NATURAL GRAPH DIFFUSION SOLVER

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing graph diffusion samplers operate under uniform time discretization, which does not reflect the non-uniform nature of distributional evolution in graph generative processes. In this paper, we establish a fundamental principle for graph diffusion generation: a natural sampler should progress at a constant rate of distributional evolution. We show that this principle admits a geometric characterization, revealing that equal distributional change is equivalent to traversing equal Fisher-Rao arc-length increments along the reverse diffusion trajectory. This connection bridges diffusion sampling and information geometry, providing a principled foundation for information-geometry-aware generation. Inspired by this insight, we introduce the Natural Graph Diffusion Solver (NGD-solver) that enforces constant distributional evolution along the reverse diffusion trajectory. NGD-solver derives an information-geometry-aware correction that can be seamlessly incorporated into the reverse sampling process. By adaptively calibrating reverse transitions according to the local geometric sensitivity of distributions, NGD-solver promotes a constant-rate and stable evolution in distribution space throughout the diffusion. NGD-solver can be seamlessly integrated into pretrained graph diffusion models at inference time. Extensive experiments on graph generation benchmarks demonstrate that NGD-solver consistently and significantly improves structural fidelity and generation quality.

1 INTRODUCTION

Diffusion-based graph generative models have emerged as a powerful framework for graph generation (Hoogeboom et al., 2022; Jo et al., 2022; Vignac et al., 2023; Jo et al., 2024; Qin et al., 2025). By leveraging gradual denoising processes, these models provide an increasingly flexible and theoretically grounded approach for modeling complex graph-structured data.

Diffusion sampling can be viewed as a sequence of transition kernels that progressively transform a prior distribution into the target data distribution (Ho et al., 2020; Song et al., 2021b). Existing samplers typically rely on uniformly discretized reverse-time dynamics, implicitly assuming that equal time steps correspond to comparable generative progress. However, the induced distributional trajectory does not evolve in a flat Euclidean space; it lies on a statistical manifold with inherently non-uniform geometry. As a result, uniform time discretization fails to reflect the heterogeneous nature of distributional evolution (Lu et al., 2022).

To improve diffusion sampling efficiency and accuracy, prior work has explored higher-order numerical solvers (Karras et al., 2022; Lu et al., 2022; Zhao et al., 2023) and adaptive time-step allocation strategies (Sabour et al., 2024; Li et al., 2024; Park et al., 2025). While these approaches reduce discretization error or improve step allocation, they remain grounded in Euclidean dynamics and do not explicitly account for the intrinsic geometry of probability distributions. This limitation is particularly pronounced in graph diffusion models. Unlike continuous image domains, graph distributions are discrete, combinatorial and highly heterogeneous, resulting in significantly more irregular geometric structures (Zhang & Chen, 2023; Fu et al., 2024; Gao et al., 2025). Consequently, equal sampling steps do not correspond to equal distributional progress throughout the reverse process, leading to imbalanced evolution, where certain transitions induce excessive distributional shifts while others contribute only limited refinement.

This motivates a more fundamental question: what constitutes a natural notion of progress in diffusion-based generation? We argue that a natural sampling trajectory should evolve at a con-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

stant rate of distributional change. From an information-geometric perspective, diffusion sampling inherently traces a trajectory on the statistical manifold of probability distributions, where the Fisher-Rao metric, induced by Fisher information, provides an intrinsic measure of distributional variation (Girolami & Calderhead, 2011; Amari, 2016). Under the Fisher-Rao geometry, the proposed principle of natural sampling corresponds to traversing the statistical manifold with constant speed, which is equivalent to maintaining equal Fisher-Rao arc-length increments along the reverse trajectory (Pascanu & Bengio, 2014).

Guided by this insight, we propose the Natural Graph Diffusion Solver (NGD-solver), which adaptively calibrates reverse transitions according to the local geometric sensitivity of distributions by enforcing a Fisher-Rao arc-length constraint. Consequently, the generation trajectory progresses at an approximately constant rate of distributional evolution without altering the original time discretization. This balanced evolution prevents overly aggressive transitions in geometrically sensitive regions while maintaining efficient progress in stable regions. Importantly, NGD-solver operates entirely at sampling time, requiring neither retraining nor model modification, and can be seamlessly integrated into existing SDE-based diffusion samplers.

In summary, the main contributions are as follows:

- We propose the Natural Graph Diffusion Solver (NGD-solver). We introduce the principle of natural sampling, which advocates maintaining a constant rate of distributional evolution throughout the reverse diffusion process.
- We establish a geometric interpretation of natural sampling, where it corresponds to traversing the statistical manifold with equal Fisher-Rao arc-length increments.
- We develop a plug-and-play inference-time solver that adaptively adjusts reverse transitions based on information geometry, requiring no retraining and integrating into existing diffusion solvers.

Extensive experiments on molecular and general graph generation benchmarks demonstrate the effectiveness and generality of NGD-solver. When integrated into various graph diffusion models, it consistently improves distributional fidelity, structural validity and sampling stability.

2 Related Work

Recent diffusion-based graph generative models can be viewed under the probabilistic transport framework, where graph samples progressively evolve from simple prior distributions toward target graph distributions. Recent advances increasingly adopt diffusion-based generative frameworks due to their strong modeling flexibility and stable training behavior. Continuous diffusion-based approaches formulate graph generation through reverse-time stochastic dynamics governed by stochastic differential equations (SDEs) (Song et al., 2021b). Representative score-based diffusion models (Jo et al., 2022; Niu et al., 2020; Hoogeboom et al., 2022) learn score functions that reverse the noise perturbation process, progressively refining noisy graph states toward the target data manifold. Beyond standard reverse-time diffusion dynamics, bridge-based methods (Jo et al., 2024) further incorporate endpoint distribution constraints into the stochastic transport process, enabling guided or constrained graph generation under prescribed boundary conditions. In addition to continuous diffusion formulations, discrete diffusion models (Vignac et al., 2023; Qin et al., 2025) have also been extensively explored for graph generation. These methods construct discrete denoising processes directly over categorical graph states and typically rely on carefully designed transition matrices to govern generation dynamics in discrete state spaces. Despite substantial progress in graph generative modeling, existing approaches mainly focus on designing network architectures, diffusion formulations, or denoising objectives. Regardless of these modeling choices, diffusion-based models ultimately rely on reverse-time sampling, which plays a critical role in determining the quality of generated graphs. This motivates a closer investigation of the reverse-time sampling process itself.

However, existing diffusion samplers primarily focus on improving numerical accuracy and sampling efficiency, while largely ignoring the geometric consistency of distributional evolution. These methods typically rely on predefined fixed-step solvers, ranging from the first-order Euler-Maruyama method (Maruyama, 1955) to higher-order integrators such as the second-order Heun method (Ascher & Petzold, 1998), together with various acceleration techniques (Song et al., 2021a; Lu et al., 2022; Zhao et al., 2023). More recently, adaptive sampling strategies (Sabour et al., 2024;

Park et al., 2025; Li et al., 2024) dynamically adjust timestep schedules according to local errors, noise levels, or heuristic criteria, enabling more efficient allocation of sampling steps. Despite these advances, they still do not explicitly regulate distributional evolution according to the intrinsic, non-uniform geometry of probability distributions. From the perspective of information geometry, natural gradient methods (Pascanu & Bengio, 2014) exploit Fisher information to construct geometry-aware parameter updates. More recently, this geometric perspective has also been introduced into diffusion sampling. Some approaches employ Fisher-information-based approximations to improve conditional guidance (Song & Lai, 2024), while others leverage Fisher-Rao geometry to design adaptive timestep schedules through diffusion-time reparameterization (Lu et al., 2026). Nevertheless, these methods primarily utilize Fisher geometry for optimization, guidance, or temporal allocation, rather than explicitly controlling the geometric evolution of reverse transition distributions. In contrast, our work studies reverse-time generation from the viewpoint of transition-distribution geometry. Specifically, we employ the Fisher-Rao metric as a measure of distance between probability distributions (Amari, 2016) and explicitly constrain the geometric evolution between consecutive reverse transitions, yielding stable and balanced constant-speed evolution of distributions.

3 NGD-SOLVER

3.1 REVERSE TRANSITION DISTRIBUTION

We consider the sampling process in diffusion-based models. Formally, the generation process can be viewed as a stochastic trajectory defined over a fixed time interval $t \in [0, T]$, evolving from a noise prior distribution at $t = 0$ toward the target data distribution at $t = T$. The dynamics are governed by the reverse-time stochastic differential equation (SDE):

$$\mathrm{d}\boldsymbol{x}_t = \boldsymbol{f}_t \,\mathrm{d}t + g_t \,\mathrm{d}\overline{\boldsymbol{w}}_t \tag{1}$$

where $\boldsymbol{f}_t$ denotes the reverse-time drift field that determines the deterministic transport direction, g_t denotes the time-dependent diffusion coefficient controlling stochasticity and $\overline{\boldsymbol{w}}_t$ denotes standard Brownian motion in reverse time.

Since the reverse-time SDE is generally not analytically tractable, practical sampling relies on numerical discretization. A standard approach is the Euler-Maruyama scheme, which partitions the time interval $[0, T]$ uniformly into K steps of size $\Delta t = T/K$:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{f}_k \Delta t + g_k \sqrt{\Delta t}\, \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{2}$$

The discrete update rule in Eq. 2 induces a family of conditional transition distributions between adjacent time steps. Specifically, conditioned on the current state $\boldsymbol{x}_k$, the next state $\boldsymbol{x}_{k+1}$ follows the Gaussian transition distribution

$$p(\boldsymbol{x}_{k+1} \mid \boldsymbol{x}_k; \boldsymbol{f}_k) = \mathcal{N}\big(\boldsymbol{x}_k + \boldsymbol{f}_k \Delta t,\ g_k^2 \Delta t\, \boldsymbol{I}\big)\,. \tag{3}$$

3.2 FISHER-RAO ARC LENGTH

To quantify distributional evolution induced by each sampling step, we examine the geometry of the transition distributions in Eq. 3. Since the covariance $g_k^2 \Delta t\, \boldsymbol{I}$ is fully determined by the noise schedule, the drift $\boldsymbol{f}_k$ is the only effective degree of freedom and thus uniquely parameterizes the transition distribution (Lu et al., 2026). Consequently, each $\boldsymbol{f}_k$ defines a point on the statistical manifold $\mathcal{M} = \{p(\cdot \mid \cdot; \boldsymbol{f}) \mid \boldsymbol{f} \in \mathbb{R}^d\}$. To characterize its geometry, we equip $\mathcal{M}$ with the Fisher information matrix (Liang et al., 2019)

$$\mathcal{I}(\boldsymbol{f}) = \mathbb{E}_p\big[\nabla_{\boldsymbol{f}} \log p \nabla_{\boldsymbol{f}}^\top \log p\big]\,, \tag{4}$$

which is a Riemannian metric (Girolami & Calderhead, 2011) that measures the sensitivity of the transition distribution to variations in $\boldsymbol{f}$. A key property is that this metric is intrinsic to the distribution space: by Čencov's theorem, it is the unique metric invariant under smooth reparameterizations (Amari, 2016). Equivalently, it admits $\mathcal{I}(\boldsymbol{f}) = -\mathbb{E}_p[\nabla_{\boldsymbol{f}}^2 \log p]$ (see Appendix A.1), highlighting its second-order geometric nature. The induced Riemannian line element is $\mathrm{d}s^2 = \mathrm{d}\boldsymbol{f}^\top \mathcal{I}(\boldsymbol{f}) \mathrm{d}\boldsymbol{f}$ and the Fisher-Rao arc length between consecutive steps is

$$\Delta s^2 = \Delta \boldsymbol{f}^\top \mathcal{I}(\boldsymbol{f}) \Delta \boldsymbol{f} \tag{5}$$

where $\Delta \boldsymbol{f}$ denotes the drift variation. This arc length can be interpreted as the distance traversed on the statistical manifold between consecutive transition distributions.

Therefore, it provides a direct measure of the local magnitude of distributional change. Its connection to information geometry becomes explicit via the KL divergence: $\mathrm{KL}(p_{\boldsymbol{f}} \,\|\, p_{\boldsymbol{f}+\Delta \boldsymbol{f}}) = \frac{1}{2}\Delta s^2 + o(\|\Delta \boldsymbol{f}\|^2)$, showing that Fisher-Rao distance is an approximation to the KL divergence between consecutive transition distributions (see Appendix A.2).

For the Gaussian transition distributions in Eq. 3, the Fisher information admits a closed form. Taking the gradient of the log-likelihood with respect to $\boldsymbol{f}_k$ gives

$$\nabla_{\boldsymbol{f}_k} \log p = \frac{1}{g_k^2}(\boldsymbol{x}_{k+1} - \boldsymbol{x}_k - \boldsymbol{f}_k \Delta t), \tag{6}$$

Substituting Eq. 6 into Eq. 4 gives

$$\mathcal{I}(\boldsymbol{f}_k) = \frac{\Delta t}{g_k^2}\, \boldsymbol{I}\,. \tag{7}$$

Substituting Eq. 7 into the general Fisher-Rao arc length definition in Eq. 5 gives

$$\Delta s_k^2 = \frac{\Delta t}{g_k^2}\|\Delta \boldsymbol{f}_k\|^2. \tag{8}$$

Thus, distributional evolution is jointly determined by the Euclidean drift change and the local noise scale g_k . The derivations of Eq. 7 and Eq. 8 are detailed in Appendix A.3.

3.3 Natural Graph Diffusion Sampling

A key limitation of existing diffusion samplers is that they rely on uniform time discretization, implicitly assuming that equal steps in the sampling space correspond to comparable distributional evolution. However, this assumption is inconsistent with the intrinsic geometry of graph distribution spaces, where the sensitivity of distributional change varies significantly across different regimes.

This motivates a natural principle: sampling should proceed at a constant rate of distributional evolution along the reverse trajectory. We operationalize this idea by enforcing constant Fisher-Rao arc-length increments:

$$\Delta s_k^2 = C, \quad C > 0 \tag{9}$$

where C controls the target evolution rate.

Substituting Eq. 8 into the constraint Eq. 9 directly yields a drift increment magnitude:

$$\|\Delta \boldsymbol{f}_k\| = g_k \sqrt{\frac{C}{\Delta t}}. \tag{10}$$

We decompose $\Delta \boldsymbol{f}_k$ into its magnitude and unit direction:

$$\Delta \boldsymbol{f}_k = \|\Delta \boldsymbol{f}_k\| \cdot \boldsymbol{e} = g_k \sqrt{\frac{C}{\Delta t}} \cdot \boldsymbol{e}\,, \quad \|\boldsymbol{e}\| = 1 \tag{11}$$

where $\boldsymbol{e}$ is the unit direction vector.

Since $\Delta \boldsymbol{f}_k = \boldsymbol{f}_k - \boldsymbol{f}_{k-1}$, we have the NGD-solver update form of the drift term

$$\boldsymbol{f}_k \coloneqq \boldsymbol{f}_{k-1} + g_k \sqrt{\frac{C}{\Delta t}} \boldsymbol{e}\,. \tag{12}$$

Eq. 12 provides a justification for regulating sampling trajectories through drift-based correction.

As shown in Eq. 11, $\boldsymbol{e}$ and $\Delta \boldsymbol{f}_k$ have the same direction. To determine the update direction $\boldsymbol{e}$, we compute the model-predicted drift $\boldsymbol{f}_k^\theta$ and define a drift increment estimate by $\Delta \boldsymbol{f}^c \coloneqq \boldsymbol{f}_k^\theta - \boldsymbol{f}_{k-1}$. Then, the unit update direction is uniquely determined as

$$\boldsymbol{e} \coloneqq \Delta \boldsymbol{f}^c \,/\, \|\Delta \boldsymbol{f}^c\|\,. \tag{13}$$

For score-based diffusion models, the reverse-time drift admits the general form $\boldsymbol{f}_k^\theta = \vec{\boldsymbol{f}} - g_k^2\, \boldsymbol{s}_\theta(\boldsymbol{x}_k, t_k)$, where $\vec{\boldsymbol{f}}$ is the forward drift and $\boldsymbol{s}_\theta$ is the learned score.

This design preserves the transport direction implied by the learned reverse dynamics while regulating only the update magnitude through the Fisher-Rao arc-length constraint, thereby correcting the non-uniform geometric displacement along the sampling trajectory. In contrast to the standard sampler, where the learned model output $\boldsymbol{f}_k^\theta$ is directly used as the reverse-time drift without geometric awareness, NGD-solver replaces it with the geometry-aware drift $\boldsymbol{f}_k$ defined in Eq. 12. Consequently, NGD-solver only modifies the drift construction and can be incorporated into existing SDE-based samplers without altering either the stochastic component or the generative model.

The resulting drift correction is then applied to the Euler-Maruyama update in Eq. 2. As a result, the drift update becomes intrinsically modulated by the local Fisher geometry, reflecting the sensitivity of transition distributions. When g_k is small, the Fisher information becomes large, indicating that the transition distribution is highly sensitive to drift perturbations. In such regions, NGD-solver automatically reduces the allowable drift increment to prevent excessive distributional evolution. Conversely, when g_k is large, larger drift variations are permitted. Consequently, the sampling trajectory adapts its update magnitude according to the local geometry, yielding approximately uniform Fisher-Rao arc-length increments throughout the reverse diffusion process.

3.4 NGD-Solver Algorithm

The full procedure is summarized in Algorithm 1. We instantiate the proposed NGD-solver for graph generation, where a graph is represented as $\mathbf{G} = (\mathbf{X}, \mathbf{A})$ with node features $\mathbf{X} \in \mathbb{R}^{N\times F}$ and adjacency matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$. The sampling process evolves from a Gaussian prior p_{prior} to the target data distribution p_{data}.

Algorithm 1 NGD-solver algorithm.

Input: Time T, step number K, and the constant C.
Initialize: $\Delta t = T/K$, $(\mathbf{X}_0, \mathbf{A}_0) \sim p_{\text{prior}}$,
$\boldsymbol{f}_{X,0} \leftarrow \boldsymbol{f}^\theta_{X,0}$, and $\boldsymbol{f}_{A,0} \leftarrow \boldsymbol{f}^\theta_{A,0}$.

1: **for** $k = 1$ **to** $K-1$ **do**
2: $\quad t_k \leftarrow k\Delta t$;
3: $\quad \boldsymbol{f}^\theta_{X,k} = \vec{\boldsymbol{f}}_{X,k} - g_k^2\, \boldsymbol{s}_\theta(\mathbf{X}_k, t_k)$,
$\quad \boldsymbol{f}^\theta_{A,k} = \vec{\boldsymbol{f}}_{A,k} - g_k^2\, \boldsymbol{s}_\theta(\mathbf{A}_k, t_k)$;
4: $\quad \Delta \boldsymbol{f}^c_X = \boldsymbol{f}^\theta_{X,k} - \boldsymbol{f}_{X,k-1}$,
$\quad \Delta \boldsymbol{f}^c_A = \boldsymbol{f}^\theta_{A,k} - \boldsymbol{f}_{A,k-1}$;
5: $\quad \boldsymbol{e}_X = \Delta \boldsymbol{f}^c_X / \|\Delta \boldsymbol{f}^c_X\|$,
$\quad \boldsymbol{e}_A = \Delta \boldsymbol{f}^c_A / \|\Delta \boldsymbol{f}^c_A\|$;
6: $\quad \boldsymbol{f}_{X,k} = \boldsymbol{f}_{X,k-1} + g_k \sqrt{\frac{C}{\Delta t}}\, \boldsymbol{e}_X$,
$\quad \boldsymbol{f}_{A,k} = \boldsymbol{f}_{A,k-1} + g_k \sqrt{\frac{C}{\Delta t}}\, \boldsymbol{e}_A$;
7: $\quad \mathbf{X}_{k+1} \leftarrow \mathbf{X}_k + \boldsymbol{f}_{X,k}\Delta t + g_k\sqrt{\Delta t}\, \boldsymbol{\epsilon}^X$,
$\quad \mathbf{A}_{k+1} \leftarrow \mathbf{A}_k + \boldsymbol{f}_{A,k}\Delta t + g_k\sqrt{\Delta t}\, \boldsymbol{\epsilon}^A$;
8: **end for**

Output: Graph $(\mathbf{X}_T, \mathbf{A}_T)$.

At each reverse step, the score network provides the score functions for node and edge structures, from which the reverse drift fields are derived via the SDE formulation. NGD-solver then constructs the drift increment for the current step. To determine its direction, NGD-solver takes the difference between the current predicted drift and the previously applied drift, capturing the local evolution suggested by the learned dynamics. Its magnitude, instead of being inherited from this difference, is set to satisfy the constant Fisher-Rao arc-length constraint. The resulting increment is added to the previous drift field, yielding the corrected drift for the current step. This ensures every update advances by the same amount in distribution space. After the geometry-aware calibration, the corrected drifts are applied to update the joint state:

$$\begin{aligned} \mathbf{X}_{k+1} &= \mathbf{X}_k + \boldsymbol{f}_{X,k}\Delta t + g_k\sqrt{\Delta t}\boldsymbol{\epsilon}^X, \\ \mathbf{A}_{k+1} &= \mathbf{A}_k + \boldsymbol{f}_{A,k}\Delta t + g_k\sqrt{\Delta t}\boldsymbol{\epsilon}^A\,. \end{aligned} \tag{14}$$

For graph generation, a key challenge is that node and edge dynamics evolve at different rates: the corresponding drift fields differ substantially in scale and variation. If a single geometric constraint were applied to the joint state, one component would dominate the evolution. To address this, NGD-solver applies the constant arc-length constraint independently to the node and edge components, allowing each to adapt its update magnitude according to its own geometric sensitivity. Importantly, both components share the same target arc-length parameter C, ensuring that they progress at a consistent rate in distribution space and maintaining a balanced joint sampling trajectory.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Models. We evaluate NGD-solver on two representative graph generative models: GDSS (Jo et al., 2022), a score-based diffusion model and GruM (Jo et al., 2024), a graph diffusion bridge model. For both models, we directly use the pretrained checkpoints released by the original papers without modifying their architectures or training procedures. The only difference among the compared methods lies in the sampling solver. To further demonstrate the generality of our geometric principle beyond continuous diffusion models, we derive an extension of NGD-solver for discrete graph diffusion models in Appendix D.

Sampling. We apply the proposed Fisher-Rao geometric correction to both first-order and second-order numerical schemes. The correction operates exclusively during inference by adaptively rescaling the learned reverse-time drift update according to the Fisher-Rao arc length. It requires neither retraining nor model modification and incurs only negligible computational overhead. For the first-order setting, we compare the proposed NGD-solver with the original Euler-Maruyama solver (Maruyama, 1955) used in GDSS and GruM. In addition, we apply the same correction to the second-order Heun numerical schemes, yielding the second-order NGD-solver (2nd NGD-solver) for comparison against the standard 2nd Heun solver (Ascher & Petzold, 1998) (see Appendix B.1 for the 2nd NGD-solver algorithm). This experimental design serves two purposes. First, it demonstrates that the proposed Fisher-Rao correction is applicable to numerical schemes of different orders. Second, if the correction continues to improve performance even when applied to a second-order baseline, it would confirm that regulating sampling dynamics on the probability manifold provides complementary advantages beyond what can be achieved by simply increasing the order of numerical integration. For fair comparison, we preserve the original sampling configurations, including Langevin correction (Song et al., 2021b) when adopted in the baseline implementations. For all quantitative evaluations, we generate 10,000 samples from each method. Additional implementation details and experimental settings are provided in Appendix B.2.

Hyperparameters. NGD-solver introduces only one hyperparameter C, which controls the target Fisher-Rao arc length of each sampling step. For each dataset, we tune C within a predefined range and report the best-performing value. We find that performance improvements are consistently observed across a wide range of C values, suggesting that NGD-solver is robust to hyperparameter choices and does not require delicate tuning. A detailed sensitivity analysis is provided in Ablation Studies. The optimal values used in our experiments are listed in Appendix B.3 Table 5.

4.2 GENERAL GRAPH GENERATION

Datasets and Metrics. We evaluate NGD-solver on four widely used graph generation benchmarks covering geometric graphs, community-structured graphs and real-world networks. Specifically, Planar (Kipf & Welling, 2017) evaluates graph generation under geometric constraints, SBM (Holland et al., 1983) and Community-small (You et al., 2018) focus on community-structured graphs at different scales and Ego-small (Sen et al., 2008) provides a real-world citation-network benchmark. Planar contains 200 synthetic planar graphs with 64 nodes per graph. SBM (Stochastic Block Model) consists of 200 synthetic graphs with 2-5 communities and 20-40 nodes per community, where the intra-community and inter-community edge probabilities are 0.3 and 0.05, respectively. Ego-small contains 200 ego-centric subgraphs extracted from the Citeseer citation network, with graph sizes ranging from 4 to 18 nodes. Community-small consists of 100 synthetic community graphs with 12-20 nodes. We evaluate generation quality using the maximum mean discrepancy (MMD) (Costa & De Grave, 2010) between generated graphs and test set over four graph statistics: degree distribution (`Deg.`), clustering coefficient (`Clus.`), 4-node orbit counts (`Orbit`) and graph spectrum (`Spec.`) (Martinkus et al., 2022). These statistics characterize structural properties ranging from local connectivity patterns to global graph organization and lower MMD values indicate better agreement with the target graph distribution.

Results. Tables 1 and 2 summarize the graph generation results on all benchmark datasets. For the first-order setting, replacing the original Euler-Maruyama solver with the proposed NGD-solver consistently improves generation quality across all datasets under both GDSS and GruM. The improvements span multiple graph statistics, including degree distributions, clustering coefficients,

Table 1: Comparison of generation performance on general graph datasets (Planar and SBM) using GDSS (Jo et al., 2022) and GruM (Jo et al., 2024). We compare the proposed NGD-solver against the first-order Euler-Maruyama solver and its second-order variant against the standard 2nd Heun solver. Both NGD-solver variants consistently improve graph structural metrics across both models.

Method	Solver	Planar				SBM			
		Deg.↓	Clus.↓	Orbit.↓	Spec.↓	Deg.↓	Clus.↓	Orbit.↓	Spec.↓
GDSS	Euler-Maruyama solver	0.3076	0.3418	0.3375	0.5887	0.2549	0.4308	0.1228	0.2065
	NGD-solver (Ours)	0.2713	0.3367	0.2758	0.5815	0.2720	0.3950	0.1216	0.2009
	2nd Heun solver	0.3120	0.3400	0.3433	0.5868	0.2577	0.4446	0.1216	0.2081
	2nd **NGD-solver** (Ours)	0.2877	0.3363	0.3088	0.5814	0.2886	0.3985	0.1202	0.2062
GruM	Euler-Maruyama solver	0.0002	0.0300	0.0013	0.0060	0.0006	0.0498	0.0455	0.0051
	NGD-solver (Ours)	0.0001	0.0290	0.0009	0.0052	0.0006	0.0488	0.0420	0.0043
	2nd Heun solver	0.0001	0.0340	0.0010	0.0054	0.0008	0.0489	0.0444	0.0052
	2nd **NGD-solver** (Ours)	0.0001	0.0216	0.0007	0.0050	0.0007	0.0485	0.0000	0.0042

Table 2: Comparison of generation performance on general graph datasets (Community and Ego-small) using GDSS (Jo et al., 2022) and GruM (Jo et al., 2024). We compare the NGD-solver against the first-order Euler-Maruyama solver and its second-order variant against the standard 2nd Heun solver. Both NGD-solver variants consistently improve structural metrics across both models.

Method	Solver	Community				Ego-small			
		Deg.↓	Clus.↓	Orbit↓	Spec.↓	Deg.↓	Clus.↓	Orbit↓	Spec.↓
GDSS	Euler-Maruyama solver	0.1328	0.1167	0.0179	0.2022	0.0229	0.0098	0.0131	0.0409
	NGD-solver (Ours)	0.0698	0.0583	0.0097	0.2014	0.0219	0.0087	0.0133	0.0359
	2nd Heun solver	0.1321	0.1237	0.0193	0.2021	0.0229	0.0098	0.0131	0.0409
	2nd **NGD-solver** (Ours)	0.0850	0.0534	0.0101	0.2017	0.0219	0.0087	0.0133	0.0359
GruM	Euler-Maruyama solver	0.0044	0.1343	0.0609	0.0419	0.0080	0.0610	0.0231	0.0145
	NGD-solver (Ours)	0.0036	0.1181	0.0574	0.0350	0.0059	0.0601	0.0237	0.0132
	2nd Heun solver	0.0032	0.2264	0.0811	0.0621	0.0072	0.0629	0.0243	0.0133
	2nd **NGD-solver** (Ours)	0.0022	0.1946	0.0751	0.0486	0.0069	0.0612	0.0238	0.0129

orbit counts and spectral properties, demonstrating that the proposed Fisher-Rao correction benefits both local structural patterns and global graph organization. Simply increasing the numerical integration order does not consistently improve graph generation performance. Unlike continuous Euclidean data, graph generation involves highly discrete and combinatorial probability distributions, whose evolution exhibits complex and non-uniform geometric characteristics. Although the 2nd Heun solver provides a higher-order approximation of the reverse dynamics, its improvements over the Euler-Maruyama solver are generally marginal and may even degrade certain graph statistics. In contrast, the proposed 2nd NGD-solver consistently improves upon the standard 2nd Heun solver across both diffusion models and all benchmark datasets. These observations indicate that the primary limitation of diffusion sampling is not the numerical integration order itself, but the mismatch between Euclidean sampling trajectories and the intrinsically curved probability manifold. They further support our central hypothesis that explicitly incorporating the Fisher-Rao geometry encourages a more uniform evolution of probability distributions throughout the reverse process, resulting in high-quality samples that more faithfully match the target graph distribution.

4.3 Molecule Graph Generation

Datasets and Metrics. We evaluate NGD-solver on two widely used molecular graph generation benchmarks: QM9 (Ramakrishnan et al., 2014) and ZINC250k (Irwin et al., 2012). QM9 contains 133,885 small organic molecules composed of C, O, N and F atoms, with at most 9 heavy atoms per molecule. ZINC250k contains 249,455 drug-like molecules with up to 38 heavy atoms and more diverse chemical structures. Molecules in both datasets are represented as graphs with three bond

Table 3: Molecule generation performance on QM9 and ZINC250k using GDSS (Jo et al., 2022) and GruM (Jo et al., 2024). NGD-solver consistently improves chemical validity (Valid), distributional and structural fidelity (FCD and NSPDK) over the Euler-Maruyama and 2nd Heun solvers.

		QM9			ZINC250k		
Method	Solver	Valid (%) ↑	FCD ↓	NSPDK ↓	Valid (%) ↑	FCD ↓	NSPDK ↓
GDSS	Euler-Maruyama solver	94.96	2.551	0.0035	96.06	15.646	0.0178
	NGD-solver (Ours)	96.84	2.009	0.0022	95.27	13.592	0.0139
	2nd Heun solver	94.86	2.548	0.0035	96.52	15.562	0.0184
	2nd **NGD-solver** (Ours)	96.82	2.030	0.0021	95.45	13.459	0.0134
GruM	Euler-Maruyama solver	99.43	0.107	0.0002	98.34	2.207	0.0016
	NGD-solver (Ours)	99.45	0.103	0.0002	98.55	2.128	0.0015
	2nd Heun solver	99.45	0.106	0.0002	98.38	2.185	0.0015
	2nd **NGD-solver** (Ours)	99.48	0.104	0.0002	98.66	2.093	0.0014

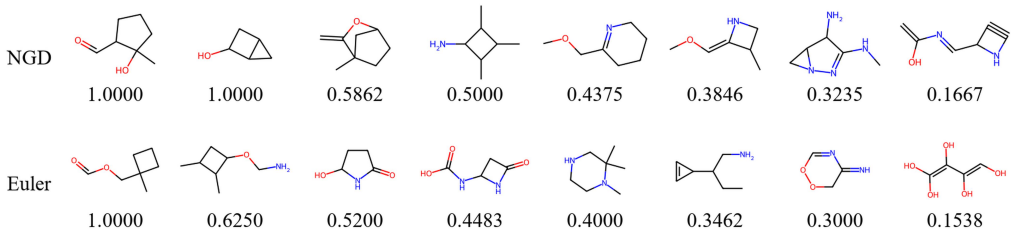

Figure 1: Visualization of molecules generated by NGD-solver and Euler-Maruyama solver on QM9 using the GDSS model. Both methods each generate 10,000 molecules and every 1,250-th molecule is selected for visualization. The similarity score shown below each molecule represents the maximum Tanimoto similarity to its closest molecule in the training set, computed using Morgan fingerprints (radius = 2, 1,024 bits).

types: single, double and triple bonds. We report three evaluation metrics: Validity (Valid) (Shi et al., 2020; Luo et al., 2021), Fréchet ChemNet Distance (FCD) (Preuer et al., 2018) and NSPDK MMD (NSPDK) (Costa & De Grave, 2010). Valid measures the percentage of chemically valid molecules generated without any post-hoc correction. FCD evaluates the distance between generated molecules and the test set in the ChemNet feature space, reflecting similarity in chemical properties. NSPDK computes the maximum mean discrepancy between generated and test molecular graphs using the Neighborhood Subgraph Pairwise Distance Kernel, assessing structural similarity from both node and edge perspectives. Lower values are better for both FCD and NSPDK, while higher values indicate better Valid.

Results. Table 3 summarizes the molecular graph generation results on QM9 and ZINC250k. For the first-order setting, replacing the original Euler-Maruyama solver with the proposed NGD-solver consistently and substantially improves molecular generation quality under both GDSS and GruM, demonstrating that the proposed sampling strategy effectively generalizes to molecular graph generation. Consistent with the observations on general graph datasets, increasing the numerical integration order alone brings only marginal improvements over the Euler-Maruyama solver. In contrast, the proposed 2nd NGD-solver consistently outperforms the standard 2nd Heun solver across both diffusion models and both molecular benchmarks, indicating that the proposed geometric correction provides complementary benefits beyond higher-order numerical integration. The improvements are consistently reflected across multiple evaluation metrics. Both NGD-solver variants reduce FCD and NSPDK, indicating improved distributional fidelity and structural similarity to the reference molecular distribution, while preserving or further improving molecular validity. This behavior is particularly important for molecular generation, where accurate distribution matching and chemically valid graph structures must be satisfied simultaneously. To further illustrate the generation quality, Figure 1 visualizes molecules generated by GDSS on QM9. Compared with the Euler-Maruyama solver, NGD-solver consistently generates molecules with higher structural similarity to the training

distribution, which is consistent with the quantitative improvements reported in Table 3. We further evaluate the robustness of NGD-solver across different random seeds in Appendix C.1.

4.4 Ablation Studies

Sampling Efficiency. To evaluate NGD-solver under different computational budgets, we conduct experiments on QM9 using GDSS with varying numbers of sampling steps. Table 4 reports the results. The same Fisher-Rao arc length parameter, selected under the 1000-step setting, is used for all step counts without any retuning, enabling a direct assessment of robustness across different discretization budgets. Across all step counts, NGD-solver consistently outperforms the original Euler-Maruyama baseline on `Valid`, `FCD`, and `NSPDK` metrics. The performance gap generally widens as the number of sampling steps increases, with NGD-solver achieving progressively better validity and distributional fidelity. Notably, with only 400 steps, NGD-solver already surpasses the 1000-step Euler-Maruyama baseline, demonstrating that geometric control can achieve comparable or better generation quality with substantially fewer sampling steps. These gains come with only marginal computational overhead, as detailed in Appendix C.2.

Table 4: Comparison of NGD-solver and Euler-Maruyama solver across different sampling steps on QM9 (GDSS).

N (Steps)	Solver	Valid (%) ↑	FCD ↓	NSPDK ↓
100	Euler-Maruyama	89.84	3.069	0.0053
	NGD-solver	89.84	3.034	0.0051
200	Euler-Maruyama	92.78	2.868	0.0044
	NGD-solver	94.14	2.662	0.0035
400	Euler-Maruyama	94.61	2.721	0.0038
	NGD-solver	95.74	2.485	0.0030
600	Euler-Maruyama	94.81	2.640	0.0037
	NGD-solver	96.26	2.263	0.0024
800	Euler-Maruyama	94.97	2.620	0.0036
	NGD-solver	96.54	2.152	0.0022
1000	Euler-Maruyama	94.96	2.551	0.0035
	NGD-solver	96.84	2.009	0.0022

Sensitivity Analysis. We evaluate the sensitivity of NGD-solver to the Fisher-Rao arc length C on QM9 using GDSS. As shown in Figure 2, performance remains stable over a broad range of C. Increasing C from small values improves generation quality, with the best `FCD` around $C = 0.0058$, while further increasing C gradually degrades performance. This trend reflects the trade-off between insufficient and overly aggressive distributional evolution: small C results in limited probability updates, while excessively large C causes overly aggressive transitions. A wide neighborhood around the optimum yields comparable results, indicating that NGD-solver is not sensitive to the exact choice of C and does not require delicate hyperparameter tuning. All tested values of C outperform the Euler-Maruyama solver (dashed line), suggesting that the performance gains stem primarily from the proposed Fisher-Rao-guided sampling strategy rather than careful hyperparameter tuning.

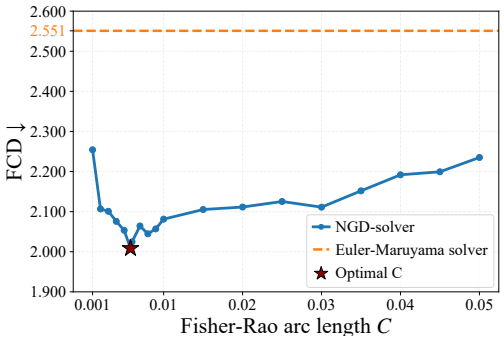

Figure 2: Sensitivity analysis of NGD-solver to the Fisher-Rao arc length C on QM9 with GDSS. The horizontal dashed line indicates the FCD of Euler-Maruyama.

5 Conclusion

In this paper, we identify a limitation of existing diffusion samplers for graph generation: uniform time discretization does not guarantee uniform distributional evolution due to the curved geometry of the probability manifold. We introduce a natural sampling principle that enforces a constant rate of distributional change, formalized through the Fisher-Rao metric as equal arc-length increments along the reverse trajectory. Based on this principle, we propose NGD-solver, a plug-and-play inference-time method that enforces balanced distributional evolution through Fisher-Rao arc-length constraints, without requiring retraining or modifications to the underlying generative models. Extensive experiments on molecular and general graph benchmarks demonstrate that NGD-solver consistently improves generation quality and structural fidelity. For future work, we aim to further explore this geometric perspective in broader graph generation scenarios.

AI USE STATEMENT

We used generative AI tools solely for language polishing. We have reviewed all AI-assisted work. Specifically, all AI-assisted language polishing was reviewed and approved by the authors. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work proposes an inference-time sampler for graph diffusion models. Experiments are conducted on publicly available graph benchmarks (QM9, ZINC250k, Planar, SBM, Community, Ego-small). No human subjects or personally identifiable information are involved. The method introduces no new data collection or privacy risks, and we foresee no direct harmful applications. All datasets and pretrained models are publicly available and used under their respective licenses.

REPRODUCIBILITY STATEMENT

The proposed NGD-solver is an inference-time method requiring no retraining; all experiments use publicly released GDSS and GruM checkpoints. Algorithm 1 provides the complete procedure. Sampling settings, including the Fisher-Rao arc-length parameter C and step counts, along with evaluation metrics, are described in the experimental section and the appendix. All assumptions and derivations are provided in the appendix. We will release the code upon acceptance to reproduce all tables and figures. Datasets are standard publicly available benchmarks, used with their official preprocessing pipelines.

REFERENCES

Shun-ichi Amari. *Information Geometry and Its Applications*, volume 194 of *Applied Mathematical Sciences*. Springer, 2016.

Uri M Ascher and Linda R Petzold. *Computer methods for ordinary differential equations and differential-algebraic equations*. SIAM, 1998.

Fabrizio Costa and Kurt De Grave. Fast neighborhood subgraph pairwise distance kernel. In *ICML*, pp. 255–262, 2010.

Xingcheng Fu, Yisen Gao, Yuecen Wei, Qingyun Sun, Hao Peng, Jianxin Li, and Xianxian Li. Hyperbolic geometric latent diffusion model for graph generation. In *ICML*, pp. 14102–14124, 2024.

Yisen Gao, Xingcheng Fu, Qingyun Sun, Jianxin Li, and Xianxian LI. Toward a unified geometry understanding : Riemannian diffusion framework for graph generation and prediction. In *NeurIPS*, volume 38, pp. 74249–74275, 2025.

Mark Girolami and Ben Calderhead. Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, pp. 123–214, 2011.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, volume 33, pp. 6840–6851, 2020.

Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983.

Emiel Hoogeboom, Vıctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3D. In *ICML*, pp. 8867–8887, 2022.

John J Irwin, Teague Sterling, Michael M Mysinger, Erin S Bolstad, and Ryan G Coleman. ZINC: a free tool to discover chemistry for biology. *Journal of Chemical Information and Modeling*, 52 (7):1757–1768, 2012.

Jaehyeong Jo, Seul Lee, and Sung Ju Hwang. Score-based generative modeling of graphs via the system of stochastic differential equations. In *ICML*, pp. 10362–10383, 2022.

Jaehyeong Jo, Dongki Kim, and Sung Ju Hwang. Graph generation with diffusion mixture. In *ICML*, volume 41, pp. 22371–22405, 2024.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *NeurIPS*, volume 35, pp. 26565–26577, 2022.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

Mingxiao Li, Tingyu Qu, Ruicong Yao, Wei Sun, and Marie-Francine Moens. Alleviating exposure bias in diffusion models through sampling with shifted time steps. In *ICLR*, volume 12, pp. 16816–16838, 2024.

Tengyuan Liang, Tomaso Poggio, Alexander Rakhlin, and James Stokes. Fisher-Rao metric, geometry, and complexity of neural networks. In *AISTATS*, pp. 888–896, 2019.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In *NeurIPS*, volume 35, pp. 5775–5787, 2022.

Yuhui Lu, Wenjing Liu, and Kun Zhan. Information-geometric adaptive sampling for graph diffusion. In *ICML*, volume 43, 2026.

Youzhi Luo, Keqiang Yan, and Shuiwang Ji. GraphDF: A discrete flow model for molecular graph generation. In *ICML*, pp. 7192–7203, 2021.

Karolis Martinkus, Andreas Loukas, Nathanaël Perraudin, and Roger Wattenhofer. Spectre: Spectral conditioning helps to overcome the expressivity limits of one-shot graph generators. In *ICML*, pp. 15159–15179, 2022.

Gisiro Maruyama. Continuous markov processes and stochastic equations. *Rendiconti del Circolo Matematico di Palermo*, 4(1):48–90, 1955.

Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, and Stefano Ermon. Permutation invariant graph generation via score-based generative modeling. In *AISTATS*, pp. 4474–4484, 2020.

Yong-Hyun Park, Chieh-Hsin Lai, Satoshi Hayakawa, Yuhta Takida, and Yuki Mitsufuji. Jump your steps: Optimizing sampling schedule of discrete diffusion models. In *ICLR*, volume 13, pp. 96272–96300, 2025.

Razvan Pascanu and Yoshua Bengio. Revisiting natural gradient for deep networks. In *ICLR*, 2014.

Kristina Preuer, Philipp Renz, Thomas Unterthiner, Sepp Hochreiter, and Gunter Klambauer. Fréchet chemnet distance: a metric for generative models for molecules in drug discovery. *Journal of Chemical Information and Modeling*, 58(9):1736–1741, 2018.

Yiming Qin, Manuel Madeira, Dorina Thanou, and Pascal Frossard. DeFoG: discrete flow matching for graph generation. In *ICML*, volume 42, pp. 50269–50326, 2025.

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.

Amirmojtaba Sabour, Sanja Fidler, and Karsten Kreis. Align your steps: Optimizing sampling schedules in diffusion models. In *ICML*, volume 41, pp. 42947–42975, 2024.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–93, 2008.

Chence Shi, Minkai Xu, Zhaocheng Zhu, Weinan Zhang, Ming Zhang, and Jian Tang. GraphAF: A flow-based autoregressive model for molecular graph generation. In *ICLR*, 2020.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021a.

Kaiyu Song and Hanjiang Lai. Fisher information improved training-free conditional diffusion model. *arXiv preprint arXiv:2404.18252*, 2024.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021b.

Clément Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. DiGress: Discrete denoising diffusion for graph generation. In *ICLR*, 2023.

Jiaxuan You, Rex Ying, Xiang Ren, William Hamilton, and Jure Leskovec. Graphrnn: Generating realistic graphs with deep auto-regressive models. In *ICML*, pp. 5694–5703, 2018.

Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. In *ICLR*, 2023.

Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. UniPC: A unified predictor-corrector framework for fast sampling of diffusion models. In *NeurIPS*, volume 36, pp. 49842–49869, 2023.

Appendix

This appendix provides detailed derivations, experimental configurations, additional results, and an extension of the proposed NGD-solver to discrete diffusion models. Specifically, Section A present the theoretical derivations underlying the Fisher-Rao geometry used in the main paper. Section B describes the experimental setup, including the second-order solver variant, implementation details, and hyperparameter configurations. Section C provide additional experimental results on statistical significance and runtime overhead. Finally, Section D extends the proposed method to discrete diffusion models, demonstrating the generality of the natural sampling principle.

A Detailed Derivations

A.1 Derivation of the Fisher Information Equivalence

We derive the equivalence between the two common forms of the Fisher information matrix:

$$\mathcal{I}(\boldsymbol{f}) = \mathbb{E}_p\left[\nabla_{\boldsymbol{f}} \log p \, \nabla_{\boldsymbol{f}}^\top \log p\right] = -\mathbb{E}_p\left[\nabla_{\boldsymbol{f}}^2 \log p\right]. \tag{4}$$

Let $p(\boldsymbol{x}; \boldsymbol{f})$ be a probability density parameterized by $\boldsymbol{f} \in \mathbb{R}^d$, satisfying $\int p(\boldsymbol{x}; \boldsymbol{f})\, \mathrm{d}\boldsymbol{x} = 1$ for all $\boldsymbol{f}$.

First, differentiating the normalization condition with respect to $\boldsymbol{f}$ gives

$$\nabla_{\boldsymbol{f}} \int p(\boldsymbol{x}; \boldsymbol{f})\, \mathrm{d}\boldsymbol{x} = \int \nabla_{\boldsymbol{f}} p(\boldsymbol{x}; \boldsymbol{f})\, \mathrm{d}\boldsymbol{x} = 0. \tag{15}$$

Since $\nabla_{\boldsymbol{f}} p = p \cdot \nabla_{\boldsymbol{f}} \log p$, we obtain the score expectation identity

$$\mathbb{E}_p\left[\nabla_{\boldsymbol{f}} \log p\right] = 0. \tag{16}$$

Next, differentiating the normalization condition a second time yields

$$\nabla_{\boldsymbol{f}}^2 \int p(\boldsymbol{x}; \boldsymbol{f})\, \mathrm{d}\boldsymbol{x} = \int \nabla_{\boldsymbol{f}}^2 p(\boldsymbol{x}; \boldsymbol{f})\, \mathrm{d}\boldsymbol{x} = 0. \tag{17}$$

Using $\nabla_{\boldsymbol{f}}^2 p = p\left(\nabla_{\boldsymbol{f}}^2 \log p + (\nabla_{\boldsymbol{f}} \log p)(\nabla_{\boldsymbol{f}}^\top \log p)\right)$, we have

$$\mathbb{E}_p\left[\nabla_{\boldsymbol{f}}^2 \log p\right] + \mathbb{E}_p\left[(\nabla_{\boldsymbol{f}} \log p)(\nabla_{\boldsymbol{f}}^\top \log p)\right] = 0. \tag{18}$$

Rearranging yields the equivalence in Eq. equation 4.

A.2 Derivation of the KL Divergence Approximation

We derive the local approximation

$$\mathrm{KL}\big(p_{\boldsymbol{f}} \,\|\, p_{\boldsymbol{f}+\Delta\boldsymbol{f}}\big) \approx \frac{1}{2}\Delta s^2, \tag{19}$$

$$\Delta s^2 = \Delta\boldsymbol{f}^\top \mathcal{I}(\boldsymbol{f}) \Delta\boldsymbol{f}. \tag{5}$$

Let $p_{\boldsymbol{f}} := p(\cdot; \boldsymbol{f})$ and $p_{\boldsymbol{f}+\Delta\boldsymbol{f}} := p(\cdot; \boldsymbol{f} + \Delta\boldsymbol{f})$. The KL divergence from $p_{\boldsymbol{f}}$ to $p_{\boldsymbol{f}+\Delta\boldsymbol{f}}$ is

$$\mathrm{KL}\big(p_{\boldsymbol{f}} \,\|\, p_{\boldsymbol{f}+\Delta\boldsymbol{f}}\big) = \mathbb{E}_{p_{\boldsymbol{f}}}\left[\log p_{\boldsymbol{f}} - \log p_{\boldsymbol{f}+\Delta\boldsymbol{f}}\right]. \tag{20}$$

Expanding $\log p_{\boldsymbol{f}+\Delta\boldsymbol{f}}$ around $\boldsymbol{f}$ to second order gives

$$\log p_{\boldsymbol{f}+\Delta\boldsymbol{f}} = \log p_{\boldsymbol{f}} + \frac{1}{2}\Delta\boldsymbol{f}^\top (\nabla_{\boldsymbol{f}}^2 \log p_{\boldsymbol{f}}) \Delta\boldsymbol{f} + o(\|\Delta\boldsymbol{f}\|^2). \tag{21}$$

Substituting into the KL definition yields

$$\begin{aligned}\mathrm{KL}\big(p_{\boldsymbol{f}} \,\|\, p_{\boldsymbol{f}+\Delta\boldsymbol{f}}\big) &\approx \mathbb{E}_{p_{\boldsymbol{f}}}\left[-(\nabla_{\boldsymbol{f}} \log p_{\boldsymbol{f}})^\top \Delta\boldsymbol{f} - \frac{1}{2}\Delta\boldsymbol{f}^\top (\nabla_{\boldsymbol{f}}^2 \log p_{\boldsymbol{f}}) \Delta\boldsymbol{f}\right] \\ &= -\left(\mathbb{E}_{p_{\boldsymbol{f}}}[\nabla_{\boldsymbol{f}} \log p_{\boldsymbol{f}}]\right)^\top \Delta\boldsymbol{f} - \frac{1}{2}\Delta\boldsymbol{f}^\top \left(\mathbb{E}_{p_{\boldsymbol{f}}}[\nabla_{\boldsymbol{f}}^2 \log p_{\boldsymbol{f}}]\right) \Delta\boldsymbol{f}. \end{aligned} \tag{22}$$

By the score expectation identity $\mathbb{E}_{p_{\boldsymbol{f}}}[\nabla_{\boldsymbol{f}} \log p_{\boldsymbol{f}}] = 0$ from Eq. equation 16, the first-order term vanishes, leaving

$$\mathrm{KL}\big(p_{\boldsymbol{f}} \,\|\, p_{\boldsymbol{f}+\Delta\boldsymbol{f}}\big) \approx -\frac{1}{2}\Delta\boldsymbol{f}^\top \left(\mathbb{E}_{p_{\boldsymbol{f}}}[\nabla^2_{\boldsymbol{f}} \log p_{\boldsymbol{f}}]\right) \Delta\boldsymbol{f}. \tag{23}$$

Using the Fisher information equivalence $\mathcal{I}(\boldsymbol{f}) = -\mathbb{E}_{p_{\boldsymbol{f}}}[\nabla^2_{\boldsymbol{f}} \log p_{\boldsymbol{f}}]$ established in Appendix A.1, we obtain

$$\mathrm{KL}\big(p_{\boldsymbol{f}} \,\|\, p_{\boldsymbol{f}+\Delta\boldsymbol{f}}\big) \approx \frac{1}{2}\Delta\boldsymbol{f}^\top \mathcal{I}(\boldsymbol{f}) \Delta\boldsymbol{f}. \tag{24}$$

Finally, by the definition of the Fisher-Rao arc length in Eq. (5) , we have

$$\mathrm{KL}\big(p_{\boldsymbol{f}} \,\|\, p_{\boldsymbol{f}+\Delta\boldsymbol{f}}\big) \approx \frac{1}{2}\Delta s^2. \tag{25}$$

This result shows that the Fisher-Rao arc length locally approximates the KL divergence between neighboring transition distributions, justifying its interpretation as a principled measure of distributional evolution.

A.3 Fisher-Rao Geometry of Gaussian Transition Kernels

We derive the Fisher-Rao geometry induced by the Gaussian transition kernel in Eq. equation 3. We first consider the complete parameterization, where both the drift field and diffusion coefficient are treated as unknown parameters. Based on the resulting Fisher information matrix and its induced Fisher-Rao metric, we then show that the diffusion contribution is asymptotically negligible, leading to the reduced drift-only metric adopted in the main text.

Fisher-Rao metric under joint parameterization. For notational simplicity, we omit the step index and define the residual

$$\boldsymbol{z} = \boldsymbol{x}_{k+1} - \boldsymbol{x}_k - \boldsymbol{f}_k \Delta t,$$

which satisfies $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, g^2 \Delta t\, \boldsymbol{I})$.

The Gaussian transition density in Eq. equation 3 is then written as

$$p(\boldsymbol{z}) = \frac{1}{(2\pi g^2 \Delta t)^{D/2}} \exp\!\left(-\frac{\boldsymbol{z}^\top \boldsymbol{z}}{2g^2 \Delta t}\right), \tag{26}$$

where D denotes the dimension of the state variable $\boldsymbol{x}$.

The corresponding log-likelihood is

$$L = \log p(\boldsymbol{z}) = -\frac{D}{2}\log(2\pi g^2 \Delta t) - \frac{\boldsymbol{z}^\top \boldsymbol{z}}{2g^2 \Delta t}. \tag{27}$$

We treat both the drift and diffusion coefficient as parameterized quantities and define the joint parameter vector

$$\theta = [\boldsymbol{f}^\top, g]^\top.$$

Differentiating the log-likelihood with respect to the drift parameter gives

$$\nabla_{\boldsymbol{f}} L = -\frac{1}{g^2 \Delta t}\left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{f}}\right)^\top \boldsymbol{z} = -\frac{1}{g^2 \Delta t}(-\Delta t\, \boldsymbol{I})\boldsymbol{z} = \frac{\boldsymbol{z}}{g^2}. \tag{28}$$

The score function with respect to the diffusion coefficient is obtained as

$$\nabla_g L = \frac{\partial}{\partial g}\left(-\frac{D}{2}\log(2\pi g^2 \Delta t) - \frac{\boldsymbol{z}^\top \boldsymbol{z}}{2g^2 \Delta t}\right) = -\frac{D}{g} + \frac{\boldsymbol{z}^\top \boldsymbol{z}}{g^3 \Delta t}. \tag{29}$$

Combining this result with the drift score in Eq. equation 6, the joint score vector becomes

$$\nabla_\theta L = \begin{bmatrix} \nabla_{\boldsymbol{f}} L \\ \nabla_g L \end{bmatrix} = \begin{bmatrix} \dfrac{\boldsymbol{z}}{g^2} \\ -\dfrac{D}{g} + \dfrac{\boldsymbol{z}^\top \boldsymbol{z}}{g^3 \Delta t} \end{bmatrix}. \tag{30}$$

The Fisher information matrix with respect to the joint parameter is defined as

$$\mathcal{I}(\theta) = \mathbb{E}_p\left[\nabla_\theta L \nabla_\theta^\top L\right]. \tag{31}$$

Evaluating the drift-drift, drift-diffusion and diffusion-diffusion blocks separately using

$$\mathbb{E}_p[\boldsymbol{z}] = \mathbf{0}, \qquad \mathbb{E}_p[\boldsymbol{z}\boldsymbol{z}^\top] = g^2\Delta t\, \boldsymbol{I}, \qquad \mathbb{E}_p[(\boldsymbol{z}^\top \boldsymbol{z})^2] = D(D+2)g^4\Delta t^2,$$

yields

$$\mathcal{I}(\theta) = \begin{bmatrix} \dfrac{\Delta t}{g^2}\boldsymbol{I} & \mathbf{0} \\ \mathbf{0} & \dfrac{2D}{g^2} \end{bmatrix}, \tag{32}$$

where the upper-left block corresponds to the drift parameter, the lower-right block corresponds to the diffusion coefficient and the off-diagonal blocks vanish because they involve odd-order moments of the centered Gaussian residual.

According to information geometry, the Fisher information matrix induces the local Riemannian metric on the statistical manifold. The resulting Fisher-Rao line element is therefore

$$\begin{aligned} \mathrm{d}s^2 &= \mathrm{d}\theta^\top \mathcal{I}(\theta)\mathrm{d}\theta \\ &= \frac{\Delta t}{g^2}\|\mathrm{d}\boldsymbol{f}\|_2^2 + \frac{2D}{g^2}(\mathrm{d}g)^2. \end{aligned} \tag{33}$$

Asymptotic dominance of the drift contribution. Equation equation 33 shows that the Fisher-Rao metric consists of two components associated with the drift field and the diffusion coefficient. To determine whether the diffusion-related component contributes to the leading-order Fisher geometry, we analyze the asymptotic behavior of the two parameter variations in the infinitesimal limit $\mathrm{d}t \to 0$, corresponding to the small-step regime of numerical discretization.

A key distinction between the two variables is that the diffusion coefficient is prescribed by the noise schedule and depends only on time, whereas the drift field is a state-dependent function

$$g = g_t, \qquad \boldsymbol{f} = \boldsymbol{f}(\boldsymbol{x}_t, t), \tag{34}$$

where the state $\boldsymbol{x}_t$ evolves according to the reverse-time SDE in Eq. (1).

Since the diffusion coefficient depends only on time,

$$\mathrm{d}g = \dot{g}_t\,\mathrm{d}t, \tag{35}$$

its infinitesimal variation satisfies

$$|\mathrm{d}g| = \mathcal{O}(\mathrm{d}t). \tag{36}$$

In contrast, the variation of the drift field is affected by both its explicit time dependence and the stochastic evolution of $\boldsymbol{x}_t$. Therefore, its infinitesimal variation is characterized by Itô's lemma. Applying Itô's lemma gives

$$\mathrm{d}\boldsymbol{f} = \underbrace{\left(\frac{\partial \boldsymbol{f}}{\partial t} + \mathbf{J}_{\boldsymbol{f}}\boldsymbol{f} + \frac{1}{2}g^2\Delta_{\boldsymbol{x}}\boldsymbol{f}\right)}_{\text{deterministic component}}\mathrm{d}t + \underbrace{g\mathbf{J}_{\boldsymbol{f}}\mathrm{d}\overline{\boldsymbol{w}}}_{\text{stochastic component}}, \tag{37}$$

where $\mathbf{J}_{\boldsymbol{f}} = \nabla_{\boldsymbol{x}}\boldsymbol{f}$ denotes the Jacobian of the drift field. The deterministic component is of order $\mathcal{O}(\mathrm{d}t)$. Its squared magnitude is therefore of order $\mathcal{O}(\mathrm{d}t^2)$. For the stochastic component, using $\mathbb{E}[\mathrm{d}\overline{\boldsymbol{w}}\,\mathrm{d}\overline{\boldsymbol{w}}^\top] = \boldsymbol{I}\,\mathrm{d}t$, together with Itô's isometry yields

$$\mathbb{E}\left[\|g\mathbf{J}_{\boldsymbol{f}}\mathrm{d}\overline{\boldsymbol{w}}\|_2^2\right] = g^2\,\mathrm{Tr}(\mathbf{J}_{\boldsymbol{f}}^\top\mathbf{J}_{\boldsymbol{f}})\mathrm{d}t + \mathcal{O}(\mathrm{d}t^2), \tag{38}$$

whose mean-square magnitude is therefore of order $\mathcal{O}(\mathrm{d}t)$. Therefore, the deterministic component contributes at order $\mathcal{O}(\mathrm{d}t^2)$, whereas the stochastic component contributes at order $\mathcal{O}(\mathrm{d}t)$. Since $\mathrm{d}t \gg \mathrm{d}t^2$ as $\mathrm{d}t \to 0$, the stochastic component dominates the infinitesimal variation of the drift field, yielding

$$\|\mathrm{d}\boldsymbol{f}\|_2 = \mathcal{O}(\sqrt{\mathrm{d}t}). \tag{39}$$

Algorithm 2 Second-order NGD-solver algorithm (2nd NGD-solver).

Input: Time T, step number K, and the constant C.
Initialize: $\Delta t = T/K$, $(\mathbf{X}_0, \mathbf{A}_0) \sim p_{\text{prior}}$, $\boldsymbol{f}_{X,0} \leftarrow \boldsymbol{f}^{\theta}_{X,0}$, and $\boldsymbol{f}_{A,0} \leftarrow \boldsymbol{f}^{\theta}_{A,0}$.

for $k = 1$ **to** $K-1$ **do**

$t_k = k\Delta t$;

$\boldsymbol{f}^{\theta,(1)}_{X,k} = \vec{\boldsymbol{f}}_{X,k} - g_k^2\, \boldsymbol{s}_\theta(\mathbf{X}_k, t_k),\ \boldsymbol{f}^{\theta,(1)}_{A,k} = \vec{\boldsymbol{f}}_{A,k} - g_k^2\, \boldsymbol{s}_\theta(\mathbf{A}_k, t_k)$;

$\tilde{\mathbf{X}}_{k+1} = \mathbf{X}_k + \boldsymbol{f}^{\theta,(1)}_{X,k}\Delta t,\ \tilde{\mathbf{A}}_{k+1} = \mathbf{A}_k + \boldsymbol{f}^{\theta,(1)}_{A,k}\Delta t$;

$\boldsymbol{f}^{\theta,(2)}_{X,k} = \vec{\boldsymbol{f}}_{X,k} - g_k^2\, \boldsymbol{s}_\theta(\tilde{\mathbf{X}}_{k+1}, t_k + \Delta t),\ \boldsymbol{f}^{\theta,(2)}_{A,k} = \vec{\boldsymbol{f}}_{A,k} - g_k^2\, \boldsymbol{s}_\theta(\tilde{\mathbf{A}}_{k+1}, t_k + \Delta t)$;

$\bar{\boldsymbol{f}}_{X,k} = \frac{1}{2}\big(\boldsymbol{f}^{\theta,(1)}_{X,k} + \boldsymbol{f}^{\theta,(2)}_{X,k}\big),\ \bar{\boldsymbol{f}}_{A,k} = \frac{1}{2}\big(\boldsymbol{f}^{\theta,(1)}_{A,k} + \boldsymbol{f}^{\theta,(2)}_{A,k}\big)$;

$\Delta\boldsymbol{f}^{c}_{X} = \bar{\boldsymbol{f}}_{X,k} - \boldsymbol{f}_{X,k-1},\ \Delta\boldsymbol{f}^{c}_{A} = \bar{\boldsymbol{f}}_{A,k} - \boldsymbol{f}_{A,k-1}$;

$\boldsymbol{e}_X = \Delta\boldsymbol{f}^{c}_{X}/\|\Delta\boldsymbol{f}^{c}_{X}\|,\ \boldsymbol{e}_A = \Delta\boldsymbol{f}^{c}_{A}/\|\Delta\boldsymbol{f}^{c}_{A}\|$;

$\boldsymbol{f}_{X,k} = \boldsymbol{f}_{X,k-1} + g_k\sqrt{\frac{C}{\Delta t}}\,\boldsymbol{e}_X,\ \boldsymbol{f}_{A,k} = \boldsymbol{f}_{A,k-1} + g_k\sqrt{\frac{C}{\Delta t}}\,\boldsymbol{e}_A$;

$\mathbf{X}_{k+1} \leftarrow \mathbf{X}_k + \boldsymbol{f}_{X,k}\Delta t + g_k\sqrt{\Delta t}\,\boldsymbol{\epsilon}^X,\ \mathbf{A}_{k+1} \leftarrow \mathbf{A}_k + \boldsymbol{f}_{A,k}\Delta t + g_k\sqrt{\Delta t}\,\boldsymbol{\epsilon}^A$;

end for

Output: Graph $(\mathbf{X}_T, \mathbf{A}_T)$.

The relative magnitude between the two parameter variations therefore satisfies

$$\frac{|\mathrm{d}g|}{\|\mathrm{d}\boldsymbol{f}\|_2} = \mathcal{O}(\sqrt{\mathrm{d}t}) \to 0, \qquad \mathrm{d}t \to 0. \tag{40}$$

Hence, $|\mathrm{d}g| = o\left(\|\mathrm{d}\boldsymbol{f}\|_2\right)$, which further implies $(\mathrm{d}g)^2 = o\left(\|\mathrm{d}\boldsymbol{f}\|_2^2\right)$. Substituting this asymptotic relation into Eq. equation 33 shows that the diffusion-related component is a higher-order infinitesimal and therefore does not affect the leading-order Fisher geometry, so only the drift block of the Fisher information matrix needs to be retained, yielding the reduced Fisher-Rao metric used in the main text.

Drift-induced Fisher-Rao metric. Retaining only the leading-order drift contribution, the Fisher information matrix reduces to

$$\mathcal{I}(\boldsymbol{f}) = \frac{\Delta t}{g^2}\boldsymbol{I}. \tag{7}$$

The corresponding Fisher-Rao line element becomes

$$\mathrm{d}s^2 = \mathrm{d}\boldsymbol{f}^\top \mathcal{I}(\boldsymbol{f})\mathrm{d}\boldsymbol{f} = \frac{\Delta t}{g^2}\|\mathrm{d}\boldsymbol{f}\|_2^2. \tag{41}$$

Replacing the infinitesimal variation with its discrete counterpart yields the discrete Fisher-Rao arc length

$$\Delta s^2 = \Delta\boldsymbol{f}^\top \mathcal{I}(\boldsymbol{f})\Delta\boldsymbol{f} = \frac{\Delta t}{g^2}\|\Delta\boldsymbol{f}\|_2^2, \tag{8}$$

which corresponds to the discrete Fisher-Rao arc-length constraint adopted by the proposed adaptive sampling strategy in the main text.

B Experimental Details

B.1 Second-Order NGD-Solver Algorithm

Algorithm 2 presents the pseudo-code for the second-order variant of NGD-solver. The algorithm extends the first-order NGD-solver by incorporating a Heun step: the drift used for computing the geometric increment is the averaged network output from the current and predicted states. The Fisher-Rao constraint is then applied to the drift increment, ensuring constant distributional progress per step.

Table 5: Optimal Fisher-Rao arc-length parameter C for NGD-solver and its second-order variant 2nd NGD-solver on all datasets.

Model	Solver	QM9	ZINC250k	Planar	SBM	Community	Ego-small
GDSS	NGD-solver	0.0058	370	0.56	0.615	0.00005	0.0002
	2nd NGD-solver	0.0058	370	0.73	0.5	0.00005	0.0002
GruM	NGD-solver	0.05	0.3	0.007	0.39	0.00047	0.00014
	2nd NGD-solver	0.046	0.32	0.254	0.287	0.301	0.0014

Table 6: Statistical significance analysis on QM9 using the GDSS model. Results are averaged over 10 independent runs (seeds 1–10). "Improvement" denotes the paired performance difference between NGD-solver and Euler-Maruyama.

Metric	Euler	NGD-solver	Improvement	95% CI	p
`Valid` (%) ↑	95.147 ± 0.174	96.591 ± 0.166	$+1.444$	$[1.325,\ 1.563]$	< 0.001
`FCD` ↓	2.551 ± 0.053	2.085 ± 0.034	-0.466	$[-0.492,\ -0.439]$	< 0.001
`NSPDK` ↓	0.0034 ± 0.0001	0.0021 ± 0.0001	-0.0013	$[-0.0014,\ -0.0012]$	< 0.001

B.2 Implementation Details

All experiments are conducted on NVIDIA RTX 3090 GPUs with 24GB memory. Unless otherwise specified, all other experimental configurations, including operating system, software library versions, and evaluation protocols, strictly follow the original settings of the respective baseline implementations.

For the main experiments on continuous diffusion models (GDSS and GruM), we adopt the default sampling configurations from the official codebases. Specifically, each run uses $K = 1000$ reverse steps, with random seeds and other hyperparameters inherited from the original implementations. All reported results are based on a single run per configuration, following the standard practice of the baselines. In addition, we provide statistical validation over 10 independent runs on the QM9 dataset in Appendix C.1, including standard deviations, confidence intervals, and significance tests.

B.3 Parameter Configuration

The Fisher-Rao arc-length parameter C in Eq. equation 9 controls the target distributional change per sampling step. A larger C permits more aggressive drift updates, while a smaller C enforces more conservative transitions. For each dataset and backbone model, we select C by grid search. The optimal values are reported in Table 5.

Its optimal value varies across datasets because different datasets exhibit different transition-distribution geometries and reverse-time dynamics, leading to different evolution rates that best balance sampling stability and efficiency. Even on the same dataset, GDSS and GruM require different values of C because they adopt different diffusion formulations and learn different reverse-time dynamics, resulting in different optimal rates of distributional evolution. NGD-solver remains robust to the choice of C within a reasonable range.

C Additional Experiments

C.1 Different random seeds

To evaluate the robustness of NGD-solver with respect to sampling randomness, we repeat molecule generation on QM9 using the GDSS model with ten different random seeds (seeds 1–10). Table 6 reports the mean and standard deviation of each metric, along with paired t-test results comparing NGD-solver against the Euler-Maruyama baseline.

As shown in Table 6, NGD-solver consistently outperforms the baseline across all evaluation metrics, with particularly pronounced improvements in `FCD` and `NSPDK`. Beyond the mean improve-

Table 7: Runtime comparison between NGD-solver and Euler-Maruyama on QM9 and ZINC250k using the GDSS model.

Dataset	Method	Total Time (s)	Overhead
QM9	Euler-Maruyama	335.21	–
	NGD-solver	337.64	+0.72%
ZINC250k	Euler-Maruyama	2479.59	–
	NGD-solver	2523.65	+1.78%

ments, we observe that NGD-solver also yields consistently lower variance across seeds for both FCD and NSPDK, with standard deviations reduced by over 36% and 29%, respectively. This indicates that NGD-solver not only achieves better average performance but also provides more stable and reliable generation quality across different random initializations.

The paired t-tests indicate that all improvements are highly statistically significant ($p < 0.001$ for all metrics). Moreover, the 95% confidence intervals of the performance improvements do not cross zero, confirming that the observed gains are consistent across different random seeds rather than being artifacts of favorable stochastic sampling.

C.2 Runtime analysis

We evaluate the computational overhead of NGD-solver compared to the Euler-Maruyama baseline on both QM9 and ZINC250k using the GDSS model. Both methods generate 10,000 molecules with the same number of sampling steps ($K = 1000$). All runtime measurements are performed on a single GeForce RTX 3090 GPU.

As shown in Table 7, NGD-solver incurs only negligible overhead on both datasets: approximately 0.72% on QM9 and 1.78% on ZINC250k. The slightly higher overhead on ZINC250k is expected, as the larger molecular graphs require more computation per step, yet the relative increase remains marginal. This minimal overhead is achieved because NGD-solver only introduces lightweight vector operations for computing the drift difference and the unit direction vector, without requiring additional neural network evaluations. Since the dominant cost of diffusion sampling lies in the forward passes through the network, the per-step overhead of NGD-solver remains minimal regardless of dataset size.

More importantly, NGD-solver enables efficient sampling with significantly fewer steps. As shown in Table 4, NGD-solver with only 400 steps achieves a Valid of 95.74% and FCD of 2.485, already surpassing the Euler-Maruyama baseline with 1000 steps (94.96% Valid, 2.551 FCD). This means NGD-solver can match or exceed the baseline's generation quality using fewer than half the sampling steps, leading to substantial reductions in wall-clock time.

These results confirm that the geometry-aware drift correction in NGD-solver achieves consistent generation quality improvements without compromising sampling efficiency.

D Discrete Extension of NGD-solver

This appendix shows that the geometric principle underlying NGD-solver naturally extends to discrete diffusion models. While the main paper derives NGD-solver from the continuous reverse-time SDE, the same geometric principle can also be formulated directly on discrete transition kernels. Specifically, by viewing each reverse transition kernel as a point on the probability simplex equipped with the Fisher-Rao metric, NGD-solver can be interpreted as enforcing approximately constant Fisher-Rao arc-length increments between consecutive reverse transitions. This demonstrates that the proposed natural sampling principle is independent of the continuous SDE parameterization and is equally applicable to discrete diffusion models.

D.1 Reverse Transition Kernel

Unlike continuous diffusion models whose dynamics are governed by a reverse-time stochastic differential equation, discrete diffusion models generate samples by iteratively applying reverse transition kernels.

Let $t \in [0, T]$ denote the reverse diffusion time, where $t = 0$ corresponds to the noise prior and $t = T$ corresponds to the data distribution, which is consistent with the notation used in the main paper. For the discrete reverse process, the interval $[0, T]$ is divided into K steps with step size $\Delta t = T/K$.

During reverse sampling, the denoising network predicts the posterior distribution of the clean graph

$$p_\phi(\boldsymbol{x}_K \mid \boldsymbol{x}_k) \tag{42}$$

where $\boldsymbol{x}_K = \boldsymbol{x}_T$ denotes the clean graph state at the terminal iteration $k = K$.

For instance, following the discrete diffusion formulation of DiGress, the transition kernel can be obtained by marginalizing over the clean graph state

$$p_\phi(\boldsymbol{x}_{k+1} \mid \boldsymbol{x}_k) = \sum_{\boldsymbol{x}_K} q(\boldsymbol{x}_{k+1} \mid \boldsymbol{x}_k, \boldsymbol{x}_K) p_\phi(\boldsymbol{x}_K \mid \boldsymbol{x}_k), \tag{43}$$

where $q(\boldsymbol{x}_{k+1} \mid \boldsymbol{x}_k, \boldsymbol{x}_K)$ is determined by the forward diffusion process.

At each reverse step, a graph state is sampled from the predicted transition kernel. Repeating this procedure progressively transforms the noisy graph into the final clean graph.

Since reverse generation is governed by these transition kernels, understanding their evolution provides a natural perspective for analyzing and regulating the reverse diffusion process through the induced probability distributions $\boldsymbol{p}_k$.

D.2 Transition Kernels on the Probability Simplex

Assume each node or edge variable takes values from a finite categorical state space

$$\mathcal{X} = \{1, 2, \ldots, D\}, \tag{44}$$

where D denotes the number of possible discrete categories.

For a reverse diffusion step, the transition kernel conditioned on the current state $\boldsymbol{x}_k$ defines a categorical distribution over the next state $\boldsymbol{x}_{k+1}$:

$$p_{k,i} = p(\boldsymbol{x}_{k+1} = i | \boldsymbol{x}_k), \quad i \in \mathcal{X}\,. \tag{45}$$

For a fixed current state $\boldsymbol{x}_k$, this transition distribution can be represented as a probability vector

$$\boldsymbol{p}_k = [p_{k,1}, p_{k,2}, \ldots, p_{k,D}], \tag{46}$$

where

$$p_{k,i} \geqslant 0, \qquad \sum_{i=1}^{D} p_{k,i} = 1.$$

Therefore, each reverse transition distribution is naturally a point on the probability simplex rather than an ordinary Euclidean space.

For example, in the QM9 dataset, each atom type belongs to a finite categorical space $\mathcal{X}_{\text{node}} = \{\text{C}, \text{N}, \text{O}, \text{F}\}$, where $D = 4$ denotes the number of possible atom categories. For a node whose current state is O at step k, the reverse transition distribution is $p(\boldsymbol{x}_{k+1} = i \mid \boldsymbol{x}_k = \text{O})$, which can be represented as $\boldsymbol{p}_k = [p(\text{C} \mid \text{O}), p(\text{N} \mid \text{O}), p(\text{O} \mid \text{O}), p(\text{F} \mid \text{O})]$.

D.3 Fisher-Rao Geometry

Note that in this appendix, θ denotes the parameter of the probability distribution and the coordinates on the statistical manifold, rather than the network parameters of the denoising model.

For a discrete probability distribution, the Fisher information matrix is

$$\mathcal{I}(\theta) = \sum_{i=1}^{D} \frac{(\nabla_\theta p_{k,i})(\nabla_\theta^\top p_{k,i})}{p_{k,i}}. \tag{47}$$

The corresponding Fisher-Rao metric is

$$\mathrm{d}s^2 = \mathrm{d}\theta^\top \mathcal{I}(\theta)\mathrm{d}\theta. \tag{48}$$

Using

$$\mathrm{d}p_{k,i} = (\nabla_\theta^\top p_{k,i})\mathrm{d}\theta, \tag{49}$$

the explicit dependence on the network parameters disappears, yielding

$$\mathrm{d}s^2 = \sum_{i=1}^{D} \frac{(\mathrm{d}p_{k,i})^2}{p_{k,i}}. \tag{50}$$

Therefore, the Fisher-Rao metric depends only on the probability distribution itself. In the following derivation, we directly regard the transition probability vector $\boldsymbol{p}_k$ as the point on the statistical manifold.

Unlike the Euclidean metric $\sum_i (\mathrm{d}p_{k,i})^2$, Fisher-Rao assigns different geometric costs to probability variations according to their statistical significance. Consequently, identical Euclidean perturbations correspond to different geometric distances on the probability simplex. Variations occurring in low-probability regions induce larger Fisher-Rao distances than those in high-probability regions, reflecting the intrinsic curvature of the statistical manifold.

Most importantly, Fisher-Rao measures the geometric length of the probability increment $\mathrm{d}\boldsymbol{p}$ rather than the probability distribution itself. Therefore, the natural sampling principle should regulate the evolution of $\mathrm{d}\boldsymbol{p}$ throughout the reverse process.

D.4 Discrete Fisher-Rao Arc Length

We now discretize the differential Fisher-Rao metric to obtain a computable distance between two consecutive reverse transition kernels.

Consider two consecutive reverse transition kernels

$$\boldsymbol{p}_k = [p(\mathbf{x}_{k+1} = i \mid \mathbf{x}_k)]_{i=1}^{D}, \tag{51}$$

and

$$\boldsymbol{p}_{k-1} = [p(\mathbf{x}_k = i \mid \mathbf{x}_{k-1})]_{i=1}^{D}. \tag{52}$$

Using the first-order finite-difference approximation

$$\Delta\boldsymbol{p} \approx \boldsymbol{p}_k - \boldsymbol{p}_{k-1}, \tag{53}$$

the Fisher-Rao arc length between two consecutive reverse transitions becomes

$$\Delta s^2 = \sum_i \frac{(p_{k,i} - p_{k-1,i})^2}{p_{k-1,i}}. \tag{54}$$

where the denominator is evaluated at the previous transition kernel following the first-order finite-difference approximation. This quantity measures the amount of distributional evolution performed during one reverse sampling step.

D.5 Discrete NGD-solver

Let $\boldsymbol{p}_k^\phi$ denote the reverse transition kernel induced by the posterior distribution predicted by the denoising network.

Algorithm 3 Discrete NGD-solver.

Input: Time T, step number K, and the constant C.
Initialize: $\Delta t = T/K$, $(\mathbf{X}_0, \mathbf{A}_0) \sim p_{\text{prior}}$, $\boldsymbol{p}_{X,0} \leftarrow \boldsymbol{p}^{\phi}_{X,0}$, $\boldsymbol{p}_{A,0} \leftarrow \boldsymbol{p}^{\phi}_{A,0}$.
1: $\mathbf{X}_1 \sim \boldsymbol{p}_{X,0}$, $\mathbf{A}_1 \sim \boldsymbol{p}_{A,0}$;
2: **for** $k = 1, 2, \ldots, K-1$ **do**
3: $\boldsymbol{p}^{\phi}_{X,k} = \sum_{\mathbf{X}_K} q(\mathbf{X}_{k+1} \mid \mathbf{X}_k, \mathbf{X}_K) p_\phi(\mathbf{X}_K \mid \mathbf{X}_k)$, $\boldsymbol{p}^{\phi}_{A,k} = \sum_{\mathbf{A}_K} q(\mathbf{A}_{k+1} \mid \mathbf{A}_k, \mathbf{A}_K) p_\phi(\mathbf{A}_K \mid \mathbf{A}_k)$;
4: $\rho_{X,k} = \sqrt{\frac{C}{\sum_{i=1}^{D} \frac{(p^{\phi}_{X,k,i} - p_{X,k-1,i})^2}{p_{X,k-1,i}}}}$, $\rho_{A,k} = \sqrt{\frac{C}{\sum_{i=1}^{D} \frac{(p^{\phi}_{A,k,i} - p_{A,k-1,i})^2}{p_{A,k-1,i}}}}$;
5: $\boldsymbol{p}_{X,k} = \boldsymbol{p}_{X,k-1} + \rho_{X,k}\left(\boldsymbol{p}^{\phi}_{X,k} - \boldsymbol{p}_{X,k-1}\right)$
$\boldsymbol{p}_{A,k} = \boldsymbol{p}_{A,k-1} + \rho_{A,k}\left(\boldsymbol{p}^{\phi}_{A,k} - \boldsymbol{p}_{A,k-1}\right)$;
6: $\mathbf{X}_{k+1} \sim \boldsymbol{p}_{X,k}$, $\mathbf{A}_{k+1} \sim \boldsymbol{p}_{A,k}$;
7: **end for**
Output: $(\mathbf{X}_T, \mathbf{A}_T)$.

Since Fisher-Rao measures the length of the probability increment, NGD-solver preserves the learned reverse transition direction while only rescaling its geometric length,

$$\boldsymbol{p}_k - \boldsymbol{p}_{k-1} = \rho_k(\boldsymbol{p}^{\phi}_k - \boldsymbol{p}_{k-1}). \tag{55}$$

This construction keeps the learned reverse dynamics unchanged while adaptively adjusting the Fisher-Rao arc length of each reverse transition.

We require every reverse diffusion step to traverse a constant Fisher-Rao arc length,

$$\Delta s^2 = C. \tag{56}$$

Substituting the scaled update yields

$$\sum_i \frac{\rho_k^2 (p^{\phi}_{k,i} - p_{k-1,i})^2}{p_{k-1,i}} = C. \tag{57}$$

Since ρ_k is independent of i, we obtain

$$\rho_k = \sqrt{\frac{C}{\sum_i \frac{(p^{\phi}_{k,i} - p_{k-1,i})^2}{p_{k-1,i}}}}. \tag{58}$$

Given the scaling factor ρ_k, the geometry-aware reverse transition kernel is obtained by

$$\boldsymbol{p}_k = \boldsymbol{p}_{k-1} + \rho_k \left(\boldsymbol{p}^{\phi}_k - \boldsymbol{p}_{k-1}\right). \tag{59}$$

The updated transition kernel $\boldsymbol{p}_k$ replaces the original prediction $\boldsymbol{p}^{\phi}_k$ during sampling. The next graph state is then sampled from the corrected transition distribution $\mathbf{x}_k \sim \boldsymbol{p}_k$ and the reverse diffusion process continues recursively until the final clean graph $\mathbf{x}_T$ is obtained.

Therefore, NGD-solver preserves the reverse transition direction predicted by the diffusion model while adaptively controlling its Fisher-Rao arc length, resulting in a more balanced evolution of probability distributions throughout reverse diffusion.

Algorithm 3 summarizes how the proposed NGD-solver is applied to discrete graph diffusion sampling.

D.6 Experiment

We first evaluate NGD-solver on DiGress (Vignac et al., 2023), a discrete denoising diffusion model for graph generation, on the QM9 dataset with hydrogen atoms removed (heavy atoms only). In

Table 8: Molecular generation results on QM9 (heavy atoms only) using DiGress (Vignac et al., 2023) and DeFoG (Qin et al., 2025).

Solver	DiGress		DeFoG	
	FCD ↓	NSPDK ↓	FCD ↓	NSPDK ↓
Base-solver	0.1601	0.00064	0.1338	0.00024
NGD-solver	0.1116	0.00042	0.0995	0.00017

addition, since discrete flow matching models such as DeFoG (Qin et al., 2025) also generate graphs via a sequence of transition probabilities on the discrete simplex, they are similarly amenable to the Fisher-Rao geometric correction proposed in this work. We therefore conduct the same set of experiments on DeFoG to verify the generalizability of NGD-solver beyond discrete diffusion models. For both models, we compare the default sampling procedure with NGD-solver applied during the reverse process, keeping all other hyperparameters and model configurations identical to the original implementations, using $K = 500$ reverse steps. Results are averaged over 5 independent runs.

In addition, the corrected transition kernel $\boldsymbol{p}_k$ is smoothed with its raw prediction $\boldsymbol{p}_k^\phi$ via exponential moving average, $\boldsymbol{p}_k \leftarrow (1-\alpha)\boldsymbol{p}_k + \alpha \boldsymbol{p}_k^\phi$, to stabilize the sampling trajectory. We set $\alpha = 0.8$ and $C = 400$ for DiGress, and $\alpha = 0.9$ and $C = 0.001$ for DeFoG.

Results. Table 8 summarizes the molecular generation results on QM9. NGD-solver consistently improves generation quality across both DiGress and DeFoG, demonstrating that the proposed geometric correction is not limited to continuous diffusion models but generalizes to discrete generative frameworks as well. For DiGress, NGD-solver reduces FCD, indicating better distributional fidelity. For DeFoG, which already benefits from efficient sampling due to its flow matching formulation, NGD-solver further enhances structural quality without compromising sampling efficiency. These results confirm that the Fisher-Rao geometric principle is independent of the specific parameterization of the reverse process and can be effectively applied whenever the generation trajectory is governed by a sequence of transition distributions on the probability simplex. The observed improvements hold consistently across both continuous and discrete generative frameworks. In continuous diffusion models, NGD-solver enforces uniform distributional evolution by adaptively rescaling the drift update; in discrete models, it does so by directly modulating the transition probabilities on the probability simplex. The fact that NGD-solver improves generation quality in both settings, despite their fundamentally different parameterizations of the reverse process, supports our core theoretical claim: it is the uniformity of distributional evolution, rather than the specific form of the generative dynamics, that fundamentally governs sample quality.