# PropMolFlow: Property-Guided Molecule Generation with Geometry-Complete Flow Matching

Cheng Zeng[1,2, 11], Jirui Jin [1,2, 11], Connor Ambrose[1,2], George Karypis[3], Mark Transtrum[4], Ellad B. Tadmor[5], Richard G. Hennig[2, 6], Adrian Roitberg[1,2], Stefano Martiniani[7,8,9,10,*], and Mingjie Liu[1,2,*]

[1]Department of Chemistry, University of Florida, Gainesville, FL 32611, USA
[2]Quantum Theory Project, University of Florida, Gainesville, FL 32611, USA
[3]Department of Computer Science & Engineering, University of Minnesota, Minneapolis, MN 55455, USA
[4]Department of Physics & Astronomy, Brigham Young University, Provo, UT 84602, USA
[5]Department of Aerospace Engineering and Mechanics, University of Minnesota, Minneapolis, MN 55455, USA
[6]Department of Materials Science & Engineering, University of Florida, Gainesville, FL 32611, USA
[7]Center for Soft Matter Research, Department of Physics, New York University, New York 10003, USA
[8]Simons Center for Computational Physical Chemistry, Department of Chemistry, New York University, New York 10003, USA
[9]Courant Institute of Mathematical Sciences, New York University, New York 10003, USA
[10]Center for Neural Science, New York University, New York 10003, USA
[11]These authors contribute equally: Cheng Zeng, Jirui Jin
[*]e-mail: sm7683@nyu.edu, mingjieliu@ufl.edu

December 16, 2025

## Abstract

Molecule generation is advancing rapidly in chemical discovery and drug design. Flow matching methods have recently set the state of the art (SOTA) in unconditional molecule generation, surpassing score-based diffusion models. However, diffusion models still lead in property-guided generation. In this work, we introduce PropMolFlow, an approach for property-guided molecule generation based on geometry-complete SE(3)-equivariant flow matching. Integrating five different property embedding methods with a Gaussian expansion of scalar properties, PropMolFlow achieves competitive performance against previous SOTA diffusion models in conditional molecule generation while maintaining high structural stability and validity. Additionally, it enables faster sampling speed with fewer time steps compared to baseline models. We highlight the importance of validating the properties of generated molecules through DFT calculations. Furthermore, we introduce a task to assess the model's ability to propose molecules with underrepresented property values, assessing its capacity for out-of-distribution generalization.

## Introduction

Deep generative models are promising to accelerate chemical discovery by statistically sampling molecular structures, reducing reliance on costly physics-based simulations [1]. State-of-the-art (SOTA) 3D molecule generation primarily uses diffusion with equivariant graph neural networks (EGNN), enabling accurate sampling of molecular geometries. [2, 3, 4]. Flow matching is an emerging alternative to diffusion models [5, 6, 7], offering flexibility over priors and probability paths while improving sampling efficiency across materials [8], proteins [9], and small molecules [10].

Seminal diffusion [2, 4] and flow-matching models [11] treat molecules as point clouds with continuous representations for discrete molecular modalities, and parameterize the generative process via E(3)-EGNN. However, continuous encodings disrespect inherently discrete features, such as atom types, and E(3) models cannot capture chirality [12]. Recent studies also found bond orders in molecular graphs to be important in improving validity of generated molecules [13, 10]. FlowMol [10] achieves SOTA unconditional molecule generation, by using geometry-complete SE(3) generative processes, explicit bond-order modeling, and discrete flow matching for atom types, formal charges and bond orders, though its extension to property-guided generation remains unaddressed.

Property-guided molecule generation aims to design molecules that satisfy target properties, often by concatenating property values with node features in molecular graphs [2, 4, 14] — a strategy mainly explored on the single-molecule QM9 data [15, 16]. Although effective, this approach may oversimplify how properties interact with molecular structures; for instance, Gebauer et al. [17] suggest Gaussian expansions can extract property information more robustly. Systematic studies of property embedding methods are lacking, and optimal strategies may vary by properties. To validate properties of generated molecules, prior methods use separate property predictors trained on the fully relaxed QM9 molecules, which may struggle to make accurate predictions on non-relaxed, generated molecules. Moreover, property-guided generation is typically evaluated in an 'in-distribution' setting, where property values and atom counts sampled within the QM9 distribution are used to generate molecules [2]. Chemical discovery requires generating molecules with underrepresented or out-of-distribution (OOD) property values.

We introduce PropMolFlow, a property-guided molecule generation framework that integrates various property embedding methods with an SE(3)-equivariant flow-matching process [10]. We evaluate PropMolFlow on QM9, demonstrating its competitive performance versus previous methods, while achieving faster inference. Besides, we propose an out-of-distribution generation task, and validate properties of generated molecules using extensive density functional theory (DFT) calculations. Finally, we introduce the close-shell ratio and revised stability metrics to complement existing metrics and ensure more comprehensive evaluations for molecule generation.

## Results and Discussion

### Overview of PropMolFlow

PropMolFlow builds on top of the FlowMol architecture. [10] The model generates samples by integrating over NN-parameterized conditional velocity fields [6, 7, 5]. An SE(3)-EGNN based on geometric vector perceptrons (GVP) [18] is used and molecules are represented by fully-connected graphs (Figure 1a). The generation is achieved by interacting property embeddings and node scalar features. Property embeddings are constructed by a non-trainable Gaussian expansion, followed with a shallow multilayer perceptron (MLP). This Gaussian expansion is optional, and its utility depends on property types and evaluation tasks. Once trained, the NN defines a joint flow matching that denoises the molecular graph by simultaneously updating all modalities—such as atom types, bond orders, and 3D coordinates—to generate molecules (Figure 1b). Details of flow matching and neural networks are provided in the 'Joint Flow Matching' and 'Model Architecture' sections.

Five embedding methods and usage of Gaussian expansion are explored (Figure 1c,d). We evaluate the property-guided generation over six molecular properties in QM9. Details for property embeddings are provided in the 'Property Embedding Operations' and 'Gaussian Expansion' sections. PropMolFlow is trained on a revised QM9 SDF data with explicit hydrogen atoms, bond orders and formal charges [15, 16]. This revised SDF file fixed the bond and charge inconsistencies of the original SDF file distributed by DeepChem [16]. Data details are provided in the 'Data' section.

### Competitive property-guided generation with PropMolFlow

We evaluated PropMolFlow on its ability to generate molecules with target properties, molecular structural validity, and inference speeds. These assessments were conducted under the in-distribution (ID) task. This task assesses the model's capability to generate structures spanning a spectrum of joint target properties and atom counts within data distributions and has been adopted in prior studies [4, 11, 14, 19]. Generative models sample molecules by property inputs, and a separate property predictor evaluates properties of generated molecules. Predicted values are compared to input values, and the mean absolute error (MAE) was used as the evaluation metric. A lower MAE indicates better model performance. Further details of how to sample molecules and GVP regressors are provided in the 'Inference' and 'GVP regressor details' sections.

We compared PropMolFlow with five baseline models, including EEGSDE [19], EquiFM [11], GeoLDM [4], GCDM [14] and JODO [20], all trained on QM9. Two intuitive baselines are also included to represent the respective upper and lower bounds of MAEs, namely "Random (Upper-bound)" and "QM9 (Lower-bound)", along with another baseline "# Atoms" (see the 'Baselines' section for more details).

Overall, PropMolFlow achieves competitive performance against the SOTA models (Table 1). It shows the lowest MAE for $\alpha$ and $\mu$, and comparable performance for $C_v$ and $\epsilon_{\text{HOMO}}$ versus the SOTA JODO, and the second best performance for $\epsilon_{\text{LUMO}}$ and $\Delta\epsilon$. The superior performance of PropMolFlow benefits from the thorough exploration of property-dependent optimal embedding methods (Supplementary Table 4 and Supplementary Section 2.2). For instance,

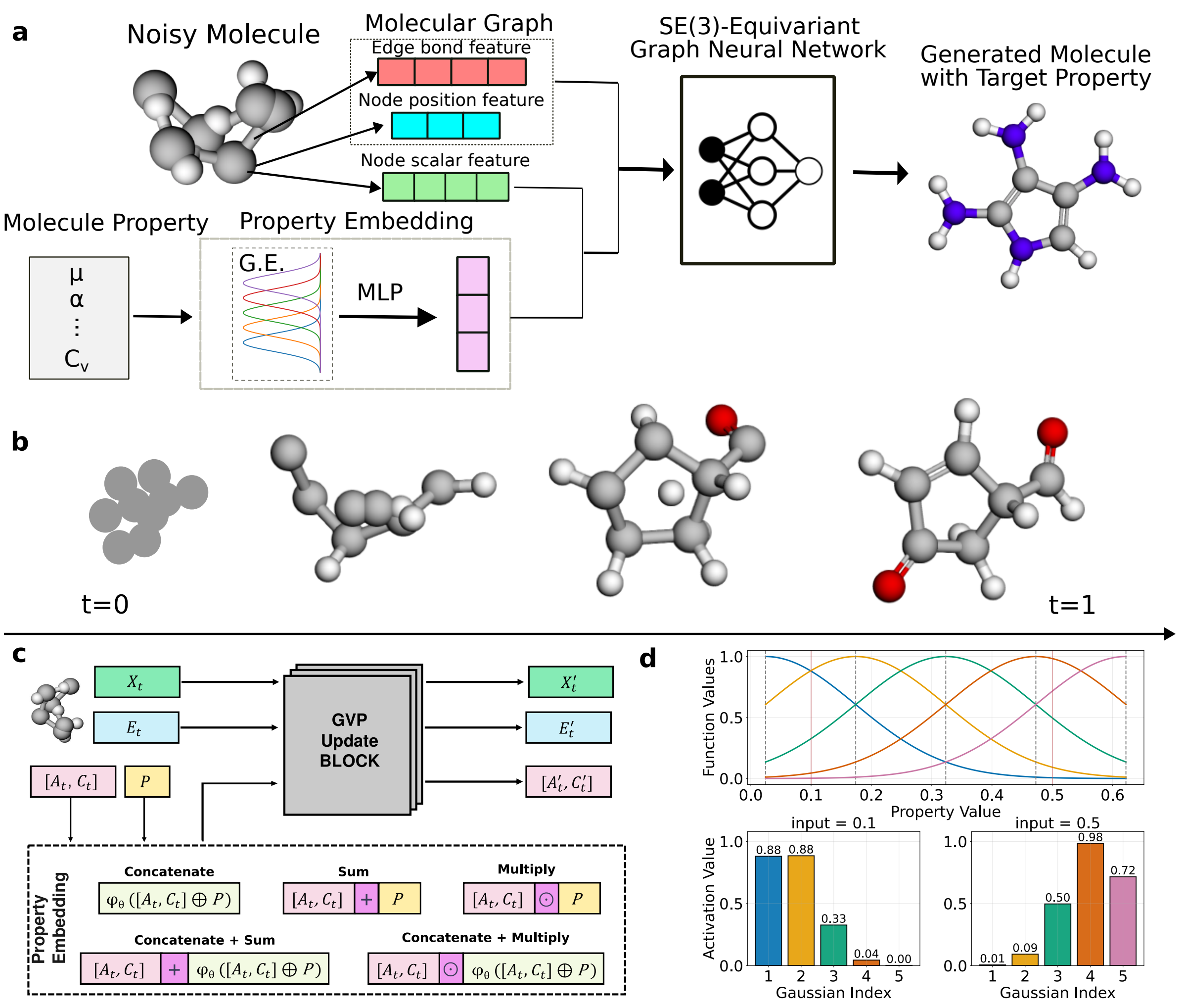

Figure 1: Overview of the PropMolFlow methodology. **a**, PropMolFlow models are jointly trained on a molecular graph and property embedding. A molecular graph includes node scalar features, node position features and edge bond features. A property embedding comprises an optional Gaussian expansion mapping (G.E.) followed by a multilayer perceptron (MLP) that projects a scalar property to a high-dimensional embedding space. Conditioning on the property is achieved by the interaction between the property embedding and node scalar features. **b**, A joint flow matching process is used to generate different molecule modalities together. **c**, Five interaction types between a property embedding ‘P’ and a molecular graph. Node scalar features $[A_t, C_t]$ include atom type $A$ and formal charge $C$ at time $t$. An MLP transformation $\varphi_\theta$ is applied to convert the dimension back to that of the original $[A_t, C_t]$, where necessary. ‘$\odot$’ represents an element-wise Hadamard product, and ‘$\oplus$’ indicates a ‘Concatenate’ operation. $E_t$ and $X_t$ represent respective bond edge features and node position features. **d**, Gaussian expansion as augmented property embedding. Curves in the top panel correspond to five Gaussian basis functions that are evenly spaced between the minimum and maximum property values. Centers of Gaussians are marked by gray dashed lines, and the red solid lines represent two example inputs. The bottom panels show the function values for each Gaussian for two inputs. In a molecule configuration, white, gray, red, and blue indicate H, C, O, and N, respectively.

Table 1: **Performance of PropMolFlow with respect to property alignment.** Results for the baseline models use an equivariant graph neural network (EGNN) as the regressor, whereas PropMolFlow results use our pretrained SE(3) GVP regressor as well as an EGNN regressor. JODO results were reported on our sampled molecules using publicly available model checkpoints.

| Property | $\alpha$ | $\Delta\epsilon$ | $\epsilon_{\mathrm{HOMO}}$ | $\epsilon_{\mathrm{LUMO}}$ | $\mu$ | $C_v$ |
|---|---|---|---|---|---|---|
| Units | Bohr$^3$ | meV | meV | meV | Debye | cal/(mol·K) |
| QM9 (Lower-bound) | 0.10 | 64 | 39 | 36 | 0.043 | 0.040 |
| Random (Upper-bound) | 9.01 | 1470 | 645 | 1457 | 1.616 | 6.857 |
| # Atoms | 3.86 | 866 | 426 | 813 | 1.053 | 1.971 |
| EEGSDE | 2.62 | 542 | 302 | 496 | 0.858 | 1.037 |
| EquiFM | 2.41 | 591 | 337 | 530 | 1.106 | 1.033 |
| GeoLDM | 2.37 | 587 | 340 | 522 | 1.108 | 1.025 |
| GCDM | 1.97 | 602 | 344 | 479 | 0.844 | 0.689 |
| JODO | 1.44 | 333 | 231 | 260 | 0.620 | 0.580 |
| PropMolFlow, GVP | 1.31 | 391 | 254 | 315 | 0.620 | 0.626 |
| PropMolFlow, EGNN | 1.36 | 391 | 246 | 312 | 0.632 | 0.640 |

the MAE for $\Delta\epsilon$ can vary by more than 40%, from 391 meV using a 'Concatenate_Sum' embedding to 551 meV using a 'Concatenate' embedding, both without Gaussian expansions. To avoid bias of comparison using different predictors, we also report MAEs using EGNN predictors released by Bao et al. [19] in Table 1. It confirms that different predictors produce consistent MAEs.

## Rapid molecule generation with high structural fidelity

Structural validity and inference speeds are critical in success of chemical discovery with generative models. Structural validity refers to whether generated molecules conform to chemical rules. This was measured using five metrics, including atomic stability, molecule stability, RDKit validity [21], 'uniqueness and validity', PoseBusters validity [22], and closed-shell ratio. Inference speeds are evaluated by the wall-clock time for generating 10,000 molecules. Details of structural metrics and computational settings for sampling are provided in respective 'Evaluation metrics' and 'Computational settings' section.

PropMolFlow consistently outperforms baseline models across all structural metrics, except for closed-shell ratios, where it slightly underperforms JODO (Fig. 2a–e and Supplementary Table. 5). Flow matching is normally much faster than diffusion models due to its shorter, deterministic probability paths and optimal transport (Supplementary Section 3) [11]. PropMolFlow requires only 100 time steps against 1000 steps for diffusion models, hence a speedup at least $8\times$ over diffusion-based models and nearly $2\times$ faster than EquiFM (Fig. 2f). We also report per-property comparisons in Supplementary Section 6. To further assess novelty of generated molecules, Supplementary Figure 6 shows the maximum Tanimoto similarity of generated molecules to the training set using Morgan fingerprints [23]. Applying a cutoff of 0.8, novelty ratios are 66–72% across all properties, indicating the model's ability to produce a substantial number of previously unseen molecules.

## Systematic inductive bias of property predictors

A property predictor is needed to evaluate the generated molecules. Typically, this predictor shares the same architecture as the generative model, which may introduce inductive biases in predictions. In this work, DFT calculations were carried out on selected and filtered molecules and compared to input ('Target') and GVP-predicted values, without and with ('-R') structural relaxations. For example, we show orders of pairwise MAE distances for $\alpha$ in Fig. 3a. We show results for $\alpha$, $\Delta\epsilon$ and $C_v$ in Fig. 3b–d, and for the other properties in Supplementary Figure 7, which are similar to $\Delta\epsilon$. DFT settings are included in the 'DFT' section, the filtering procedure are provided in the 'Molecule filtering' section, and numbers of filtered molecules are listed in Supplementary Table 14.

First, we examine GVP on the raw generated molecules (Fig. 3b). MAEs of "Target vs. DFT" are comparable to "Target vs. GVP" for $\alpha$ and $\Delta\epsilon$, suggesting that GVP predictors are reliable for estimating "Target vs. DFT" MAEs. However, GVP consistently underestimates the MAEs, revealing an inductive bias. For $C_v$, the "Target vs. DFT" MAE is much larger than "Target vs. GVP". This is because DFT-computed $C_v$ depends on vibrational frequencies, which are highly sensitive to geometry. Relaxation resolves this issue, reducing the MAE to 0.68 cal/(mol $\cdot$ K), still higher than, but closer to "Target vs. GVP" (0.61). For chemical discovery, it is also important to relax the structures. Fig. 3c shows a much closer agreement between DFT and GVP after relaxation for $C_v$, while a notable offset is observed for $\alpha$. It

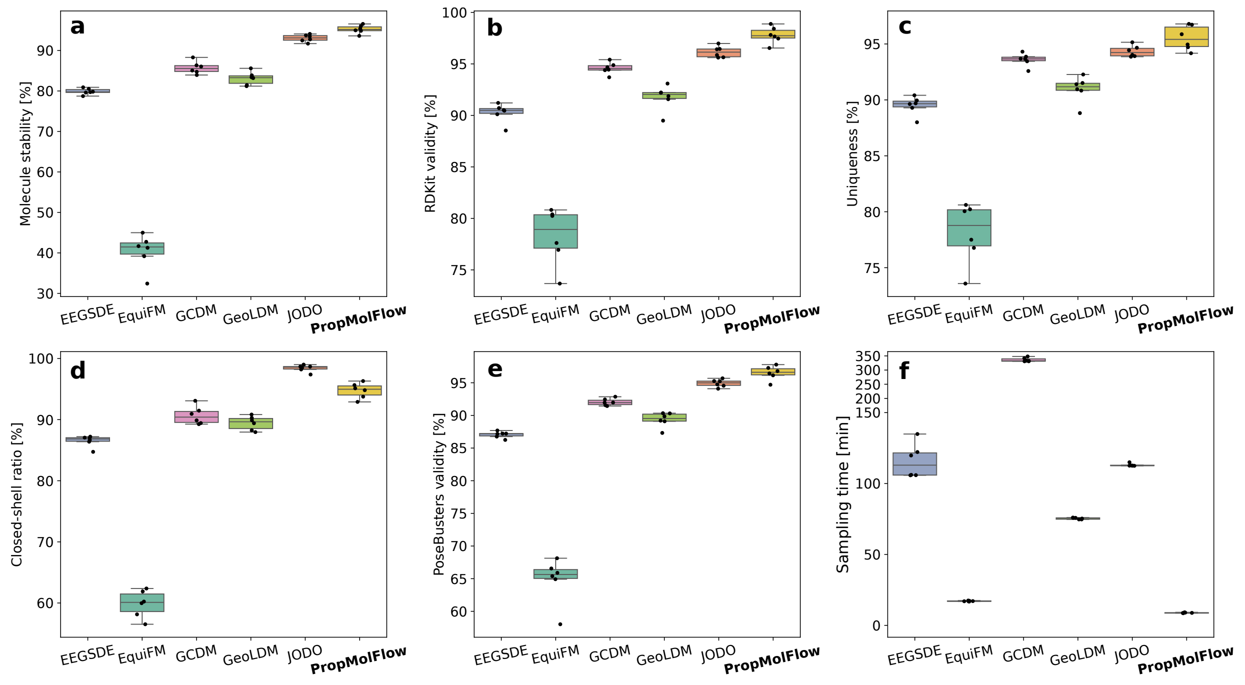

Figure 2: **Chemical validity and sampling efficiency of PropMolFlow against five baseline models. a,** Molecule stability. **b,** RDKit validity. **c,** Uniqueness. **d,** Closed-shell ratio. **e,** PoseBusters validity. **f,** Sampling time. The y-axis for sampling time uses a broken scale to expand 0–150 min and compress 150–360 min by a ratio of 5 for visual clarity. Each box plot summarizes the metric values computed for six molecular properties ($n = 6$) and 10,000 sampled molecules. The median is shown as a solid line. The edges of the box correspond to the first and third quartiles, and the whiskers extend to values within $1.5\times$ interquartile ranges. All individual data points are overlaid as black dots. PropMolFlow results use the top-performing models of each property in the ID tasks for property alignment.

indicates that GVP is more reliable on unrelaxed structures, perhaps due to its inductive bias. This is also true for other properties (Supplementary Figure 8).

To further quantify the relaxation effect, we calculated the root mean square distances (RMSDs) between atomic positions before and after relaxation. Fig. 3d shows that RMSD distributions are left-skewed toward 0 Å, meaning most unrelaxed structures are close to their relaxed counterparts. To estimate the property sensitivity to relaxation, we calculated the normalized property sensitivity to allow comparisons between different properties. Definitions of RMSDs and normalized property sensitivity are provided in the 'Evaluation metrics' section. Despite similar RMSD distributions, the differences are more pronounced in the property sensitivity, following the order of $C_v > \alpha > \Delta\epsilon$. Fig. 3d also shows configurations with the highest RMSDs, their target, DFT-calculated, and GVP-predicted values. DFT values with relaxation can be either closer or farther to Target. Despite high RMSDs, GVP values are nearly unaltered with relaxation, showing a much weaker structural dependence of GVP *versus* DFT. Overall, GVP predictions are close to DFT values for all cases, hence it can offer a statistically reliable evaluation of PropMolFlow models despite the inductive biases.

## Interpolated structures aligned with chemical intuition

To assess whether PropMolFlow has learned a smooth structure–property relationship, we performed interpolation by varying target values while fixing atom counts of molecules at 19. For each value, ten molecules were sampled, and the one whose DFT-calculated property is closest to the input was selected. We show the configurations for selected molecules, and minimum and maximum target values of each property along with corresponding DFT values in Fig. 4. Numeric results for all configurations are provided in Supplementary Table 12.

The interpolated molecules exhibit chemically intuitive structural changes when target values increase. For instance, higher $\alpha$ values largely correspond to more elongated molecular shapes. Higher $\mu$ and $C_v$ tend to show respective polar functional groups (-C≡N and -COO) and extended structures with reduced steric hindrance. High HOMO-LUMO gap, HOMO, LUMO energies correspond to more polarized $\sigma$ bonds, $\pi$ conjugated systems, and saturated $\sigma$ bonds,

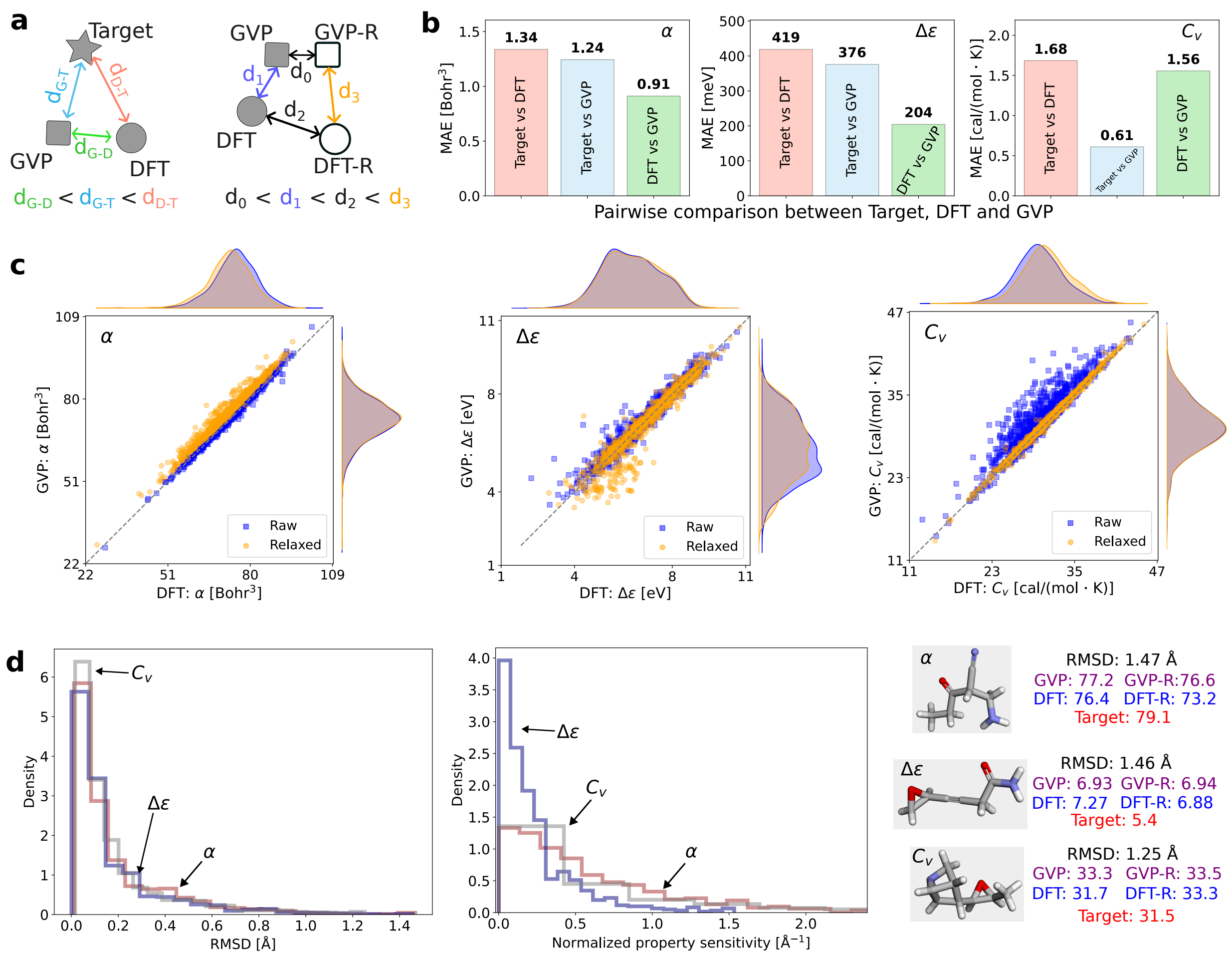

Figure 3: **Performance of GVP property predictors without and with DFT relaxation. a,** Comparison between Target, DFT and GVP shows the reliability of GVP in evaluating MAE metrics commonly used for property-guided generation. Comparison between GVP and GVP-R, and between DFT and DFT-R shows the structural dependence of GVP-predicted and DFT-predicted properties, respectively. Comparison between GVP and DFT with and without relaxation shows the reliability of GVP in capturing ground-truth DFT values for both raw and relaxed structures. 'Target', 'DFT' and 'GVP' denote input, DFT-calculated and GVP-predicted property values on raw molecules, respectively. 'DFT-R' and 'GVP-R' refer to values evaluated on DFT-relaxed molecules. The $d$ indicates the MAE distances between two property-value vectors. **b,** Pairwise comparison between Target, DFT and GVP on raw molecules. **c,** GVP versus DFT for both raw and DFT-relaxed molecules. **d,** Root mean squared distances (RMSDs) and normalized property sensitivity due to DFT relaxation. Molecules with the highest RMSDs for each property and their corresponding DFT and GVP values are shown. In molecular representations, gray, red, blue and white indicate C, O, N and H atoms, respectively. Property values for $\alpha$, $\Delta\epsilon$ and $C_v$ are in units of Bohr$^3$, eV and cal mol$^{-1}$ K$^{-1}$, respectively.

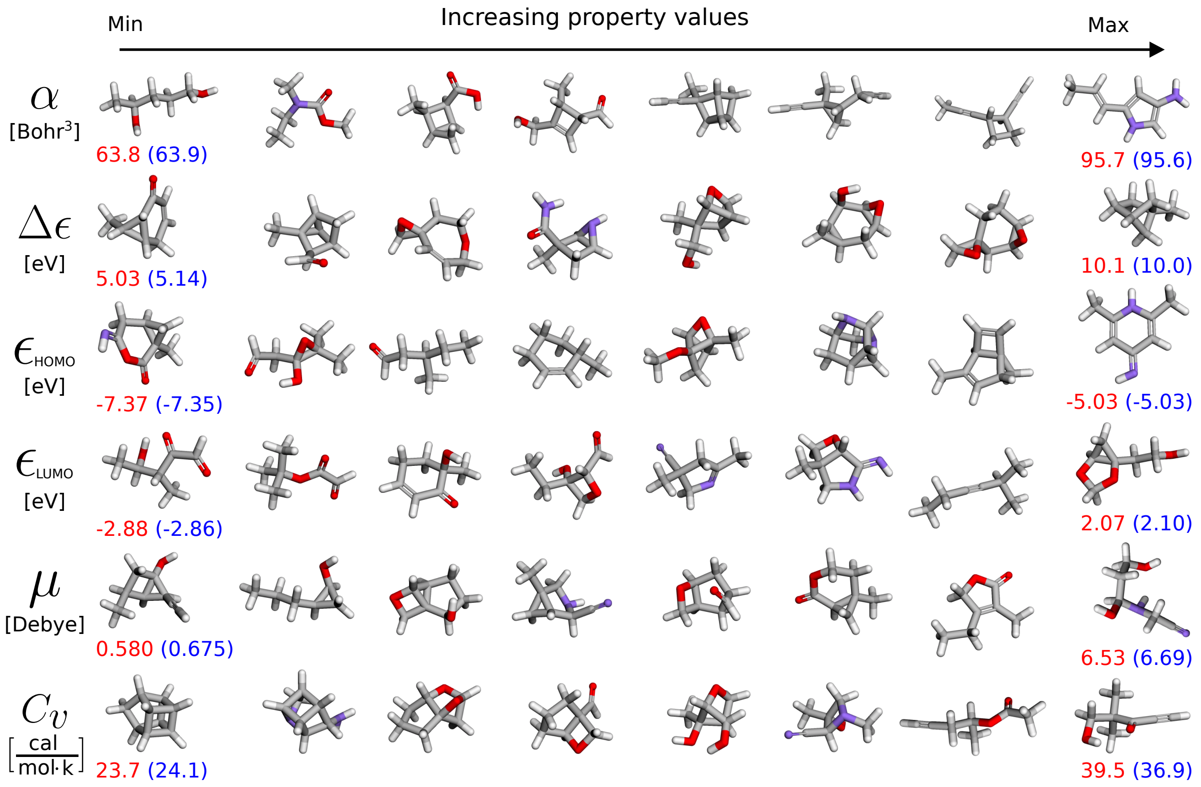

Figure 4: **Interpolation study by varying property values.** The minimum and maximum target properties (red) and the corresponding DFT-calculated properties (blue) are shown below each configuration. All molecules shown pass the filtering criteria and have DFT values closest to the target properties. In molecular representations, gray, white, red and blue indicate C, H, O and N atoms, respectively. Property units are provided in square brackets under each property symbol.

respectively. Yet, the generated molecule may not correspond to the most stable isomer, probably due to data limitations or sampling stochasticity. For example, conditioned on HOMO, the generated '2,6-Dimethyl-1H-pyridin-4-imine' is likely less stable than its tautomer '5-Amino-3-methyl-2H-pyridin-4-imine'. 2D molecular graph, SMILES and energy comparison of these two isomers are provided in Supplementary Figure 9.

## Toward out-of-distribution generation

We propose an OOD task to test generation for underrepresented property values. The 99th quantile of the training data distribution was chosen as the target value and the atom counts were chosen according to the 'Inference' section. Specific target values are provided in Supplementary Table 13. 1,000 molecules were sampled with top-performing models in this task (Supplementary Table 3), and we filtered structures using the same procedure as that used in the ID task. We compare DFT and GVP property distributions against that of the QM9 training data (Fig. 5a). DFT distributions shift toward the targets (dashed vertical lines). The distribution mode is very close to the target for $\alpha$. For $\Delta\epsilon$ and $C_v$, DFT distribution modes are off the targets by a small margin, for example 0.41 eV for $\Delta\epsilon$ (8.95 versus target 9.36 eV). This deviation is likely due to the scarcity of training data in the corresponding property region; when conditioning on the 50th quantile (medians), modes are closer to the targets (Supplementary Figure 11). Moreover, GVP distributions are almost overlapped with DFT ones, suggesting their desired extrapolation performance.

To examine the novelty of generated molecules, Fig. 5b presents three generated molecules whose SMILES representations are not part of QM9 but are found in the larger PubChem database [24]. We also evaluated the novelty of generated molecules by Tanimoto similarity to the training data. Fig. 5c indicates that 20–35% of the molecules exist in the training data. Using 0.8 as a cutoff, novelty ratios are 58–75% across all properties, demonstrating substantial exploration beyond the training data distribution. Results for other properties are provided in Supplementary Figure 10, and additional extrapolation to more extreme property values or atom counts can be found in respective Supplementary Figure 12 and Supplementary Figure 13.

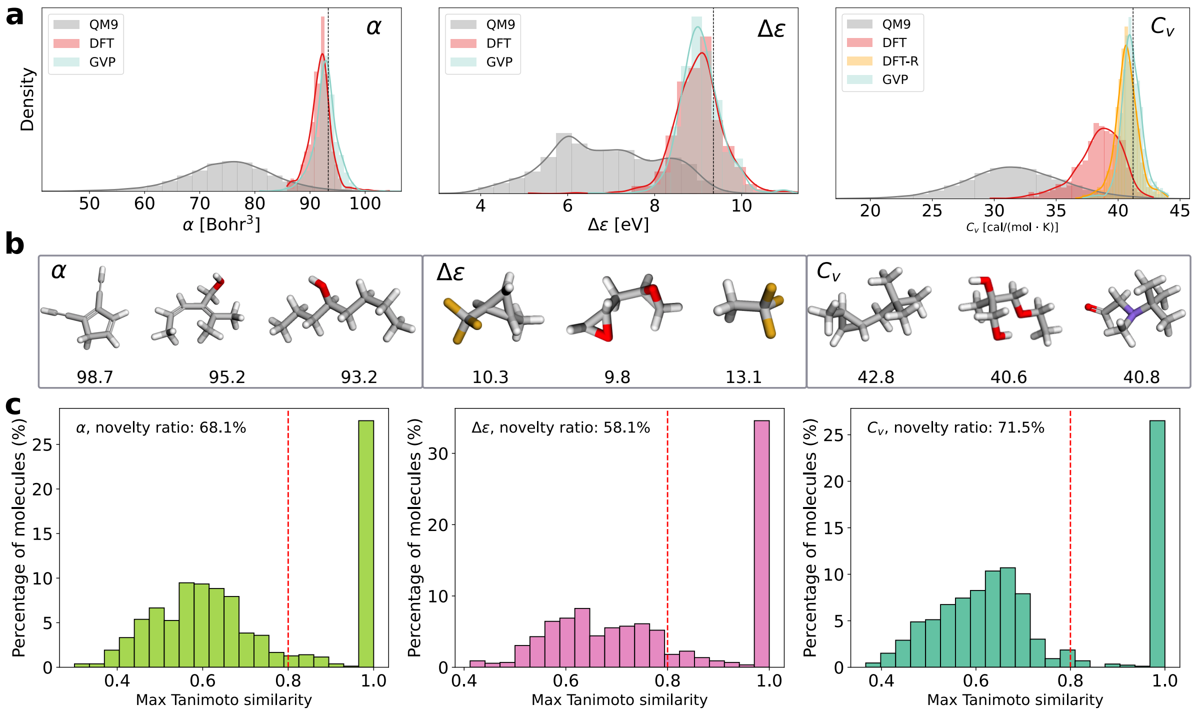

Figure 5: **Toward out-of-distribution generation.** **a,** Distribution of DFT-calculated and GVP-predicted property values for molecules generated by PropMolFlow; the property distribution of the QM9 training data is also shown. The vertical black dashed line in the histograms denotes the target property value $q_{0.99}$, corresponding to the 99th percentile of the training-data distribution. Curves overlaid on the histograms are kernel density estimation fits. **b,** Three example molecules absent from QM9 but present in a larger PubChem dataset are shown in the left panel. Numbers below the configurations indicate DFT-calculated property values for raw molecules generated by PropMolFlow. In molecular representations, gray, white, red, blue and yellow indicate C, H, O, N and F atoms, respectively. Property values for $\alpha$, $\Delta\epsilon$ and $C_v$ are in units of Bohr$^3$, eV and cal mol$^{-1}$ K$^{-1}$, respectively. **c,** Maximum Tanimoto similarity between generated, filtered molecules and the training data computed using Morgan fingerprints. Dashed lines indicate the 0.8 similarity cutoff used to define novel molecules.

## Discussion

While PropMolFlow can generate chemically valid and novel molecules with strong property alignment and fast inference, its ability to extrapolate could be further enhanced, potentially by integrating the model into an active-learning or reinforcement-learning framework. The current implementation is limited to the small-molecule QM9 dataset; extending PropMolFlow to larger molecular datasets and conditioning on properties of greater practical relevance represents an important future direction. In some cases, the property distribution of generated molecules exhibits deviations from target values. Property alignment could be improved by incorporating advanced guidance mechanisms with tunable guidance weights [25, 26]. Moreover, despite faster inference than diffusion models, flow matching still requires multi-step integration . Additional efficiency gains may be possible by learning flow maps that support direct jumps along probability paths [27]. However, these approaches have shown effectiveness only on continuous data; learning flow maps for multi-modal 3D molecules remains a challenge.

Although extending PropMolFlow to multiple properties is conceptually straightforward, the optimal strategy for synthesizing different properties and for managing potential correlations remains unclear. Another challenge concerns conformational stability: PropMolFlow does not guarantee the generation of the most stable conformers. Injecting energies into *de novo* generation is an open question, though recent advances in conformer generation with fixed compositions, such as adjoint Schrödinger bridge sampling [28], may provide useful insights.

Furthermore, several techniques, such as fake atoms and self-conditioning, were recently introduced in the unconditional FlowMol model [29] while this work was under development, and may further benefit the PropMolFlow conditional model. Despite these limitations and challenges, PropMolFlow demonstrates reliable generative modeling in chemical discovery and provides a strong foundation for future advancements.

## Methods

### Joint flow matching

Molecules are represented by fully-connected graphs $G$. Graph nodes encode the atomic types $A$, charges $C$ and positions $X$ of a molecule, and graph edges include bond orders $E$. Ideally, new molecules would be generated by sampling directly from the probability distribution over valid molecular graphs $q(x)$, where $x$ is an arbitrary sample. However, this target distribution lacks a tractable analytical form and can only be approximated through a generative process. To this end, we adopt flow matching to learn an approximation of $q(x)$. This flow matching is parameterized by an SE(3) equivariant graph neural network built on top of FlowMol architecture [10], details of which are provided in the subsequent 'Model architecture' section. In flow matching, the objective is to learn a time-dependent velocity field $u_t^\theta$ parameterized by a neural network $\theta$, which defines a probability path $p_t(x) : [0,1] \times x_0 \to x_1$ from a base distribution, $p_0(x)$ to a target distribution, $p_1(x) = q(x)$ (subscripts denote time dependence, for example, $p_t(x)$) The flow induced by the velocity field is given by the ODE $\dot{x} = u_t(x)$, which describes the evolution of samples along the probability paths. Designing a flow matching process thus involves two key steps: 1) specifying a probability path that satisfies the boundary conditions $x_0 \sim p_0$ and $x_1 \sim p_1$, and 2) training a neural network $\theta$ to approximate the velocity field that transports samples along this path.

In practice, while accurately modeling the exact velocity field is challenging, conditioning it on a random variable $z$ leads to a more tractable formulation. Importantly, optimizing the conditional velocity field is equivalent to optimizing the marginal velocity field [6]. The endpoint reformulation of [30] reparametrizes the flow matching problem by focusing on learning a denoising function rather than the full time-dependent velocity field $u_t^\theta$. Specifically, the method learns a denoising function $x_{1|t}^\theta$ that predicts the final state $x_1$ from the intermediate state $x_t$. The training objective then minimizes the discrepancy between the predicted final sample and the true final sample:

$$\mathcal{L} = \mathbb{E}_{t, p_{t|1}(x_t|z), p_z}\left[\|x_{1|t}^\theta - x_1\|\right], \tag{1}$$

where $z$ is a conditioning variable that can be either the final graph $x_1$ or a pair $z = (x_0, x_1)$ representing both the initial and final states.

Discrete graph features are handled via CTMC as developed by Campell *et al.* [9] and Gat *et al.* [31]. In this setting, Eq. (1) is reformulated to minimize a cross-entropy loss over discrete variables:

$$\mathcal{L}_{\text{CE}} = \mathbb{E}_{t, p_{t|1}(x_t|z), p_z}\left[-\log p_{1|t}^\theta(x_1^i|x_t)\right] \tag{2}$$

We factorize the conditional probability path of the molecular graph into the individual paths for each modality. A joint flow matching process is then defined by learning a velocity field (or, equivalently, a denoiser)—parameterized by a GNN—that minimizes the total loss expressed as a weighted sum of the losses for each modality:

$$\mathcal{L} = \eta_X \mathcal{L}_X + \eta_A \mathcal{L}_A + \eta_C \mathcal{L}_C + \eta_E \mathcal{L}_E \tag{3}$$

Previous studies suggest that atomic positions should have a higher weight, followed by bond orders, atomic charges, and atom types. The loss weights are chosen to be $(\eta_X, \eta_A, \eta_C, \eta_E) = (3.0, 0.4, 1.0, 2.0)$. The factorization of the flow into its components allows for the flexible design of individual interpolants for each modality, as detailed below.

**Interpolant for atomic positions.** The atomic position flow is conditioned on the pair of initial and final states $z_X = (X_0, X_1)$, and is given by the linear interpolant

$$X_t = \alpha_t X_0 + \beta_t X_1 \tag{4}$$

where $\alpha_0 = \beta_1 = 1$ and $\alpha_1 = \beta_0 = 0$. The functions $\alpha_t$ and $\beta_t$, known as the interpolant schedule, control the rate of mixing between the base and target distributions. A typical and effective choice of interpolant schedule for both molecule and materials generation is a linear schedule: $\alpha_t = (1-t)$ and $\beta_t = t$ [10, 8]. Other variants of interpolants are discussed in detail in [7, 8]. The base distribution for the atomic position flow is a standard Gaussian distribution $p_0(X) = \prod_{i=1}^N \mathcal{N}(X_0^i|\mathbf{0}, \mathbb{I}_3)$. A denoising network is then trained to produce the final atomic positions $X_{1|t}^\theta$, minimizing the following loss:

$$\mathcal{L}_X = \mathbb{E}_{t, p_t(X_t|X_0,X_1), \pi(X_0,X_1)}\left[\|X_{1|t}^\theta - X_1\|\right] \tag{5}$$

Where the joint distribution $\pi(X_0, X_1)$ defines the optimal transport coupling between $(X_0, X_1)$. Details of the optimal transport formulation are provided in Supplementary Section 3.

**Interpolant for atom type, charge, and bond order.** Atom types, charge and bond orders are modeled using CTMC-based discrete flow matching [9]. For each categorical variable, a 'mask' token is added as an additional discrete state. For example, consider atomic identities $A = \{A^1, A^2, \ldots, A^N\}$ where each $A^i$ takes values in the set $\{1, 2, \ldots, n_A, M\}$. Here, $n_A$ is the number of predefined atom types and $M$ is the mask token. The joint probability path over all atomic identities is factorized into independent contributions from each atom. The conditional probability path for atom $i$ is defined as:

$$p_t(A_t^i|A_0, A_1) = \alpha_t \delta(A_t^i, A_1^i) + \beta_t \delta(A_t^i, M) \tag{6}$$

where $\alpha_t$ and $\beta_t$ are the same interpolant schedules defined in Eq. (4). A linear interpolant is effective for categorical variables as well [10, 8]. Here, $\delta(i,j)$ denotes the Kronecker delta, which is 1 if $i = j$ and 0 otherwise. The base distribution corresponds to all atoms being initialized in the masked state. This formulation implies that at time $t$, the atom identity $A_t^i$ has a probability $\alpha_t$ of being in its final state $A_1^i$ and a probability $\beta_t$ of being in the masked state $M$.

In contrast to continuous variables, discrete flows are propagated via CTMC, where state transitions are stochastic rather than deterministic. The transition probability for atom $i$ over a short time interval $\Delta t$ can be expressed in terms of individual atomic contributions:

$$p^i_{t+\Delta t}(j|A_t^i) = \begin{cases} R^i(A_t^i, j) & \text{for} \quad j \neq A_t^i \\ 1 + R^i(A_t^i, A_t^i) & \text{for} \quad j = A_t^i \end{cases} \tag{7}$$

$$= \delta(j, A_t^i) + R^i(A_t^i, j)\Delta t \tag{8}$$

where $R(A_t^i, j)$ is the rate matrix specifying the transition probability from atom type $A_t^i$ to $j$ [9]. To ensure proper normalization of the transition probabilities at time $t+\Delta t$, the rate matrix must satisfy $R^i(A_t^i, A_t^i) = -\sum_{j \neq A_t^i} R(A_t^i, j)$ so that the total probability mass remains normalized.

An additional stochastic term $\eta$ can be introduced into the rate matrix to form the modified rate matrix $R_t^\eta := R_t^* + \eta R_t^{\mathrm{DB}}$. This new rate matrix generates the same marginal probability path if $R_t^{\mathrm{DB}}$ satisfies detailed balance ('DB') over the conditional probability path $p_{t|1}$. By including the stochasticity term and choosing a linear interpolant ($\alpha_t = 1 - t$ and $\beta_t = t$), the rate matrix for atom $i$ becomes:

$$R^i(A_t^i, j) = \frac{1 + \eta t}{1 - t} p^\theta_{1|t}(j|A_t^i)\delta(A_t^i, M) + \eta \left(1 - \delta(A_t^i, M)\right)\delta(j, M) \tag{9}$$

This formulation implies that if atom $i$ is in the masked state at time $t$, the probability of transitioning to atom type $j$ at time step $t + \Delta t$ is $\Delta t \frac{1+\eta t}{1-t}$. Conversely, if atom $i$ is in an unmasked state, the probability of transitioning back to a masked state is $\eta \Delta t$. In Eq. (9), the rate matrix can diverge as $t \to 1$. To address this issue, we employ a time-dependent loss function that avoids division-by-zero [32].

Additionally, low-temperature sampling—which rescales the prediction logits to sharpen the output distribution—is found to be critical for improving model performance [10]. Specifically, the denoiser network's output $p^\theta_{1|t}(A_1^i|A_t)$ is transformed as follows:

$$\widehat{p}^\theta_{1|t}(A_1^i|A_t) = \mathrm{softmax}\left(\nu^{-1} \log p^\theta_{1|t}(A_1^i|A_t)\right) \tag{10}$$

where $\nu$ is a hyperparameter. The modified probabilities $\widehat{p}^\theta_{1|t}(A_1^i|A_t)$ are then employed in the discrete loss function defined in Eq. (2). When training PropMolFlow, we set $\eta = 10$ and $\nu = 0.05$. During inference, we observe that turning off the stochastic term leads to better performance. To ensure that the generated molecules satisfy the required symmetry constraints, the joint flow matching process is constructed with an invariant base distribution and an equivariant transition probability path parameterized by the SE(3)-GVP architecture. A self-contained formal derivation of probability invariance and equivariant generative process is provided in respective Supplementary Section 4 and Supplementary Section 5.

## Model architecture

We utilize the FlowMol architecture, which is implemented with PyTorch and Deep Graph Library [10]. Molecule updates are achieved through layers comprising Geometric Vector Perceptrons (GVP). Within each GVP, the molecule graph passes through a sequential steps of node feature update, node position update and edge feature update. Each node $i$ consists of a position $x_i \in \mathbb{R}^3$, scalar features $s_i \in \mathbb{R}^d$, and vector features $v_i \in \mathbb{R}^{c \times 3}$. The scalar feature is a concatenation of atom type and charge vectors; that is, $s_i := [a_i : c_i]$ where ':' defines a concatenation operation. Vector features are initialized at zeros, and despite not existing in the final outputs, their updates employ the vector cross-products that is not equivariant to reflections, hence SE(3) equivariant [30]. Each edge feature corresponds to the bond order and the permutation invariance of the scaler bond order is ensured by taking the sum of learned bond features from $i \to j$ and $j \to i$;*i.e.*, $\widehat{e}^{ij} = \mathrm{MLP}(e_{ij} + e_{ji})$.

**Node feature update.** The node feature uses graph convolution to update node scalar and vector features $s_i, v_i$. It follows two steps: the first step generates scalar $m_{i\to j}^{(s)}$ and vector messages $m_{i\to j}^{(v)}$ by a function $\psi_M$ which is a chain of two GVPs:

$$m_{i\to j}^{(s)}, m_{i\to j}^{(v)} = \psi_M\left(\left[s_i^{(l)} : e_{ij}^{(l)} : d_{ij}^{(l)}\right], \left[v_i : \frac{x_i^{(l)} - x_j^{(l)}}{d_{ij}^{(l)}}\right]\right) \tag{11}$$

Where $d_{ij}^{(l)}$ is the distance between nodes $i$ and $j$ at update block $l$. In the FlowMol implementation, the distance $d_{ij}$ is replaced with a radial basis distance embedding before being fed into the GVPs or MLPs. Following the message generation step, an update of node scalar and vector features can be accomplished by aggregating generated messages:

$$s_i^{(l+1)}, v_i^{(l+1)} = \mathrm{LN}\left(\left[s_i^{(l)}, v_i^{(l)}\right] + \psi_N\left(\frac{1}{|\mathcal{N}(i)|}\sum_{j\in\mathcal{N}(i)}\right)\left[m_{j\to i}^{(s)}, m_{j\to i}^{(v)}\right]\right) \tag{12}$$

Where $LN$ stands for LayerNormalization operation, $\psi_N$ is a chain of three GVPs.

**Node position update.** NPU block updates node positions by taking node-wise operations on the updated node scalar and vector features:

$$x_i^{(l+1)} = x_i^{(l)} + \psi_P\left(s_i^{(l+1)}, v_i^{(l+1)}\right) \tag{13}$$

Where $\psi_P$ is a chain of three GVPs in which the final output has 1 vector and 0 scalar features.

**Edge feature update.** Edge features are updated by edge-wise operations that take the updated node scalar features and node distance as the inputs:

$$e_{ij}^{(l+1)} = \mathrm{LN}\left(e_{ij}^{(l)} + \mathrm{MLP}\left(s_i^{(l+1)}, s_j^{(l+1)}, d_{ij}^{(l+1)}\right)\right) \tag{14}$$

Models are trained using the settings detailed in the 'Hyperparameters and training settings' section.

## Property embedding operations

Conditional generation can be framed as sampling a property-conditional target distribution $p_1(x|k)$ over molecular graphs, where $k$ is a property. Previous approaches to conditional generation concatenate a property value to the node features of a molecular graph, training the model jointly on the property and the graph [2, 4, 11]. As a result, the learned denoising process is conditioned on the property. This approach requires modifying the input dimension of the denoiser to include the additional property value, which is subsequently discarded in the final output. However, directly appending the property to the node features may oversimplify the interaction between property values and molecular graphs. As shown in Figure 1, our implementation maps a scalar property, $k$, to a high-dimensional property embedding space $P = \phi_{\text{prop}}(k)$ using a MLP, $\phi_{\text{prop}}$. The dimensionality of the property embedding matches that of the node scalar features, allowing seamless integration without requiring additional dimensional matching. To encode interactions between the property embedding and node scalar features in molecualr graphs, we propose five distinct operations, including 'Concatenation', 'Sum', 'Multiply', 'Concatenate + Sum', and 'Concatenate + Multiply'. Depending on the operation, if the combined feature vector $[A, C]$ maintains its original dimensionality after interacting with the property embedding, no further processing is applied; otherwise, an MLP is used to project it back to the original dimensionality. Since the property embedding interacts exclusively with node scalar features, leaving position vectors unchanged, the resulting property-conditioned graph retains the SE(3) equivariance. Moreover, the loss function preserves its original form, and the trained SE(3) GNN defines a valid conditional generative process.

Specifically, in the 'Sum' operation, the property embedding is simply added to the node scalar features. The 'Multiply' operation performs an element-wise Hadamard product '$\odot$'. To ensure the multiplicative factors remain within a bounded range, the property embedding is passed through a sigmoid function and then shifted by 0.5, resulting in multipliers in the interval $[0.5, 1.5]$. Both 'Sum' and 'Multiply' preserve the original feature dimensionality. In contrast, operations involving 'Concatenate' increase the dimensionality of the node scalar features, and an MLP transformation $\varphi_\theta$ is required to project the combined features back to the original dimension. For the composite operations, 'Concatenate + Sum' and 'Concatenate + Multiply', the property embedding is first concatenated to the node features, followed by the respective arithmetic operation. In both cases, the final output has the desired dimensionality, either by design or via the subsequent MLP transformation.

## Gaussian expansion

We incorporate a fixed, non-trainable Gaussian expansion layer to enrich the property representations prior to being mapped to a property embedding via an MLP $\phi_{\mathrm{GE}}$ [17]. The Gaussian expansion maps a scalar property value into a fixed-length vector using a set of Gaussian basis functions. Given a property $k$ with value $\tau_k$, the expanded representation takes the form:

$$f_{\mathrm{k}} = \phi_{\mathrm{GE}}\left(\left[\exp\left(-\frac{(\tau_{\mathrm{k}} - (\tau_{\min} + n_g d))^2}{2d^2}\right)\right]_{0 \le n_g \le \frac{\tau_{\max} - \tau_{\min}}{d}}\right) \tag{15}$$

where $\tau_{\min}$ and $\tau_{\max}$ are the minimum and maximum property values, $d$ is the uniform spacing between the Gaussian centers, and $n_g$ indexes the basis functions. The output vector encodes the proximity of the input $\tau_k$ to each Gaussian center, producing a smooth, localized representation that is refined by the MLP $\phi_{\mathrm{GE}}$.

Figure 1d shows an example of a Gaussian expansion for the HOMO-LUMO gap. In this example, five Gaussian basis functions ($n_g = 5$) are used, with centers evenly spaced between the minimum and maximum gap values in the QM9 dataset: $\tau_{\min} = 0.0246$ and $\tau_{\max} = 0.6221$ Hartree. The width of each Gaussian is set by the spacing parameter $d = (\tau_{\max} - \tau_{\min})/(n_g - 1)$. The top panel of Figure 1d shows the five Gaussian basis functions, with their centers marked by vertical dashed gray lines. In the bottom panel, we pick two example property values, namely 0.1, and 0.5 Hartree, and show the expanded feature vectors as bar plots indicating the activation of each Gaussian basis function. For instance, if the gap is 0.1 Hartree, the first two Gaussian functions (centered closest to 0.1) have higher responses than the remaining functions, whose centers are farther away. Similar activation patterns are observed for the other two property values.

## Inference

**ID tasks.** To generate paired samples of property values and numbers of atoms, we employed a two-stage sampling procedure. First, atom counts were sampled according to their empirical distribution in the training dataset. Specifically, the frequency of each unique atom count was computed and used to construct a categorical probability distribution, from which atom counts were drawn.

Conditional on a given atom count, the associated property values were sampled using a histogram-based approximation of their empirical distribution. For each unique atom count, the observed property values were discretized into a fixed number of bins like 1000. The counts within each bin were normalized to form a categorical distribution. During sampling, a bin index was first drawn from this categorical distribution, after which a continuous property value was assigned by uniformly sampling within the range of the selected bin.

This approach yields paired samples of atom counts and property values that respect both the marginal distribution of atom counts and the conditional distribution of properties given the atom count. To make a fair comparison across different property embedding methods, the same sampled target values and atom counts were used.

**OOD tasks.** The property values in the OOD tasks are chosen as the $99^{\mathrm{th}}$ quantile of the QM9 training data and the numbers of atoms are selected for the property ranges bounded by the 97th quantile and the maximum (100th quatile) property value. During inference, PropMolFlow models use Euler's method with 100 evenly spaced time steps to integrate the learned velocity field.

## Datasets, baselines and evaluation metrics

### Datasets

We trained PropMolFlow on the QM9 data with explicit hydrogen atoms [15, 16]. We did not evaluate PropMolFlow on the GEOM-Drugs dataset, which is commonly used for unconditional generation and includes larger molecules, because it only provides quantum-mechanical energy values and lacks other relevant properties [33]. The original QM9 data contains 133,885 molecules, and each molecule contains up to 9 heavy atoms and consists of 3–29 atoms (average: 18 atoms), composed of up to 5 elements: C, N, O, F, and H. Molecules are fully optimized and molecular properties are obtained with DFT at the level of B3LYP/6-31G(2df,p) [34, 35]. For property-guided generation, we considered six molecule properties: polarizability ($\alpha$), HOMO-LUMO gap, HOMO energy, LUMO energy, dipole moment ($\mu$), and heat capacity ($C_v$). Precise definitions of each property can be found in Supplementary Section 8. Upon inspection, we discovered numerous inconsistencies: invalid bond orders and non-zero net charges in roughly 30,000 molecules, despite QM9's requirement for charge-neutral, closed-shell valency [15]. We corrected these discrepancies by reassigning bond orders or charges to enforce valency-charge consistency for the vast majority of entries and used the corrected data to train our PropMolFlow models. After the correction, we improved the structural validity of generated molecules substantially (Supplementary Table 1). Previous works that used bond orders overlooked this issue [13, 10]; instead, they modified evaluation metrics for models trained on the problematic SDF data against the one used by Hoogeboom et al. [2] to tolerate chemically inconsistent bond–charge combinations, hence inflating molecule stability estimates contrary to QM9's intended constraints. A detailed comparison between models using previous problematic data and the corrected data, together with the comparison between previous stability metrics and the revised ones, can be found in Supplementary Table 1. A detailed description of our procedure and the alignment with the original QM9 XYZ data is provided in Supplemental Section 1.2. The revised QM9 SDF data is provided on Zenodo [36]. We used RDKit's sanitization function [21] to filter chemically valid molecules on the QM9 SDF file with bond and charge fix, resulting in a curated set of 133k molecules. After random shuffling, we split the dataset into 100k training, 20k validation, and 13k test samples. PropMolFlow models and GVP property predictors are trained on disjoint 50k and 50k datasets from the 100k training data.

## Baselines

We compare PropMolFlow against several recent property-guided molecular generation methods based on diffusion models and flow matching, including EEGSDE, GCDM, GeoLDM, JODO, and EquiFM [19, 14, 4, 20, 11]. EEGSDE (Equivariant Energy Guided Stochastic Differential Equations) use an energy function to guide the diffusion for property-guided generation [19]. EquiFM is the first equivariant flow matching model for 3D molecule generation [11]. GeoLDM represents the first latent diffusion model for molecular geometry generation [4]. GCDM constructs an SE(3) equivariant geometry-complete diffusion model using geometry-complete perceptrons for molecule generation [14, 37]. JODO uses a diffusion graph transformer for joint generation of 2D molecular graphs and 3D geometries. EEGSDE, EquiFM, GeoLDM, and GCDM do not include bond orders in their molecular graphs, while JODO incorporates both bond existence and bond orders. All five baseline models omit atomic charges in their conditional generation, in contrast to PropMolFlow, which can generate molecules with non-zero atomic charges.

In this work, we focus exclusively on conditional generation results; unconditional generation performance is reported in earlier studies. Language-model-based approaches are excluded from this comparison, as state-of-the-art models in that category rely on fundamentally different molecule representations and training data regimes. For EEGSDE, we benchmark PropMolFlow against the canonical conditional model with a scaling factor of 1 ($s = 1$) [19]. Although EEGSDE provides DFT-based validation for five out of six properties, it does so on only 100 generated molecules, which limits the statistical significance of those results. Because EEGSDE, JODO and GCDM released their model checkpoints, we sampled molecules from those and computed all structural validity metrics on the resulting structures; for GeoLDM, we used the checkpoints bundled with GCDM. EquiFM checkpoints were unavailable, so we trained its conditional models by ourselves following information in [11] (training details in Supplementary Section 8).

Three intuitive baselines are included for comparison in the ID task. The "Random (Upper-bound)" intuitive baseline fully shuffles property values, thereby removing any correlation between molecular structures and properties, and serves as an upper bound on error. The "# Atoms" baseline uses only the number of atoms as a predictor for molecular properties, capturing coarse size-dependent trends. It is implemented as a simple neural network with one hidden layer [2]. The "QM9 (Lower-bound)" baseline corresponds to a pretrained EGNN regressor provided by Hoogeboom *et al.* [2], trained directly on QM9 and serving as a lower bound on achievable error. Improvement over the "Random" baseline indicates that the conditional generation effectively integrates property information. Surpassing the "# Atoms" baseline suggests that the generative model captures structural features beyond simple atom count when generating new molecules.

## Evaluation metrics

**Structural validity metrics.** Our comparison includes both structural-validity, inference-efficiency and property-specific evaluation metrics. Structural validity metrics include atomic stability, molecule stability, RDKit validity, 'uniqueness and validity', PoseBusters validity, and closed-shell ratio. Atom stability is given by the proportion of atoms with correct valency. To account for atomic charges, an atom is considered to have the correct valency if its formal charge balances its explicit valency, as defined in RDKit [21]. For instance, a nitrogen atom should carry a +1 formal charge if it has a valency of 4. Molecule stability defines the proportion of molecules in which all atoms are stable and the molecule is charge neutral (zero net formal charges). RDKit validity refers to the ratios of molecules that pass RDKit's sanitization check. 'Uniqueness and validity' is defined as the proportion of molecules that are RDKit valid and unique in their SMILES representation. PoseBusters validity corresponds to the fraction of molecules that pass many *de novo* chemical and structural validation tests, such as all atom connectivity, valid bond lengths and angles, and absence of internal clashes [22]. We used the first 10 columns of the PoseBusters outputs in CSV formats as the criteria for results shown in Fig. 2 and Supplementary Table 5. A molecule is closed-shell valid if it has an even number of valence electrons. Since all molecules in QM9 are closed-shell, any generated open-shell species are considered invalid. Except for the closed-shell ratio, all structural validity metrics depend on bond-order information, which baseline models (other than JODO) do not output. We therefore assigned bond orders using the distance-based, optimized cutoffs introduced by Hoogeboom [2]. Although recent work [13] suggests that the bond orders can probably be optimized through the OpenBabel program [38], we did not apply such refinements to the baseline samples. To assess the chemical similarity between generated molecules and a reference training set, we employed a fingerprint-based approach using RDKit. For each molecule, a Morgan fingerprint (circular fingerprint) was computed with a radius of 2 and a 2048-bit representation. Pairwise similarity between each generated molecule and all training molecules was computed using the Tanimoto similarity metric on their respective Morgan fingerprints. For each generated molecule, the maximum similarity score and the corresponding most similar training molecule were recorded. Lastly, inference efficiency is given by the wall-clock time required to generate 10,000 molecules.

Property metrics. PropMolFlow is directly compared to prior methods on the ID tasks. In this task, a conditional generative model was first used to generate molecules conditioned jointly on property values and atom counts sampled from the same distribution as the training data (for example, QM9). These sampled property values and atom counts served as the input conditions. A separate property predictor/regressor was then applied to estimate the properties of the generated molecules. In PropMolFlow, we used an SE(3) GVP network for molecule generation and, correspondingly, trained a separate GVP regressor for each of the six molecular properties. 10000 molecules are generated to evaluated the performance of conditional generation. OOD results are not directly comparable to baseline models, but we validate our results with DFT calculations to assess physical fidelity.

**RMSD and property sensitivity.** Given the same chemical compositions and element-type ordering, the RMSD between any two molecules ($\mathrm{M}_0$ and $\mathrm{M}_1$) is the minimum distance between these two structures considering the translational and rotational symmetry operations, which can be expressed as:

$$\mathrm{RMSD}(\mathrm{M}_0, \mathrm{M}_1) = \min_{\mathrm{P} \in \mathrm{SE}(3)} \{\mathrm{d}\left(\mathrm{M}_0, \mathrm{P}(\mathrm{M}_1)\right)\} \tag{16}$$

Where SE(3) represents the group of operations that respect the translational and rotational symmetry. To estimate the property sensitivity to structural relaxation, we selected the structures whose RMSDs are larger than 0.03 Å—RMSDs lower than this threshold are considered very close to the relaxed structures, and in Eq. (17), we define the normalized property sensitivity '$\chi_{\mathrm{DFT}}$' to allow comparisons between different properties.

$$\chi_{\mathrm{DFT}} = \frac{|\delta q_{\mathrm{DFT}}|}{(q_{0.99} - q_{0.01}) \cdot \mathrm{RMSD}} \tag{17}$$

where $\delta q_{\mathrm{DFT}}$ is the change of DFT-calculated property upon relaxation, and the normalizer ($q_{0.99} - q_{0.01}$) is the difference between the 99th quantile and the 1th quantile of the property distribution of QM9 training data. This range is chosen over the min-max range because it is robust to outliers.

## Computational settings

### GVP regressor details

To be self-consistent, we trained property regressors using Graph Neural Networks based on GVPs. GVP property predictors are trained on a disjoint dataset versus that used for training PropMolFlow models, hence avoiding data leakage between the generative models and property prediction models. We appended an MLP layer that takes the final node scalar features as input to predict the target property. The parameters of GVP regressors are optimized by minimizing a mean squared error loss function. Separate models were trained for each of the six molecular properties.

### DFT

All DFT calculations were performed using the Gaussian 16 package (Gaussian 16, Revision C.01) [39]. The B3LYP hybrid functional [34, 35] together with the 6-31G(2df,p) basis set was employed for single-point calculations, and geometric optimizations. Single-point calculations were used unless otherwise specified. Structural relaxations were performed for the property $C_v$ in most cases, except in the interpolation study. Relaxations were performed to evaluate changes in both properties and geometries, as shown in Fig. 3. DFT calculations that failed to converge were discarded. We released the DFT-evaluated molecules in both the ID and OOD tasks with and without structural optimization at the Zenodo repository [36].

### Molecule filtering

We selected 1000 molecules out of 10000 samples structures in the ID task or use the directly generated 1000 structures in the OOD task for DFT evaluations. multi-step filtering is essential to ensure that the structures are chemically valid and suitable for downstream applications. We filtered the 1000 molecules by five criteria: Molecule stability, RDKit validity, PoseBusters validity, closed-shell validity, and multi-fragment check. The multi-fragment check filters out molecules that with more than one disconnected fragment. For molecules passing all filters, property values are evaluated using both a GVP property predictor and DFT calculations. We also filtered out molecules whose DFT calculations did not converge, resulting in no less than 90% of and 78–94% of the molecules remaining for the subsequent analyses in the respective ID and OOD tasks (Supplementary Table 14).

### Hyperparameters and training details

PropMolFlow models on the QM9 dataset were trained with 8 molecule update blocks. Atoms contain 256 hidden scalar features and 16 hidden vector features. Edges contain 128 hidden features. QM9 PropMolFlow models were trained with 2000 epochs. For each property embedding method, 6 model checkpoints with lowest validation losses were saved and the top-3 model checkpoints were deposited to the Zenodo repository [36]. Inference for PropMolFlow models and all baseline models used a single NVIDIA A100-SXM4 graphic card with 80GB of memory, with a batch size of 128. Training of PropMolFlow models use both Nvidia A100-SXM4 and 2080ti graphic cards. PropMolFlow models can be trained in 2–3 days with A100 and 4–5 days with 2080Ti cards, and EquiFM models were trained with 2500 epochs on A100, which took around 3.5 days.

## Data availability

All data can be found in the Zenodo repository [36], including revised QM9 SDF data, SDF files for generated raw molecules, DFT-calculated structures and their molecular properties saved in the extxyz format, full property MAE and structural validity results in CSV format for all saved PropMolFlow checkpoint models, sampled structures and their full PoseBusters results for all baseline models, and EGNN property-predictor pretrained models and notebook examples to use the property predictors.

## Code availability

Our PropMolFlow implementation is available at https://github.com/Liu-Group-UF/PropMolFlow and on Zenodo [40].

## Acknowledgments

The authors acknowledge funding from NSF Grant OAC-2311632 and from the AI and Complex Computational Research Award of the University of Florida. S.M also acknowledges support from the Simons Center for Computational Physical Chemistry (Simons Foundation grant 839534). The authors also gracefully acknowledge UFIT Research Computing for computational resources and consultation, as well as the NVIDIA AI Technology Center at UF. We would like to thank Alex Morehead (University of Missouri) for providing the checkpoint models for the conditional generation of GeoLDM.

## References

[1] Benjamin Sanchez-Lengeling and Alán Aspuru-Guzik. Inverse molecular design using machine learning: Generative models for matter engineering. *Science*, 361(6400):360–365, July 2018.

[2] Emiel Hoogeboom, Victor Garcia Satorras, Clement Vignac, and Max Welling. Equivariant Diffusion for Molecule Generation in 3D. In *Proceedings of the 39th International Conference on Machine Learning*, pages 8867–8887. PMLR, June 2022.

[3] Víctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) equivariant graph neural networks. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9323–9332. PMLR, 18–24 Jul 2021.

[4] 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, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 38592–38610. PMLR, 23–29 Jul 2023.

[5] Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023.

[6] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

[7] Michael Samuel Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations*, 2023.

[8] Philipp Höllmer, Thomas Egg, Maya Martirossyan, Eric Fuemmeler, Zeren Shui, Amit Gupta, Pawan Prakash, Adrian Roitberg, Mingjie Liu, George Karypis, Mark Transtrum, Richard Hennig, Ellad B. Tadmor, and Stefano Martiniani. Open materials generation with stochastic interpolants. In *Forty-second International Conference on Machine Learning*, 2025.

[9] Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. In *Forty-first International Conference on Machine Learning*, 2024.

[10] Ian Dunn and David R. Koes. Exploring discrete flow matching for 3d de novo molecule generation. *arXiv* preprint arXiv:2411.16644, 2024.

[11] Yuxuan Song, Jingjing Gong, Minkai Xu, Ziyao Cao, Yanyan Lan, Stefano Ermon, Hao Zhou, and Wei-Ying Ma. Equivariant Flow Matching with Hybrid Probability Transport for 3D Molecule Generation. *Advances in Neural Information Processing Systems*, 36:549–568, December 2023.

[12] Alexandru Dumitrescu, Dani Korpela, Markus Heinonen, Yogesh Verma, Valerii Iakovlev, Vikas Garg, and Harri Lähdesmäki. E(3)-equivariant models cannot learn chirality: Field-based molecular generation. In *The Thirteenth International Conference on Learning Representations*, 2025.

[13] 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, editors, *Machine Learning and Knowledge Discovery in Databases: Research Track*, pages 560–576, Cham, 2023. Springer Nature Switzerland.

[14] Alex Morehead and Jianlin Cheng. Geometry-complete diffusion for 3D molecule generation and optimization. *Commun Chem*, 7(1):1–11, July 2024.

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

[16] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

[17] Niklas W. A. Gebauer, Michael Gastegger, Stefaan S. P. Hessmann, Klaus-Robert Müller, and Kristof T. Schütt. Inverse design of 3d molecular structures with conditional generative neural networks. *Nat Commun*, 13(1):973, February 2022.

[18] Bowen Jing, Stephan Eismann, Patricia Suriana, Raphael John Lamarre Townshend, and Ron Dror. Learning from protein structure with geometric vector perceptrons. In *International Conference on Learning Representations*, 2021.

[19] Fan Bao, Min Zhao, Zhongkai Hao, Peiyao Li, Chongxuan Li, and Jun Zhu. Equivariant energy-guided SDE for inverse molecular design. In *The Eleventh International Conference on Learning Representations*, 2023.

[20] Han Huang, Leilei Sun, Bowen Du, and Weifeng Lv. Learning Joint 2-D and 3-D Graph Diffusion Models for Complete Molecule Generation. *IEEE Transactions on Neural Networks and Learning Systems*, 35(9):11857–11871, September 2024.

[21] Greg Landrum et al. Rdkit: open-source cheminformatics software. RDKit, 2016.

[22] Martin Buttenschoen, Garrett M. Morris, and Charlotte M. Deane. PoseBusters: AI-based docking methods fail to generate physically valid poses or generalise to novel sequences. *Chemical Science*, 15(9):3130–3139, 2024.

[23] H. L. Morgan. The Generation of a Unique Machine Description for Chemical Structures-A Technique Developed at Chemical Abstracts Service. *Journal of Chemical Documentation*, 5(2):107–113, May 1965.

[24] Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, Leonid Zaslavsky, Jian Zhang, and Evan E Bolton. PubChem in 2021: new data content and improved web interfaces. *Nucleic Acids Research*, 49(D1):D1388–D1395, January 2021.

[25] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

[26] Tero Karras, Miika Aittala, Tuomas Kynkäänniemi, Jaakko Lehtinen, Timo Aila, and Samuli Laine. Guiding a diffusion model with a bad version of itself. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[27] Nicholas Matthew Boffi, Michael Samuel Albergo, and Eric Vanden-Eijnden. How to build a consistency model: Learning flow maps via self-distillation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[28] Guan-Horng Liu, Jaemoo Choi, Yongxin Chen, Benjamin Kurt Miller, and Ricky T. Q. Chen. Adjoint schrödinger bridge sampler. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[29] Ian Dunn and David R. Koes. Flowmol3: Flow matching for 3d de novo small-molecule generation. *arXiv* preprint arXiv:2508.12629, 2025.

[30] Ian Dunn and David Ryan Koes. Mixed continuous and categorical flow matching for 3d de novo molecule generation. *CoRR*, abs/2404.19739, 2024.

[31] Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T. Q. Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[32] Tuan Le, Julian Cremer, Frank Noe, 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*, 2024.

[33] Simon Axelrod and Rafael Gómez-Bombarelli. GEOM, energy-annotated molecular conformations for property prediction and molecular generation. *Sci Data*, 9(1):185, April 2022.

[34] Axel D. Becke. Density-functional thermochemistry. III. The role of exact exchange. *The Journal of Chemical Physics*, 98(7):5648–5652, April 1993.

[35] Chengteh Lee, Weitao Yang, and Robert G. Parr. Development of the Colle-Salvetti correlation-energy formula into a functional of the electron density. *Physical Review B*, 37(2):785–789, January 1988.

[36] Cheng Zeng, Jirui Jin, and Mingjie Liu. Propmolflow data: Version v3. Zenodo. https://doi.org/10.5281/zenodo.17726328, 2025.

[37] Michael Moret, Irene Pachon Angona, Leandro Cotos, Shen Yan, Kenneth Atz, Cyrill Brunner, Martin Baumgartner, Francesca Grisoni, and Gisbert Schneider. Leveraging molecular structure and bioactivity with chemical language models for de novo drug design. *Nat Commun*, 14(1):114, January 2023.

[38] Noel M O'Boyle, Markus Banck, Craig A James, Chris Morley, Tim Vandermeersch, and Geoffrey R Hutchison. Open babel: An open chemical toolbox. *Journal of Cheminformatics*, 3:33, 2011.

[39] M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, G. A. Petersson, H. Nakatsuji, X. Li, M. Caricato, A. V. Marenich, J. Bloino, B. G. Janesko, R. Gomperts, B. Mennucci, H. P. Hratchian, J. V. Ortiz, A. F. Izmaylov, J. L. Sonnenberg, D. Williams-Young, F. Ding, F. Lipparini, F. Egidi, J. Goings, B. Peng, A. Petrone, T. Henderson, D. Ranasinghe, V. G. Zakrzewski, J. Gao, N. Rega, G. Zheng, W. Liang, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, K. Throssell, J. A. Montgomery Jr., J. E. Peralta, F. Ogliaro, M. J. Bearpark, J. J. Heyd, E. N. Brothers, K. N. Kudin, V. N. Staroverov, T. A. Keith, R. Kobayashi, J. Normand, K. Raghavachari, A. P. Rendell, J. C. Burant, S. S. Iyengar, J. Tomasi, M. Cossi, J. M. Millam, M. Klene, C. Adamo, R. Cammi, J. W. Ochterski, R. L. Martin, K. Morokuma, O. Farkas, J. B. Foresman, and D. J. Fox. Gaussian 16 revision c.01, 2016.

[40] Cheng Zeng, Jirui Jin, and Liu Mingjie. Propmolflow code version 1. Zenodo. https://doi.org/10.5281/zenodo.17702415, 2025.