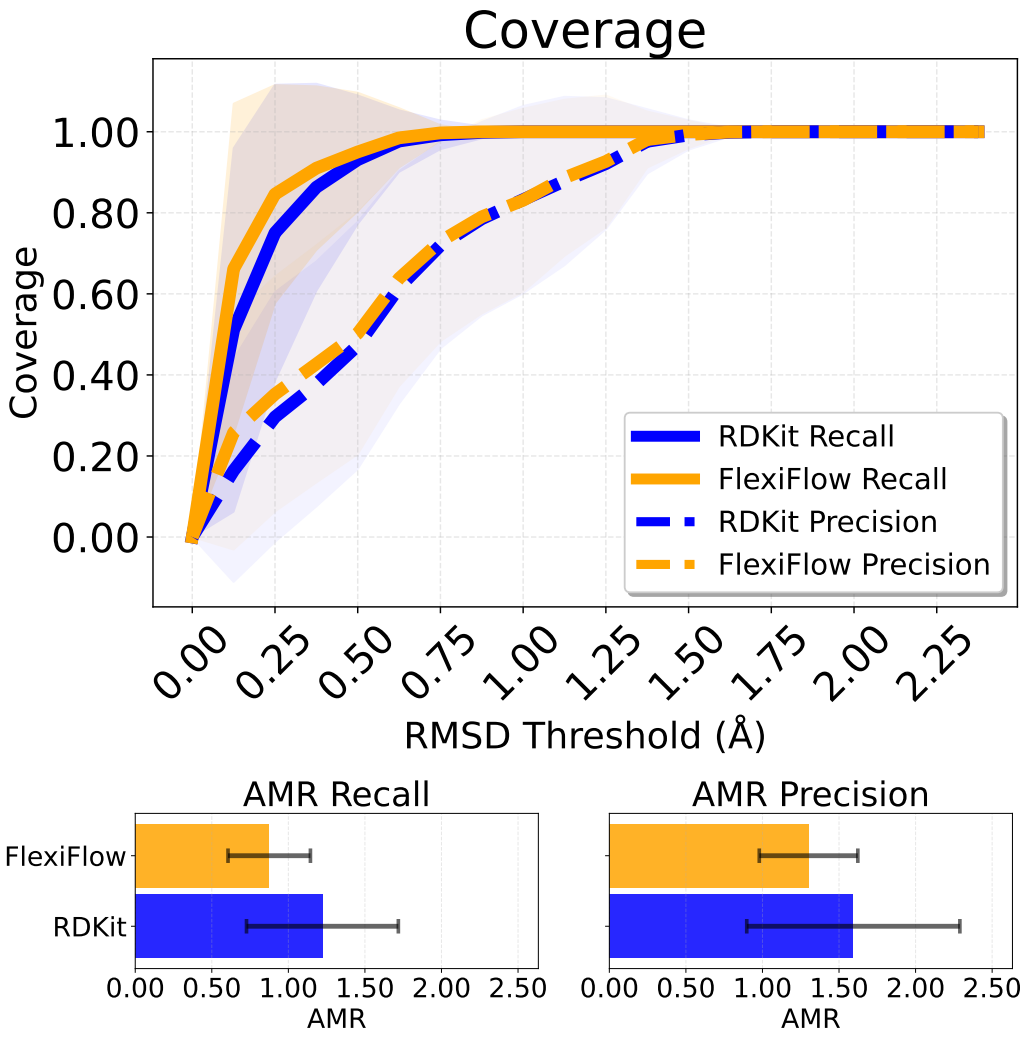

Figure 10: Coverage (top) and AMR (down) precision and recall for conformer generation on QM9. As the target data distribution of the conformers is derived from RDKit, FlexiFlow achieves comparable results to RDKit in terms of AMR and Cov, with a slight improvement. Specifically, on Coverage recall we can note that with a threshold of 0.75 Å FlexiFlow already achieves a Cov of 1.0, meaning that for all the molecules generated we can find a conformer that is at most 0.75 Å away from a CREST-reference conformer. This suggests that FlexiFlow learns the conformational distribution of the training data and is able to generate conformers that closely resemble the CREST-reference conformers. Conversely, coverage precision decreases because FlexiFlow deliberately explores a broader conformational space, causing some samples to lie farther from the CREST references. This trade-off is expected for a diversity-oriented generator. Overall, FlexiFlow yields high-quality, physically plausible conformers that remain close to reference structures while retaining diversity.

D.4 GEOM DRUGS STRATIFIED RMSD DISTRIBUTION BEFORE AND AFTER ENERGY MINIMIZATION

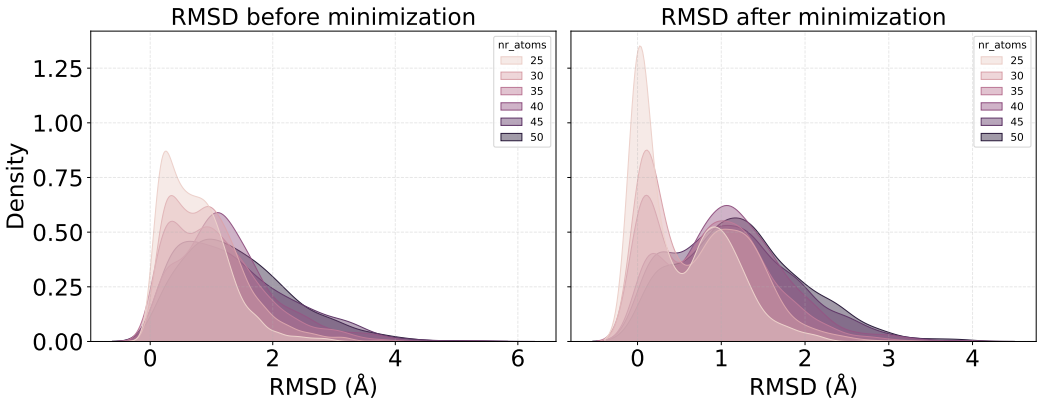

Figure 11: Results on GEOM Drugs. We use the metric in Equation 14 to test whether generated conformers, after minimization, occupy distinct local minima of the potential energy surface. For each atom-count setting, we sample 20 molecules; each yields 50 conformers. For every molecule we record the minimum pairwise RMSD (closest conformers). We then minimize each conformer for 100 steps with the MMFF94 force field (Halgren, 1996) and recompute the minimum RMSD. Results show that FlexiFlow's conformers, once minimized, fall into different local minima, indicating captured conformational diversity. The effect strengthens with molecular size: larger molecules exhibit larger pre/post minimization closest-conformer RMSD gaps.

D.5 QUALITATIVE RESULTS CONFORMERS GEOM DRUGS

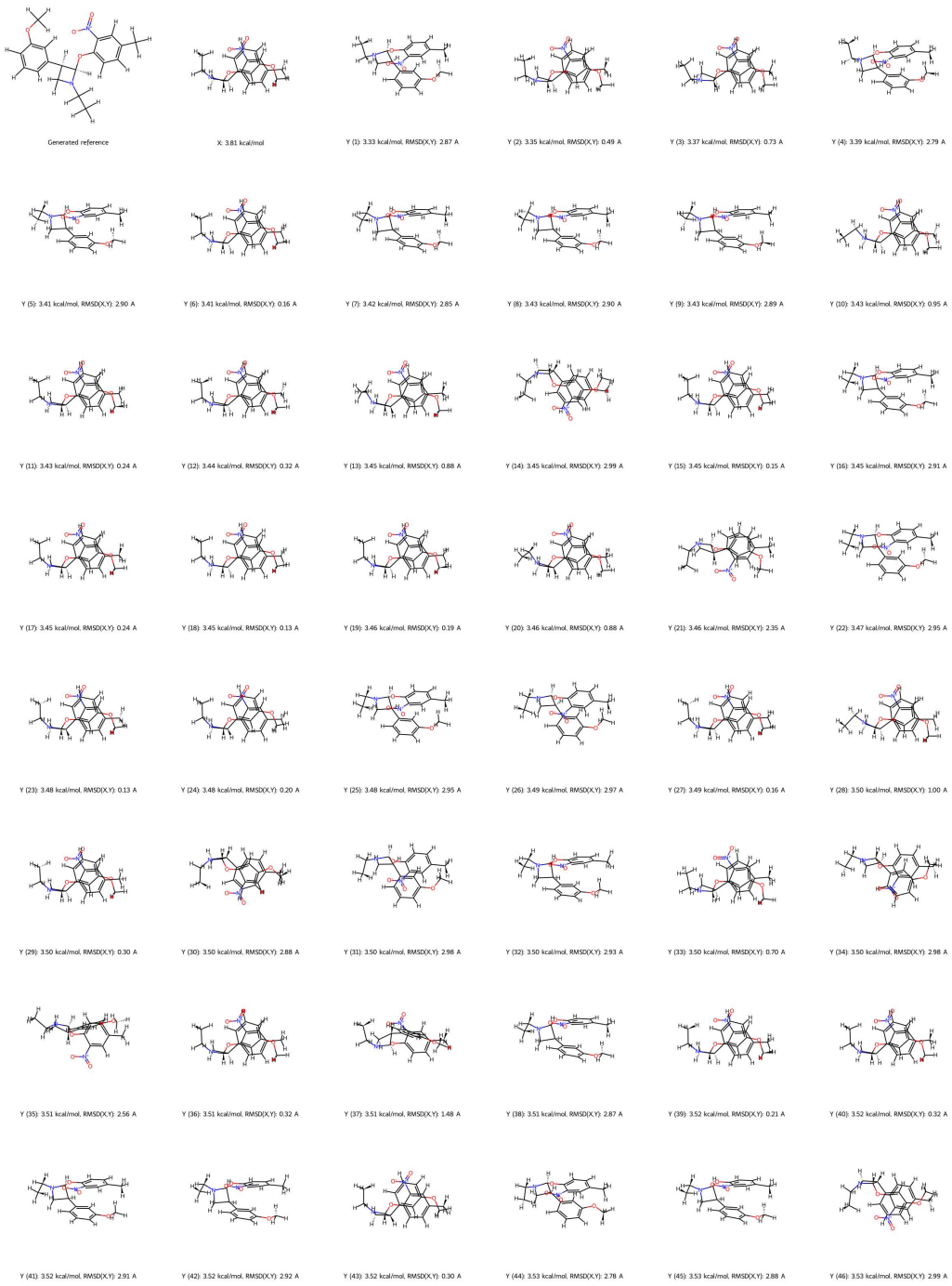

Figure 12: Qualitative results of a molecule sampled with FlexiFlow along with its generated conformers. We denote with x the target conformation, and with y_1, y_2, ..., y_{46} the generated conformers. For each molecule we report the energy state and for each y_i the RMSD with respect to x.

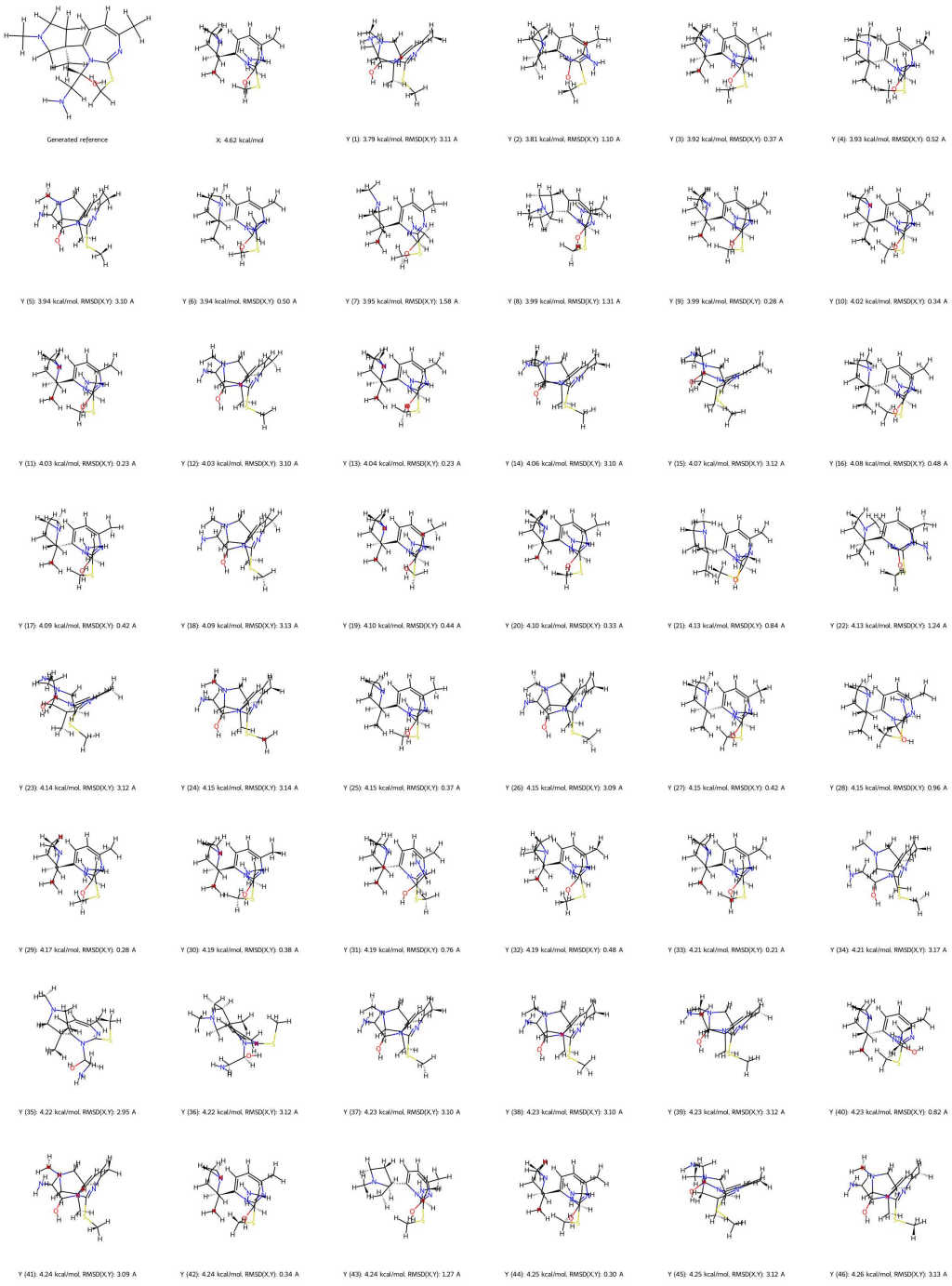

Figure 13: Qualitative results of a molecule sampled with FlexiFlow along with its generated conformers. We denote with x the target conformation, and with y_1, y_2, ..., y_{46} the generated conformers. For each molecule we report the energy state and for each y_i the RMSD with respect to x.

Figure 14: Qualitative results of a molecule sampled with FlexiFlow along with its generated conformers. We denote with x the target conformation, and with y_1, y_2, ..., y_{46} the generated conformers. For each molecule we report the energy state and for each y_i the RMSD with respect to x.

D.6 PROTEIN CONDITIONING WITH AND WITHOUT MOLECULAR FLEXIBILITY DURING TRAINING

For this experiment, we generate 10 x molecular structures along with 42 y conformers each for each testset protein in PDBBind using both models with and without GEOM Drugs training. Due to GPU vRAM memory constrains our machine could not handle more than 42 y conformers at the same time for each x with protein conditioning. We compute the MMFF strain energy for all generated conformers as the difference between the MMFF energy before and after MMFF optimization, divided by the number of atoms in the molecule. Training the model with GEOM Drugs consistently yields lower and tighter MMFF strain energy distributions than the training without it. The violin plots in Figure 15 illustrate this trend across different top-k y conformers, where the average is performed on the MMFF strain. We better quantify the gap (in terms of MMFF strain between the model with and without GEOM Drugs training) in Figure 16, where we show the difference between the two models' mean MMFF strain.

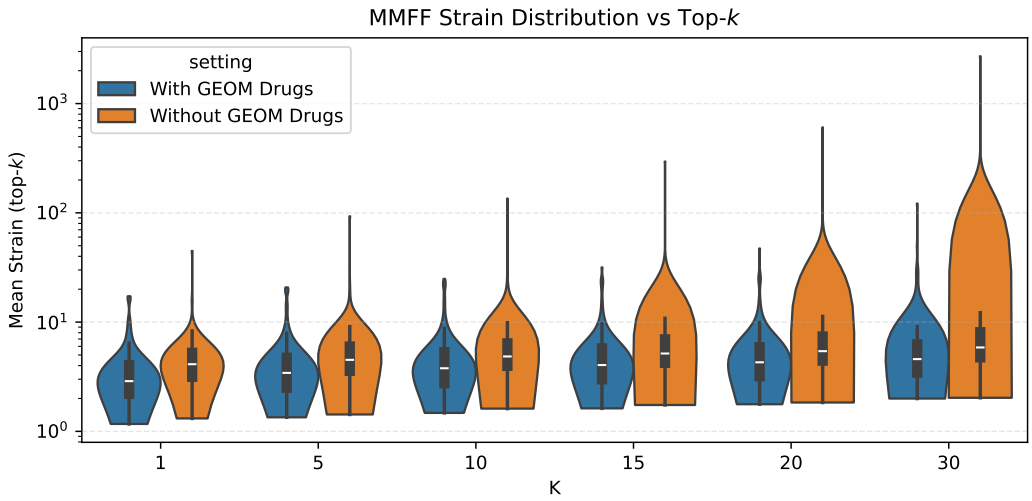

Figure 15: Violin plots report the mean MMFF strain (per-atom conformational strain = (unoptimized MMFF energy - optimized MMFF energy) / atom count) for the top-k y ligand conformations (k = 1, 5, 10, 15, 20, 30) with or without GEOM Drugs training. Boxes inside violins mark mean and interquartile range. The log y-axis highlights a heavy right tail, without GEOM Drugs exhibit progressively higher mean strain as k increases, while with GEOM Drugs maintains lower and more compact distributions across all k. This indicates including the flexibility during training (from GEOM Drugs) consistently produces lower-strain (more relaxed) top-ranked conformations. The magnitude growth of this advantage is quantified in the accompanying Delta-strain plot (Figure 16).

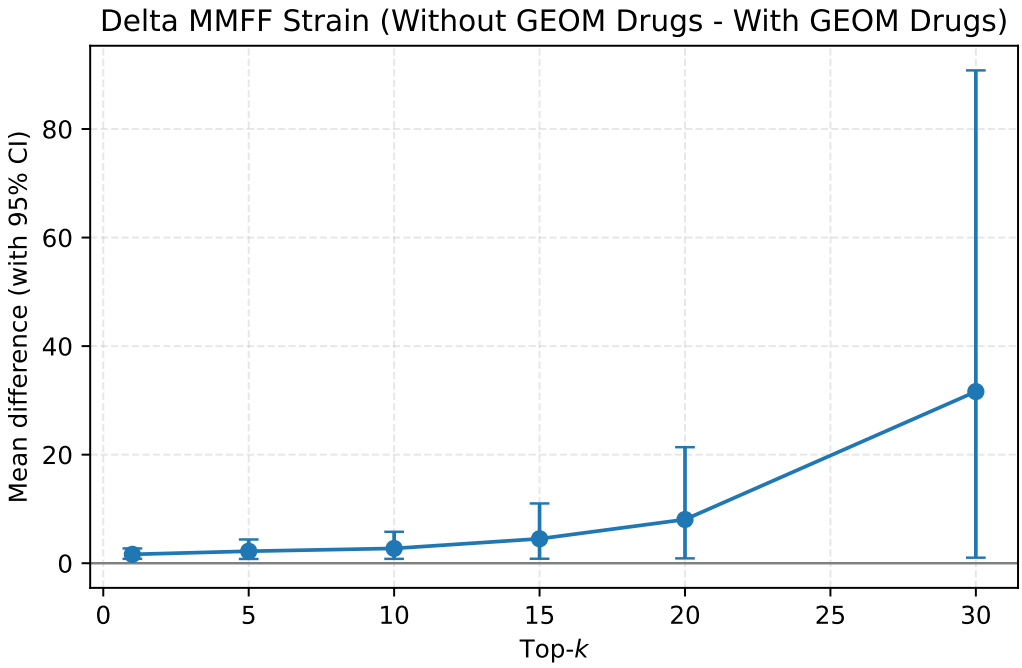

Figure 16: Figure reports the mean difference in MMFF strain between without and with GEOM Drugs during training for the top-k y ligand conformations (k = 1, 5, 10, 15, 20, 30) with 95% confidence intervals (error bars).

D.7 ADDITIONAL RESULTS ON PROTEIN CONDITIONING

Although FlexiFlow is neither trained nor explicitly conditioned to optimize Vina scores on the conformer coordinates y, some generated conformers nevertheless obtain higher Vina scores. This implies the model implicitly captures protein-ligand interaction geometry. Figures 17, 18, 19 and 20 (complex 6e5s) illustrate a qualitative example: conditioned on the protein pocket, we generate the target ligand x and multiple conformers y; three y conformers are shown alongside the reference ligand pose.

Additionally, we provide additional illustrations of other generated ligands on the test set including both target x and conformer y, see Figures 21, 22 and 23.

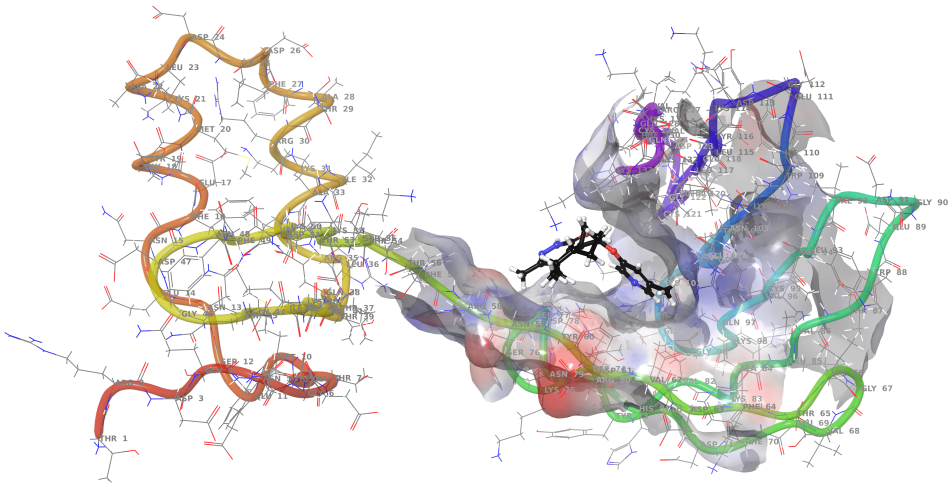

Figure 17: Binding surface electrostatic potential for complex 6e5s mapped onto the ligand's generated reference conformation x (Vina score: -4.1 kcal/mol; QED: 0.94).

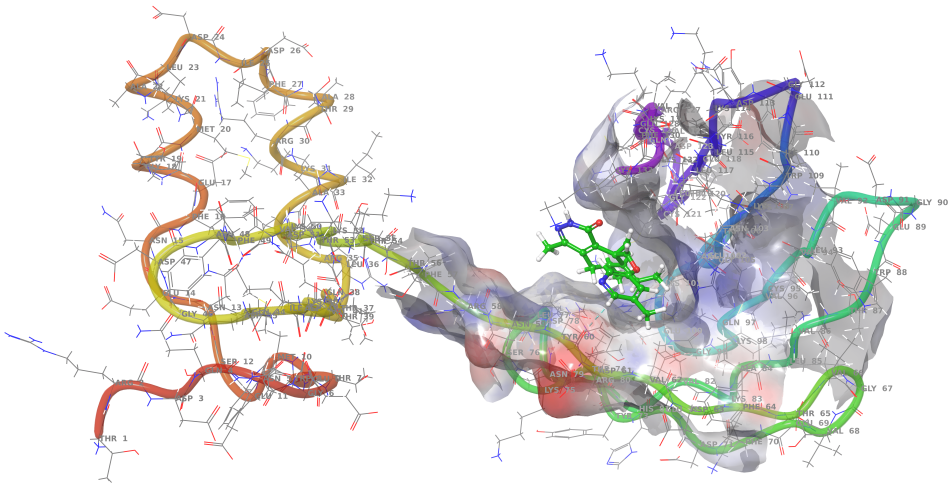

Figure 18: Binding surface electrostatic potential for complex 6e5s mapped onto the ligand's generated reference conformation y_1 (Vina score: -5.5 kcal/mol; QED: 0.94).

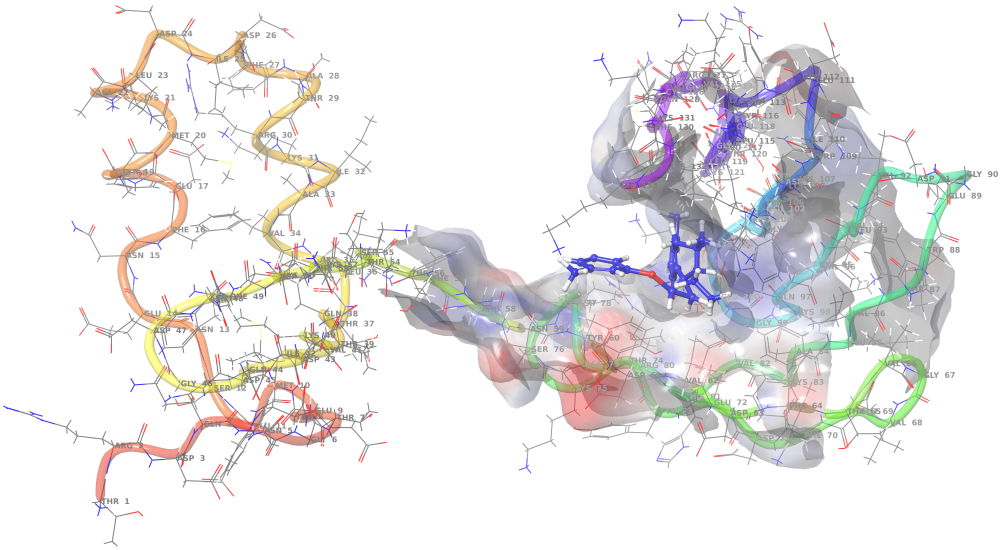

Figure 19: Binding surface electrostatic potential for complex 6e5s mapped onto the ligand's generated reference conformation y_2 (Vina score: -5.3 kcal/mol; QED: 0.94).

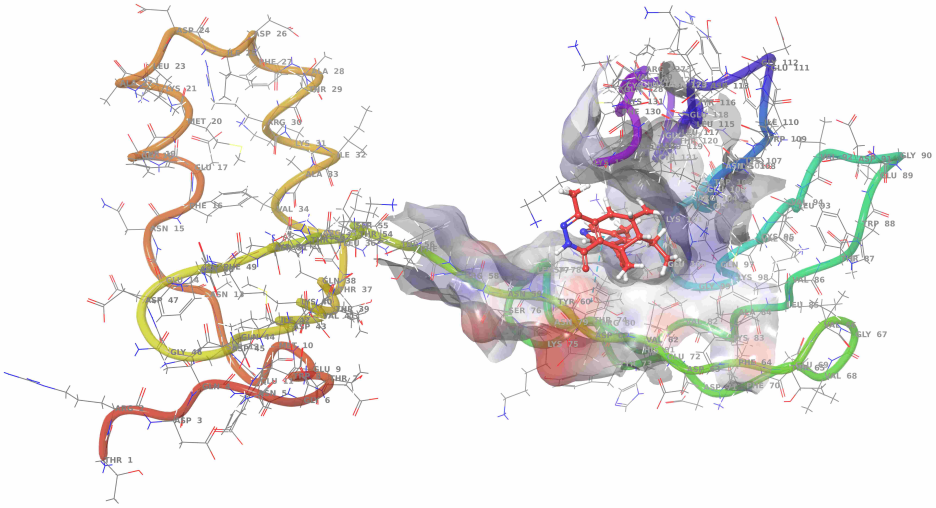

Figure 20: Binding surface electrostatic potential for complex 6e5s mapped onto the ligand's generated reference conformation y_3 (Vina score: -5.0 kcal/mol; QED: 0.94).

D.8 QUALITATIVE RESULTS ON PROTEIN CONDITIONING

.

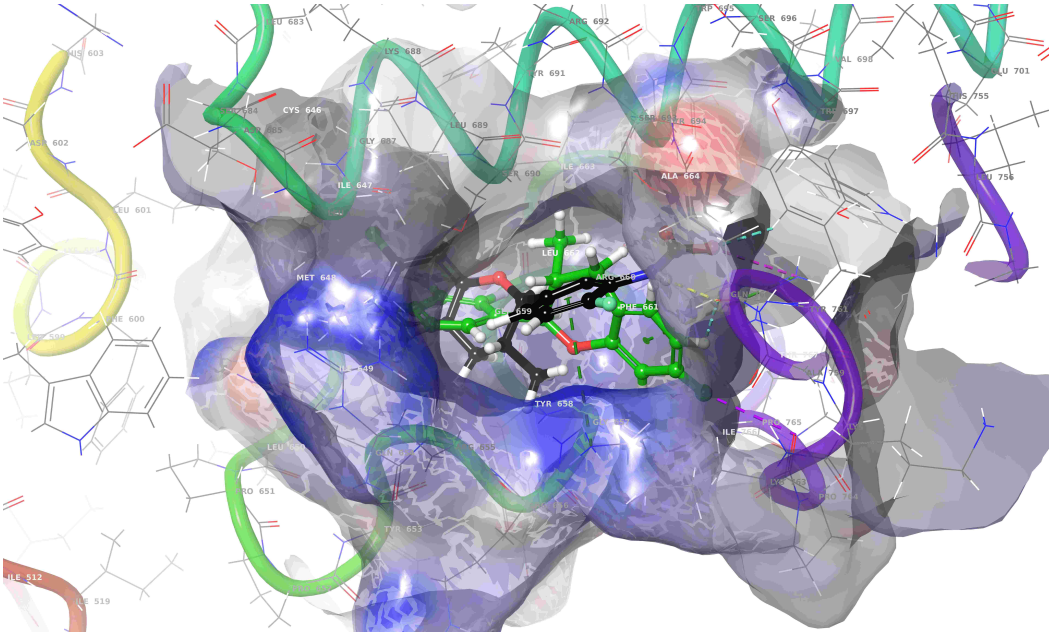

Figure 21: Conformations x (Vina score: -3.1 kcal/mol) and y (Vina score: -6.0 kcal/mol) for the complex 6oin (QED ligand: 0.91).

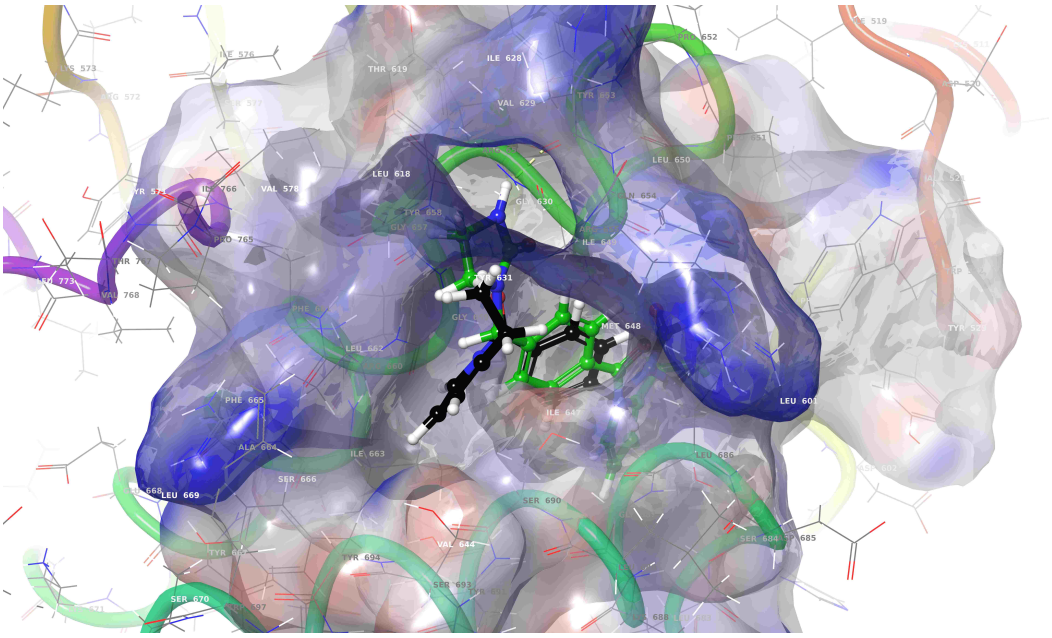

Figure 22: Conformations x (Vina score: -2.3 kcal/mol) and y (Vina score: -4.8 kcal/mol) for the complex 6oiq (QED ligand: 0.94).

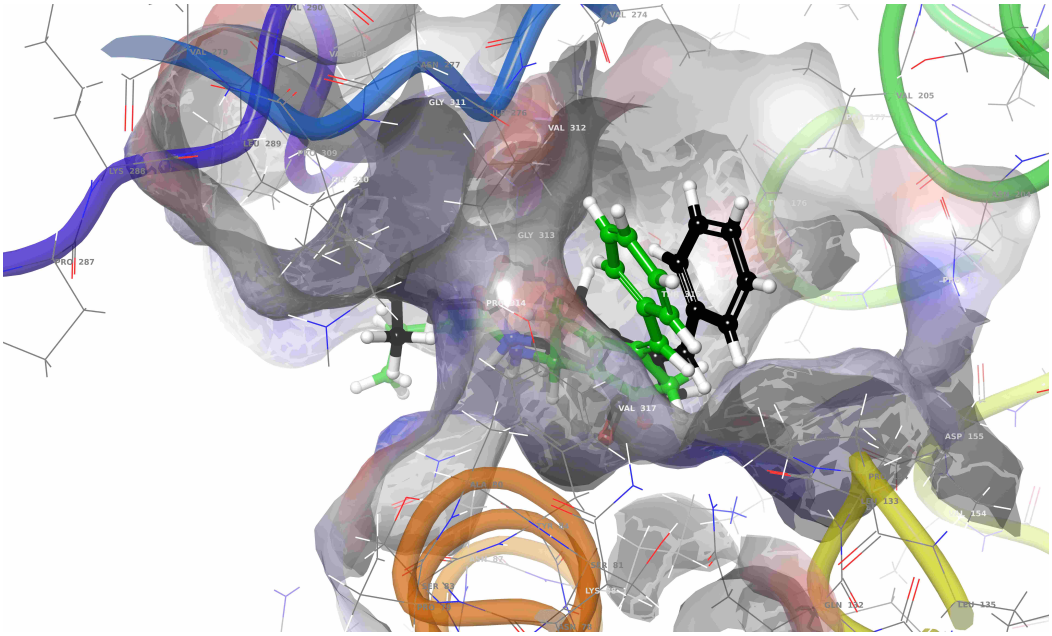

Figure 23: Conformations x (Vina score: -4.5 kcal/mol) and y (Vina score: -4.6 kcal/mol) for the complex 6jib (QED ligand: 0.92).

E FLOW DECOMPOSITION ON MNIST

E.1 DATA PROCESSING & TRAINING SETUP

Since MNIST does not have implicitly colored images for the digits, we color the digits using the following procedure. First, we uniformly sample a color among red, green and blue and added some white noise over it. Then, this is multiplied by the grayscale value of the digit, so that the background remains black.

In order to independently sample the color based on the digit x that is being generated, we use the sum between the grayscale digit x and RGB color y to constrain the generation of a colored digit z, i.e. $z = x + y$. In this way, the model can learn to generate both the grayscale digit and the color independently (see Figure 24).

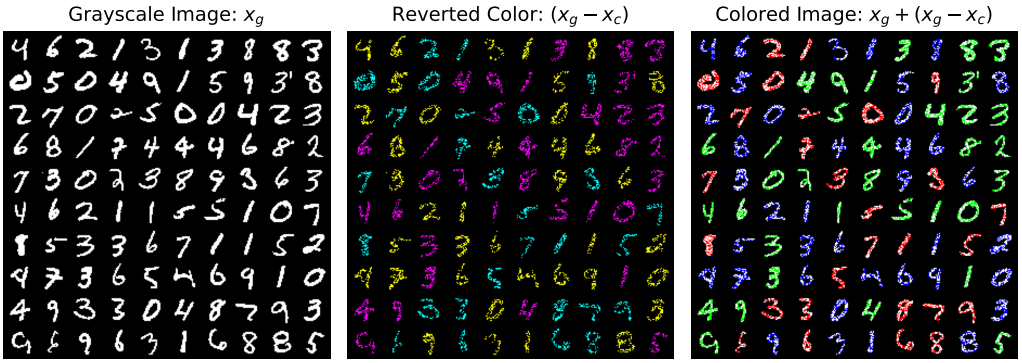

Figure 24: Data processing steps used to derive the colored digit $x + y$ from the grayscale digit $x = x_g$ and the $y = x_g - x_c$ vector field.

E.2 DUAL-UNET MNIST ARCHITECTURE

The Dual-Unet architecture $\mathcal{F}_{\text{Dual-Unet}}(x_t^g, x_t^y, t) = (s_g, s_c)$ is a modified U-Net that takes as input the noisy grayscale image $x_t^g \in \mathbb{R}^{1\times H\times W}$, and noisy reversed color image $x_t^y \in \mathbb{R}^{3\times H\times W}$ and time t, and outputs the score functions s_g and s_y for the grayscale and reversed color images (rescaled by a factor 0.001). In this setting, we train the network to directly estimate the target data distribution from the noisy inputs.

To obtain x_t^g and x_t^y, we sample a x_0^g and x_0^y independently from a $\mathcal{N}(0, I)$ distribution, $t \sim \text{Uniform}(0, 1)$ and we use the following interpolation scheme: $x_t^g = (1-t)x_0^g + tx_1^g$ and $x_t^y = (1-t)x_0^y + tx_1^y$, where x_1^g and x_1^y are the grayscale and reversed colored images from the training set, respectively. During training we minimize the objective in Equation 9.

In the architecture, x_t^g and x_t^y are flatten and process by $x_{g,0} = \text{Conv}_g(x_t^g) \in \mathbb{R}^{D\times H\times W}$, for the colored counter part $x_{y,0} = \text{Conv}_c(x_t^y) \in \mathbb{R}^{3D\times H\times W}$ and $\tau(t) = \text{MLP}(\text{SinusoidalEmbed}(t)) \in \mathbb{R}^{4D}$ for the time component, where Conv stands for convolutional layer. We apply L encoder layers $\mathbf{E}_i$, $i \in 1, \dots, L$:

$$h_{g,i}, x_{g,i} = \mathbf{E}_{g,i}(x_{g,(i-1)}, \tau) \quad h_{y,i}, x_{y,i} = \mathbf{E}_{y,i}(x_{y,(i-1)}, \tau) \tag{77}$$

where $\mathbf{E}_{g,i}$ and for the colored $\mathbf{E}_{y,i}$ are encoder blocks with ResNet and downsampling blocks, producing features $x_{g,L}$, $x_{y,L}$ and skip connections $h_{g,L}$, $h_{y,L}$. We finally concatenate the colored to the grayscale bottleneck features and from these start decoding the images $x_{g,m} = \mathcal{M}_g(x_{g,L}, \tau)$ and $x_{y,m} = \mathcal{M}_c([x_{y,L}, x_{g,m}], \tau)$ where $[\cdot, \cdot]$ denotes channel-wise concatenation. We reverse the process with a decoder $\mathbf{D}_i$ for $i \in L, \dots, 1$ layers for both $x_{g,m}$ and $x_{y,m}$:

$$x_i^u = \mathbf{D}_{g,i}([x_{g,(i+1)}^u, h_{g,i}], \tau) \quad x_{y,i}^u = \mathbf{D}_{y,i}([x_{y,(i+1)}^u, h_{y,i}, h_{g,i}], \tau) \tag{78}$$

where $\mathbf{D}_{g,i}$ and $\mathbf{D}_{y,i}$ are decoder blocks with ResNet and upsampling blocks. The output is the result of a ResBlock and final convolution

$$s_g = \text{Conv-l}_g(\text{ResBlock}_g([x_{g,1}^u, x_{g,0}], \tau)), \; s_c = \text{Conv-l}_y(\text{ResBlock}_y([x_{y,1}^u, x_{y,0}, x_{g,0}], \tau)) \tag{79}$$

E.3 MNIST INFERENCE

At inference time, we sample $\hat{x}_0^g$ and $\hat{x}_0^y$ independently from a Gaussian distribution $\mathcal{N}(0, I)$, and we use the euler ODE solver to integrate both vector fields from $t = 0$ to $t = 1$ independently. Finally, we obtain the grayscale image $\hat{x}_1^g$ and reversed color image $\hat{x}_1^c$, to obtain the final colored image we simply apply the operation we show in Figure 24, $\hat{x}_1^g$ + ($\hat{x}_1^g$ - $\hat{x}_1^y$).

E.4 QUALITATIVE RESULTS ON MNIST

For Figure 25, the noise for x_g is held fixed while varying the noise for x_c, producing distinct color textures conditioned on an unchanged grayscale digit structure. Figure 26 provides results varying the noise for both x_g and x_c yields diverse grayscale digit shapes together with corresponding variations in color texture. This demonstrates that FlexiFlow can independently modulate structure (grayscale) and appearance (color or texture).

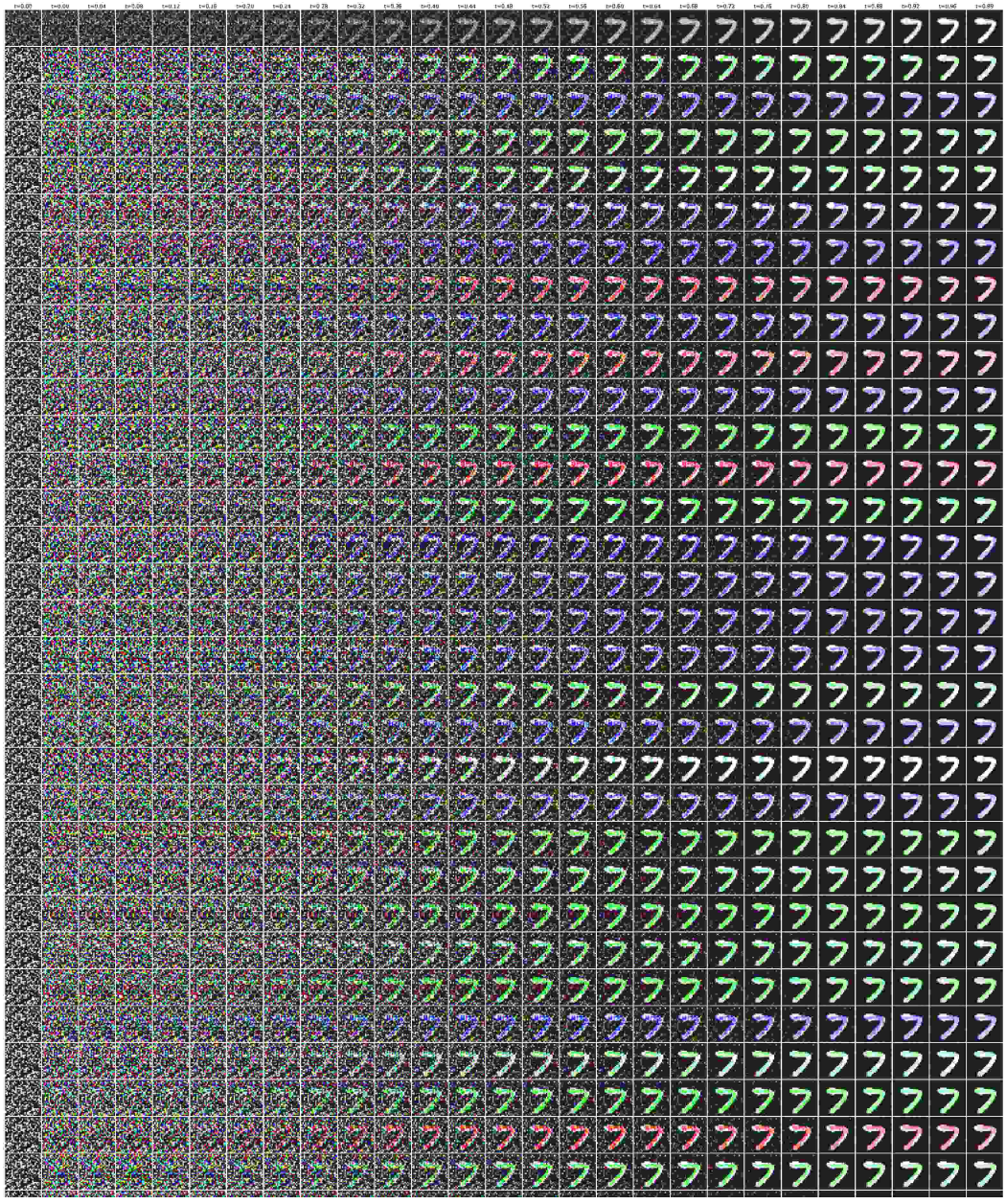

Figure 25: Integration of different x_c noise for a fixed x_g noise, where we can observe that FlexiFlow is able to generate different color textures for the same grayscale image. The x_g reference is shown in the top row, while all the other rows show different x_c samples.

Figure 26: Integration of different x_g and x_c noise, where we can observe that FlexiFlow is able to generate diverse grayscale images along with different color textures. The x_g reference is shown every other row, while all the other rows show different x_c samples given the same x_g noise shown on top.

F TRAINING DATASET ENERGIES STATISTICS ON GEOM DRUGS CONSTRUCTED (X, Y) PAIRS

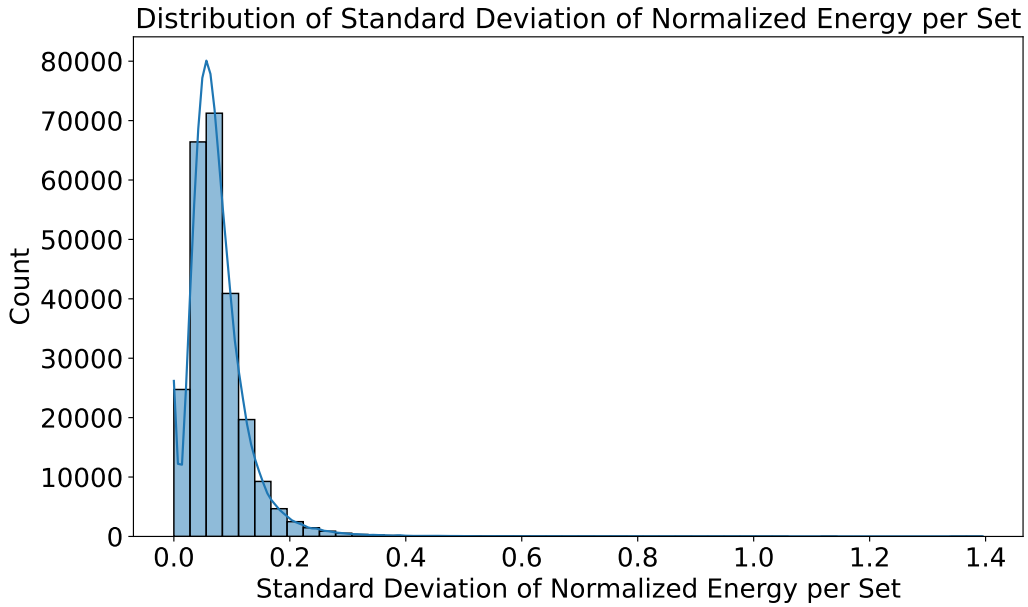

Figure 27: Standard deviations of the normalized energies (computed as described in Appendix D.1) of the molecules within each set of conformers $\mathcal{S}$. This plot shows that all the conformations that we have in GEOM Drugs, in terms of energy, are almost at the energy minima configuration.

Energy Minima Location	**Count**
y	224265
x	19425

Table 5: Count of how many times the energy minima conformation is located in x or y by selecting the closest conformer to the average as reference.

Energy difference	**Count**
$y < x$	224265
$y > x$	15378
$x = y$	4047

Table 6: Times the energy minima conformation in x or y is lower, or equal, by selecting the closest conformer to the average as reference.