EFFICIENT SPECTRAL GRAPH DIFFUSION BASED ON SYMMETRIC NORMALIZED LAPLACIAN

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph generative modeling has seen rapid progress, yet existing approaches often trade off between fidelity, scalability, and stability. Continuous and discrete diffusion models capture complementary aspects but remain hampered by either structural distortion or heavy computational costs. We introduce Efficient Spectral Graph Diffusion (ESGD), a lightweight yet highly competitive framework that revisits score-based diffusion from the perspective of the Symmetric Normalized Laplacian. By compressing eigenvalues into a bounded symmetric spectral domain, ESGD guarantees balanced information flow, provable stability, and faster convergence. A novel degree-matrix recovery algorithm closes the gap between spectral representations and graph reconstruction, enabling faithful generation at scale. Theoretically, ESGD reduces condition numbers and eliminates hub-node dominance; empirically, a model with one of the smallest parameter counts among diffusion approaches, yet capable of matching or surpassing state-of-the-art performance. ESGD converges up to 100× faster in training and requires 6–10× fewer sampling steps, delivering scalable generation across synthetic and citation benchmarks and achieves competitive results across molecular benchmarks. Our findings suggest that progress in graph generation may come less from heavier engineering, and more from principled reformulations that unlock both efficiency and fidelity

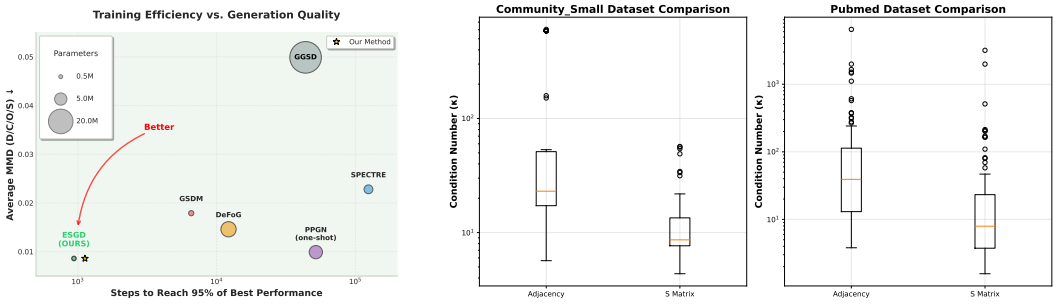
1 INTRODUCTION

Graph distribution learning and generation have become central research topics with broad applications in drug discovery, materials science, and network analysis. The goal is to capture the underlying distribution of graphs and model their intrinsic structural properties, including the interplay between nodes, edges, and features. Early generative models such as variational autoencoders (GraphVAE Simonovsky & Komodakis (2018)) and generative adversarial networks De Cao & Kipf (2018); Miyato et al. (2018) demonstrated feasibility, but VAEs struggle with posterior estimation on large graphs, while GANs are prone to mode collapse Jo et al. (2022). These limitations highlight the need for more scalable and robust paradigms.

Diffusion-based approaches have recently shown remarkable promise. Early models operate directly on adjacency matrices or their eigenspaces, applying Gaussian perturbations to both node features and graph structure Niu et al. (2020b); Jo et al. (2022). To preserve sparsity and improve efficiency, discrete diffusion models such as DiGress Vignac et al. (2023b) and DeFoG Qin et al. (2025) introduce edit-based noise processes. In addition to discrete models, Laplacian Martinkus et al. (2022b); Bergmeister et al. (2024a) and spectral Luo et al. (2024); Minello et al. (2025) methods which explore diffusion over eigenvalues and eigenvectors, capturing global structural properties but often suffering from eigenvalue imbalance or added model complexity.

In this paper, we propose Efficient Spectral Graph Diffusion (ESGD), a framework that advances spectral graph generation along three key dimensions. (i) By compressing eigenvalues of the **S**ymmetric **N**ormalized **L**aplacian (SNL) into a bounded domain, ESGD eliminates spectrum imbalance and dependence, with theoretical guarantees of stability and faster convergence. (ii) We design a degree-matrix recovery algorithm that overcomes the reconstruction bottleneck of prior spectral approaches Martinkus et al. (2022b). (iii) ESGD scales to citation networks datasets with thousands of nodes, where other models become prohibitively slow or unstable.

Compared with prior spectral and Laplacian methods, ESGD achieves strictly better fidelity with orders-of-magnitude faster training and sampling. In contrast to discrete and other diffusion models, which often require massive training epochs, large parameter counts (see Figure 1a), and long sampling chains, ESGD delivers comparable or better accuracy while remaining compact, efficient, and theoretically grounded. ESGD transforms the adjacency matrix into an SNL matrix, effectively improving the spectral properties of the dataset (see Figure 1b). This optimization manifests as a universal reduction in condition numbers, a more concentrated condition number distribution, and fewer outliers, which collectively reduces the learning difficulty associated with data distribution patterns and minority outlier features. This complementary profile highlights ESGD as the best spectral diffusion method that is not only state-of-the-art within spectral paradigm, but also competitive across different paradigms in both quality and scalability.



(a) Model comparison on Planar dataset (b) Condition Number Distributions: Adjacency vs SNL Matrix

Figure 1: Model performance comparison and spectrum improvements by SNL

Our contributions are threefold:

- Spectral compression with theory: ESGD diffuses in a bounded Laplacian spectral space, ensuring provable stability, faster training, and efficient sampling.
- Degree-matrix recovery: A principled algorithm reconstructs adjacency matrices from spectral representations, closing a key gap in spectral models.
- Large-graph scalability: Robust performance on large graphs with thousands of nodes, where existing discrete and spectral models are too slow or fail to converge.

2 PRELIMINARIES

2.1 SCORE-BASED GENERATIVE MODELS

Diffusion-based generative modeling has emerged as a powerful paradigm for high-dimensional data generation. In score-based generative models Song et al. (2021c), the key idea is to learn the score function $\nabla_{\boldsymbol{z}} \log p_t(\boldsymbol{z})$, the gradient of the log-density of a perturbed data distribution at time t. This allows one to simulate the reverse-time stochastic differential equation (SDE) to transform noise into data samples.

Formally, the forward noising process is defined by an SDE

$$d\boldsymbol{z}_t = f(t, \boldsymbol{z}_t)dt + g(t)d\boldsymbol{w}_t, \quad t \in [0, 1], \tag{1}$$

where f and g denote drift and diffusion coefficients, and $\boldsymbol{w}_t$ is a standard Wiener process. As $t \to 1$, $\boldsymbol{z}_t$ converges to a simple prior distribution (e.g., Gaussian). The reverse-time SDE takes the form

$$d\boldsymbol{z}_t = \Big(f(t, \boldsymbol{z}_t) - g(t)^2 \nabla_{\boldsymbol{z}} \log p_t(\boldsymbol{z}_t)\Big)dt + g(t)d\bar{\boldsymbol{w}}_t, \tag{2}$$

where $\bar{\boldsymbol{w}}_t$ is a reverse-time Wiener process. In practice, the score function is unknown and must be approximated by a neural network $s_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t)$.

2.2 Graph Convolutional Networks (GCN)

Graph Convolutional Networks (GCNs) Kipf & Welling (2017) are a fundamental building block for learning on graphs. Given an undirected graph with adjacency matrix $\boldsymbol{A}$ and degree matrix $\boldsymbol{D}$, GCN defines a layer-wise propagation rule that aggregates information from neighbors:

$$\boldsymbol{H}^{(\ell+1)} = \sigma\Big(\hat{\boldsymbol{A}}\,\boldsymbol{H}^{(\ell)}\,\boldsymbol{W}^{(\ell)}\Big), \hat{\boldsymbol{A}} = \tilde{\boldsymbol{D}}^{-1/2}\,\tilde{\boldsymbol{A}}\,\tilde{\boldsymbol{D}}^{-1/2}, \quad \tilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}. \tag{3}$$

where $\boldsymbol{H}^{(\ell)}$ is the hidden representation at layer ℓ, $\boldsymbol{W}^{(\ell)}$ is a trainable weight matrix, σ is a non-linear activation

From a spectral perspective, GCN layers correspond to applying a low-pass filter in the Laplacian eigenbasis, which smooths node features across connected nodes. This dual view message passing in the spatial domain and filtering in the spectral domain makes GCNs a good choice for parameterizing score functions in graph diffusion models.

3 Methodology

3.1 Problem Setup and ESGD framework

We study the graph generation problem where each graph $G = (\boldsymbol{X}, \boldsymbol{A})$ consists of a node-feature matrix $\boldsymbol{X} \in \mathbb{R}^{n\times d}$ and an adjacency matrix $\boldsymbol{A}$. We define the SNL operator

$$\boldsymbol{S} = \boldsymbol{L} - \boldsymbol{I} = -\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}},$$

Let $\boldsymbol{S} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$ be its eigen-decomposition. ESGD performs diffusion in a fixed eigen-space, we keep $\boldsymbol{U}$ fixed and only diffuse the eigenvalues $\boldsymbol{\Lambda}$ together with node features $\boldsymbol{X}$. The forward process is given by two coupled SDEs:

$$d\boldsymbol{X}_t = f_{\boldsymbol{X}}(\boldsymbol{X}_t, t)dt + g_{\boldsymbol{X}}(t)d\boldsymbol{W}_t^{\boldsymbol{X}}, \quad d\boldsymbol{\Lambda}_t = f_{\boldsymbol{\Lambda}}(\boldsymbol{\Lambda}_t, t)dt + g_{\boldsymbol{\Lambda}}(t)d\boldsymbol{W}_t^{\boldsymbol{\Lambda}},$$

with independent Wiener processes for $\boldsymbol{X}$ and $\boldsymbol{\Lambda}$. The reverse SDEs follow standard score-based formulations using score networks $s_{\boldsymbol{\theta}}$ and $s_{\boldsymbol{\phi}}$.

Score Networks We use GCN-based architectures for both $s_{\boldsymbol{\theta}}$ (node features) and $s_{\boldsymbol{\phi}}$ (eigenvalues). To avoid double normalization and self loop, the message-passing operator is replaced by $-\boldsymbol{S}$:

$$\boldsymbol{H}^{(\ell+1)} = \sigma(-\boldsymbol{S}\boldsymbol{H}^{(\ell)}\boldsymbol{W}^{(\ell)}).$$

Objectives We minimize denoising score matching losses:

$$\hat{E}(\boldsymbol{\theta}) = \mathbb{E}\|s_{\boldsymbol{\theta}}(\boldsymbol{X}_t, \boldsymbol{\Lambda}_t, \boldsymbol{U}) - \nabla_{\boldsymbol{X}_t}\log p_t(\boldsymbol{X}_t|\boldsymbol{X}_0)\|^2,$$
$$\hat{E}(\boldsymbol{\phi}) = \mathbb{E}\|s_{\boldsymbol{\phi}}(\boldsymbol{X}_t, \boldsymbol{\Lambda}_t, \boldsymbol{U}) - \nabla_{\boldsymbol{\Lambda}_t}\log p_t(\boldsymbol{\Lambda}_t|\boldsymbol{\Lambda}_0)\|^2.$$

Sampling After training, we reverse the diffusion to obtain $(\hat{\boldsymbol{X}}_0, \hat{\boldsymbol{\Lambda}}_0)$. The adjacency matrix is reconstructed using the recovered eigenvectors $\hat{\boldsymbol{U}}$:

$$\hat{\boldsymbol{A}} = -\boldsymbol{D}^{1/2}\hat{\boldsymbol{S}}\boldsymbol{D}^{1/2}, \quad \hat{\boldsymbol{S}} = \boldsymbol{U}\hat{\boldsymbol{\Lambda}}\boldsymbol{U}^\top,$$

where the degree matrix $\boldsymbol{D}$ is uniquely recoverable from $\hat{\boldsymbol{S}}$ by a provable algorithm 1. This guarantees exact graph reconstruction up to numerical thresholds. The full details of the ESGD model architecture can be found in B.2 and supplementary code project files.

3.2 Theoretical Properties

The theoretical guarantees of ESGD can be understood from two complementary perspectives. First, diffusion in the symmetric normalized Laplacian (SNL) domain enjoys uniformly bounded signal-to-noise ratio (SNR) and mutual information, independent of the maximum degree. In contrast, adjacency-based formulations scale with $\Delta_{\max}^2$ and thus suffer from severe information imbalance dominated by hub nodes. This boundedness provides several concrete advantages: (i) stable training across diverse degree distributions, (ii) efficient sampling with fewer function evaluations, (iii) robustness across heterogeneous graphs without dataset-specific tuning, and (iv) a direct explanation

of the empirical improvements in convergence and fidelity shown in Section 4. Second, the Lipschitz properties of the score and drift functions further ensure well-conditioned optimization and stable numerical solvers. In the SNL domain, curvature grows only linearly with n, whereas in the adjacency domain it is amplified by $\Delta_{\max}^2$, making ESGD inherently robust to graphs with extreme degree heterogeneity. See appendix A for details.

3.2.1 SPECTRAL REGULARIZATION

Theorem 3.1 (Spectral boundedness)**.** *For an undirected graph, eigenvalues of SNL satisfy $|\lambda_i(\boldsymbol{L})| \leq 1$, while adjacency eigenvalues satisfy $\sqrt{\Delta_{\max}} \leq |\lambda_i(\boldsymbol{A})| \leq \Delta_{\max}$, where $\Delta_{\max}$ is the max degree of the graph. Chung (1997)*

Theorem 3.2 (Node Permutation Invariance of ESGD)**.** *Let $G = (\boldsymbol{X}, \boldsymbol{A})$ be an undirected graph with adjacency $\boldsymbol{A}$ and node features $\boldsymbol{X}$. For any permutation matrix $\boldsymbol{P}$, define $\boldsymbol{X}' = \boldsymbol{P}\boldsymbol{X}$ and $\boldsymbol{A}' = \boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^\top$. Let $\boldsymbol{S} = -\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$ and $\boldsymbol{S}' = -\boldsymbol{D}'^{-1/2}\boldsymbol{A}'\boldsymbol{D}'^{-1/2}$. Then the forward and reverse diffusion processes satisfy*

$$(\boldsymbol{X}_t', \boldsymbol{\Lambda}_t') \overset{d}{=} (\boldsymbol{P}\boldsymbol{X}_t, \boldsymbol{\Lambda}_t), \quad \boldsymbol{A}_0' = \boldsymbol{P}\boldsymbol{A}_0\boldsymbol{P}^\top,$$

Theorem 3.3 (Spectral SNR and Information Retention)**.** *Let $\boldsymbol{X}_0 \in \mathbb{R}^n$ be the spectral embedding of a graph. Consider the forward process $\boldsymbol{X}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{X}_0 + \sigma_t\varepsilon$, $\varepsilon \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_n)$, with $\rho_t = \bar{\alpha}_t/\sigma_t^2$. Then:*

1. *(**SNR bound**) For any fixed initial data $\boldsymbol{x}_0$,*

$$\mathrm{SNR}(t) \;\leq\; \begin{cases} \rho_t, & \textit{SNL domain } \boldsymbol{S}, \\ \Delta_{\max}^2\,\rho_t, & \textit{adjacency domain } \boldsymbol{A}. \end{cases}$$

2. *(**Mutual information**) For different initial data $\boldsymbol{X}_0$ is random with covariance $\boldsymbol{\Sigma}_0$, then*

$$I(\boldsymbol{X}_0; \boldsymbol{X}_t) \leq \tfrac{1}{2}\log\det(\boldsymbol{I} + \rho_t\boldsymbol{\Sigma}_0) \;\leq\; \tfrac{1}{2}\rho_t\,\mathbb{E}\|\boldsymbol{X}_0\|^2$$

which scales as $O(\rho_t n)$ in domain $\boldsymbol{S}$ and $O(\rho_t n\Delta_{\max}^2)$ in domain $\boldsymbol{A}$.

3.2.2 STABILITY AND EFFICIENCY

Theorem 3.4 (Score Lipschitz Bound)**.** *For the score $s(\boldsymbol{x}, t) = \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})$ we have*

$$\|\nabla_{\boldsymbol{x}} s(\boldsymbol{x}, t)\|_{\mathrm{op}} \leq \sigma_t^{-2} + \frac{\bar{\alpha}_t}{\sigma_t^4} \cdot \frac{D_\bullet^2}{4},$$

where $D_\bullet$ is the spectral diameter. Consequently,

$$\|\nabla_{\boldsymbol{x}} s(\boldsymbol{x}, t)\|_{\mathrm{op}} \leq \begin{cases} \sigma_t^{-2} + \bar{\alpha}_t n/\sigma_t^4, & \boldsymbol{S}, \\ \sigma_t^{-2} + \bar{\alpha}_t n\Delta_{\max}^2/\sigma_t^4, & \boldsymbol{A}. \end{cases}$$

Theorem 3.5 (Drift Lipschitz and EM error)**.** *The reverse-time SDE drift $b(\boldsymbol{x}, t) = -\frac{1}{2}\beta(t)\boldsymbol{x} - \beta(t)s(\boldsymbol{x}, t)$ has Lipschitz constant*

$$L_b(t) \;\leq\; \tfrac{1}{2}\beta(t) + \beta(t)\Big(\sigma_t^{-2} + \tfrac{\bar{\alpha}_t}{4\sigma_t^4}D_\bullet^2\Big).$$

The Euler–Maruyama strong error satisfies

$$\big(\mathbb{E}\|\boldsymbol{X}^{\mathrm{EM}} - \boldsymbol{X}\|^2\big)^{1/2} \leq C_{\mathrm{EM}}\Big(\int_0^1 L_b(t)^2 dt\Big)^{1/2}\Delta t^{1/2}.$$

Remark 3.6 (Sampling efficiency). Combining Theorems 3.1 and 3.5, we conclude that SNL domain diffusion requires asymptotically fewer function evaluations than adjacency domain. This theoretical gain matches the $\Delta_{\max}^2\times$ reduction in sampling steps observed in Table 7 and Section 4.

Theorem 3.7 (Fisher spectrum and conditioning)**.** *Let $\boldsymbol{F} = \mathbb{E}[\nabla_{\boldsymbol{\theta}}\ell\nabla_{\boldsymbol{\theta}}\ell^\top]$ be the Fisher matrix of the score matching loss. Assume the network Jacobian satisfies $\|\boldsymbol{J}_{\boldsymbol{\theta}}(\boldsymbol{x}, t)\| \leq C_{\mathrm{net}}(t)\|\boldsymbol{x}\|$ for all $\boldsymbol{x}, t$. Then:*

Algorithm 1 Degree Matrix Recovery from $\boldsymbol{S}$

Require: SNL matrix $\boldsymbol{L}_{mod}$, threshold parameter $\delta > 0$
Ensure: Unweighted degree matrix $\boldsymbol{D}'$ and weighted adjacency matrix $\boldsymbol{A}$
1: **Step 1:** $\hat{\boldsymbol{A}}_{ij} \leftarrow \mathbf{1}_{|(\boldsymbol{S})_{ij}|>\delta}$ for all (i,j) ▷ Identify graph structure by thresholding
2: **Step 2:** $d_i \leftarrow \sum_{j=1}^{n} \hat{\boldsymbol{A}}_{ij}$ for all i ▷ Compute unweighted node degrees
3: **Step 3:** $\boldsymbol{D}' \leftarrow \mathrm{diag}(d_1, \ldots, d_n)$ ▷ Construct unweighted degree matrix
4: **Step 4 (For weighted graphs):** Recover the weighted adjacency matrix
5: **for all** (i,j) with $\hat{\boldsymbol{A}}_{ij} = 1$ **do** ▷ Edge weight recovery for connected pairs
6: $\quad \boldsymbol{A}_{ij} \leftarrow -(\boldsymbol{S})_{ij} \cdot \sqrt{d_i d_j}$
7: **end for**
8: $\boldsymbol{A}_{ij} \leftarrow 0$ for all (i,j) with $\hat{\boldsymbol{A}}_{ij} = 0$ ▷ Zero weights for disconnected pairs
9: **return** $\boldsymbol{D}'$, $\boldsymbol{A}$

1. *(**Spectral bound**) The largest eigenvalue scales as*

$$\lambda_{\max}(\boldsymbol{F}) = \begin{cases} O(n), & \textit{SNL domain}, \\ O(n\Delta_{\max}^2), & \textit{adjacency domain}. \end{cases}$$

2. *(**Condition number**) If in addition $\lambda_{\min}(\boldsymbol{F}) \geq \gamma > 0$, then*

$$\kappa(\boldsymbol{F}) = \frac{\lambda_{\max}(\boldsymbol{F})}{\lambda_{\min}(\boldsymbol{F})} = \begin{cases} O(n/\gamma), & \boldsymbol{S}, \\ O(n\Delta_{\max}^2/\gamma), & \boldsymbol{A}. \end{cases}$$

4 Graph Generation Results

4.1 Generic graph generation

Datasets: We test ESGD on *Community-small*, *Enzymes*, *Grid*, *Ego-small*, *Tree*, *Sbm*, and *Planar*. More details of these datasets are provided in Appendix C.1.

Metrics: We evaluate the maximum mean discrepancy (MMD) between equal numbers of generated and test graphs by measuring degree, clustering coefficient, 4-node orbit occurrences, their average, and spectral (Deg., Clus., Orbit, Avg., and Spec. in Table 1 and Table 2). See Appendix C.3 for details.

Table 1: Generic graph generation on Community-small, Enzymes, Grid, and Ego-small. * The results were obtained by executing the published source code. Other results are taken from the published papers Luo et al. (2024); Wen et al. (2024); Jang et al. (2024); Eijkelboom et al. (2024). Hyphen (-) denotes that results are not provided and were not applicable due to memory issues. The best results are highlighted in **bold**, and the underline denotes the second best. We provide the standard deviations in Appendix D.1 due to page limit.

	Community-small				Enzymes				Grid				Ego-small			
	Synthetic, ($12 \leq V \leq 20$)				Real, ($10 \leq V \leq 125$)				Synthetic, ($100 \leq V \leq 400$)				Real, ($4 \leq V \leq 18$)			
	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Deg.↓	Clus.↓	Orbit↓	Avg.↓
DeepGMG Li et al. (2018)	0.220	0.950	0.400	0.053	-	-	-	-	-	-	-	-	0.040	0.100	0.020	0.053
GraphRNN You et al. (2018)	0.080	0.120	0.040	0.080	0.017	0.062	0.046	0.042	0.064	0.043	0.021	0.043	0.090	0.220	0.003	0.104
GraphAF Shi et al. (2020)	0.180	0.200	0.020	0.133	1.669	1.283	0.266	1.073	-	-	-	-	0.030	0.110	**0.001**	0.047
GraphDF Luo et al. (2021)	0.060	0.120	0.030	0.070	1.503	1.061	0.202	0.922	-	-	-	-	0.040	0.130	0.010	0.060
GraphVAE	0.350	0.980	0.540	0.623	1.369	0.629	0.191	0.730	1.619	**0.000**	0.919	0.846	0.130	0.170	0.050	0.117
GNF Liu et al. (2019)	0.200	0.200	0.110	0.170	-	-	-	-	-	-	-	-	0.030	0.100	**0.001**	0.044
EDP-GNN Niu et al. (2020a)	0.053	0.144	0.026	0.074	0.023	0.268	0.082	0.124	0.455	0.238	0.328	0.340	0.052	0.093	0.007	0.051
WSGM Guth et al. (2022)	0.039	0.084	0.009	0.044	0.034	0.097	0.013	0.048	0.083	0.006	0.065	0.051	-	-	-	-
GDSS Jo et al. (2022)	0.045	0.086	0.007	0.046	0.026	0.102	0.009	0.046	0.111	0.005	0.070	0.062	0.021	0.024	0.007	0.017
HGDM Wen et al. (2024)	0.014	0.050	0.005	0.024	0.045	0.049	**0.003**	0.032	0.137	0.004	0.048	0.063	0.015	0.023	0.003	0.014
GSDM* Luo et al. (2024)	0.016	0.027	0.004	0.020	0.098	0.091	0.085	0.091	0.001	**0.000**	**0.000**	**0.000**	0.027	0.034	0.004	0.023
GEEL Jang et al. (2024)	-	-	-	-	**0.005**	**0.018**	0.006	**0.010**	**0.000**	**0.000**	**0.000**	**0.000**	-	-	-	-
CatFlow Eijkelboom et al. (2024)	0.018	0.086	0.007	0.037	-	-	-	-	-	-	-	-	0.013	0.024	0.008	0.015
ESGD (ours)	**0.007**	**0.010**	**0.001**	**0.006**	0.007	0.064	0.009	0.027	**0.000**	**0.000**	**0.000**	**0.000**	**0.009**	**0.022**	**0.001**	**0.011**

Results: As shown in Table 1, ESGD achieves state-of-the-art performance on generic graph generation. On Community-small and Ego-small, ESGD delivers substantial average MMD reductions of

70% and 21.4% respectively compared to existing methods, ranking first across all evaluation metrics. ESGD achieves perfect performance on Grid, matching GSDM and GEEL while significantly outperforming other baselines. On the challenging Enzymes dataset, ESGD ranks second overall, remaining competitive with GEEL while offering considerably lower computational overhead. Notably, GSDM's poor performance on complex datasets (Enzymes, Ego-small) confirms our hypothesis that diffusion in normalized Laplacian spectral space provides superior scalability compared to adjacency matrix spectral approaches. These results establish ESGD as a new state-of-the-art method that effectively addresses scalability limitations of existing spectral diffusion approaches.

Table 2: Generic graph generation on Planar, SBM, and Tree. Results are taken from the published papers Jang et al. (2024); QIN et al. (2025); Bergmeister et al. (2024b).

	Planar				SBM				Tree			
	Synthetic, ($\|V\| = 64$)				Synthetic, ($31 \leq V \leq 187$)				Synthetic, ($\|V\| = 64$)			
	Deg.↓	Clus.↓	Orbit↓	Spec.↓	Deg.↓	Clus.↓	Orbit↓	Spec.↓	Deg.↓	Clus.↓	Orbit↓	Spec.↓
GraphRNN You et al. (2018)	0.0049	0.2779	1.2543	0.0459	0.0055	0.0584	0.0785	0.0065	-	-	-	-
GRAN Liao et al. (2019)	0.0007	0.0426	0.0009	0.0075	0.0113	0.0553	0.0540	0.0054	0.1884	0.0080	0.0199	0.2751
SPECTRE Martinkus et al. (2022a)	0.0005	0.0785	0.0012	0.0112	0.0015	0.0521	0.0412	0.0056	-	-	-	-
DiGress Vignac et al. (2023a)	0.0007	0.0780	0.0079	0.0098	0.0018	0.0485	0.0415	0.0045	0.2678	0.0428	0.0097	0.0123
EDGE Chen et al. (2023b)	0.0761	0.3229	0.7737	0.0957	0.0279	0.1113	0.0854	0.0251	0.0211	0.1207	0.0374	0.0438
GDSS Jo et al. (2022)	0.2500	0.3930	0.5870	-	0.4960	0.4560	0.7170	-	-	-	-	-
GEEL* Jang et al. (2024)	0.0006	0.0458	**0.0000**	0.0070	0.0034	0.0621	**0.0000**	0.0049	-	-	-	-
DisCo Xu et al. (2024)	0.0002	0.0403	0.0009	-	0.0006	**0.0266**	0.0510	-	-	-	-	-
Cometh Siraudin et al. (2025)	0.0006	0.0434	0.0016	**0.0049**	0.0020	0.0498	0.0383	**0.0024**	-	-	-	-
DeFoG QIN et al. (2025)	0.0005	0.0501	0.0006	0.0072	0.0006	0.0517	0.0556	0.0054	0.0002	**0.0000**	**0.0000**	0.0108
Local PPGN (one-shot) Bergmeister et al. (2024b)	0.0003	**0.0245**	0.0006	0.0104	0.0141	0.0528	0.0809	0.0071	0.0004	**0.0000**	**0.0000**	**0.0080**
Local PPGN Bergmeister et al. (2024b)	0.0005	0.0626	0.0017	0.0075	0.0119	0.0517	0.0669	0.0067	**0.0001**	**0.0000**	**0.0000**	0.0117
ESGD (ours)	**0.0001**	0.0275	0.0005	0.0069	**0.0005**	**0.00273**	0.0462	0.0039	**0.0001**	**0.0000**	**0.0000**	0.0100

On Table 2, ESGD demonstrates strong performance across diverse synthetic datasets. On Planar graphs, ESGD achieves the best degree metric and consistently ranks second on clustering, orbit, and spectral metrics. For SBM graphs, ESGD attains the best degree and clustering and second-best spectral performance, though specialized methods like GEEL excel on orbit through stronger local motif modeling. Most notably, ESGD achieves state-of-the-art performance on nearly all metrics for Tree graphs, confirming that SNL diffusion aligns exceptionally well with hierarchical structures.

4.2 Large graph generation

Datasets: We evaluate ESGD on three widely used citation networks: *Cora*, *Citeseer*, and *PubMed*. These datasets contain thousands of nodes and edges, making them substantially larger and structurally more complex than synthetic benchmarks. Directly training generative models on full citation networks is computationally prohibitive due to the quadratic complexity of spectral decomposition. To address this challenge, we propose an *ego-subgraph decomposition strategy*: each large graph is decomposed into a set of k-hop ego-subgraphs, which serve as training instances. This design not only reduces computational cost but also preserves essential local structural statistics (degree distributions, clustering coefficients, orbit counts), while providing multi-view samples of the original global graph. Additional dataset statistics are provided in Appendix C.1.

Rationale for ego-subgraphs: Ego-subgraph extraction offers three key advantages. First, it improves *computational efficiency*: large graphs with tens of thousands of nodes are decomposed into subgraphs of size 50–300, significantly reducing both memory footprint and spectral computation cost. Second, it preserves *structural fidelity*: each ego-subgraph retains the k-hop neighborhood around a center node, thus capturing local motifs and degree/clustering patterns representative of the original network. Third, it enhances *generalization*: by sampling many subgraphs, the model benefits from a data augmentation effect that alleviates overfitting to a single large graph. This procedure transforms large-graph generation into a tractable and principled learning problem without compromising statistical validity (see Appendix C.2 for detailed justification and statistics).

Metrics: We use the same metrics as in Section 4.1.

Results: Table 3 shows that ESGD consistently outperforms spectral diffusion baselines on citation networks. Compared to the best prior method (GSDM), ESGD reduces average MMD by 54.1%, 63.6%, and 66.4% on Citeseer, Cora, and PubMed, respectively. Moreover, ESGD achieves uniformly better scores across all metrics, whereas existing spectral methods typically incur MMD

Table 3: Large graph generation results on Cora, Citeseer, and PubMed. The baselines (SPECTRE, GSDM, GGSD) are diffusion-based generative models in the spectral space.

	Citeseer				Cora				PubMed			
	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Deg.↓	Clus.↓	Orbit↓	Avg.↓
SPECTRE* Martinkus et al. (2022a)	1.224	1.513	1.023	1.253	1.566	1.492	1.127	1.395	1.148	1.392	0.933	1.158
GSDM* Luo et al. (2024)	1.043	0.943	0.843	0.943	0.932	1.042	0.980	0.985	0.885	0.727	0.762	0.791
GGSD* Minello et al. (2025)	1.011	1.142	1.244	1.132	1.218	1.432	1.391	1.347	0.775	0.711	1.029	0.838
ESGD (ours)	**0.329**	**0.656**	**0.314**	**0.433**	**0.311**	**0.573**	**0.192**	**0.359**	**0.215**	**0.475**	**0.109**	**0.266**

values three to five times higher. These results demonstrate that combining ego-subgraph training with SNL diffusion yields scalable and faithful large-graph generation, effectively balancing computational tractability and structural fidelity in real-world networks.

4.3 Molecules generation

Datasets: We test ESGD on two molecule benchmarks: ZINC250k (Irwin et al., 2012) and QM9 (Ramakrishnan et al., 2014).

Metrics: We evaluate the quality of 10,000 generated graphs using Frechet ChemNet Distance (FCD) (Preuer et al., 2018), Neighborhood Subgraph Pairwise Distance Kernel (NSPDK) MMD (Costa & De Grave, 2010), validity w/o correction abbreviated as Val. w/o, validity, and the generation time. Please see Appdenxi C.4 for more details.

Table 4: Results on the QM9 and ZINC250k. Results were taken from the published papers Luo et al. (2024); Wen et al. (2024); Jang et al. (2024); Eijkelboom et al. (2024); QIN et al. (2025). We provide the validity, uniqueness, and novelty values in Appendix D.2 due to page limit.

	QM9					ZINC250k				
	Validity (%)↑	Val. w/o (%)↑	NSPDK↓	FCD↓	Time(s)↓	Validity (%)↑	Val. w/o (%)↑	NSPDK↓	FCD↓	Time(s)↓
GraphAF (Shi et al., 2020)	**100**	67	0.020	5.268	$2.28e^3$	**100**	68	0.044	16.289	$5.72e^3$
GraphAF+FC	**100**	74.43	0.021	5.625	$2.32e^3$	**100**	68.47	0.044	16.023	$5.91e^3$
GraphDF (Luo et al., 2021)	**100**	82.67	0.063	10.816	$5.08e^4$	**100**	89.03	0.176	34.202	$5.87e^4$
GraphDF+FC	**100**	93.88	0.064	10.928	$4.72e^4$	**100**	90.61	0.177	33.546	$5.79e^4$
MoFlow (Zang & Wang, 2020)	**100**	91.36	0.017	4.467	**4.58**	**100**	63.11	0.046	20.931	**25.9**
EDP-GNN (Niu et al., 2020a)	**100**	47.52	0.005	2.680	$4.13e^3$	**100**	82.97	0.049	16.737	$8.41e^3$
GDSS (Jo et al., 2022)	**100**	95.72	0.003	2.900	$1.06e^2$	**100**	97.01	0.019	14.656	$2.11e^3$
HGDM (Wen et al., 2024)	**100**	98.04	0.002	2.131	$1.23e^2$	**100**	93.51	0.016	17.69	$2.23e^3$
GSDM* (Luo et al., 2024)	**100**	99.81	0.009	3.191	18.5	**100**	93.0	0.016	12.07	86.3
GEEL Jang et al. (2024)	**100.0**	**100**	**0.0002**	**0.089**	-	**100**	**99.31**	**0.0068**	**0.401**	-
CatFlow Eijkelboom et al. (2024)	**100**	99.81	-	0.441	-	**100**	99.21	-	13.211	-
DeFog QIN et al. (2025)	-	-	-	-	-	**100**	99.22	**0.0008**	1.425	-
ESGD (ours)	**100**	99.20	0.002	1.425	14.6	**100**	98.29	0.010	8.80	72.1

Results: As shown in Table 4, ESGD achieves competitive performance on molecular generation benchmarks while delivering substantial improvements over existing spectral methods. Compared to GSDM, ESGD improves FCD by 55.3% and 27.1% on QM9 and ZINC250k respectively, and enhances NSPDK by 77.8% and 37.5%. ESGD also generates molecules substantially faster than other spectral methods, highlighting its computational efficiency. While specialized methods like GEEL and DeFoG achieve superior performance through molecular-specific inductive biases, ESGD establishes new state-of-the-art results among spectral approaches, confirming its effectiveness on complex graphs with multiple node types and weighted edges.

4.4 Efficiency evaluation

Training Efficiency: We report the number of model parameters, training configuration, and the required training steps to reach 95% of the best performance on the *Planar* dataset. The results are summarized in Table 5.

From Table 5, it is evident that ESGD requires significantly fewer training steps to reach near-optimal performance. Despite having the smallest parameter size (0.381M), ESGD converges within only 938 steps, which is **7**$\times$ **faster** than GSDM and over **100**$\times$ **faster** than SPECTRE and GGSD. Interestingly, even compared to DeFoG—a discrete model—ESGD achieves convergence with nearly

Table 5: Training efficiency comparison on the *Planar* dataset. Steps denote the number of training iterations required to achieve 95% of the best performance, computed as Steps $=$ Epochs $\times \left\lceil \frac{\text{Training Graphs}}{\text{Batch Size}} \right\rceil$. The best results are highlighted in **bold**.

Model	Parameters (M)	Training Graphs	Batch Size	Steps to 95% Best Perf.
SPECTRE	1.64	200	10	124,160
GSDM	0.568	200	64	6,563
GGSD	21.09	200	16	43,750
DeFoG	4.92	200	64	12,188
PPGN (one-shot)	3.72	200	16	51,875
ESGD (ours)	**0.381**	200	64	**938**

an order-of-magnitude reduction in training cost. This highlights ESGD's ability to balance parameter efficiency and convergence speed, making it particularly suitable for scenarios where both computational resources and training time are constrained.

Sampling efficiency: To show our efficiency gains, we compare ESGD with other state-of-the-art spectral diffusion models. GGSD, which also operates in the spectral space, requires 100-200 sampling steps to achieve comparable quality. The recently proposed DeFoG model demonstrates impressive efficiency by achieving good validity with only 5-10% of traditional diffusion steps, yet ESGD matches or exceeds this efficiency on structured graph datasets while maintaining superior graph quality metrics.

Figure 2 illustrates how ESGD's performance metrics stabilize rapidly across four diverse datasets. On the Enzymes dataset, the MMD metrics converge by 150 steps, while on the Tree dataset, near-optimal performance is achieved with just 50 steps. For molecular graphs (QM9), ESGD achieves 99.08% validity at 500 steps, demonstrating excellent sampling efficiency without sacrificing generation quality.

These experimental findings confirm our theoretical analysis in Section 3, which explains that ESGD enhances sampling efficiency by eliminating edge-number dependencies in eigenvalues, constraining the eigenvalue space to the bounded interval (e.g., [-1,1] for generic graph datasets). The rapid convergence observed across all datasets validates that our normalized Laplacian spectral representation provides a more efficient diffusion trajectory compared to adjacency-based approaches.

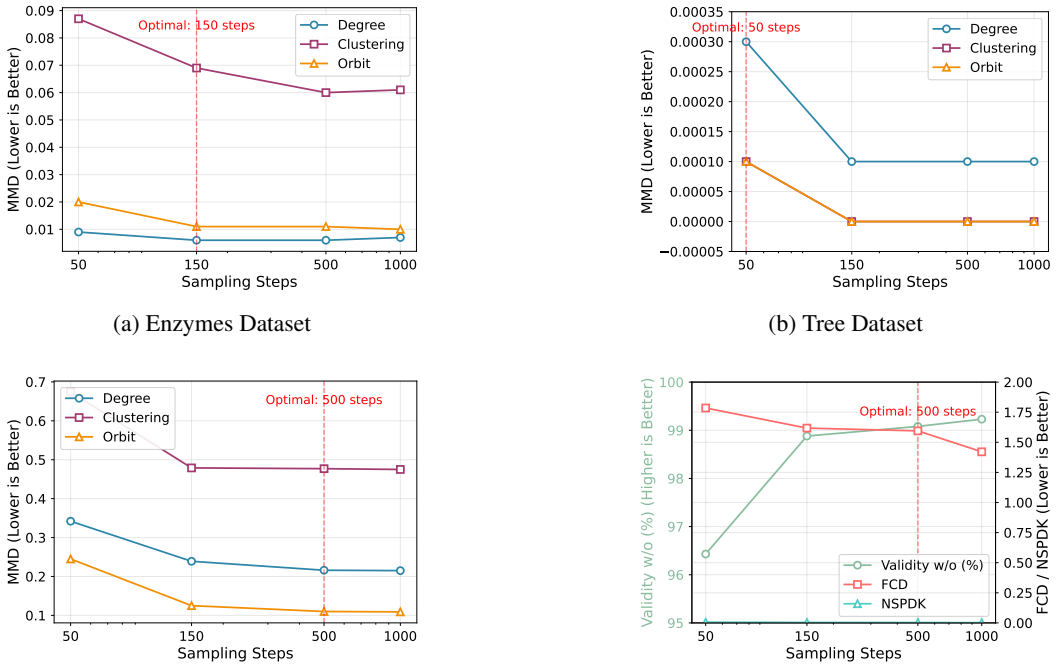
5 Related Work

Early graph generative models. Early approaches relied on VAEs and GANs. GraphVAE Simonovsky & Komodakis (2018) and MolGAN De Cao & Kipf (2018); Miyato et al. (2018) showed that deep generative learning on graphs is possible, but inherited the weaknesses of their backbones: VAEs suffer from inaccurate posterior approximation on large graphs, while GANs are prone to instability and mode collapse Jo et al. (2022). These limitations motivated the search for more stable generative paradigms.

Diffusion-based generative modeling. Diffusion models—including DDPM Ho et al. (2020), DDIM Song et al. (2021a), score-based diffusion Song et al. (2021b), stable diffusion Rombach et al. (2022), and flow-based variants Lipman et al. (2023); Liu et al. (2022)—have since emerged as a powerful family of generative methods, overcoming many of the weaknesses of VAEs and GANs in high-dimensional domains. Their success in images and molecules has spurred growing interest in graphs, where two main directions have been explored:

Continuous diffusion. These models Niu et al. (2020b); Jo et al. (2022) apply Gaussian perturbations to adjacency matrices and node features. While effective, the injected noise often produces dense graphs, degrading sparsity and structural fidelity.

Discrete diffusion. In contrast, models such as DiGress Vignac et al. (2023b), EDGE Chen et al. (2023a), local-PPGN Bergmeister et al. (2024b) GEEL Jang et al. (2024) DeFoG Qin et al. (2025)



(a) Enzymes Dataset

(b) Tree Dataset

(c) PubMed Dataset

(d) QM9 Dataset

Figure 2: Sampling efficiency of ESGD across different datasets. The figures show how evaluation metrics change with different sampling steps. Red dashed lines indicate the optimal sampling steps for each dataset.

define edit-based noise processes on nodes and edges, preserving sparsity and graph structure. However, they require long training schedules and slow sampling, which limits scalability.

Spectral and Laplacian approaches. A complementary line of work leverages graph spectra. SPECTRE Martinkus et al. (2022b) models dominant Laplacian eigencomponents to capture global structure but introduces significant architectural complexity. GSDM Luo et al. (2024) improves efficiency via low-rank spectral diffusion, yet remains sensitive to eigenvalue scaling. GGSD Minello et al. (2025) increases complexity by sampling eigencomponents. These methods illustrate the potential of spectral domains, but also reveal persistent limitations in stability, reconstruction, and scalability.

Our approach. Existing methods for graph diffusion each suffer from fundamental trade-offs: continuous approaches are effective but local information, discrete approaches preserve structure but are computationally expensive. Spectral approaches trying to combine eigenspace and eigenvalues to capture local information but enlarge the model size and increase complexity. ESGD provides a different perspective: it reinterprets score-based diffusion in the spectral domain, where eigenvalue compression of the normalized Laplacian ensures stable score estimation with theoretical guarantees. This spectral view makes it capture more local information or mitigate the effects of matrix heterogeneity. As a result, ESGD unifies the advantages of stability, efficiency, and scalability, and demonstrates empirically that spectral diffusion can compete not only within its own paradigm but also with discrete models on both fidelity and large-graph generalization.

6 Conclusions

We have presented a spectral perspective on graph diffusion that achieves both theoretical soundness and practical efficiency. The broader lesson is that progress in generative modeling may not always come from additional layers of engineering, but from revisiting the core formulations that govern stability and scalability. Subtle adjustments to these foundations can sometimes prove more effective than increasingly intricate designs, a direction our work illustrates for graph generation.

REFERENCES

Andreas Bergmeister, Karolis Martinkus, Nathanaël Perraudin, and Roger Wattenhofer. Efficient and scalable graph generation through iterative local expansion. In *International Conference on Learning Representations (ICLR)*, 2024a. URL https://openreview.net/forum?id=2XkTz7gdpc.

Andreas Bergmeister, Karolis Martinkus, Nathanaël Perraudin, and Roger Wattenhofer. Efficient and scalable graph generation through iterative local expansion. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview.net/forum?id=2XkTz7gdpc.

Xiaohui Chen, Jiaxing He, Xu Han, and Li-Ping Liu. Efficient and degree-guided graph generation via discrete diffusion modeling. In *Proceedings of the 40th International Conference on Machine Learning*. PMLR, 2023a.

Xiaohui Chen, Jiaxing He, Xu Han, and Liping Liu. Efficient and degree-guided graph generation via discrete diffusion modeling. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 4585–4610. PMLR, 23–29 Jul 2023b.

Fan R. K. Chung. *Spectral Graph Theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society, 1997.

Fabrizio Costa and Kurt De Grave. Fast neighborhood subgraph pairwise distance kernel. In *Proceedings of the 26th International Conference on Machine Learning*, pp. 255–262. Omnipress; Madison, WI, USA, 2010.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 2nd edition, 2006.

Nicola De Cao and Thomas Kipf. MolGAN: An implicit generative model for small molecular graphs. *arXiv preprint arXiv:1805.11973*, 2018.

Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011.

Floor Eijkelboom, Grigory Bartosh, Christian A. Naesseth, Max Welling, and Jan-Willem van de Meent. Variational flow matching for graph generation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 11735–11764. Curran Associates, Inc., 2024.

Florentin Guth, Simon Coste, Valentin De Bortoli, and Stephane Mallat. Wavelet score-based generative modeling. In *Advances in neural information processing systems*, volume 35, pp. 478–491, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

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. doi: 10.1021/ci3001277. PMID: 22587354.

Yunhui Jang, Seul Lee, and Sungsoo Ahn. A simple and scalable representation for graph generation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=nO344avRib.

Jaehyeong Jo, Seul Lee, and Sung Ju Hwang. Score-based generative modeling of graphs via the system of stochastic differential equations. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 10362–10383. PMLR, 2022.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

Peter E. Kloeden and Eckhard Platen. *Numerical Solution of Stochastic Differential Equations*. Springer, 1992.

Yujia Li, Oriol Vinyals, Chris Dyer, Razvan Pascanu, and Peter Battaglia. Learning deep generative models of graphs. *arXiv preprint arXiv:1803.03324*, 2018.

Renjie Liao, Yujia Li, Yang Song, Shenlong Wang, Will Hamilton, David K Duvenaud, Raquel Urtasun, and Richard Zemel. Efficient graph generation with graph recurrent attention networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Yaron Lipman, Ricky T. Q. Chen, and Max Welling. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

Jenny Liu, Aviral Kumar, Jimmy Ba, Jamie Kiros, and Kevin Swersky. Graph normalizing flows. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Xuanqing Liu, Yuxin Wen, Michael K. Ng, and Cho-Jui Hsieh. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Tianze Luo, Zhanfeng Mo, and Sinno Jialin Pan. Fast graph generation via spectral diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(5):3496–3508, 2024.

Youzhi Luo, Keqiang Yan, and Shuiwang Ji. Graphdf: A discrete flow model for molecular graph generation. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 7192–7203. PMLR, 18–24 Jul 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 *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pp. 15159–15179. PMLR, 17–23 Jul 2022a.

Karolis Martinkus, Andreas Loukas, Nathanaël Perraudin, and Roger Wattenhofer. SPECTRE: Spectral conditioning helps to overcome the expressivity limits of one-shot graph generators. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 15159–15179. PMLR, 2022b.

Giorgia Minello, Alessandro Bicciato, Luca Rossi, Andrea Torsello, and Luca Cosmo. Graph generation via spectral diffusion. In *International Conference on Learning Representations*, 2025.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, and Stefano Ermon. Permutation invariant graph generation via score-based generative modeling. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 4474–4484. PMLR, 26–28 Aug 2020a.

Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, and Stefano Ermon. Permutation invariant graph generation via score-based generative modeling. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 4474–4484. PMLR, 2020b.

Bernt Øksendal. *Stochastic Differential Equations: An Introduction with Applications*. Springer, 6th edition, 2003.

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 *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=KPRIwWhqAZ.

Yiming Qin, Manuel Madeira, Dorina Thanou, and Pascal Frossard. DeFoG: Discrete flow matching for graph generation. In *Proceedings of the 42nd International Conference on Machine Learning*. PMLR, 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.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Chence Shi, Minkai Xu, Zhaocheng Zhu, Weinan Zhang, Ming Zhang, and Jian Tang. Graphaf: a flow-based autoregressive model for molecular graph generation. In *International Conference on Learning Representations*, 2020.

Martin Simonovsky and Nikos Komodakis. GraphVAE: Towards generation of small graphs using variational autoencoders. In *International Conference on Artificial Neural Networks and Machine Learning*, pp. 412–422. Springer, 2018.

Antoine Siraudin, Fragkiskos D. Malliaros, and Christopher Morris. Cometh: A continuous-time discrete-state graph diffusion model. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations (ICLR)*, 2021a.

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

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021c.

Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. Digress: Discrete denoising diffusion for graph generation. In *The Eleventh International Conference on Learning Representations*, 2023a. URL https://openreview.net/forum?id=UaAD-Nu86WX.

Clément Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. DiGress: Discrete denoising diffusion for graph generation. In *International Conference on Learning Representations*, 2023b.

Ulrike von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

Lingfeng Wen, Xuan Tang, Mingjie Ouyang, Xiangxiang Shen, Jian Yang, Daxin Zhu, Mingsong Chen, and Xian Wei. Hyperbolic graph diffusion model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 15823–15831, 2024.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations (ICLR)*, 2019.

Zhe Xu, Ruizhong Qiu, Yuzhong Chen, Huiyuan Chen, Xiran Fan, Menghai Pan, Zhichen Zeng, Mahashweta Das, and Hanghang Tong. Discrete-state continuous-time diffusion for graph generation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 79704–79740. Curran Associates, Inc., 2024.

Jiaxuan You, Rex Ying, Xiang Ren, William Hamilton, and Jure Leskovec. GraphRNN: Generating realistic graphs with deep auto-regressive models. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5708–5717. PMLR, 10–15 Jul 2018.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabás Póczos, Russ R. Salakhutdinov, and Alexander J. Smola. Deep sets. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 3391–3401, 2017.

Chengxi Zang and Fei Wang. Moflow: an invertible flow model for generating molecular graphs. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 617–626, 2020.

LLM USAGE DECLEARATION

LLM involvement: restricted to language refinement and document formatting

A THEORETICAL SUPPORT

Theorem A.1 (Permutation invariance)**.** *Let $G = (\boldsymbol{X}, \boldsymbol{A})$ be an undirected graph with adjacency $\boldsymbol{A}$ and node features $\boldsymbol{X}$. For any permutation matrix $\boldsymbol{P}$, set $\boldsymbol{X}' = \boldsymbol{P}\boldsymbol{X}$ and $\boldsymbol{A}' = \boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^\top$. Let $\boldsymbol{S} = -\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$ and $\boldsymbol{S}' = -\boldsymbol{D}'^{-1/2}\boldsymbol{A}'\boldsymbol{D}'^{-1/2}$. Then the ESGD forward and reverse diffusion processes satisfy*

$$(\boldsymbol{X}'_t, \boldsymbol{\Lambda}'_t) \overset{d}{=} (\boldsymbol{P}\boldsymbol{X}_t, \boldsymbol{\Lambda}_t), \qquad \boldsymbol{A}'_0 = \boldsymbol{P}\boldsymbol{A}_0\boldsymbol{P}^\top,$$

so the generative distribution is invariant to node permutations.

Proof. **(1) Operator similarity.** Since $\boldsymbol{D}' = \mathrm{diag}(\boldsymbol{A}'\mathbf{1}) = \mathrm{diag}(\boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^\top\mathbf{1}) = \boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^\top$ and $\boldsymbol{P}$ is orthogonal, we have

$$\boldsymbol{S}' = -\boldsymbol{D}'^{-1/2}\boldsymbol{A}'\boldsymbol{D}'^{-1/2} = -(\boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^\top)^{-1/2}(\boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^\top)(\boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^\top)^{-1/2} = \boldsymbol{P}\boldsymbol{S}\boldsymbol{P}^\top.$$

Hence $\boldsymbol{S}$ and $\boldsymbol{S}'$ are similar and share eigenvalues; their eigenvectors transform as $\boldsymbol{U}' = \boldsymbol{P}\boldsymbol{U}$ (e.g., (Chung, 1997, Ch. 1)). **(2) Equivariance of message passing layers.** Consider a standard (normalized) GCN/MPNN layer

$$\Phi(\boldsymbol{H}; \boldsymbol{S}) = \sigma(-\boldsymbol{S}\boldsymbol{H}\boldsymbol{W}),$$

with elementwise activation σ and weight matrix $\boldsymbol{W}$. Using $\boldsymbol{P}\boldsymbol{H}\boldsymbol{W} = (\boldsymbol{P}\boldsymbol{H})\boldsymbol{W}$ and $\boldsymbol{P}\boldsymbol{S}\boldsymbol{P}^\top\boldsymbol{P}\boldsymbol{H} = \boldsymbol{P}(-\boldsymbol{S}\boldsymbol{H}\boldsymbol{W})$,

$$\Phi(\boldsymbol{P}\boldsymbol{H}; \boldsymbol{P}\boldsymbol{S}\boldsymbol{P}^\top) = \sigma\big(-(\boldsymbol{P}\boldsymbol{S}\boldsymbol{P}^\top)(\boldsymbol{P}\boldsymbol{H})\boldsymbol{W}\big) = \boldsymbol{P}\,\sigma(-\boldsymbol{S}\boldsymbol{H}\boldsymbol{W}) = \boldsymbol{P}\,\Phi(\boldsymbol{H}; \boldsymbol{S}).$$

Thus each layer is permutation-equivariant; stacked networks and the score nets inherit equivariance (see also Zaheer et al. (2017); Xu et al. (2019)). **(3) Forward SDE equivariance.** The forward SDEs read

$$\mathrm{d}\boldsymbol{X}_t = f_{\boldsymbol{X}}(\boldsymbol{X}_t, \boldsymbol{\Lambda}_t, t; \boldsymbol{S})\,\mathrm{d}t + g_{\boldsymbol{X}}(t)\,\mathrm{d}\boldsymbol{W}_t^{(\boldsymbol{X})}, \quad \mathrm{d}\boldsymbol{\Lambda}_t = f_{\boldsymbol{\Lambda}}(\boldsymbol{\Lambda}_t, t)\,\mathrm{d}t + g_{\boldsymbol{\Lambda}}(t)\,\mathrm{d}\boldsymbol{W}_t^{(\boldsymbol{\Lambda})}.$$

Define $\boldsymbol{X}'_t = \boldsymbol{P}\boldsymbol{X}_t$, $\boldsymbol{\Lambda}'_t = \boldsymbol{\Lambda}_t$, and $\boldsymbol{W}_t^{(\boldsymbol{X})'} = \boldsymbol{P}\boldsymbol{W}_t^{(\boldsymbol{X})}$. Since Brownian motion is invariant under orthogonal transforms and $f_{\boldsymbol{X}}(\cdot; \boldsymbol{S})$ is permutation-equivariant by (2), we obtain

$$\mathrm{d}\boldsymbol{X}'_t = f_{\boldsymbol{X}}(\boldsymbol{X}'_t, \boldsymbol{\Lambda}'_t, t; \boldsymbol{S}')\,\mathrm{d}t + g_{\boldsymbol{X}}(t)\,\mathrm{d}\boldsymbol{W}_t^{(\boldsymbol{X})'}.$$

Therefore the forward process is permutation-equivariant (e.g., (Øksendal, 2003, Ch. 3)).
(4) Scores and reverse SDE. Let p_t be the joint density of $(\boldsymbol{X}_t, \boldsymbol{\Lambda}_t)$. For any permutation $\boldsymbol{P}$, $p_t^{(\boldsymbol{P})}(\boldsymbol{x}, \lambda) = p_t(\boldsymbol{P}^\top\boldsymbol{x}, \lambda)$

$$\Rightarrow \quad \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{P}\boldsymbol{x}, \lambda) = \boldsymbol{P}\,\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}, \lambda), \;\; \nabla_\lambda \log p_t(\boldsymbol{P}\boldsymbol{x}, \lambda) = \nabla_\lambda \log p_t(\boldsymbol{x}, \lambda).$$

Hence the ground-truth scores are permutation-equivariant and so are consistent score networks trained by score matching. The reverse-time SDE (variance-exploding case) is

$$\mathrm{d}\boldsymbol{X}_t = \big(-\tfrac{1}{2}\beta(t)\boldsymbol{X}_t - \beta(t)\,s(\boldsymbol{X}_t, t)\big)\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\bar{\boldsymbol{W}}_t,$$

which remains permutation-equivariant when replacing $(\boldsymbol{X}_t, s)$ by $(\boldsymbol{P}\boldsymbol{X}_t, \boldsymbol{P}s)$.
(5) Reconstruction. At termination, $(\boldsymbol{X}_0, \boldsymbol{\Lambda}_0) \mapsto (\boldsymbol{P}\boldsymbol{X}_0, \boldsymbol{\Lambda}_0)$ and

$$\boldsymbol{S}'_0 = (\boldsymbol{P}\boldsymbol{U}_0)\boldsymbol{\Lambda}_0(\boldsymbol{P}\boldsymbol{U}_0)^\top = \boldsymbol{P}\boldsymbol{S}_0\boldsymbol{P}^\top.$$

With $\boldsymbol{A}_0 = -\boldsymbol{D}^{1/2}\boldsymbol{S}_0\boldsymbol{D}^{1/2}$ and $\boldsymbol{A}'_0 = -\boldsymbol{D}'^{1/2}\boldsymbol{S}'_0\boldsymbol{D}'^{1/2}$, and $\boldsymbol{D}' = \boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^\top$, we get $\boldsymbol{A}'_0 = \boldsymbol{P}\boldsymbol{A}_0\boldsymbol{P}^\top$. This proves the claim. □

Let $\boldsymbol{S}_t = \boldsymbol{U}_0 \boldsymbol{\Lambda}_t \boldsymbol{U}_0^\top$ with fixed $\boldsymbol{U}_0$ chosen once by eigendecomposition. For any block-orthogonal rotation $\boldsymbol{R}$ acting within degenerate eigenspaces of $\boldsymbol{U}_0$, set $\boldsymbol{U}_0' = \boldsymbol{U}_0 \boldsymbol{R}$. Then

$$\boldsymbol{U}_0' \boldsymbol{\Lambda}_t \boldsymbol{U}_0'^\top = \boldsymbol{U}_0 \boldsymbol{R} \boldsymbol{\Lambda}_t \boldsymbol{R}^\top \boldsymbol{U}_0^\top = \boldsymbol{U}_0 \boldsymbol{\Lambda}_t \boldsymbol{U}_0^\top,$$

so the reconstructed operator and hence the generated distribution are independent of the particular basis within degenerate subspaces (cf. von Luxburg (2007)).

Definition A.2 (Spectral diameter)**.** Let $\mathbb{X}_\bullet \subset \mathbb{R}^n$ be the feasible spectral set in domain $\bullet \in \{\boldsymbol{S}, \boldsymbol{A}\}$ (SNL $\boldsymbol{S}$ or adjacency $\boldsymbol{A}$). There exist absolute constants

$$D_{\boldsymbol{S}} = 2\sqrt{n}, \qquad D_{\boldsymbol{A}} = 2\Delta_{\max}\sqrt{n}$$

such that any spectral embedding $\boldsymbol{x}_0 \in \mathbb{X}_\bullet$ satisfies $\|\boldsymbol{x}_0\|^2 \le D_\bullet^2/4$. For $\boldsymbol{S}$, this follows from the spectrum lying in $[-1, 1]$; for $\boldsymbol{A}$, from $\|\boldsymbol{A}\| \le \Delta_{\max}$ (see Chung (1997)).

Theorem A.3 (Spectral SNR and information retention)**.** *Let $\boldsymbol{X}_0 \in \mathbb{R}^n$ be a spectral embedding. Consider*

$$\boldsymbol{X}_t = \sqrt{\bar{\alpha}_t}\, \boldsymbol{X}_0 + \sigma_t \varepsilon, \qquad \varepsilon \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_n), \qquad \rho_t := \bar{\alpha}_t/\sigma_t^2.$$

Then:

1. *(**SNR bound**) For any fixed $\boldsymbol{x}_0$,*

$$\mathrm{SNR}(t) := \frac{\bar{\alpha}_t \|\boldsymbol{x}_0\|^2}{n\sigma_t^2} \le \begin{cases} \rho_t, & \bullet = \boldsymbol{S}, \\ \Delta_{\max}^2\, \rho_t, & \bullet = \boldsymbol{A}. \end{cases}$$

2. *(**Mutual information**) If $\boldsymbol{X}_0$ has covariance $\boldsymbol{\Sigma}_0$, then*

$$I(\boldsymbol{X}_0; \boldsymbol{X}_t) \le \tfrac{1}{2} \log\det(\boldsymbol{I} + \rho_t \boldsymbol{\Sigma}_0) \;\le\; \tfrac{1}{2}\rho_t\, \mathbb{E}\|\boldsymbol{X}_0\|^2,$$

with $\mathbb{E}\|\boldsymbol{X}_0\|^2 = O(n)$ in domain $\boldsymbol{S}$ and $O(n\Delta_{\max}^2)$ in domain $\boldsymbol{A}$.

Proof. (1) By Definition A.2, $\|\boldsymbol{x}_0\|^2 \le D_\bullet^2/4$, hence

$$\mathrm{SNR}(t) = \rho_t \frac{\|\boldsymbol{x}_0\|^2}{n} \le \rho_t \cdot \frac{D_\bullet^2}{4n} = \begin{cases} \rho_t, & D_{\boldsymbol{S}}^2/4 = n, \\ \Delta_{\max}^2 \rho_t, & D_{\boldsymbol{A}}^2/4 = n\Delta_{\max}^2. \end{cases}$$

(2) Since $\boldsymbol{X}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{X}_0 + \sigma_t\varepsilon$ with $\varepsilon \perp \boldsymbol{X}_0$, the Gaussian channel formula gives $I(\boldsymbol{X}_0; \boldsymbol{X}_t) = \frac{1}{2}\log\det(\boldsymbol{I} + \rho_t\boldsymbol{\Sigma}_0)$ (e.g., (Cover & Thomas, 2006, Ch. 9)). Using $\log\det(\boldsymbol{I} + \boldsymbol{M}) \le \mathrm{tr}(\boldsymbol{M})$ for $\boldsymbol{M} \succeq \boldsymbol{0}$,

$$I(\boldsymbol{X}_0; \boldsymbol{X}_t) \le \tfrac{1}{2}\,\mathrm{tr}(\rho_t \boldsymbol{\Sigma}_0) = \tfrac{1}{2}\rho_t\, \mathbb{E}\|\boldsymbol{X}_0\|^2.$$

By Definition A.2, any $\boldsymbol{X}_0$ supported on $\mathbb{X}_\bullet$ satisfies $\mathbb{E}\|\boldsymbol{X}_0\|^2 \le D_\bullet^2/4$, which yields the stated domain scalings. □

Theorem A.4 (Score Lipschitz)**.** *Let p_t be the density of $\boldsymbol{X}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{X}_0 + \sigma_t\varepsilon$ with $\varepsilon \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_n)$ and define the score $s(\boldsymbol{x}, t) = \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})$. Then*

$$\|\nabla_{\boldsymbol{x}} s(\boldsymbol{x}, t)\|_{\mathrm{op}} \le \sigma_t^{-2} + \frac{\bar{\alpha}_t}{\sigma_t^4} \cdot \frac{D_\bullet^2}{4} \le \begin{cases} \sigma_t^{-2} + \bar{\alpha}_t n/\sigma_t^4, & \bullet = \boldsymbol{S}, \\ \sigma_t^{-2} + \bar{\alpha}_t n\Delta_{\max}^2/\sigma_t^4, & \bullet = \boldsymbol{A}. \end{cases}$$

Proof. Fix t and write $\bar{\alpha} = \bar{\alpha}_t$, $\sigma = \sigma_t$. Denote $m(\boldsymbol{x}) := \mathbb{E}[\boldsymbol{X}_0 \mid \boldsymbol{X} = \boldsymbol{x}]$. By Tweedie's identity for additive Gaussian noise (see Efron (2011) and also (Song et al., 2021b, Sec. 3)),

$$\sqrt{\bar{\alpha}}\, m(\boldsymbol{x}) = \boldsymbol{x} + \sigma^2 s(\boldsymbol{x}, t) \quad \Longleftrightarrow \quad s(\boldsymbol{x}, t) = \frac{\sqrt{\bar{\alpha}}\, m(\boldsymbol{x}) - \boldsymbol{x}}{\sigma^2}. \tag{4}$$

Differentiating equation 4 in $\boldsymbol{x}$ yields

$$\nabla s(\boldsymbol{x}, t) = \frac{\sqrt{\bar{\alpha}}}{\sigma^2}\, \nabla m(\boldsymbol{x}) - \frac{1}{\sigma^2}\boldsymbol{I}. \tag{5}$$

For the Gaussian corruption channel $\boldsymbol{X} = \sqrt{\bar{\alpha}}\boldsymbol{X}_0 + \sigma\varepsilon$, a standard covariance identity (Stein's lemma / Bayes rule differentiation) gives

$$\nabla m(\boldsymbol{x}) = \frac{\sqrt{\bar{\alpha}}}{\sigma^2}\, \mathrm{Cov}(\boldsymbol{X}_0 \mid \boldsymbol{X} = \boldsymbol{x}), \tag{6}$$

see, e.g., (Efron, 2011, Sec. 2). Substituting equation 6 into equation 5,

$$\nabla s(\boldsymbol{x}, t) = \frac{\bar{\alpha}}{\sigma^4} \operatorname{Cov}(\boldsymbol{X}_0 \mid \boldsymbol{X} = \boldsymbol{x}) - \frac{1}{\sigma^2} \boldsymbol{I}. \tag{7}$$

Since $\boldsymbol{X}_0 \in \mathbb{X}_\bullet$ almost surely, any conditional distribution $\boldsymbol{X}_0 \mid \boldsymbol{X} = \boldsymbol{x}$ is supported on $\mathbb{X}_\bullet$. Hence, by a diameter (Popoviciu-type) bound for bounded random vectors,

$$\|\operatorname{Cov}(\boldsymbol{X}_0 \mid \boldsymbol{X} = \boldsymbol{x})\|_{\mathrm{op}} \leq \frac{D_\bullet^2}{4}.$$

Taking operator norms in equation 7 and using the triangle inequality yields

$$\|\nabla s(\boldsymbol{x}, t)\|_{\mathrm{op}} \leq \frac{1}{\sigma^2} + \frac{\bar{\alpha}}{\sigma^4} \cdot \frac{D_\bullet^2}{4},$$

and substituting $D_{\boldsymbol{S}}^2/4 = n$ and $D_{\boldsymbol{A}}^2/4 = n\Delta_{\max}^2$ completes the proof. □

Theorem A.5 (Drift Lipschitz and EM error)**.** *Consider the reverse-time SDE in variance-exploding form*

$$\mathrm{d}\boldsymbol{X}_t = \underbrace{\Big(-\tfrac{1}{2}\beta(t)\boldsymbol{X}_t - \beta(t) s(\boldsymbol{X}_t, t) \Big)}_{=: b(\boldsymbol{X}_t, t)} \mathrm{d}t + \sqrt{\beta(t)}\, \mathrm{d}\bar{\boldsymbol{W}}_t.$$

Then, for each t,

$$L_b(t) := \sup_{\boldsymbol{x}} \|\nabla_{\boldsymbol{x}} b(\boldsymbol{x}, t)\|_{\mathrm{op}} \leq \tfrac{1}{2}\beta(t) + \beta(t)\Big(\sigma_t^{-2} + \tfrac{\bar{\alpha}_t}{4\sigma_t^4} D_\bullet^2 \Big).$$

Moreover, the Euler–Maruyama (EM) strong error with step size Δt satisfies

$$\big(\mathbb{E}\|\boldsymbol{X}_1^{\mathrm{EM}} - \boldsymbol{X}_1\|^2\big)^{1/2} \leq C_{\mathrm{EM}} \left(\int_0^1 L_b(t)^2 \, \mathrm{d}t \right)^{1/2} \Delta t^{1/2}.$$

Proof. The Jacobian of $b(\cdot, t)$ is

$$\nabla_{\boldsymbol{x}} b(\boldsymbol{x}, t) = -\tfrac{1}{2}\beta(t)\boldsymbol{I} - \beta(t)\nabla_{\boldsymbol{x}} s(\boldsymbol{x}, t).$$

Hence

$$\|\nabla_{\boldsymbol{x}} b(\boldsymbol{x}, t)\|_{\mathrm{op}} \leq \tfrac{1}{2}\beta(t) + \beta(t)\, \|\nabla_{\boldsymbol{x}} s(\boldsymbol{x}, t)\|_{\mathrm{op}}.$$

Applying Theorem A.4 gives the bound on $L_b(t)$. For EM, consider the time-inhomogeneous SDE $\mathrm{d}\boldsymbol{X}_t = b(\boldsymbol{X}_t, t)\, \mathrm{d}t + \sigma(t)\, \mathrm{d}\bar{\boldsymbol{W}}_t$ with $\sigma(t) = \sqrt{\beta(t)}\boldsymbol{I}$ independent of $\boldsymbol{x}$. Under global $\boldsymbol{x}$-Lipschitz continuity of $b(\cdot, t)$ with modulus $L_b(t)$ and linear growth (both satisfied here), the classical EM estimate (e.g., (Kloeden & Platen, 1992, Thm. 10.2.2)) yields

$$\big(\mathbb{E}\|\boldsymbol{X}_1^{\mathrm{EM}} - \boldsymbol{X}_1\|^2\big)^{1/2} \leq C_{\mathrm{EM}} \Big(\int_0^1 (L_b(t)^2 + L_\sigma(t)^2)\, \mathrm{d}t \Big)^{1/2} \Delta t^{1/2}.$$

Because σ does not depend on $\boldsymbol{x}$, $L_\sigma(t) = 0$, which gives the stated bound. □

Theorem A.6 (Fisher spectrum and conditioning)**.** *Let $\boldsymbol{F} = \mathbb{E}[\nabla_{\boldsymbol{\theta}} \ell \nabla_{\boldsymbol{\theta}} \ell^\top]$ be the Fisher (or generalized Gauss–Newton) matrix associated with the score matching loss. Assume the score network $S_{\boldsymbol{\theta}}(\cdot, t)$ has input Jacobian $\boldsymbol{J}_{\boldsymbol{\theta}}(\boldsymbol{x}, t) = \partial S_{\boldsymbol{\theta}}(\boldsymbol{x}, t)/\partial \boldsymbol{x}$ satisfying*

$$\|\boldsymbol{J}_{\boldsymbol{\theta}}(\boldsymbol{x}, t)\| \;\leq\; C_{\mathrm{net}}(t)\, \|\boldsymbol{x}\| \quad \forall \boldsymbol{x}, t.$$

Then:

1. *(**Spectral bound**) The largest eigenvalue of $\boldsymbol{F}$ scales as*

$$\lambda_{\max}(\boldsymbol{F}) = \begin{cases} O(n), & \text{normalized Laplacian domain } \boldsymbol{S}, \\ O(n\Delta_{\max}^2), & \text{adjacency domain } \boldsymbol{A}. \end{cases}$$

2. *(**Condition number**) If in addition $\lambda_{\min}(\boldsymbol{F}) \geq \gamma > 0$, then*

$$\kappa(\boldsymbol{F}) = \frac{\lambda_{\max}(\boldsymbol{F})}{\lambda_{\min}(\boldsymbol{F})} = \begin{cases} O(n/\gamma), & \boldsymbol{S}, \\ O(n\Delta_{\max}^2/\gamma), & \boldsymbol{A}. \end{cases}$$

Proof. Let $\ell(\boldsymbol{\theta}; \boldsymbol{X}_t, t)$ denote the score-matching loss at time t, with gradient

$$\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}; \boldsymbol{X}_t, t) = \boldsymbol{J}_{\boldsymbol{\theta}}(\boldsymbol{X}_t, t)^\top \big(S_{\boldsymbol{\theta}}(\boldsymbol{X}_t, t) - s(\boldsymbol{X}_t, t)\big),$$

where $s(\cdot, t)$ is the ground-truth score. **Step 1 (upper bound).** For any unit vector $\boldsymbol{u}$, the Rayleigh–Ritz principle gives

$$\boldsymbol{u}^\top \boldsymbol{F} \boldsymbol{u} = \mathbb{E}\big[\langle \nabla_{\boldsymbol{\theta}} \ell, \boldsymbol{u} \rangle^2\big] \le \mathbb{E}\|\nabla_{\boldsymbol{\theta}} \ell\|^2.$$

Hence $\lambda_{\max}(\boldsymbol{F}) \le \mathbb{E}\|\nabla_{\boldsymbol{\theta}} \ell\|^2$. By submultiplicativity,

$$\|\nabla_{\boldsymbol{\theta}} \ell\| \le \|\boldsymbol{J}_{\boldsymbol{\theta}}(\boldsymbol{X}_t, t)\| \, \|S_{\boldsymbol{\theta}}(\boldsymbol{X}_t, t) - s(\boldsymbol{X}_t, t)\|.$$

Using the Jacobian bound, this yields

$$\|\nabla_{\boldsymbol{\theta}} \ell\|^2 \le C_{\mathrm{net}}(t)^2 \, \|\boldsymbol{X}_t\|^2 \, \|S_{\boldsymbol{\theta}}(\boldsymbol{X}_t, t) - s(\boldsymbol{X}_t, t)\|^2.$$

Taking expectations and bounding the training error term by a finite constant $C_{\mathrm{err}} = \sup_t \mathbb{E}\|S_{\boldsymbol{\theta}}(\boldsymbol{X}_t, t) - s(\boldsymbol{X}_t, t)\|^2$ a.e, we obtain

$$\lambda_{\max}(\boldsymbol{F}) \;\le\; C_{\mathrm{err}} \, \mathbb{E}[C_{\mathrm{net}}(t)^2 \, \|\boldsymbol{X}_t\|^2]. \tag{8}$$

Step 2 (domain scaling of $\mathbb{E}\|\boldsymbol{X}_t\|^2$). The forward corruption process is $\boldsymbol{X}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{X}_0 + \sigma_t \varepsilon$, $\varepsilon \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_n)$. Then

$$\mathbb{E}\|\boldsymbol{X}_t\|^2 = \bar{\alpha}_t \, \mathbb{E}\|\boldsymbol{X}_0\|^2 + n\sigma_t^2.$$

By Definition A.2, $\mathbb{E}\|\boldsymbol{X}_0\|^2 = O(n)$ in domain $\boldsymbol{S}$ and $O(n\Delta_{\max}^2)$ in domain $\boldsymbol{A}$. Thus the scaling of $\lambda_{\max}(\boldsymbol{F})$ in equation 8 matches the theorem. **Step 3 (condition number).** If $\lambda_{\min}(\boldsymbol{F}) \ge \gamma > 0$, then

$$\kappa(\boldsymbol{F}) = \frac{\lambda_{\max}(\boldsymbol{F})}{\lambda_{\min}(\boldsymbol{F})} = \begin{cases} O(n/\gamma), & \boldsymbol{S}, \\ O(n\Delta_{\max}^2/\gamma), & \boldsymbol{A}. \end{cases}$$

This completes the proof. □

B ADDITIONAL INFORMATION OF ESGD

B.1 DEGREE MATRIX RECOVERY

We begin by analyzing the structure of the SNL $\boldsymbol{S} = -(\boldsymbol{D}')^{-1/2}\boldsymbol{A}(\boldsymbol{D}')^{-1/2}$. For an undirected, weighted graph with no self-loops:

$$\boldsymbol{S}_{i,j} = \begin{cases} 0 & \text{if } i = j \text{ (since } \boldsymbol{A}_{i,i} = 0 \text{ for no self-loops)} \\ -\frac{A_{i,j}}{\sqrt{d_i d_j}} & \text{if } i \neq j \text{ and } (i,j) \in \boldsymbol{E} \\ 0 & \text{if } i \neq j \text{ and } (i,j) \notin \boldsymbol{E} \end{cases} \tag{9}$$

where d_i represents the unweighted degree of node i, which is simply the number of edges connected to node i (regardless of their weights), $A_{i,j}$ is the weight of the edge between nodes i and j, and $\boldsymbol{E}$ is the set of edges. For a weighted graph with unweighted degree matrix, when nodes i and j are adjacent:

$$S_{i,j} = -\frac{A_{i,j}}{\sqrt{d_i d_j}} \tag{10}$$

This means that for any edge $(i,j) \in \boldsymbol{E}$, the product of the degrees d_i and d_j is related to $\boldsymbol{S}$ and the edge weight $A_{i,j}$:

$$d_i d_j = \frac{A_{i,j}^2}{S_{i,j}^2} \tag{11}$$

For any node i with at least two neighbors $j, k \in \mathcal{N}(i)$, we have:

$$\frac{d_j}{d_k} = \frac{S_{i,k}^2 \cdot A_{i,j}^2}{S_{i,j}^2 \cdot A_{i,k}^2} \tag{12}$$

Since the graph is connected, we can establish proportional relationships between all node degrees by traversing the graph. This gives us a system of equations that determines the degrees up to a constant factor. To resolve this remaining degree of freedom, we use the fact that the sum of all unweighted degrees equals twice the number of edges:

$$\sum_{i=1}^{n} d_i = 2|\mathrm{E}| \tag{13}$$

The number of edges $|\mathrm{E}|$ can be determined from the structure of $\boldsymbol{S}$ by counting the number of non-zero off-diagonal elements and dividing by 2. This yields a system of equations that uniquely determines the degree matrix $\boldsymbol{D}'$.

Practical algorithm for estimating unweighted degree matrix: In practical applications, the generated $\boldsymbol{S}$ may contain numerical errors or noise. Theoretically, elements corresponding to non-edges should be exactly zero, but in practice, they might appear as small non-zero values due to stochastic sampling process. Therefore, we introduce a thresholding parameter δ to distinguish between actual edges and numerical artifacts. The threshold parameter δ may need to be tuned based on the specific characteristics of the graph.

B.2 ESGD MODEL ARCHITECTURE

ESGD (Efficient Spectral Graph Diffusion) is a spectral graph diffusion model based on symmetric normalized Laplacian matrices for graph generation tasks.

B.2.1 CORE COMPONENTS

SDE Framework: The model employs Variance Preserving SDE (VPSDE):

$$d\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}dt + \sqrt{\beta(t)}d\mathbf{w} \tag{14}$$

$$\beta(t) = \beta_{min} + t(\beta_{max} - \beta_{min}) \tag{15}$$

where $\beta_{min} = 0.1$, $\beta_{max} = 20$, and $t \in [0, 1]$.

Score Networks: Two main networks predict scores for node features and adjacency matrices:

- *ScoreNetworkX*: Uses modified GCN layers with $\boldsymbol{S}$ convolution
- *ScoreNetworkA_eigen*: Operates in eigenvalue space with pooled node representations

Modified GCN Layer: Unlike traditional GCN, uses symmetric normalized Laplacian:

$$\mathbf{H}^{(l+1)} = \tanh\left(\boldsymbol{S}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right) \tag{16}$$

Graph Multi-Head Attention: Enhances representation with attention mechanism:

$$\mathbf{A}_{att} = \tanh\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right) \tag{17}$$

B.2.2 Loss Function

Score matching loss in both node and spectral domains:

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) + \frac{\boldsymbol{\epsilon}}{\sqrt{1-\alpha_t}}\right\|^2\right] \tag{18}$$

B.2.3 Key Features

- Spectral domain diffusion for stability
- Support for both generic graphs and molecular generation
- Multiple $\beta(t)$ scheduling (linear, exponential, cosine)
- Computational complexity: $O(N^2d + Nd^2)$
- Datasets: Community-small, Grid, Enzymes, Ego-small, QM9, ZINC250k

C EXPERIMENT DETAILS

In this section, we provide the detailed experimental settings. The hyperparameters of ESGD in this paper are provided in Table 6.

Table 6: Hyperparameters of ESGD used in the generic graph generation tasks and the molecule generation tasks. We provide the hyperparameters of the score-based models (s_θ and s_ϕ), the diffusion processes (SDE for X and A), the SDE solver, and the training.

	Hyperparameter	Ego-small	Community-small	Enzymes	Grid	Planar	SBM	Tree	QM9	ZINC250k
s_θ	Number of GCN layers	4	3	5	5	5	4	4	4	3
	Hidden dimension	32	32	32	32	32	32	32	16	16
s_ϕ	Number of attention heads	4	4	4	4	4	4	4	4	4
	Number of initial channels	2	2	2	2	2	2	2	2	2
	Number of hidden channels	8	8	8	8	8	8	8	8	8
	Number of final channels	4	4	4	4	4	4	4	4	4
	Number of GCN layers	5	5	7	7	7	6	7	6	6
	Hidden dimension	32	32	32	32	32	32	32	16	16
SDE for X	Type	VP	VP	VP	VP	VP	VP	VP	VP	VP
	Number of sampling steps	1000	1000	1000	1000	1000	1000	1000	1000	1000
	β_{min}	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	β_{max}	1.0	1.0	1.0	1.0	1.0	1.0	1.0	10.0	4.0
SDE for A	Type	VP	VP	VP	VP	VP	VP	VP	VP	VP
	Number of sampling steps	1000	1000	1000	1000	1000	1000	1000	1000	1000
	β_{min}	0.1	0.1	0.1	0.2	0.2	0.1	0.1	0.1	0.2
	β_{max}	1.0	1.0	1.0	0.8	0.9	1.0	1.0	1.0	1.0
Solver	Type	EM	EM + Langevin	EM + Langevin	EM + Langevin	EM + Langevin	EM + Langevin	EM + Langevin	EM + Langevin	EM + Langevin
	SNR	–	0.05	0.2	0.1	0.10	0.15	0.10	0.2	0.2
	Scale coefficient	–	0.8	0.9	0.7	0.8	0.6	0.6	0.9	0.9
Train	Optimizer	Adam	Adam	Adam	Adam	Adam	Adam	Adam	Adam	Adam
	Learning rate	1×10^{-2}	1×10^{-2}	1×10^{-2}	1×10^{-2}	1×10^{-2}	1×10^{-3}	1×10^{-2}	5×10^{-3}	5×10^{-3}
	Weight decay	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}
	Batch size	128	128	64	8	64	32	128	1024	1024
	Number of epochs	5000	200	5000	5000	1000	3000	1000	300	500
	EMA	–	–	0.999	0.999	0.999	0.999	0.999	–	–

C.1 DETAILS OF DATASETS

In this section, we provide key statistics of the datasets employed in the experiments, as shown in Table 7, for a better illustration of the experimental results. The statistics include the graph number in each dataset, the range of node numbers, the range of edge numbers for each node, the number of edge types, and the maximum eigenvalue.

Table 7: Statistics for the datasets in our experiments.

	Name	Graph Number	Node range	Edge number of node	Number of edge types	Maximum eigenvalue
Generic	Ego-small	200	[4, 18]	[1, 16]	1	9.036
	Community-small	100	[12, 20]	[1, 9]	1	6.6145
	Enzymes	587	[10, 125]	[1, 9]	1	5.3045
	Grid	100	[100, 400]	[1, 4]	1	3.9454
	Planar	200	[64, 64]	[2, 12]	1	6.1230
	SBM	200	[44, 187]	[1, 23]	1	14.1320
	Tree	200	[64, 64]	[1, 8]	1	3.0510
Molecule	QM9	133,885	[2, 9]	[1, 4]	3	3.7063
	ZINC250k	249,455	[6, 38]	[1, 4]	3	3.5823
Large	Cora	1	[2708, 2708]	[5429, 5429]	1	-
	Citeseer	1	[3312, 3312]	[4715, 4715]	1	-
	PubMed	1	[19717, 19717]	[44338, 44338]	1	-

Note: For large citation networks, "Graph Number" is 1 since each dataset consists of a single giant graph. Node and edge ranges reduce to single values (the total counts).

C.2 DETAILS OF EGO-SUBGRAPH DECOMPOSITION

To make training on large graphs feasible, we employ an ego-subgraph decomposition strategy implemented with NetworkX's `ego_graph` function. Given a center node and a radius r, an ego-subgraph contains the center and all nodes within r-hop distance, together with induced edges. We

apply size filters ($50 \leq |V| \leq 400$) to control computational complexity, remove self-loops, and relabel nodes to contiguous IDs. Datasets are split into training and test sets with an 80/20 ratio.

Table 8 reports the aggregated statistics of the constructed ego-subgraph datasets.

Table 8: Statistics of ego-subgraph datasets derived from large citation networks.

Dataset	Num. subgraphs	Node range	Avg. nodes	Edge range	Avg. edges	Avg. degree
Cora	100	50–219	112.8	65–428	207.2	3.67
Citeseer	80	51–300	141.7	65–788	271.0	3.82
PubMed	100	50–282	112.5	60–1177	236.8	4.21

This decomposition provides three main benefits:

- **Efficiency**: smaller subgraphs reduce quadratic spectral costs and fit within GPU memory.
- **Structural fidelity**: local neighborhood motifs and degree/clustering statistics are preserved.
- **Generalization**: sampling multiple ego-subgraphs introduces data augmentation, mitigating overfitting to a single global graph.

C.3 Implementation details for the experiments on generic datasets

To evaluate the generated graphs, we employ the maximum mean discrepancy (MMD) to compare distributions of graph statistics between generated and test graphs. The evaluated statistics include degree, clustering coefficient, and occurrences of 4-node orbits. We compute the MMDs using the Gaussian Earth Mover's Distance (EMD) kernel on Ego-small, Community-small, Enzymes, and Grid following (Jo et al., 2022) and using the Gaussian Total Variation Distance (TV) kernel on Planar, SBM, and Tree following QIN et al. (2025).

As the setting from (Jo et al., 2022), we report the results of ESGD and GSDM on the Ego-small and Community-small datasets by 15 runs, 3 runs for 5 independently trained models, and on the Enzymes and Grid datasets by 3 runs. For GSDM, we use the hyperparameters given by the original paper and further search for the best performance if specific parameters do not exist. To get the best hyperparameters, we perform a grid search to choose the best signal-to-noise ratio (SNR) in $\{0.05, 0.1, 0.15, 0.2, 0.25, 0.3\}$ and the scale coefficient in the $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$. We select the best MMD with the lowest average performance in Deg., Clus., and Orbit, respectively. Following (Jo et al., 2022), we quantize the value of each edge in the sampled adjacency matrix with the operator $1_{x>0.5}$ to get the 0-1 adjacency matrix. The specific hyperparameters are shown in Table 6.

C.4 Implementation details for the experiments on molecule datasets

We assess the quality of 10,000 generated graphs using multiple metrics. Frechet ChemNet Distance (FCD) leverages activations from ChemNet's penultimate layer to calculate the distance between test and generated graphs (Preuer et al., 2018). Neighborhood Subgraph Pairwise Distance Kernel (NSPDK) MMD measures the maximum mean discrepancy between test and generated graphs, accounting for both node and edge features (Costa & De Grave, 2010). Additionally, we report validity metrics: validity w/o correction and Validity represent the fractions of valid molecules without and with valency correction or edge resampling, respectively.

As the setting of (Jo et al., 2022), we report the results of ESGD and GSDM on QM9 and ZINC250k by 3 runs. We preprocess each molecule into a graph with the node features $X \in \{0, 1\}^{N \times F}$ and the adjacency matrix $A \in \{0, 1, 2, 3\}^{N \times N}$, where N is the maximum number of atoms and F is the number of atom types. We also use the grid search for the best SNR in $\{0.5, 1, 1.5, 2, 2.5, 3\}$ and the scale coefficient in $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$. The specific hyperparameters are shown in Table 6. We select the hyperparameters for the best FCD value. We quantize the entries of the adjacency matrices to $\{0, 1, 2, 3\}$ by clipping the value $(-\infty, 0.5)$ to 0, $[0.5, 1.5)$ to 1, $[1.5, 2.5)$ to 2, and $[2.5, \infty)$ to 3 following (Jo et al., 2022).

C.5 COMPUTING RESOURCES

For all experiments, we use PyTorch to implement ESGD and train the score models on an NVIDIA RTX A4000 GPU with intel i7-14700K CPU.

D ADDITIONAL EXPERIMENTAL RESULTS

In this section, we provide additional experimental results.

D.1 GENERIC GRAPH GENERATION

We report the standard deviation of the generation results of Table 1 in Table 9 and Table 10. We provide sampling acceleration of ESGD on the Community-small and Ego-small datasets in Table 11.

Table 9: Generation results of ESGD on Ego-small and Community-small. * denotes that the results are obtained by running open-source codes. The results of GDSS and HGDM are taken from (Luo et al., 2024; Wen et al., 2024; Eijkelboom et al., 2024). The best results are highlighted in **bold** (lower is better), and the underline denotes the second best. We report the MMD distance between the test datasets and the generated graphs with the standard deviation.

	Ego-small			Community-small		
	Deg.↓	Clus.↓	Orbit↓	Deg.↓	Clus.↓	Orbit↓
GDSS (Jo et al., 2022)	0.021±0.008	0.024±0.007	0.007±0.005	0.045±0.028	0.086±0.022	0.007±0.004
HGDM (Wen et al., 2024)	0.015±0.005	0.023±0.006	0.003±0.005	0.017±0.029	0.050±0.018	0.005±0.003
GSDM* (Luo et al., 2024)	0.027±0.000	0.034±0.007	0.004±0.001	0.016±0.018	0.027±0.026	0.004±0.005
CatFlow Eijkelboom et al. (2024)	0.013±0.007	0.024±0.009	**0.001**±0.005	0.018±0.012	0.086±0.021	0.007±0.005
ESGD (Ours)	**0.009**±0.003	**0.022**±0.002	**0.001**±0.000	**0.007**±0.003	**0.010**±0.004	**0.001**±0.000

Table 10: Generation results of ESGD on Enzymes and Grid. * denotes that the results are obtained by running open-source codes. The results of GDSS and HGDM are taken from (Luo et al., 2024; Wen et al., 2024; Eijkelboom et al., 2024). The best results are highlighted in **bold** (lower is better), and the underline denotes the second best. We report the MMD distance between the test datasets and the generated graphs with the standard deviation.

	Enzymes			Grid		
	Deg.↓	Clus.↓	Orbit↓	Deg.↓	Clus.↓	Orbit↓
GDSS (Jo et al., 2022)	0.026±0.008	0.102±0.010	0.009±0.005	0.111±0.012	0.004±0.000	0.070±0.044
HGDM (Wen et al., 2024)	0.045±0.008	**0.049**±0.011	**0.003**±0.001	0.137±0.019	0.004±0.000	0.070±0.044
GSDM* (Luo et al., 2024)	0.098±0.010	0.091±0.003	0.085±0.010	0.001±0.000	**0.000**±0.000	**0.000**±0.000
CatFlow Eijkelboom et al. (2024)	0.013±0.012	0.062±0.011	0.008±0.007	0.115±0.010	0.004±0.002	0.075±0.071
ESGD (ours)	**0.007**±0.001	0.064±0.002	0.009±0.001	**0.000**±0.000	**0.000**±0.000	**0.000**±0.000

Table 11: Sampling efficiency of ESGD by 1 run on Community-small and Ego-small.

Dataset	Steps	Deg.↓	Clus.↓	Orbit↓	Time (s)↓
Community-small	1000	0.011	0.015	0.001	1.51
	500	0.011	0.015	0.001	0.94
	250	0.011	0.015	0.001	0.58
	200	0.058	0.106	0.012	0.48
Ego-small	1000	0.012	0.019	0.001	1.13
	800	0.014	0.014	0.001	0.9
	750	0.015	0.014	0.001	0.86
	700	0.018	0.029	0.003	0.82

D.2 MOLECULE GENERATION

We additionally report the validity, uniqueness, and novelty of the generated molecules aside from the results in Table 4 to comprehensively illustrate the performance of molecule generation. Validity is the fraction of the generated molecules that do not violate the chemical valency rule. Uniqueness is the fraction of the valid molecules that are unique. Novelty is the fraction of the valid molecules that are not in the training set. Moreover, the standard deviation of each metric is also provided in this section. The results of molecule generation are shown in Table 12 and Table 13.

As shown in Table 12, the generated molecules of ESGD have lower novelty and comparable uniqueness. As discussed in (Wen et al., 2024), high novelty does not necessarily represent good generation quality due to the property of the QM9 dataset, such as the generated molecules of GraphDF and GraphEBM. In other words, the models that can generate molecules with high novelty fail to capture adequate properties of the dataset.

Table 12: Generation results on QM9. * denotes that the results are obtained by running open-source codes. Other results of the baselines are taken from the published papers Luo et al. (2024); Wen et al. (2024); Jang et al. (2024); Eijkelboom et al. (2024). The best results are highlighted in **bold**, and the underline denotes the second best.

Method	Val. w/o (%)↑	NSPDK MMD↓	FCD↓	Validity↑	Uniqueness↑	Novelty↑
GraphAF (Shi et al., 2020)	67	0.020±0.003	5.268±0.403	**100.00**	94.51	88.83
GraphDF (Luo et al., 2021)	82.67	0.063±0.001	10.816±0.020	**100.00**	97.62	**98.10**
MoFlow (Zang & Wang, 2020)	91.36±1.23	0.017±0.003	4.467±0.595	**100.00**±0.00	98.65±0.57	94.72±0.77
EDP-GNN (Niu et al., 2020a)	47.52±3.60	0.005±0.001	2.680±0.221	**100.00**±0.00	99.25±0.05	86.58±1.85
GDSS (Jo et al., 2022)	95.79±1.93	0.003±0.000	2.813±0.278	**100.00**±0.00	98.02±0.63	82.55±3.11
HGDM (Wen et al., 2024)	98.04±1.27	0.002±0.000	2.13±0.254	**100.00**±0.00	97.27±0.71	69.63±2.75
GSDM* (Luo et al., 2024)	99.81±0.08	0.009±0.000	3.191±0.014	**100.00**±0.00	94.7±0.15	68.5±0.47
CatFlow Eijkelboom et al. (2024)	99.81 ± 0.03	-	0.441 ± 0.023	**100.00** ± 0.00	**99.95** ± 0.02	-
GEEL Jang et al. (2024)	**100.0**	**0.0002**	**0.089**	**100.00**	96.08	22.30
ESGD (ours)	99.20±0.02	0.002±0.000	1.425±0.009	**100.00**±0.00	96.61±0.16	60.64±0.00

Table 13: Generation results on ZINC250k. * denotes that the results are obtained by running open-source codes. Other results of the baselines are taken from the published papers Luo et al. (2024); Wen et al. (2024); Jang et al. (2024); Eijkelboom et al. (2024); QIN et al. (2025). The best results are highlighted in **bold**, and the underline denotes the second best.

Method	Val. w/o (%)↑	NSPDK MMD↓	FCD↓	Validity↑	Uniqueness↑	Novelty↑
GraphAF (Shi et al., 2020)	68	0.044±0.006	16.289±0.482	**100.00**	99.10	**100.00**
GraphDF (Luo et al., 2021)	89.03	0.176±0.001	34.202±0.160	**100.00**	99.16	**100.00**
MoFlow (Zang & Wang, 2020)	63.11±5.17	0.046±0.002	20.931±0.184	**100.00**±0.00	99.99±0.01	**100.00**±0.00
EDP-GNN (Niu et al., 2020a)	82.97±2.73	0.049±0.006	16.737±1.300	**100.00**±0.00	99.79±0.08	**100.00**±0.00
GDSS (Jo et al., 2022)	95.90±1.01	0.019±0.001	16.621±1.213	**100.00**±0.00	99.67±0.14	**100.00**±0.00
HGDM (Wen et al., 2024)	93.51±0.87	0.016±0.001	17.69±1.146	**100.00**±0.00	99.82±0.18	**100.00**±0.00
GSDM* (Luo et al., 2024)	93.0±0.04	0.016±0.000	12.07±0.062	**100.00**±0.00	99.97±0.09	**100.00**±0.00
CatFlow Eijkelboom et al. (2024)	99.21 ± 0.04	-	13.211 ± 0.012	**100.00** ± 0.00	**100.00** ± 0.00	-
GEEL Jang et al. (2024)	**99.31**	0.0068	**0.401**	**100.00**	99.97	99.89
DeFoG QIN et al. (2025)	99.22 ± 0.08	**0.0008** ± 0.0001	1.425 ± 0.0001	**100.00** ± 0.00	99.99 ± 0.01	-
ESGD (ours)	98.29±0.58	0.010±0.000	8.80±0.132	**100.00**±0.00	99.76±0.12	**100.00**±0.00

We present acceleration results for ESGD sampling with 1,000-step training on both QM9 and ZINC250k datasets in Table 14. For QM9, ESGD maintains comparable accuracy when using 600 sampling steps. Although validity scores without correction decrease at lower step counts, both NSPDK and FCD metrics remain robust even when using as few as 500 sampling steps. This demonstrates that ESGD maintains excellent sampling efficiency when applied to molecule datasets.

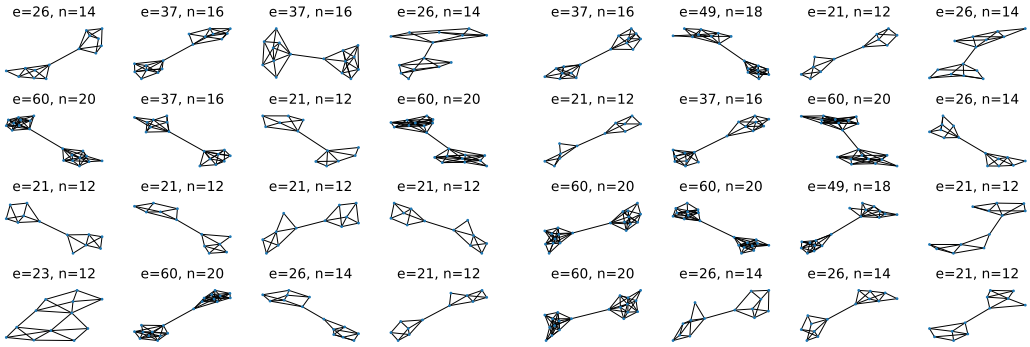
Table 14: Sampling efficiency of ESGD by 1 run on QM9 and ZINC250k.

Dataset	Steps	Val. w/o (%)↑	FCD↓	NSPDK MMD↓	Time (s)↓
QM9	1000	99.23	1.421	0.002	14.3
	800	99.35	1.427	0.002	11.2
	600	99.22	1.485	0.002	8.1
	500	99.08	1.595	0.003	5.4
ZINC250k	1000	98.81	8.856	0.011	71.6
	800	97.34	8.623	0.011	60.2
	500	95.94	8.813	0.010	37.5
	400	95.44	8.989	0.010	30.6

E VISUALIZATION

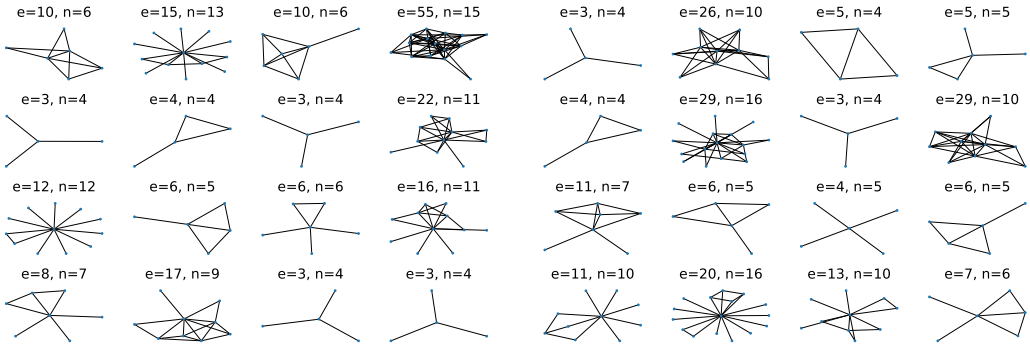
E.1 GENERIC GRAPH GENERATION

We visualize a randomly selected subset of samples from the training datasets and the generated graph set in Figures 3-9.



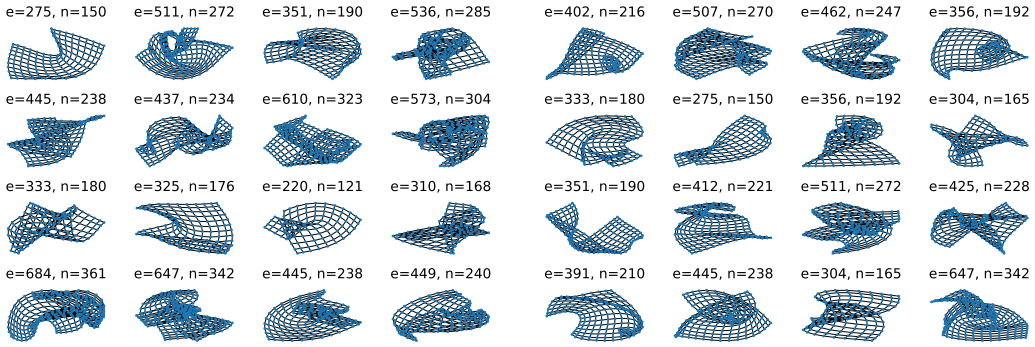
(a) Training Data (b) ESGD (ours)

Figure 3: Visualization of the graphs from the Community-small dataset and the generated graphs of ESGD.



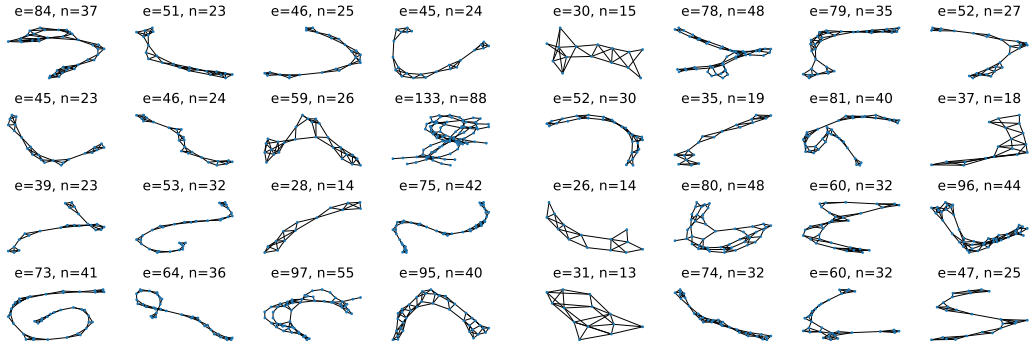
(a) Training Data (b) ESGD (ours)

Figure 4: Visualization of the graphs from the Ego-small dataset and the generated graphs of ESGD.



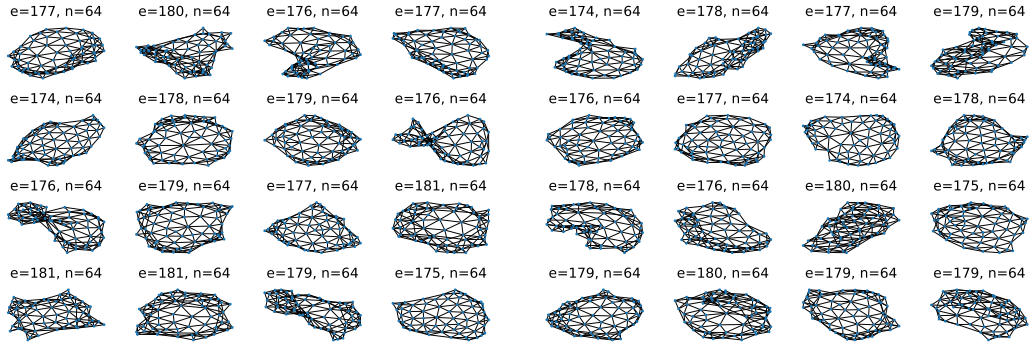
(a) Training Data (b) ESGD (ours)

Figure 5: Visualization of the graphs from the Grid dataset and the generated graphs of ESGD.



(a) Training Data (b) ESGD (ours)

Figure 6: Visualization of the graphs from the Enzymes dataset and the generated graphs of ESGD.

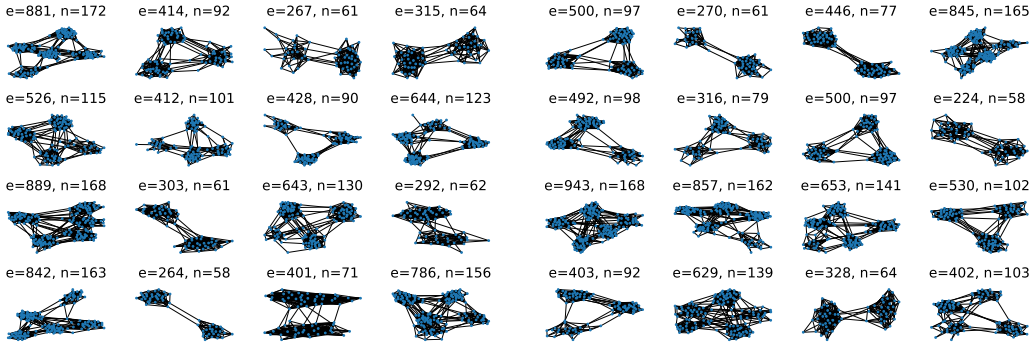


(a) Training Data (b) ESGD (ours)

Figure 7: Visualization of the graphs from the Planar dataset and the generated graphs of ESGD.

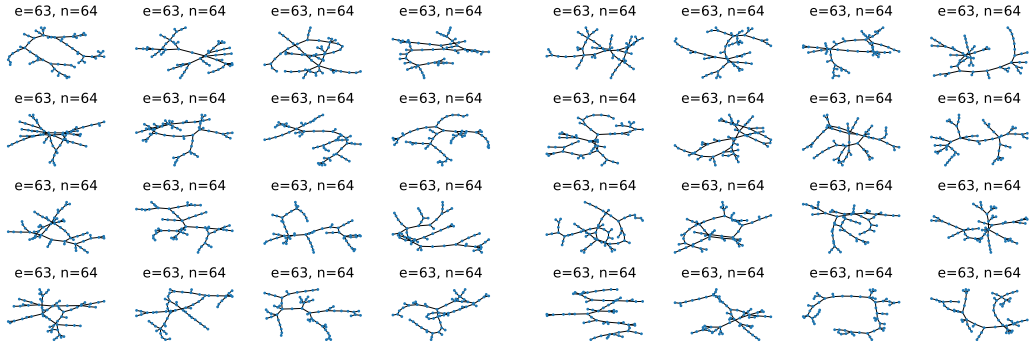
E.2 Molecule graph generation

We visualize a randomly selected subset of the generated graph set in Figures 10-11.



(a) Training Data (b) ESGD (ours)

Figure 8: Visualization of the graphs from the SBM dataset and the generated graphs of ESGD.



(a) Training Data (b) ESGD (ours)

Figure 9: Visualization of the graphs from the Tree dataset and the generated graphs of ESGD.

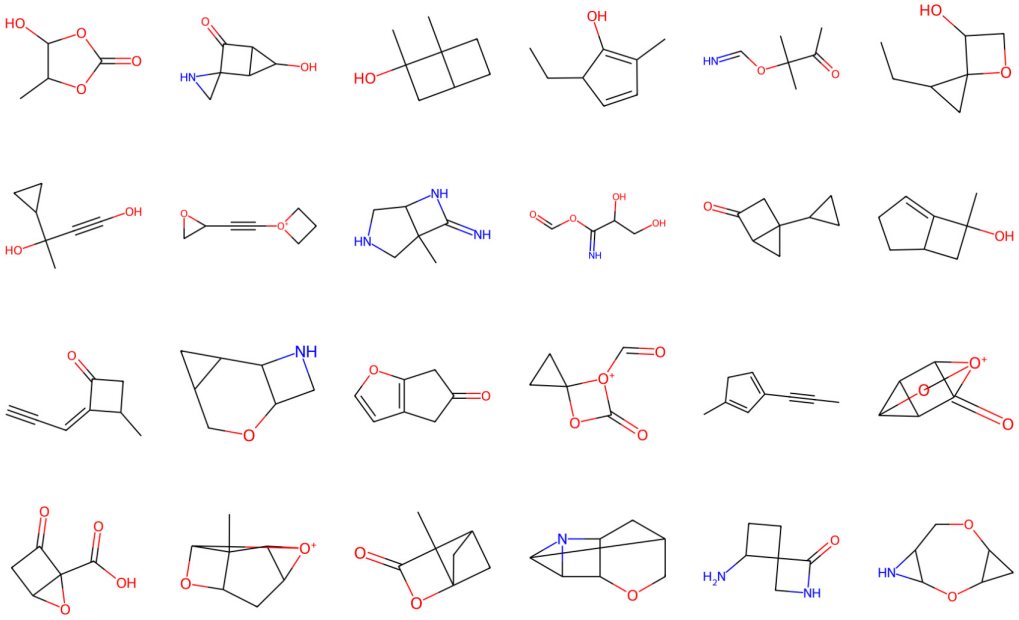


Figure 10: Visualization of the random samples generated by ESGD trained on QM9.

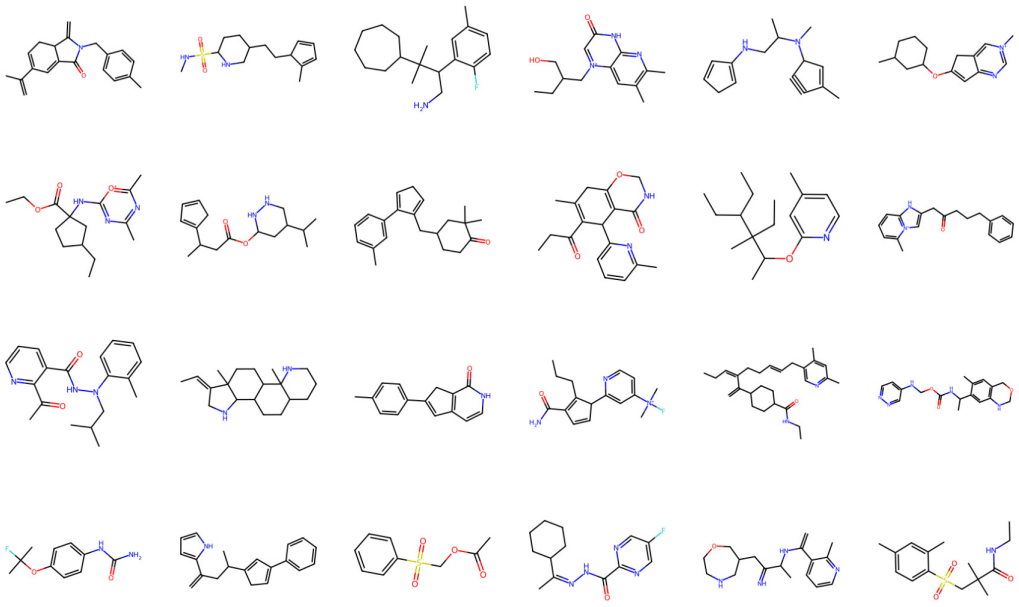


Figure 11: Visualization of the random samples generated by ESGD trained on ZINC250k.