Fast Graph Generation via Spectral Diffusion

Tianze Luo ⓘ, Zhanfeng Mo ⓘ, and Sinno Jialin Pan ⓘ, *Senior Member, IEEE*

Abstract—**Generating graph-structured data is a challenging problem, which requires learning the underlying distribution of graphs. Various models such as graph VAE, graph GANs, and graph diffusion models have been proposed to generate meaningful and reliable graphs, among which the diffusion models have achieved state-of-the-art performance. In this paper, we argue that running full-rank diffusion SDEs on the whole graph adjacency matrix space hinders diffusion models from learning graph topology generation, and hence significantly deteriorates the quality of generated graph data. To address this limitation, we propose an efficient yet effective Graph Spectral Diffusion Model (GSDM), which is driven by low-rank diffusion SDEs on the graph spectrum space. Our spectral diffusion model is further proven to enjoy a substantially stronger theoretical guarantee than standard diffusion models. Extensive experiments across various datasets demonstrate that our proposed GSDM turns out to be the SOTA model, by exhibiting both significantly higher generation quality and much less computational consumption than the baselines.**

Index Terms—**Graph diffusion, graph generative model, stochastic differential equations.**

I. INTRODUCTION

LEARNING to generate graph-structural data not only requires knowing the nodes' feature distribution but also a deep understanding of the underlying graph topology, which is essential to modeling various graph instances, such as social networks [1], [2], molecule structures [3], [4], neural architectures [5], recommender systems [6], etc. Conventional likelihood-based graph generative models, e.g. GraphGAN [7], GraphVAE [8] and GraphRNN [9], have demonstrated great strength on graph generation tasks. In general, a likelihood-based model is designed to learn the likelihood function of the underlying graph data distribution, with which one can draw new samples with preserved graph properties from the distribution of interest. However, most likelihood-based generative models
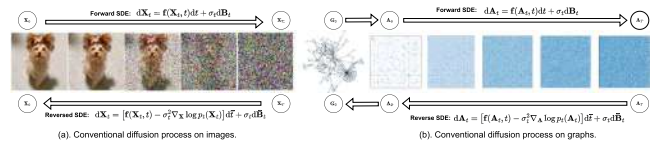
Fig. 1.   Illustration of the difference between applying conventional SDE diffusion on images (a) and graphs (b).

suffer from either limited quality of modeling graph structures, or considerable computational burden [10].

Recently, a series of diffusion-based generative models have been proposed to overcome the limitations of likelihood-based models. Although being originally established for image generation [11], diffusion models exhibit great success in graph generation tasks with complex graph structural properties [10], [12]. Roughly speaking, diffusion is a mathematical technique that involves a Stochastic Differential Equation (SDE) used to smoothly transform real data into pure noise by adding more noise. Diffusion models are a class of probabilistic generative models that use this technique in reverse to generate representative data samples from noise. These models are trained to learn the underlying structure of a dataset by simulating how data points gradually become blurry due to added noise. Therefore, a well-trained diffusion model allows us to restore the original data by reversing the diffusion process and gradually removing the noise from the blurred sample. The first graph diffusion model through SDEs, coined Graph Diffusion via SDE Systems (GDSS) [10], is designed to simultaneously generate node features and adjacency matrix via reversed diffusion. Similar to image diffusion models [11], [13], at each diffusion step, GDSS directly inserts standard Gaussian noise to both node features and the adjacency matrix. Meanwhile, two separate neural networks are trained to learn the score functions of the node features and adjacency matrix, respectively.

However, unlike the densely distributed image data, graph adjacency matrices can be highly sparse, which makes isotropic Gaussian noise insertion incompatible with graph structural data. In these circumstances, as shown in Fig. 1, there is a stark difference between the diffusion process on images and on graph adjacency matrices. As can be seen from the figure, the image corrupted by full-rank Gaussian noise exhibits recognizable numerical patterns along the early- and middle-stage of forward diffusion. However, the corrupted sparse graph adjacency matrix degenerates into a dense matrix with uniformly distributed entries in a few diffusion steps. In intuition, Fig. 1 implies that standard diffusion SDEs with full-rank isotropic noise insertion are destructive of learning graph topology and feature representations. Theoretically speaking, for extremely

| (a) Real data | (b) GSDM samples | (c) GDSS samples |

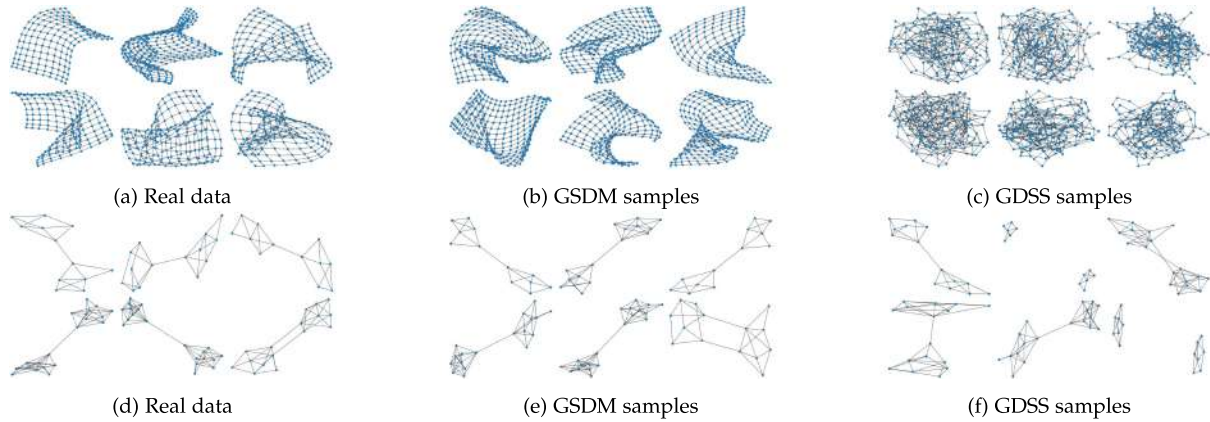| (d) Real data | (e) GSDM samples | (f) GDSS samples |

Fig. 2. Non-cherry-picked random samples from the testing set as well as samples generated by GSDM (ours) and GDSS [10], on Grid (top row) and Community-small (bottom row) datasets. For GDSS, we use the authors' released code and checkpoints to generate the samples.

sparse graphs (e.g. social networks, molecules) with low-rank adjacency matrices, the adjacency score functions are supported on a low-dimensional manifold embedded in the full adjacency matrix space. Thus, directly applying diffusion models on graph topology generation is not desirable: once the diffusion SDE is run in the full space of the adjacency matrix, lethal noise will be injected into the out-of-support regions and drives the signal-to-noise ratio to be essentially zero, which is fatal for training score networks.

Even for densely connected graphs, the standard diffusion model is problematic for topology generation. Unlike image pixels that are merely locally correlated, an adjacency matrix governs the message-passing pattern of the whole graph. Thus, isotropic Gaussian noise insertion severely distorts the message-passing pattern, by blindly encouraging message passing on sparsely connected parts, which impedes the representation learning of sparse regions.

In order to establish a graph-friendly diffusion model, one should design an appropriate diffusion scheme that is compatible with the graph topology structure. To this end, we propose the Graph Spectral Diffusion Model (GSDM), which is driven by diffusion SDEs on both the node feature space and the graph spectrum space. At each diffusion step, instead of corrupting the entire adjacency matrix, our method confines the Gaussian insertion to the graph spectrum space, i.e. the eigenvalue matrix of the adjacency matrix. This novel diffusion scheme enables us to perform smooth transformations on graph data during both the training and sampling phases. As illustrated in Fig. 2, our proposed GSDM significantly outperforms the standard graph diffusion model (GDSS [10]) in terms of graph generation quality and plausibility. For the Grid dataset shown in the top row of the figure, GDSS samples seem to be merely chaotic clusters, while GSDM samples exhibit smooth surface-like patterns that are visually similar to real data. For the Community-small dataset shown in the bottom row of the figure, GDSS fails to capture the link between two communities on some samples, while GSDM is able to capture dumbbell-like patterns (two clusters connected by one edge) as well as butterfly-like patterns (two clusters connected by two edges). This implies that GDSS's generation not only fails to mimic the observed topology distribution but

also suffers from capturing challenging details of the data such as links between communities. In contrast, GSDM is capable of generating high-quality graphs that are topologically similar to the real data, while retaining critical details.

We empirically evaluate the capability of our proposed GSDM on generic graph generation tasks, by evaluating the generation quality on both synthetic and real-world graph datasets. As shown in Section IV, GSDM outperforms existing one-shot generative models on various datasets, while achieving competitive performance to autoregressive models. Further molecule generation experiments show that our GSDM outperforms the state-of-the-art baselines, demonstrating that our proposed spectral diffusion model is capable of capturing complicated dependency between nodes and edges. Our main contributions are 3 folds:

- We propose a novel Graph Spectral Diffusion Model (GSDM) for fast and quality graph generation. Our method overcomes the limitations of existing graph diffusion models by leveraging diffusion SDEs on both the node feature and graph spectrum spaces.
- Through the lens of stochastic analysis, we prove that GSDM enjoys a substantially stronger performance guarantee than the standard graph diffusion model. Our proposed spectral diffusion sharpens the reconstruction error bound from $\mathcal{O}(n^2 \exp(n^2))$ to $\mathcal{O}(n \exp(n))$, where n is the number of nodes.
- We evaluate GSDM on both synthetic and real-world graph generation tasks. GSDM outperforms all existing graph generative models and also achieves evidently higher computational efficiency compared to existing graph diffusion models.

II. RELATED WORK

In recent years, there have been several advanced graph generation strategies proposed, as outlined in [3], [8], [9], [12], [14], [15] and [10]. Among these, GraphRNN [9] and GraphVAE [8] generate nodes and edges sequentially with validity checks, while GAN-based models [16], VAE-based models [15], flow-based models [3], and score-based models [10], [12] generate the entire graph in an integrative way and exhibit high computational

efficiency due to their node permutation-invariant property. Our proposed GSDM, which generates data by reversing a spectral diffusion SDE with a learned spectral score function, falls into the score-based model category. Specifically, GSDM accepts the destination of the reversed diffusion SDE as the final generated samples, without requiring any additional refinements.

A recently proposed score-based model, GDSS [10], is the first state-of-the-art diffusion-based generative model that simultaneously conducts nodes and edges generation. In essence, GDSS recasts the image diffusion paradigm [13] for graph generation. During the forward diffusion process, GDSS injects Gaussian noise into both the node features and the adjacency matrix at each diffusion step. Then, a neural network is trained to learn the score function by minimizing the score-matching objective, which enables a reversion of graph data from noise via a reversed time diffusion process. However, such a directly borrowed diffusion model is incompatible with graph topology generation: unlike images that are feature-rich, the graph adjacency matrix is generally sparse and low-rank. Hence, injecting isotropic Gaussian noise into the sparsely connected parts of the adjacency matrix severely harms the graph data distribution and makes it hard to be recovered from Gaussian noise.

Recent advancements in the field of image generation and molecule generation have been exploring ideas that are relevant to our work. Subspace Diffusion [17] adopts image downsampling to restrict the image diffusion process via projections onto a sequence of progressively smaller subspaces as the data evolves towards noise. Wavelet-Score Diffusion (WSGM) [18] adopts wavelet transformation on the image to construct multi-level subspaces, and conducts diffusion training and sampling on the corresponding subspaces. In the field of molecule generation, Torsional diffusion [19], DiffDock [20], and FoldingDiff [21] explore diffusion method for 3D molecule generation with a similar principle: they utilize diffusions on the flexible reparameterized space of 3D molecules instead of canonical 3D Cartesian space. Corresponding reparameterized spaces include: internal angles of atoms [21], chirality tags, bond lengths and torsion angles of atoms [19], as well as translations and rotations of molecules [20].

III. PRELIMINARIES

In this paper, we denote the probability space of interest as $(\Omega, \mathcal{F}, \mathbb{P})$ and $(\mathcal{F}_t)_{t \in \mathbb{R}}$ be a filtration, i.e. a sequence of increasing sub-σ-algebra of $\mathcal{F}$. Without specification, we denote $(\mathbf{B}_t)_{t \in \mathbb{R}}$ as the d-dimensional standard Brownian motion on the filtered probabilistic space $(\Omega, \mathcal{F}, \mathbb{P}, (\mathcal{F}_t)_{t \in \mathbb{R}})$. The distribution, support set, and expectation of a random variable $\mathbf{z}$ are defined as $\mathrm{law}(\mathbf{z})$, $\mathrm{supp}(\mathbf{z})$ and $\mathbb{E}[\mathbf{z}]$. $\mathbf{y}|\mathbf{z}$ denotes the distribution of $\mathbf{y}$ conditioned on $\mathbf{z}$. $\mathrm{Unif}(A)$ denotes the uniform distribution on a set A. $\|\cdot\|$ denotes the standard Euclid norm. $\|\cdot\|_\infty$ and $\|\cdot\|_{\mathrm{lip}}$ denotes the supremum norm and Lipschitz norm of a function. $\lfloor\cdot\rfloor$ denotes the flooring function.

A. Score-Based Generative Diffusion Model

A generative model refers to a mapping $g_{\boldsymbol{\theta}} : \mathbb{R}^d \mapsto \mathbb{R}^d$, which maps a simple known priori π to a complicated data distribution

$\mathcal{D}$. Once the model $g_{\boldsymbol{\theta}}$ is sufficiently trained on N i.i.d samples from $\mathcal{D}$, denoted by $\mathcal{S}$, it enables us to generate plausible instances from $\mathcal{D}$ directly, by sampling from $g_{\boldsymbol{\theta}}(\varepsilon)$, $\varepsilon \sim \pi$. Unlike conventional generative models, e.g. VAE and GAN, which treat $\mathcal{D}$ as a unilateral transformation of π, diffusion models consider the bilateral relation between $\mathcal{D}$ and π from the perspective of SDE. Given an SDE traveling from $\mathcal{D}$ to π, the corresponding reversed time SDE enables us to backtrack from noisy priori to the distribution of interest.

Lemma III.1 (Forward Diffusion and Reversed Time SDE [22]):

The Forward Diffusion refers to the following SDE

$$\mathbf{z}_0 \sim \mathcal{D}, \ \mathrm{d}\mathbf{z}_t = \mathbf{f}(\mathbf{z}_t, t)\mathrm{d}t + \sigma_t \mathrm{d}\mathbf{B}_t, \ t \in [0, 1], \quad (1)$$

where $\mathbf{f}(\cdot, t) : \mathbb{R}^d \mapsto \mathbb{R}^d$ is the drift function, $\sigma_t : [0, 1] \mapsto \mathbb{R}$ be a scalar diffusion function. Let $p_t(\cdot)$ be the probability density function of $\mathbf{z}_t$, then the Reversed Time SDE is given by

$$\mathrm{d}\bar{\mathbf{z}}_t = (\mathbf{f}(\bar{\mathbf{z}}_t, t) - \sigma_t^2 \nabla \log p_t(\bar{\mathbf{z}}_t))\mathrm{d}\bar{t} + \sigma_t \mathrm{d}\bar{\mathbf{B}}_t, \quad (2)$$

where $\bar{\mathbf{z}}_1 \sim \mathbf{z}_1, t \in [0, 1], \mathrm{d}\bar{t} = -\mathrm{d}t$ is the negative infinitesimal time step, $(\bar{\mathbf{B}}_t)_{t \in \mathbb{R}}$ is a reversed time Brownian motion w.r.t $(\Omega, \mathcal{F}, \mathbb{P}, (\bar{\mathcal{F}}_t)_{t \in \mathbb{R}})$, and $(\bar{\mathcal{F}}_t)_{t \in \mathbb{R}}$ is the corresponding decreasing filtration, and $\nabla \log p_t(\cdot)$ is the score function.

During the forward diffusion process, with a carefully designed $\mathbf{f}(\cdot, t)$, the original data is perturbed by Gaussian noise with increasing magnitude, and it is assumed to be gradually corrupted to a truly noisy signal (priori), i.e. $\mathrm{law}(\mathbf{z}_1) = \pi$. In order to draw new data from $\mathcal{D}$ via the reversed time SDE, one needs to learn the unknown score function $\nabla \log p_t(\cdot)$ with a neural network $s_{\boldsymbol{\theta}}(\cdot) : \mathbb{R}^d \mapsto \mathbb{R}^d$, by minimizing the following explicit score matching error:

$$\mathcal{E}(\boldsymbol{\theta}) \triangleq \mathbb{E}_{\mathbf{z} \sim \mathcal{D}} \mathbb{E}_{\mathbf{z}_t|\mathbf{z}} \|s_{\boldsymbol{\theta}}(\mathbf{z}_t) - \nabla \log p_t(\mathbf{z}_t)\|^2. \quad (3)$$

In practice, we employ a Gaussian priori π. For each sample $\mathbf{z}^i \in \mathcal{S}$, we first generate a sequence of corrupted data $\{\mathbf{z}^i_{t_j}\}_{j=1}^{T}$ by discretizing (1). To learn the score network $s_{\boldsymbol{\theta}}(\cdot)$, one can minimize a more tractable denoising score matching objective $\hat{\mathcal{E}}(\boldsymbol{\theta})$

$$\hat{\mathcal{E}}(\boldsymbol{\theta}) \triangleq \mathbb{E}_{\mathbf{z} \sim \mathrm{Unif}(\mathcal{S})} \mathbb{E}_{\mathbf{z}_t|\mathbf{z}} \|s_{\boldsymbol{\theta}}(\mathbf{z}_t) - \nabla \log p(\mathbf{z}_t|\mathbf{z})\|^2, \quad (4)$$

which has been proven to be equivalent to $\mathcal{E}(\boldsymbol{\theta})$ in [23]. Given a well trained score network $s_{\boldsymbol{\theta}^*}(\cdot)$, one is able to generate plausible data from π via solving the learned reversed-time SDE

$$\mathrm{d}\hat{\mathbf{z}}_t = (\mathbf{f}(\hat{\mathbf{z}}_t, t) - \sigma_t^2 s_{\boldsymbol{\theta}^*}(\hat{\mathbf{z}}_t))\mathrm{d}\bar{t} + \sigma_t \mathrm{d}\bar{\mathbf{B}}_t, \quad (5)$$

where $\hat{\mathbf{z}}_1 \sim \pi$ and $t \in [0, 1]$. Ideally, the learned reversed time SDE should lead us towards $\mathcal{D}$, i.e. $\mathrm{law}(\hat{\mathbf{z}}_0) = \mathcal{D}$.

IV. METHODOLOGY

In this section, we establish the Graph Spectral Diffusion Model (GSDM) for fast and effective graph data generation. In Section IV-A, we briefly review the standard score-based graph diffusion model [10]. In Section IV-B, we formally introduce our GSDM algorithm and its α-quantile variants. In Section IV-C, we provide theoretical analyses to justify the efficacy of GSDM on graph data generation.

A. Standard Graph Diffusion Model

A graph with n nodes is defined as $\mathbf{G} \triangleq (\mathbf{X}, \mathbf{A}) \in \mathbb{R}^{n \times d} \times \mathbb{R}^{n \times n}$, where $\mathbf{X} \in \mathbb{R}^{n \times d}$ is the node feature matrix with dimension d and $\mathbf{A} \in \mathbb{R}^{n \times n}$ denotes the adjacency matrix. A graph generative model aims to learn the underlying data distribution, say $\mathbf{G} \sim \mathcal{G}$, which is a joint distribution of both $\mathbf{X}$ and $\mathbf{A}$. Note that, if $(\mathbf{X}, \mathbf{A})$ is treated as a whole and omits the intrinsic graph structure, the aforementioned score-based generation framework can be parallelly extended to the graph generation setting, which yields the standard graph diffusion model, i.e. GDSS [10]. Roughly speaking, for each graph sample $(\mathbf{X}, \mathbf{A})$, we first generate a sequence of perturbed graphs $\{(\mathbf{X}_{t_i}, \mathbf{A}_{t_i})\}_{i=1}^{T}$ via forward diffusion. Then, we train two score networks $s_{\boldsymbol{\theta}}(\cdot)$ and $s_{\boldsymbol{\phi}}(\cdot)$ to learn the score functions for both $\mathbf{X}_t$ and $\mathbf{A}_t$, with which we can generate new data from π by running the reversed time SDE.

Definition 1 (Graph Diffusion SDEs):
The Forward Graph Diffusion refers to the following SDE system

$$\begin{cases} d\mathbf{X}_t = \mathbf{f}^X(\mathbf{X}_t, t)dt + \sigma_{X,t}d\mathbf{B}_t^X, \\ d\mathbf{A}_t = \mathbf{f}^A(\mathbf{A}_t, t)dt + \sigma_{A,t}d\mathbf{B}_t^A, \end{cases} \quad (6)$$

where $(\mathbf{X}_0, \mathbf{A}_0) \sim \mathcal{G}$, $t \in [0, 1]$, $\mathbf{f}^X(\cdot, t) : \mathbb{R}^{n \times d} \mapsto \mathbb{R}^{n \times d}$ and $\mathbf{f}^A(\cdot, t) : \mathbb{R}^{n \times n} \mapsto \mathbb{R}^{n \times n}$ is the drift functions for nodes feature and adjacency matrix; $\sigma_{X,t}, \sigma_{A,t}$ are the scalar diffusion terms (a.k.a noise schedule function), and $(\mathbf{B}_t^X)_{t \in \mathbb{R}}$ and $(\mathbf{B}_t^A)_{t \in \mathbb{R}}$ are standard Brownian motions on $\mathbb{R}^{n \times d}$ and $\mathbb{R}^{n \times n}$, respectively.

To alleviate the computational burden of calculating the drift w.r.t the high dimensional $\mathbf{G}$, the drift term of forward graph diffusion is disentangled into $\mathbf{f}^X(\cdot, t)$ and $\mathbf{f}^A(\cdot, t)$. Again, Lemma III.1 guarantees the existence of the reversed time SDEs for graph diffusion.

Corollary 1 (Reversed Time SDEs for Graph Diffusion): The reversed time SDE system of (6) is given by

$$\begin{cases} d\bar{\mathbf{X}}_t = \left(\mathbf{f}^X(\bar{\mathbf{X}}_t, t) - \sigma_{X,t}^2 \nabla_{\mathbf{X}} \log p_t(\bar{\mathbf{X}}_t, \bar{\mathbf{A}}_t)\right) d\bar{t} + \sigma_{X,t} d\bar{\mathbf{B}}_t^X, \\ d\bar{\mathbf{A}}_t = \left(\mathbf{f}^A(\bar{\mathbf{A}}_t, t) - \sigma_{A,t}^2 \nabla_{\mathbf{A}} \log p_t(\bar{\mathbf{X}}_t, \bar{\mathbf{A}}_t)\right) d\bar{t} + \sigma_{A,t} d\bar{\mathbf{B}}_t^A, \end{cases} \quad (7)$$

where $(\bar{\mathbf{X}}_1, \bar{\mathbf{A}}_1) \sim \pi$, $t \in [0, 1]$, $d\bar{t} = -dt$ is the negative infinitesimal time step, $(\bar{\mathbf{B}}_t^X)_{t \in \mathbb{R}}$ and $(\bar{\mathbf{B}}_t^A)_{t \in \mathbb{R}}$ are the reversed time standard Brownian motions induced by (6).

The disentanglement of the drift functions implies the conditional independence $\mathbf{X}_t \perp \mathbf{A}_t | \mathbf{G}_0$, with which we can decompose $p_t(\bar{\mathbf{X}}_t, \bar{\mathbf{A}}_t)$ into $p_{t|0}(\mathbf{X}_t | \mathbf{X}_0) \cdot p_{t|0}(\mathbf{A}_t | \mathbf{A}_0)$, where $p_{t|0}(\cdot)$ denotes the density function of $\mathbf{G}_t | \mathbf{G}_0$. As proposed in [10], such conditional independence reduces the denoising score-matching objective to a simpler form

$$\widehat{\mathcal{E}}(\boldsymbol{\theta}) \triangleq \mathbb{E}_{\mathbf{G} \sim \text{Unif}(\mathcal{S})} \mathbb{E}_{\mathbf{G}_t | \mathbf{G}} \| s_{\boldsymbol{\theta}}(\mathbf{G}_t) - \nabla \log p_{t|0}(\mathbf{X}_t | \mathbf{X}_0) \|^2, \quad (8)$$

$$\widehat{\mathcal{E}}(\boldsymbol{\phi}) \triangleq \mathbb{E}_{\mathbf{G} \sim \text{Unif}(\mathcal{S})} \mathbb{E}_{\mathbf{G}_t | \mathbf{G}} \| s_{\boldsymbol{\phi}}(\mathbf{G}_t) - \nabla \log p_{t|0}(\mathbf{A}_t | \mathbf{A}_0) \|^2. \quad (9)$$

Hence, the training and sampling procedures can be directly borrowed from standard score-based models.

While GDSS is the first attempt at leveraging diffusion models on graph generation, its performance is hindered by the brute-force application of diffusion. For sparsely connected graphs, while the distribution of node features varies across datasets, the distribution of graph topology, i.e. adjacency matrix, resides in a low dimensional manifold. As mentioned in Section I, an evident pattern of the adjacency matrices also implies that the true distribution of $\mathbf{A}$ is of low rank. In this case, the score-matching objective fails to provide consistency estimators. Although running a full rank diffusion on $\mathbf{A} \in \mathbb{R}^{n \times n}$ alleviates this issue by extending the support of corrupted data from the manifold to the full space, it inevitably introduces lethal noise to regions of zero probability density. As a consequence, the signal-to-noise ratio of regions out of $\text{supp}(\mathbf{A})$ is essentially zero, which is a catastrophe for training the denoising score network. Note that such full-rank diffusion is also inappropriate for densely connected graph generation. This is because an isotropically corrupted adjacency matrix encourages delusive message passing on sparsely connected parts of the graph, which is destructive to the graph message passing pattern. Thus, standard diffusion can severely impair representation learning for sparse graph regions.

B. Graph Spectral Diffusion Model

To address these notorious yet ubiquitous issues, we propose a novel Graph Spectral Diffusion Model. For graph topology generation, in contrast to GDSS which is driven by a full-rank diffusion on the whole space $\mathbb{R}^{n \times n}$, our GSDM leverages low-rank diffusion SDEs on the n-dimensional spectrum manifold, e.g. the span of n-eigenvalues of $\mathbf{A}$. As we shall see later, GSDM achieves both robustness and computational efficiency, by exploiting the graph spectrum structure and running diffusion on an information-concentrated manifold.

Definition 2 (Graph Spectral Diffusion SDEs):
Let the spectral decomposition of $\mathbf{A}$ be $\mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{\top}$, where columns of $\mathbf{U}$ are the orthonormal eigenvectors and $\boldsymbol{\Lambda}$ be the diagonal eigenvalue matrix, i.e. spectrum. The Forward Spectral Diffusion refers to the following SDE system

$$\begin{cases} d\mathbf{X}_t = & \mathbf{f}^X(\mathbf{X}_t, t)dt + \sigma_{X,t}d\mathbf{B}_t^X, \\ d\boldsymbol{\Lambda}_t = & \mathbf{f}^{\Lambda}(\boldsymbol{\Lambda}_t, t)dt + \sigma_{\Lambda,t}d\mathbf{W}_t^{\Lambda}, \end{cases} \quad (10)$$

where $(\mathbf{X}_0, \mathbf{A}_0) \sim \mathcal{G}$, $\mathbf{A}_0 = \mathbf{U}_0 \boldsymbol{\Lambda}_0 \mathbf{U}_0^{\top}$, $t \in [0, 1]$, $\mathbf{f}^X(\cdot, t) : \mathbb{R}^{n \times d} \mapsto \mathbb{R}^{n \times d}$ is the drift for nodes feature; $\mathbf{f}^{\Lambda}(\cdot, t) : \mathbb{R}^n \mapsto \mathbb{R}^n$ is the drift for spectrum, which only acts on the diagonal entries; $\sigma_{X,t}, \sigma_{\Lambda,t}$ are the scalar diffusion terms (a.k.a noise schedule functions), $(\mathbf{B}_t^X)_{t \in \mathbb{R}}$ and $(\mathbf{B}_t^{\Lambda})_{t \in \mathbb{R}}$ are standard Brownian motions on $\mathbb{R}^{n \times d}$ and $\mathbb{R}^n$, respectively; and $\mathbf{W}_t^{\Lambda} \triangleq \text{diag}(\mathbf{B}_t^{\Lambda})$ is a diagonal Brownian motion.

As will be illustrated later, the evolution of $\mathbf{A}_t \triangleq \mathbf{U}_0 \boldsymbol{\Lambda}_t \mathbf{U}_0^{\top}$ is driven by a n-dimensional Gaussian process. Hence, we prevent the corrupted adjacency matrix from rampaging around the full

space. We are now ready to establish the reversed time spectral diffusion SDEs.

Corollary 2 (Reversed Time Spectral Diffusion SDEs): The reversed time Spectral Diffusion SDE system of (10) is given by

$$
\begin{cases}
d\bar{\mathbf{X}}_t = \left(\mathbf{f}^X(\bar{\mathbf{X}}_t, t) - \sigma_{X,t}^2 \nabla_{\mathbf{X}} \log p_t(\bar{\mathbf{X}}_t, \bar{\mathbf{\Lambda}}_t)\right) d\bar{t} + \sigma_{X,t} d\bar{\mathbf{B}}_t^X, \\
d\bar{\mathbf{\Lambda}}_t = \left(\mathbf{f}^\Lambda(\bar{\mathbf{\Lambda}}_t, t) - \sigma_{\Lambda,t}^2 \nabla_{\mathbf{\Lambda}} \log p_t(\bar{\mathbf{X}}_t, \bar{\mathbf{\Lambda}}_t)\right) d\bar{t} + \sigma_{\Lambda,t} d\bar{\mathbf{W}}_t^\Lambda,
\end{cases}
$$
$$(11)$$

where $(\bar{\mathbf{X}}_1, \bar{\mathbf{\Lambda}}_1) \sim \pi$, $t \in [0,1]$, $d\bar{t} = -dt$ is the negative infinitesimal time step; $(\bar{\mathbf{B}}_t^X)_{t \in \mathbb{R}}$ and $(\bar{\mathbf{B}}_t^\Lambda)_{t \in \mathbb{R}}$ are reversed time standard Brownian motions induced by (6), and $\bar{\mathbf{W}}_t^\Lambda \triangleq \mathrm{diag}(\bar{\mathbf{B}}_t^\Lambda)$.

Since $\mathbf{U}$ is no longer involved in (11), the boundary condition is only imposed on the joint distribution of $(\mathbf{X}_1, \mathbf{\Lambda}_1)$ such that $\mathrm{law}(\mathbf{X}_1, \mathbf{\Lambda}_1) = \pi$. This assumption implies that the authentic distribution $(\mathbf{X}_0, \mathbf{\Lambda}_0)$ can be recovered from a priori π by the reversed time spectral diffusion SDE. According to score matching techniques [23], we can train two score networks $s_{\boldsymbol{\theta}}(\cdot, \cdot, \cdot), s_{\boldsymbol{\phi}}(\cdot, \cdot, \cdot)$ to learn the score functions $\nabla_{\mathbf{X}} \log p_t(\cdot, \cdot), \nabla_{\mathbf{\Lambda}} \log p_t(\cdot, \cdot)$ via minimizing

$$
\widehat{\mathcal{E}}(\boldsymbol{\theta})
$$
$$
\triangleq \mathbb{E}_{\mathbf{G} \sim \mathrm{Unif}(\mathcal{S})} \mathbb{E}_{\mathbf{X}_t | \mathbf{G}} \| s_{\boldsymbol{\theta}}(\mathbf{X}_t, \mathbf{\Lambda}_t, \mathbf{U}_0) - \nabla \log p_{t|0}(\mathbf{X}_t | \mathbf{X}_0) \|^2,
$$
$$
\widehat{\mathcal{E}}(\boldsymbol{\phi})
$$
$$
\triangleq \mathbb{E}_{\mathbf{G} \sim \mathrm{Unif}(\mathcal{S})} \mathbb{E}_{\mathbf{\Lambda}_t | \mathbf{G}} \| s_{\boldsymbol{\phi}}(\mathbf{X}_t, \mathbf{\Lambda}_t, \mathbf{U}_0) - \nabla \log p_{t|0}(\mathbf{\Lambda}_t | \mathbf{\Lambda}_0) \|^2.
$$

In a nutshell, the proposed GSDM has two main steps:

1) Run forward spectral diffusion model on $(\mathbf{X}, \mathbf{\Lambda})$ by (10). Train two score networks $s_{\boldsymbol{\theta}}(\cdot, \cdot, \cdot), s_{\boldsymbol{\phi}}(\cdot, \cdot, \cdot)$ to learn score functions showing up in (11).

2) Uniformly sample $\widehat{\mathbf{U}}_0$ from the observed eigenvector matrices. Generate plausible $(\widehat{\mathbf{X}}_0, \widehat{\mathbf{\Lambda}}_0)$ by reversing the spectral diffusion SDEs from $t = 1$ to $t = 0$, with estimated score functions $s_{\boldsymbol{\theta}}(\widehat{\mathbf{X}}_t, \widehat{\mathbf{\Lambda}}_t, \widehat{\mathbf{U}}_0), s_{\boldsymbol{\phi}}(\widehat{\mathbf{X}}_t, \widehat{\mathbf{\Lambda}}_t, \widehat{\mathbf{U}}_0)$.

Since the full adjacency matrices are not involved in the computation of diffusion SDEs, GSDM achieves significant acceleration in both training and sampling. Moreover, one can further enhance the computational efficiency, by confining the spectral diffusion to the top-k largest eigenvalues of $\mathbf{A}$, where $k \triangleq [\alpha n]$. Suppose $\mathbf{\Lambda}^k$ is the truncated eigenvalue matrix, where only the top-k diagonal entries of $\mathbf{\Lambda}$ are preserved, we can define the α-quantile GSDM by substituting the occurrence of $\mathbf{\Lambda}$ in GSDM with $\mathbf{\Lambda}^{(k)}$. As will be seen in the following section, α-quantile GSDM exhibits evidently faster processing speed and comparable performance to GSDM.

The pseudo codes of training and sampling with GSDM are described in Algorithms 1 and 2. Specifically, we first train a GSDM from real data by minimizing score-matching errors. On top of that, we are able to generate graph features and eigenvalues of graph adjacency matrices by reversing the forward spectral SDE. By uniformly sampling eigenvectors from the training set, we can construct a plausible graph adjacency matrix via spectral composition.

Algorithm 1: Training of GSDM.

Input: Score networks $\mathbf{s}_{\boldsymbol{\theta},t}(\cdot, \cdot, \cdot), \mathbf{s}_{\boldsymbol{\phi},t}(\cdot, \cdot, \cdot)$, maximal diffusion time T, drift functions $\mathbf{f}^X(\cdot, t), \mathbf{f}^\Lambda(\cdot, t)$, noise schedules $\sigma_{X,t}, \sigma_{\Lambda,t}$, learning rate η, training epochs K.
Output: Optimized score network parameters $\boldsymbol{\theta}_K, \boldsymbol{\phi}_K$.
1: Initialize $\boldsymbol{\theta}_0, \boldsymbol{\phi}_0$
2:   **for** $k = 1$ **to** K **do**
3:     $(\mathbf{X}_0, \mathbf{A}_0) \sim \mathcal{G}$;
4:     $\mathbf{\Lambda}_0, \mathbf{U}_0 \leftarrow \mathrm{EigenDecomposition}(\mathbf{A}_0)$;
5:     $t \sim \mathrm{Unif}([0, T])$;
6:     $\mathbf{X}_t \sim \int_0^t f^X(\mathbf{X}_\tau, \tau) d\tau + \int_0^t \sigma_{X,\tau} d\mathbf{B}_\tau^X$
7:     $\mathbf{\Lambda}_t \sim \int_0^t f^\Lambda(\mathbf{\Lambda}_\tau, \tau) d\tau + \int_0^t \sigma_{\Lambda,\tau} d\mathbf{B}_\tau^\Lambda$;
8:     $\widehat{\mathcal{E}}(\boldsymbol{\theta}_k) \leftarrow \| s_{\boldsymbol{\theta}_k}(\mathbf{X}_t, \mathbf{\Lambda}_t, \mathbf{U}_0) - \nabla \log p_{t|0}(\mathbf{X}_t | \mathbf{X}_0) \|^2$;
9:     $\widehat{\mathcal{E}}(\boldsymbol{\phi}_k) \leftarrow \| s_{\boldsymbol{\phi}_k}(\mathbf{X}_t, \mathbf{\Lambda}_t, \mathbf{U}_0) - \nabla \log p_{t|0}(\mathbf{\Lambda}_t | \mathbf{\Lambda}_0) \|^2$;
10:    $(\boldsymbol{\theta}_{k+1}, \boldsymbol{\phi}_{k+1}) \leftarrow (\boldsymbol{\theta}_k, \boldsymbol{\phi}_k) - \eta(\nabla \widehat{\mathcal{E}}(\boldsymbol{\theta}_k), \nabla \widehat{\mathcal{E}}(\boldsymbol{\phi}_k))$;
11: **end for**
12: **Return:** $\boldsymbol{\theta}_K, \boldsymbol{\phi}_K$;

C. Theoretical Analysis

Here, we provide supportive theoretical evidence for the efficacy of GSDM. In Proposition 1, we first study the low-rank structure of the spectral diffusion SDE of adjacency matrices. In Proposition 2, we further prove that our proposed spectral diffusion enjoys a sharper reconstruction error bound than the standard graph diffusion model.

Proposition 1 (Spectral Diffusion SDEs on Adjacency Matrix): Suppose $\mathbf{f}_\Lambda(\mathbf{\Lambda}, t) \triangleq -\sigma_{\Lambda,t}^2/2\mathbf{\Lambda}$. The spectral diffusion SDE system (10) induces an n-dimensional SDE system of the adjacency matrix $\mathbf{A}$ on the full space $\mathbb{R}^{n \times n}$. Following previous notations, the forward diffusion SDE is given by

$$
\begin{cases}
d\mathbf{X}_t = \mathbf{f}^X(\mathbf{X}_t, t)dt + \sigma_{X,t}d\mathbf{B}_t^X, \\
d\mathbf{A}_t = -\frac{1}{2}\sigma_{\Lambda,t}^2 \mathbf{A}_t dt + \sigma_{\Lambda,t}d\mathbf{M}_t,
\end{cases}
$$
$$(12)$$

where $(\mathbf{M}_t)_{t \in [0,1]}$ is a n-dimensional centered Gaussian process on $\mathbb{R}^{n \times n}$, with zero mean and covariance kernel $\mathcal{K}(s, t) : [0,1] \times [0,1] \mapsto \mathbb{R}^{n \times n \times n \times n}$ as

$$
\mathcal{K}(s, t)_{i,j,k,l} = \min(s, t) \cdot \sum_{h=1}^n \mathbf{U}_0[i, h]\mathbf{U}_0[j, h]\mathbf{U}_0[k, h]\mathbf{U}_0[l, h].
$$

Hence, the conditional distribution of $\mathbf{A}_t$ on $\mathbf{A}_0$ is Gaussian

$$
\mathbf{A}_t | \mathbf{A}_0 \sim N\left(\mathbf{A}_t; \mathbf{A}_0 e^{-\frac{1}{2}\int_0^t \sigma_\tau^2 d\tau}, (1 - e^{-\int_0^t \sigma_\tau^2 d\tau})\mathcal{K}(1, 1)\right),
$$
$$(13)$$

which admits a closed-form probability density function.

Remark 1: The proof can be found in Section V-A. Proposition 1 shows that our spectral diffusion framework substantially recasts the evolution of the adjacency matrix, by driving the n^2-dimensional SDE with n-dimensional noise $(\mathbf{M}_t)_{t \in \mathbb{R}}$. Guided by the graph spectrum, the diffusion is concentrated to the salient parts of $\mathrm{supp}(\mathbf{A})$ to prevent introducing irreducible noise to the out-of-support regions.

The central question for graph generation is how to measure the quality of the synthesis data that is recovered from noise with

Algorithm 2: Sampling via GSDM.

Input: Score-based models $\mathbf{s}_{\boldsymbol{\theta},t}(\cdot,\cdot,\cdot)$ and $\mathbf{s}_{\boldsymbol{\phi},t}(\cdot,\cdot,\cdot)$, maximal diffusion time T, number of sampling steps M, Langevin-MCMC step sizes $\{\epsilon_i\}_{i=1}^M$, noise schedules $\{\beta_i\}_{i=1}^M$ and priori distribution π. Initialize eigenvectors randomness $\sigma > 0$. Empirical average eigenvectors $\bar{\mathbf{U}}$.

Output: Generated graph data $(\widehat{\mathbf{X}}_0, \widehat{\mathbf{A}}_0)$.

1: $t \leftarrow T$;
2: $(\widehat{\mathbf{X}}_T, \widehat{\boldsymbol{\Lambda}}_T) \sim \pi$
3: $\widehat{\mathbf{U}}_0 \sim \text{Unif}(\{\mathbf{U} \triangleq \text{EigenVectors}(\mathbf{A}), (\mathbf{X},\mathbf{A}) \sim \mathcal{G}\})$;
4: **for** $m = M - 1$ **to** 0 **do**
5:   $\mathbf{S}_X \leftarrow \mathbf{s}_{\boldsymbol{\theta},t}(\widehat{\mathbf{X}}_t, \widehat{\boldsymbol{\Lambda}}_t, \widehat{\mathbf{U}}_0), \mathbf{S}_\Lambda \leftarrow \mathbf{s}_{\boldsymbol{\phi},t}(\widehat{\mathbf{X}}_t, \widehat{\boldsymbol{\Lambda}}_t, \widehat{\mathbf{U}}_0)$
6:   $t' \leftarrow t - T/(2\,M)$;
7:   $\widehat{\mathbf{X}}_{t'} \leftarrow (2 - \sqrt{1 - \beta_{m+1}}\widehat{\mathbf{X}}_t + \beta_{m+1}\mathbf{S}_X) +$
      $\sqrt{\beta_{m+1}}\mathbf{z}_X, \mathbf{z}_X \sim N(\mathbf{0},\mathbf{I})$ {Prediction step: $\mathbf{X}$}
8:   $\widehat{\boldsymbol{\Lambda}}_{t'} \leftarrow (2 - \sqrt{1 - \beta_{m+1}}\widehat{\boldsymbol{\Lambda}}_t + \beta_{m+1}\mathbf{S}_\Lambda) +$
      $\sqrt{\beta_{m+1}}\mathbf{z}_\Lambda, \mathbf{z}_\Lambda \sim N(\mathbf{0},\mathbf{I})$; {Prediction step: $\boldsymbol{\Lambda}$}
9:   $\mathbf{S}_X \leftarrow \mathbf{s}_{\boldsymbol{\theta},t'}(\widehat{\mathbf{X}}_{t'}, \widehat{\boldsymbol{\Lambda}}_{t'}, \widehat{\mathbf{U}}_0)$,
10:   $\mathbf{S}_\Lambda \leftarrow \mathbf{s}_{\boldsymbol{\phi},t'}(\widehat{\mathbf{X}}_{t'}, \widehat{\boldsymbol{\Lambda}}_{t'}, \widehat{\mathbf{U}}_0)$;
11:   $t \leftarrow t' - T/(2\,M)$;
12:   $\widehat{\mathbf{X}}_t \leftarrow \widehat{\mathbf{X}}_{t'} + \epsilon_i \mathbf{S}_X + \sqrt{2\epsilon_i}\mathbf{z}_X, \mathbf{z}_X \sim N(\mathbf{0},\mathbf{I})$;
      {Correction step: $\mathbf{X}$}
13:   $\widehat{\boldsymbol{\Lambda}}_t \leftarrow \widehat{\boldsymbol{\Lambda}}_{t'} + \epsilon_i \mathbf{S}_\Lambda + \sqrt{2\epsilon_i}\mathbf{z}_\Lambda, \mathbf{z}_\Lambda \sim N(\mathbf{0},\mathbf{I})$;
      {Correction step: $\boldsymbol{\Lambda}$}
14: **end for**
15: $\widehat{\mathbf{A}}_0 = \widehat{\mathbf{U}}_0 \widehat{\boldsymbol{\Lambda}}_0 \widehat{\mathbf{U}}_0^{\mathbf{T}}$;
16: **Return:** $(\widehat{\mathbf{X}}_0, \widehat{\mathbf{A}}_0)$;

the learned score function. The key to this question is to establish the reconstruction error bound, i.e. the expected error between the data reconstructed with ground truth score $\nabla \log p_t(\cdot)$ and the learned scores $s_\phi(\cdot)$. For graph topology generation, our GSDM is proven to enjoy a sharper reconstruction bound than standard graph diffusion. Detailed proof can be found in Section V-B.

Proposition 2 (Reconstruction Bounds for Adjacency Matrix Generation): Following the previous notations, we define the estimated reversed time SDEs for standard and spectral graph diffusion models as

$$\mathrm{d}\widehat{\mathbf{A}}_t^{\text{full}} = \left(-\frac{1}{2}\sigma_t^2 \widehat{\mathbf{A}}_t^{\text{full}} - \sigma_t^2 s_\phi(\widehat{\mathbf{A}}_t^{\text{full}}, t)\right)\mathrm{d}\bar{t} + \sigma_t \mathrm{d}\bar{\mathbf{B}}_t^A,$$

$$\mathrm{d}\widehat{\mathbf{A}}_t^{\text{spec}} = \left(-\frac{1}{2}\sigma_t^2 \widehat{\mathbf{A}}_t^{\text{spec}} - \sigma_t^2 s_\varphi(\widehat{\mathbf{A}}_t^{\text{spec}}, t)\right)\mathrm{d}\bar{t} + \sigma_t \mathrm{d}\bar{\mathbf{M}}_t,$$

where both generation methods share the same drift term and noise schedule $(\sigma_t)_{t\in[0,1]}$. We further assume that $\|s_\phi(\cdot,t)\|_{\text{lip}} = \mathcal{O}(\mathbb{E}_{|\mathbf{A}_0}\|\nabla_\mathbf{A}\log p_{t|0}(\widehat{\mathbf{A}}_t^{\text{full}})\|_{\text{lip}})$ and $\|s_\varphi(\cdot,t)\|_{\text{lip}} = \mathcal{O}(\mathbb{E}_{|\mathbf{A}_0}\|\nabla_\mathbf{A}\log p_{t|0}(\widehat{\mathbf{A}}_t^{\text{spec}})\|_{\text{lip}})$ holds almost surly. By reversing both SDEs from $t = 1$ to $t = 0$, we obtain $\widehat{\mathbf{A}}_t^{\text{full}}$ and $\widehat{\mathbf{A}}_t^{\text{spec}}$, which are two reconstructions of the authentic $\mathbf{A}_0$, with expected reconstruction errors bounded by

$$\mathbb{E}\|\mathbf{A}_0 - \widehat{\mathbf{A}}_0^{\text{full}}\|^2$$
$$\leqslant M\mathcal{E}(\phi) \cdot \left(1 + n^2 K \int_0^1 \Sigma_t^{-2} \exp\left(n^2 K \int_t^1 \Sigma_s^{-2}\mathrm{d}s\right)\mathrm{d}t\right),$$

$$\mathbb{E}\|\mathbf{A}_0 - \widehat{\mathbf{A}}_0^{\text{spec}}\|^2$$
$$\leqslant M\mathcal{E}(\varphi) \cdot \left(1 + nK \int_0^1 \Sigma_t^{-2} \exp\left(nK \int_t^1 \Sigma_s^{-2}\mathrm{d}s\right)\mathrm{d}t\right),$$

where $M \triangleq C^2\|\sigma_\cdot\|_\infty^4$; $K \triangleq 2ML/\mathbb{E}\|\mathbf{A}_0\|_{2,2}$; C, L are absolute constants; $\Sigma_t^2 \triangleq 1 - e^{-\int_0^t \sigma_s^2\mathrm{d}s}$; $\mathcal{E}(\cdot)$ is the expected score matching objective defined in (3).

Remark 2: While the error bound of $\widehat{\mathbf{A}}_0^{\text{full}}$ is at the order of $\mathcal{O}(n^2 \exp(n^2))$, $\widehat{\mathbf{A}}_0^{\text{spec}}$ exhibits a much sharper bound of order $\mathcal{O}(n \exp(n))$. For large-scale graphs with a large number of nodes n, our proposed GSDM enjoys a substantially better performance guarantee, which coincides with our numerical results. Moreover, under additional conditions, the error bounds can be significantly improved for well-trained score networks ϕ^*, φ^*, by showing that $\mathcal{E}(\phi^*) \ll \mathcal{E}(\phi)$ as in Proposition 3. The detailed proof can be found in Section V-C.

Definition 3 (β-smooth): $f : \mathbb{R}^m \mapsto \mathbb{R}^n$ is called β-smooth if and only if

$$\|f(x_1) - f(x_2) - \nabla f(x_2)^\top(x_1 - x_2)\| \leqslant \frac{\beta}{2}\|x_1 - x_2\|^2$$

holds for $\forall\, x_1, x_2 \in \mathbb{R}^m$.

Proposition 3 (Convergence of Score Matching Objective Minimization): Recall that the score matching objective of a generic sample $\mathbf{Z} \in \mathbb{R}^d$ ($\widehat{\mathbf{A}}^{\text{full}}$ or $\widehat{\mathbf{A}}^{\text{spec}}$) is defined as

$$\mathcal{E}(\boldsymbol{\theta};\mathbf{Z}) \triangleq \mathbb{E}_{\mathbf{Z}_t|\mathbf{Z}}\|s_{\boldsymbol{\theta}}(\mathbf{Z}_t, t) - \nabla_\mathbf{Z}\log p_t(\mathbf{Z}_t)\|^2, \quad (14)$$

and expected and empirical score-matching errors are defined as

$$\mathcal{E}(\boldsymbol{\theta}) \triangleq \mathbb{E}_{\mathbf{Z}\sim\mathcal{D}}\mathcal{E}(\boldsymbol{\theta};\mathbf{Z}) \quad (15)$$

$$\widehat{\mathcal{E}}(\boldsymbol{\theta};\mathcal{S}_N) \triangleq \mathbb{E}_{\mathbf{Z}\sim\mathcal{S}_N}\mathcal{E}(\boldsymbol{\theta};\mathbf{Z}) \quad (16)$$

where $\mathcal{D}$ is the population distribution of $\mathbf{Z}$, and $\mathcal{S} \triangleq \{\mathbf{Z}^i\}_{i=1}^N$ is the uniform distribution over an i.i.d sampled training dataset of size N. Suppose $\mathcal{E}(\boldsymbol{\theta})$ is minimized via running standard Stochastic Gradient Descent (SGD) on training data, i.e. at the k-th iteration, $\boldsymbol{\theta}_k$ is updated on a mini-batch of size b

$$\boldsymbol{\theta}_{k+1} \triangleq \boldsymbol{\theta}_k - \eta\nabla_{\boldsymbol{\theta}}\widehat{\mathcal{E}}(\boldsymbol{\theta};\mathcal{S}_b). \quad (17)$$

Assume that almost surely (w.r.t $\mathbf{Z}$), $\mathcal{E}(\cdot;\mathbf{Z})$ is β-smooth, and the tangent kernel $\mathbf{K}_{\boldsymbol{\theta}}(\mathcal{S}) \in \mathbb{R}^{Nd\times Nd}$ of $s_{\boldsymbol{\theta}}(\cdot)$ satisfies

$$\lambda_{\min}(\mathbf{K}_{\boldsymbol{\theta}}(\mathcal{S})) \geqslant \lambda > 0, \; \boldsymbol{\theta} \in B(\boldsymbol{\theta}_0, R),$$

$$\mathbf{K}_{\boldsymbol{\theta}}(\mathcal{S})[Ni + 1 : N(i+1), Nj + 1 : N(j+1)]$$
$$\triangleq \nabla_{\boldsymbol{\theta}}s_{\boldsymbol{\theta}}(\mathbf{Z}^i)^\top\nabla_{\boldsymbol{\theta}}s_{\boldsymbol{\theta}}(\mathbf{Z}^j),$$

with $R = 2N\sqrt{2\beta\mathcal{E}(\boldsymbol{\theta}_0)}/(\mu\delta), \delta > 0$. Then, with probability $1 - \delta$ over the choice of mini-batch $\mathcal{S}_b$, SGD with a learning rate $\eta \leqslant \frac{\lambda/N}{N\beta(N^2\beta + \lambda(b-1)/N)}$ converges to a global solution in the ball $B(\boldsymbol{\theta}_0, R)$ with exponential convergence rate

$$\mathcal{E}(\boldsymbol{\theta}_k) \leqslant \left(1 - \frac{\lambda b\eta}{N}\right)^k \mathcal{E}(\boldsymbol{\theta}_0). \quad (18)$$

V. Detailed Proofs

A. Proof of Proposition 1

Proof: Plugging $\mathbf{A}_t = \mathbf{U}_0 \mathbf{\Lambda}_t \mathbf{U}_0^\top$ into (12) yields

$$\mathrm{d}\mathbf{A}_t = -\frac{1}{2}\sigma_{\Lambda,t}^2 \mathbf{A}_t \mathrm{d}t + \sigma_{\Lambda,t}\mathrm{d}\mathbf{M}_t,$$

$$\mathbf{M}_t \triangleq \mathbf{U}_0 \mathbf{W}_t^\Lambda \mathbf{U}_0^\top = \mathbf{U}_0 \cdot \mathrm{diag}(\mathbf{B}_t^\Lambda) \cdot \mathbf{U}_0^\top.$$

Since $(\mathbf{M}_t)_{t \in \mathbb{R}}$ is a linear transformation of $(\mathbf{B}_t^\Lambda)_{t \in \mathbb{R}}$, the standard Brownian motion on $\mathbb{R}^n$, hence $(\mathbf{M}_t)_{t \in \mathbb{R}}$ is a rank-n centered Gaussian process in $\mathbb{R}^{n \times n}$. Moreover, $\mathbf{M}_t$ is characterized by its covariance kernel $\mathcal{K}(s,t) : [0,1] \times [0,1] \mapsto \mathbb{R}^{n \times n \times n \times n}$, which is given by

$$\mathcal{K}(s,t)_{i,j,k,l}$$
$$\triangleq \mathrm{Cov}(\mathbf{M}_s[i,j], \mathbf{M}_t[k,l]) = \mathbb{E}[\mathbf{M}_s[i,j]\mathbf{M}_t[k,l]]$$
$$= \mathbb{E}\left(\sum_{h=1}^n \mathbf{U}_0[i,h]\mathbf{B}_s^\Lambda[h]\mathbf{U}_0[j,h]\right)$$
$$\times \left(\sum_{h=1}^n \mathbf{U}_0[k,h]\mathbf{B}_t^\Lambda[h]\mathbf{U}_0[l,h]\right)$$
$$= \sum_{h=1}^n \mathbb{E}(\mathbf{B}_s^\Lambda[h]\mathbf{B}_t^\Lambda[h])\mathbf{U}_0[i,h]\mathbf{U}_0[j,h]\mathbf{U}_0[k,h]\mathbf{U}_0[l,h]$$
$$= \min(s,t)\sum_{h=1}^n \mathbf{U}_0[i,h]\mathbf{U}_0[j,h]\mathbf{U}_0[k,h]\mathbf{U}_0[l,h].$$

Notice that (10) is an Ornstein–Uhlenbeck process, which admits a closed-form solution

$$\mathbf{\Lambda}_t = \mathbf{\Lambda}_0 e^{-\frac{1}{2}\int_0^t \sigma_\tau^2 \mathrm{d}\tau} + (1 - e^{-\int_0^t \sigma_\tau^2 \mathrm{d}\tau})\mathbf{W}_1^\Lambda. \quad (19)$$

Hence, plugging (19) into $\mathbf{A}_t = \mathbf{U}_0 \mathbf{\Lambda}_t \mathbf{U}_0^\top$ yields

$$\mathbf{A}_t = \mathbf{A}_0 e^{-\frac{1}{2}\int_0^t \sigma_\tau^2 \mathrm{d}\tau} + (1 - e^{-\int_0^t \sigma_\tau^2 \mathrm{d}\tau})\mathbf{M}_1. \quad (20)$$

This completes the proof. $\square$

B. Proof of Proposition 2

In preparation for the main proof, we first establish some technical lemmas.

Lemma V.1 (Reconstruction Bound for Generic Diffusion): We first consider the following oracle reversed time SDE on $(\mathbb{R}^d, \|\cdot\|)$

$$\mathrm{d}\bar{\mathbf{Z}}_t = \left(\mathbf{f}(\bar{\mathbf{Z}}_t, t) - \sigma_t^2 \nabla_{\mathbf{Z}} \log p_t(\bar{\mathbf{Z}}_t)\right)\mathrm{d}\bar{t} + \sigma_t \mathrm{d}\bar{\mathbf{B}}_t, \ t \in [0,1],$$

and we define the corresponding estimated reverse time SDE as

$$\mathrm{d}\widehat{\mathbf{Z}}_t = \left(\mathbf{f}(\widehat{\mathbf{Z}}_t, t) - \sigma_t^2 s_\phi(\widehat{\mathbf{Z}}_t, t)\right)\mathrm{d}\bar{t} + \sigma_t \mathrm{d}\bar{\mathbf{B}}_t, \ t \in [0,t],$$

where $s_\phi(\cdot)$ is optimized to predict the Stein score function $\nabla_{\mathbf{Z}} \log p_t(\mathbf{Z}_t)$ by minimizing the score matching objective

$$\min_\phi \mathcal{E}(\phi) \triangleq \mathbb{E}_{\mathbf{Z}}\mathbb{E}_{\mathbf{Z}_t|\mathbf{Z}}\|s_\phi(\mathbf{Z}_t, t) - \nabla_{\mathbf{Z}} \log p_t(\mathbf{Z}_t)\|^2.$$

Then for any ϕ, the construction error is bounded by
$$\mathbb{E}\|\mathbf{Z}_0 - \widehat{\mathbf{Z}}_0\|^2 \leqslant C^2 \|\sigma.\|_\infty^4 \mathcal{E}(\phi)$$
$$\cdot \left(1 + \int_0^1 F(t) \exp\left(\int_t^1 F(s)\mathrm{d}s\right)\mathrm{d}t\right), \quad (21)$$

where $F(t) \triangleq C^2\sigma_t^4 \|s_\phi(\cdot,t)\|_{\mathrm{lip}}^2 + C\|\mathbf{f}(\cdot,t)\|_{\mathrm{lip}}^2$ and C is a constant.

Proof of Proposition 2: Assumptions of Proposition 2 imply

$$\|\mathbf{f}(\cdot,t)\|_{\mathrm{lip}} = \frac{1}{2}\sigma_t^2,$$
$$\|s_\phi(\cdot,t)\|_{\mathrm{lip}} \leqslant L \cdot \mathbb{E}_{|\mathbf{A}_0}\|\nabla \log p_{t|0}(\widehat{\mathbf{A}}_t^{\mathrm{full}})\|_{\mathrm{lip}},$$
$$\|s_\varphi(\cdot,t)\|_{\mathrm{lip}} \leqslant L \cdot \mathbb{E}_{|\mathbf{A}_0}\|\nabla \log p_{t|0}(\widehat{\mathbf{A}}_t^{\mathrm{spec}})\|_{\mathrm{lip}},$$

where L is a constant. According to [11], [13], at each time step $t \in [0,1]$, the oracle conditional score function $\nabla \log p_{t|0}(\cdot)$ is equivalent to a mapping that maps the input d-dimensional random variable $\mathbf{Z}_t$ to $\Sigma_t^{-1}\boldsymbol{\varepsilon}$. The shown up $\boldsymbol{\varepsilon}$ is a d-dimensional standard Gaussian vector that is independent to $\mathbf{Z}_t$, and Σ_t is the standard deviation of $\mathbf{Z}_t$, which is given by $\Sigma_t \triangleq (1 - e^{-\int_0^t \sigma_s^2 \mathrm{d}s})^{\frac{1}{2}}$. Thus, the expected Lipschitz norm of oracle conditional score function is bounded by

$$\left(\mathbb{E}_{|\mathbf{A}_0}\|\nabla \log p_{t|0}(\mathbf{Z}_t)\|_{\mathrm{lip}}\right)^2 \leqslant \mathbb{E}_{|\mathbf{A}_0}\|\nabla \log p_{t|0}(\mathbf{Z}_t)\|_{\mathrm{lip}}^2$$
$$\leqslant \frac{\Sigma_t^{-2}}{\|\mathbf{Z}_0\|^2}\mathbb{E}\|\boldsymbol{\varepsilon}\|^2 = \frac{\Sigma_t^{-2}d}{\|\mathbf{Z}_0\|^2}. \quad (22)$$

With Lemma V.1 in hand, we only need to plug (22) into (21), by substituting the notations $(\mathbf{Z}, d, \|\cdot\|)$ with $(\widehat{\mathbf{A}}_t^{\mathrm{full}}, n^2, \|\cdot\|_{2,2})$ and $(\widehat{\mathbf{\Lambda}}_t^{\mathrm{spec}}, n, \|\cdot\|_{2,2})$ to bound the Lipschitz norm of the score networks. This leads us to

$$\mathbb{E}\|\mathbf{A}_0 - \widehat{\mathbf{A}}_0^{\mathrm{full}}\|^2$$
$$\leqslant M\mathcal{E}(\phi) \cdot \left(1 + n^2K \cdot \int_0^1 \Sigma_t^{-2}\exp\left(n^2K\int_t^1 \Sigma_s^{-2}\mathrm{d}s\right)\mathrm{d}t\right),$$
$$\mathbb{E}\|\mathbf{\Lambda}_0 - \widehat{\mathbf{\Lambda}}_0^{\mathrm{spec}}\|^2$$
$$\leqslant M\mathcal{E}(\phi) \cdot \left(1 + nK \cdot \int_0^1 \Sigma_t^{-2}\exp\left(nK\int_t^1 \Sigma_s^{-2}\mathrm{d}s\right)\mathrm{d}t\right),$$

where $M \triangleq C^2\|\sigma.\|_\infty^4$ and $K \triangleq 2ML/\mathbb{E}\|\mathbf{A}_0\|_{2,2}$. For the spectral diffusion part, the final proof step is completed by the fact that

$$\mathbb{E}\|\mathbf{A}_0 - \widehat{\mathbf{A}}_0^{\mathrm{spec}}\|^2 = \mathbb{E}\|\mathbf{U}_0\left(\mathbf{\Lambda}_0 - \widehat{\mathbf{\Lambda}}_0^{\mathrm{spec}}\right)\mathbf{U}_0^\top\|^2 = \mathbb{E}\|\mathbf{\Lambda}_0 - \widehat{\mathbf{\Lambda}}_0^{\mathrm{spec}}\|^2,$$

$$\mathbb{E}\|\mathbf{A}_0\|_{2,2} = \mathbb{E}\|\mathbf{U}_0\mathbf{\Lambda}_0\mathbf{U}_0^\top\|_{2,2} = \mathbb{E}\|\mathbf{\Lambda}_0\|_{2,2}.$$

This completes the proof. $\square$

Proof of Lemma V.1:

To bound the expected reconstruction error $\mathbb{E}\|\mathbf{Z}_0 - \widehat{\mathbf{Z}}_0\|^2$, we first analysis how $\mathbb{E}\|\bar{\mathbf{Z}}_t - \widehat{\mathbf{Z}}_t\|^2$ evolves as time t is reversed from 1 to 0. By definition, it holds that

$$\bar{\mathbf{Z}}_t - \widehat{\mathbf{Z}}_t = \int_1^t \left(\mathbf{f}(\bar{\mathbf{Z}}_s, s) - \mathbf{f}(\widehat{\mathbf{Z}}_s, s)\right)$$
$$+ \sigma_s^2\left(s_\phi(\widehat{\mathbf{Z}}_s, s) - \nabla_{\mathbf{Z}} \log p_s(\bar{\mathbf{Z}}_s)\right)\mathrm{d}\bar{s}.$$

By taking norm and expectation on both sides, we have

$$\mathbb{E}\|\bar{\mathbf{Z}}_t - \hat{\mathbf{Z}}_t\|^2$$

$$\leqslant \mathbb{E} \int_1^t \left\| \left(\mathbf{f}(\bar{\mathbf{Z}}_s, s) - \mathbf{f}(\hat{\mathbf{Z}}_s, s) \right) \right.$$
$$\left. + \sigma_s^2 \left(s_\phi(\hat{\mathbf{Z}}_s, s) - \nabla_{\mathbf{Z}} \log p_s(\bar{\mathbf{Z}}_s) \right) \right\|^2 \mathrm{d}\bar{s}$$

$$\leqslant C\mathbb{E} \int_1^t \left\| \mathbf{f}(\bar{\mathbf{Z}}_s, s) - \mathbf{f}(\hat{\mathbf{Z}}_s, s) \right\|^2 \mathrm{d}\bar{s}$$
$$+ C\mathbb{E} \int_1^t \sigma_s^4 \left\| s_\phi(\hat{\mathbf{Z}}_s, s) - \nabla_{\mathbf{Z}} \log p_s(\bar{\mathbf{Z}}_s) \right\|^2 \mathrm{d}\bar{s}$$

$$\leqslant C^2 \int_1^t \sigma_s^4 \cdot \mathbb{E} \left\| s_\phi(\hat{\mathbf{Z}}_s, s) - s_\phi(\bar{\mathbf{Z}}_s, s) \right\|^2$$
$$+ \sigma_s^4 \cdot \mathbb{E} \left\| s_\phi(\bar{\mathbf{Z}}_s, s) - \nabla_{\mathbf{Z}} \log p_s(\bar{\mathbf{Z}}_s) \right\|^2 \mathrm{d}\bar{s}$$
$$+ C \int_1^t \|\mathbf{f}(\cdot, s)\|_{\mathrm{lip}}^2 \cdot \mathbb{E} \left\| \bar{\mathbf{Z}}_s - \hat{\mathbf{Z}}_s \right\|^2 \mathrm{d}\bar{s}$$

$$\leqslant \int_1^t \underbrace{\left(C^2 \sigma_s^4 \| s_\phi(\cdot, s) \|_{\mathrm{lip}}^2 + C \| \mathbf{f}(\cdot, s) \|_{\mathrm{lip}}^2 \right)}_{F(s)} \cdot \mathbb{E} \left\| \hat{\mathbf{Z}}_s - \bar{\mathbf{Z}}_s \right\|^2 \mathrm{d}\bar{s}$$
$$+ \underbrace{C^2 \int_1^t \sigma_s^4 \mathrm{d}\bar{s} \cdot \mathcal{E}(\phi)}_{G(t)} .$$

The proof is completed by applying the Grönwall's inequality to $t \mapsto \mathbb{E}\|\bar{\mathbf{Z}}_t - \hat{\mathbf{Z}}_t\|^2$, which yields

$$\mathbb{E}\|\bar{\mathbf{Z}}_0 - \hat{\mathbf{Z}}_0\|^2$$

$$\leqslant G(0) + \int_0^1 G(t)F(t) \exp\left(\int_t^1 F(s)\mathrm{d}s \right) \mathrm{d}t$$

$$\leqslant C^2 \|\sigma.\|_\infty^4 \mathcal{E}(\phi) \cdot \left(1 + \int_0^1 F(t) \exp\left(\int_t^1 F(s)\mathrm{d}s \right) \mathrm{d}t \right).$$

C. Proof of Proposition 3

Proof: Since $\mathcal{E}(\boldsymbol{\theta}) = \mathbb{E}_{\mathcal{S}\sim\mathcal{D}^N} \hat{\mathcal{E}}(\boldsymbol{\theta}; \mathcal{S}_N)$, we only need to bound the empirical risk $\hat{\mathcal{E}}(\boldsymbol{\theta}; \mathcal{S}_N)$. By assumption, denote

$$\mathbf{h} \triangleq \begin{pmatrix} s_{\boldsymbol{\theta}}(\mathbf{Z}_0^1) - \nabla \log p(\mathbf{Z}_0^1 | \mathbf{Z}_1^1)^\top \\ \vdots \\ s_{\boldsymbol{\theta}}(\mathbf{Z}_0^N) - \nabla \log p(\mathbf{Z}_0^N | \mathbf{Z}_1^N)^\top \end{pmatrix} \in \mathbb{R}^{Nd\times 1},$$

It holds that

$$\|\nabla_{\boldsymbol{\theta}} \hat{\mathcal{E}}(\boldsymbol{\theta}; \mathcal{S}_N)\|^2 = \frac{1}{N^2} \mathbf{h}^\top \mathbf{K}_{\boldsymbol{\theta}}(\mathcal{S}) \mathbf{h} \geqslant \frac{\lambda}{N^2} \|\mathbf{h}\|^2 = \frac{\lambda}{N} \hat{\mathcal{E}}(\boldsymbol{\theta}; \mathcal{S}_N),$$

which implies that $\hat{\mathcal{E}}(\boldsymbol{\theta}; \mathcal{S}_N)$ satisfies the $\frac{\lambda}{N}$-Polyak Łojasiewicz condition [24]. Then the proof is completed by applying Theorem 7 in [24] to $\hat{\mathcal{E}}(\cdot; \mathcal{S}_N)$. □

VI. EXPERIMENTS

In this section, we evaluate our proposed GSDM with state-of-the-art graph generative baselines on two types of graph generation tasks: generic graph generation and molecule generation, over several benchmark datasets. We also conduct extensive ablation studies and visualizations to further illustrate the effectiveness and efficiency of GSDM.

A. Generic Graph Generation

Datasets: We test our model on three generic datasets with different scales, and we use N to denote the number of nodes: (1) Community-small ($12 \leq N \leq 20$): contains 100 small community graphs. (2) Enzymes ($10 \leq N \leq 125$): contains 578 protein graphs which represent the protein tertiary structures of the enzymes from the BRENDA database. (3) Grid ($100 \leq N \leq 400$): contains 100 standard 2D grid graphs. To fairly compare our model with baselines, we adopt the experimental and evaluation setting from [9] with the same train/test split.

Baselines: We compare our model with well-known or state-of-the-art graph generation methods, which can be categorized into auto-regressive models and one-shot models. The **auto-regressive model** refers to sequential generation, which constructs a graph via a set of consecutive steps, usually done nodes by nodes and edges by edges [14]. Under this category, we include DeepGMG [25], GraphRNN [9], GraphAF [4], and GraphDF [26]. The **one-shot model** refers to building a probabilistic graph model that can generate all nodes and edges in one shot [14]. Under this category, we include GraphVAE [8], Graph Normalizing Flow(GNF) [27], EDP-GNN [12], and GDSS [10]. In addition to graph generation methods, we incorporate two SOTA diffusion-based generation methods from the field of computer vision into the task of graph generation: SubspaceD-iff [17] and WSGM [18]. Implementation details of these two methods are elaborated in the Appendix, available online.

Metrics: We adopt the maximum mean discrepancy (MMD) to compare the distributions of graph statistics, such as the degree, the clustering coefficient, and the number of occurrences of orbits with 4 nodes [9], [10] between the same number of generated and test graphs.

Results: We show the results in Table I. The results show that our proposed GSDM significantly outperforms both the auto-regressive baselines and one-shot baseline methods. Specifically, GDSS is the SOTA graph diffusion model which performs the diffusion in the whole space of the graph data. Our method substantially outperforms GDSS in terms of both average performance and convergence rate (Fig. 3), which demonstrates the advantages of performing SDEs in the spectrum of the graph compared to the whole space.

B. Molecules Generation

Besides generic graph generation, our model can also generate organic molecules through our proposed reverse diffusion process. We test our model with two well-known molecule datasets: QM9 [29] and ZINC250K [30]. Following previous works [10], [26], the molecules are kekulized by the RDKit library [31] with

TABLE I
GENERATION RESULTS ON THE GENERIC GRAPH DATASETS

	Community-small				Enzymes				Grid			
	Synthetic, $12 \leq |V| \leq 20$				Real, $10 \leq |V| \leq 125$				Synthetic, $100 \leq |V| \leq 400$			
	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Deg.↓	Clus.↓	Orbit↓	Avg.↓
DeepGMG [25]	0.220	0.950	0.400	0.523	-	-	-	-	-	-	-	-
GraphRNN [9]	0.080	0.120	0.040	0.080	0.017	**0.043**	0.021	0.043				
GraphAF [4]	0.18	0.200	0.020	0.133	1.669	1.283	0.266	1.073	-	-	-	-
GraphDF [26]	0.060	0.120	0.030	0.070	1.503	1.061	0.202	0.922	-	-	-	-
GraphVAE [8]	0.350	0.980	0.540	0.623	1.369	0.629	0.191	0.730	1.619	**0.0**	0.919	0.846
GNF [27]	0.200	0.200	0.110	0.170	-	-	-	-	-	-	-	-
EDP-GNN [12]	0.053	0.144	0.026	0.074	0.023	0.268	0.082	0.124	0.455	0.238	0.328	0.340
SubspaceDiff [17]	0.057	0.098	0.012	0.056	0.037	0.099	0.018	0.051	0.124	0.013	0.090	0.076
WSGM [18]	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[1] [10]	0.045	0.086	0.007	0.046	0.026	0.102	0.009	0.046	0.111	0.005	0.070	0.062
Ours	**0.011**	**0.015**	**0.001**	**0.009**	**0.013**	0.088	**0.009**	**0.037**	**0.002**	**0.0**	**0.0**	**0.0007**

[1] The average results of the Enzymes dataset reported in the GDSS original paper is 0.032. However, the best result we can obtain using the author's released code and checkpoint with careful fine-tuning is 0.046.
We report the MMD distances between the test datasets and generated graphs. The best results are highlighted in bold (the smaller the better). Hyphen (-) denotes out-of-resources that take more than 10 days or are not applicable due to memory issues.
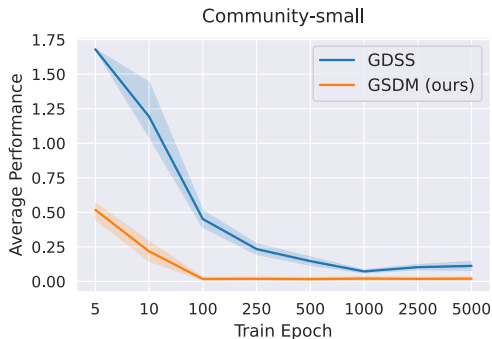

Fig. 3. GSDM enjoys a significantly faster convergence rate than GDSS. On the community-small dataset, our GSDM reaches the SOTA performance within 100 training epochs.

hydrogen atoms removed. We evaluate the quality of 10,000 generated molecules with **Frechet ChemNet Distance (FCD)** [32], **Neighborhood subgraph pairwise distance kernel (NSPDK) MMD** [33], **validity w/o correction**, and the **generation time**. FCD computes the distance between the testing and the generated molecules using the activations of the penultimate layer of the ChemNet. (NSPDK) MMD computes the MMD between the generated and the testing set which takes into account both the node and edge features for evaluation. Generally speaking, FCD measures the generation quality in the view of molecules in the chemical space, while NSPDK MMD evaluates the generation quality from the graph structure perspective. Besides, following [10], we also include the **validity w/o correction** as another metric to explicitly evaluate the quality of molecule generation prior to the correction procedure. It computes the fraction of the number of valid molecules without valency correction or edge resampling over the total number of generated molecules. In contrast, **validity** measures the fraction of the valid molecules after the correction phase. **Generation time** measures the time for generating 10,000 molecules in the form of RDKit. It is a salient measure of the practicability of molecule generation, especially for macromolecule generation.

Baselines: We compare our model with the state-of-the-art molecule generation models. The baselines include SOTA auto-regressive models: GraphAF [4] is a flow-based model, and GraphDF [26] is a flow-based model using discrete latent variables. Following GDSS [10], we modify the architecture of GraphAF and GraphDF to consider formal charges in the molecule generation, denoted as GraphAF+FC and GraphDF+FC, for fair comparisons. For the one-shot model, we include MoFlow [3], which is a flow-based model; EDP-GNN [12] and GDSS [12] which are both diffusion models.

Results: We show the results in Table II. Evidently, GSDM achieves the highest performance under most of the metrics. The highest scores in NSPDK and FCD show that GSDM is able to generate molecules that have close data distributions to the real molecules in both the chemical space and graph space. Especially, our model outperforms GDSS, in most of the metrics, verifying that our proposed spectral diffusion is not only suitable for generic graph generation but also advisable for molecule designs.

Visualization of molecule generations: Apart from the visualization of generated graphs in Fig. 2, visualizations of generated molecules are showed in Fig. 4. In the figure, we show the generated molecules that are maximally similar to certain training molecules. We compute the molecule similarity using the Tanimoto similarity based on the Morgan fingerprints, which are implemented based on REKit [31]. Higher Tanimoto scores indicate that the model is able to generate more similar molecules as the training set, which reflects the learning capability of the model. As shown in the figure, compared to the GDSS model, GSDM is able to generate molecules that have a more similar distribution as the molecules in the training set.

Time Complexity: One of the main advantages of our GSDM compared to other diffusion models (e.g. EDP-GNN and GDSS) is the efficiency in generating molecules. We show the time spent (in seconds) for generating 10,000 molecules in Table II, demonstrating that our GSDM takes a significantly lower time in the inference process compared to EDP-GNN and GDSS. Especially, compared to GDSS which requires $1.06e^3$ and $2.11e^3$

TABLE II
GENERATION RESULTS ON THE QM9 AND ZINC250 K DATASETS

Method	QM9					ZINC250k				
	Validity (%)↑	Val. w/o corr. (%)↑	NSPDK↓	FCD↓	Time (s)↓	Validity (%)↑	Val. w/o corr. (%)↑	NSPDK↓	FCD↓	Time (s)↓
GraphAF [4]	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 [26]	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 [3]	100	91.36	0.017	4.467	**4.58**	100	63.11	0.046	20.931	**25.9**
EDP-GNN [12]	100	<u>47.52</u>	0.005	2.680	$4.13e^3$	100	82.97	0.049	16.737	$8.41e^3$
GraphEBM [28]	100	<u>8.22</u>	0.030	6.143	35.33	100	<u>5.29</u>	0.212	35.471	53.72
GDSS [10]	100	95.72*	**0.003***	2.900*	$1.06e^2$	100	97.01*	0.019*	14.656*	$2.11e^3$
Ours	100	**99.9**	**0.003**	**2.650**	18.02	100	92.70	**0.017**	**12.956**	45.91

Results are the means of three different runs, and the best results are highlighted in bold. Values denoted by * are taken from the respective original papers. Other results are obtained by running open-source codes. Val. w/o corr. Denotes the validity w/o correction metric, and values that do not exceed 50% are underlined.

TABLE III
ABLATION STUDY ON THE α-QUANTILE EIGENVALUES

	Community-small					Enzymes					Grid				
	Synthetic, $12 \leq |V| \leq 20$					Real, $10 \leq |V| \leq 125$					Synthetic, $100 \leq |V| \leq 400$				
α	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Avg.%↑	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Avg.%↑	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Avg.%↑
10%	1.010	1.105	0.227	0.781	0.012	0.446	0.608	0.059	0.371	0.029	1.996	0	1.013	1.003	0.001
20%	0.450	1.102	0.102	0.551	0.016	0.056	0.220	0.012	0.096	0.395	1.996	0	1.013	1.003	0.001
30%	0.085	0.421	0.030	0.179	0.050	**0.011**	**0.093**	0.012	**0.039**	0.982	**0.107**	0	**0.044**	**0.050**	0.012
40%	**0.039**	**0.056**	**0.006**	**0.034**	0.268	**0.010**	**0.093**	**0.011**	**0.038**	1.000	**0.018**	0	**0.001**	**0.006**	0.095
50%	**0.022**	**0.024**	**0.002**	**0.016**	0.563	**0.010**	**0.093**	**0.011**	**0.038**	1.000	**0.007**	0	**0.001**	**0.002**	0.225
60%	**0.014**	**0.019**	**0.001**	**0.011**	0.794	**0.010**	**0.093**	**0.011**	**0.038**	1.000	**0.002**	0	0	**0.001**	1.000
70%	**0.012**	**0.019**	**0.001**	**0.011**	0.844	**0.010**	**0.093**	**0.011**	**0.038**	1.000	**0.002**	0	0	**0.001**	1.000
80%	**0.011**	**0.016**	**0.001**	**0.009**	0.964	**0.010**	**0.093**	**0.011**	**0.038**	1.000	**0.002**	0	0	**0.001**	1.000
90%	**0.011**	**0.015**	**0.001**	**0.009**	1.000	**0.010**	**0.093**	**0.011**	**0.038**	1.000	**0.002**	0	0	**0.001**	1.000
100%	**0.011**	**0.015**	**0.001**	**0.009**	1.000	**0.010**	**0.093**	**0.011**	**0.038**	1.000	**0.002**	0	0	**0.001**	1.000

The metrics used are the same as in Table I. The results that surpass the baseline methods in Table I are highlighted in bold (the smaller the better). Avg.% denotes the percentage of average scores achieved compared to using the whole eigenvalues and eigenvectors set.
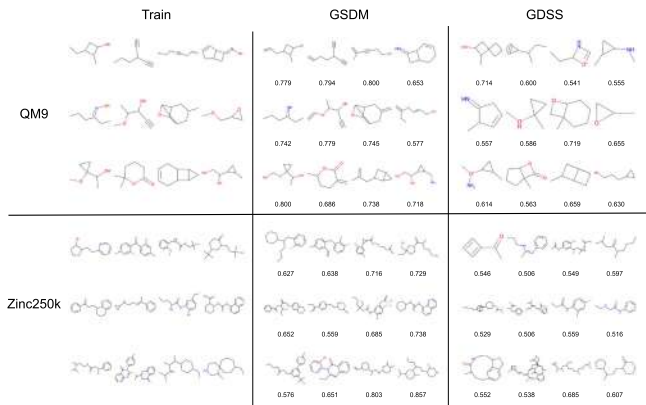

Fig. 4. Visualization of molecule generation with maximum Tanimoto similarity of GSDM comparing to GDSS. The left part shows randomly selected molecules from the training set of QM9 and Zinc250 k. For each generated molecules, we show the Tanimoto similarity value at the bottom.

Fig. 5. Ablation studies on different diffusion step numbers.

C. Ablation Studies

Ablation Studies on the Number of Diffusion Steps: To analyze the robustness of our proposed GSDM during the reconstruction procedure (i.e. reverse diffusion process), we compare the performance of our model with GDSS on varying numbers of diffusion steps using the Community small and Grid dataset, which represents small and large generic graphs respectively. The default steps for our model and GDSS during training and evaluation are 1,000 steps. Ideally, for diffusion models, with the reduction in step numbers, the performance of the models decreases, while the model of higher robustness still exhibits a higher performance under different numbers of diffusion steps. The results are shown in Fig. 5. We can observe that GSDM outperforms GDSS in both datasets under different diffusion steps. Specifically, in the Grid dataset, GSDM consistently exhibits significantly higher performance than GDSS. In the community small dataset, although GSDM and GDSS achieve

seconds to generate 10,000 molecule graphs according to QM9 and Zinc250 k, our model only takes 18.02 and 45.91 seconds, which are $\mathbf{58\times}$ and $\mathbf{46\times}$ faster respectively. This phenomenon verifies our methodology in designing the spectral diffusion that not only the spectral diffusion is more effective than the whole space diffusion, but also more efficient. Such an improvement is crucial for numerous applications such as drug design and material analysis.
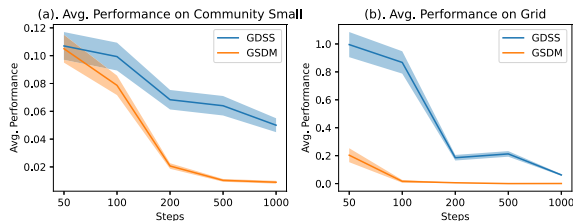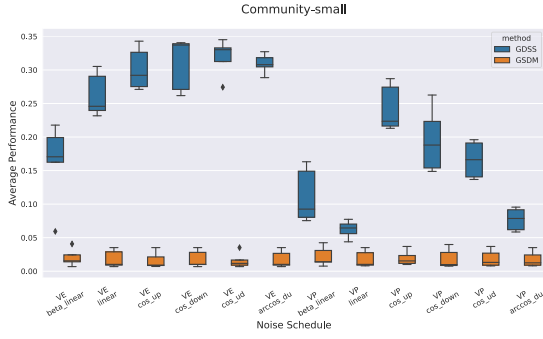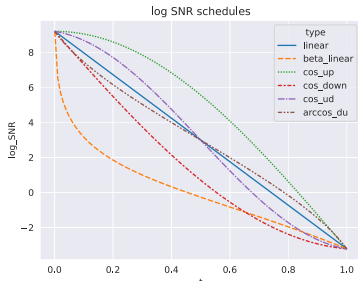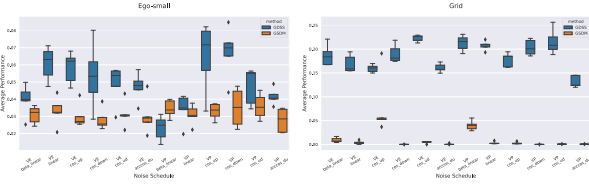
(a) Ablation studies on diffusion schedules. The y-axis denotes the average performance score (↓) and each column denotes a diffusion schedule configuration. Each box plot summarizes the results of 5 random trials.

(b) Noise schedules.

(c) Ablation study on the choice of noise schedules.

Fig. 6. Ablation studies on diffusion and noise schedules.
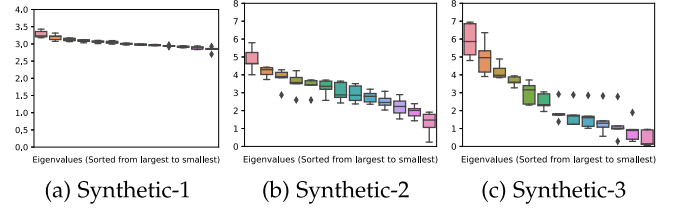

(a) Synthetic-1 (b) Synthetic-2 (c) Synthetic-3

Fig. 7. Eigenvalue distribution of three synthetic datasets.

We further empirically verify the robustness of our proposed GSDM under various types of noise schedules. As shown in Fig. 6(b), we design six representative noise schedules with the same initial and final signal-to-noise ratio (defined in [13], [34]), while exhibiting different behavior along the diffusion process. For both VP- and VE-SDE configurations, we train our proposed GSDM with six representative noise schedules and we evaluate the average performance scores. Results on Ego-small and Grid in Fig. 6(c) show that GSDM is robust to the choice of diffusion schedules, and it is able to achieve SOTA performance without deliberate noise schedule optimization.

Ablation Studies on Low-Rank Approximations of the Score-Function: As mentioned in Section IV-B, one can further accelerate the training and sampling phases, by confining diffusion to the partial spectrum of the graph adjacency matrix. Such variants are coined as α-quantile GSDM in Section IV-B. In this experiment, we select the top $[\alpha n]$ eigenvalues to conduct the sampling process with a well-trained GSDM, where α is chosen from $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$. The results are shown in Table III.

Numerical experiments prove that the low-rank α-quantile GSDM retains the state-of-the-art performance among most of the baseline methods. Especially, for the Enzymes dataset, by merely preserving the top-30% eigenvalues and eigenvectors, the 0.3-quantile GSDM can still retain 98.2% of the performance of the full GSDM. The ablation study justifies that our proposed GSDM is capable of capturing essential knowledge that resides on low-dimensional manifolds. It further demonstrates that α-quantile GSDM enables outstanding scalability for large-scale datasets.

Ablation Studies on Graphs With Different Eigenvalues Distributions: To analyze how GSDM performs under graphs with different eigenvalue distributions, we randomly generate 3 types of synthetic graphs with eigenvalues distributions shown in Fig. 7: synthetic-1 has evenly distributed eigenvalues, and synthetic-2 and synthetic-2 have moderate and high distinctions in eigenvalues. Due to the predefined eigenvalue distributions, the edge values of the synthetic graphs are not limited to $\{0, 1\}$. Thus instead of using the metrics presented in Section V-A, we directly compute the MMD of the adjacency matrices between the generated graphs and the test graphs. Corresponding results compared to GDSS are shown in Fig. 8. We also include an ablation variant of GSDM by using only top-50% quantile eigenvalues. Both GSDM and its ablation variant outperform the GDSS model under this synthetic setting.
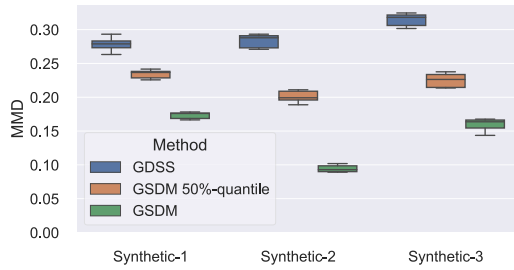
similar performance at 50 steps, GSDM becomes much more accurate when the number of steps is higher than 100.

Ablation Studies on the Type of Diffusion Schedules: As mentioned in [13], [34] and [35], the performance of diffusion models highly relies on the diffusion schedule, i.e. the scheduled noise insertion process or discretization of certain types of SDEs, which lies at the heart of both training and inference phases. According to the taxonomy proposed in [13] and [11], mainstream artificial diffusion schedules are categorized to be either Variance Exploding (VE) or Variance Preserving (VP). To improve model robustness against the choice of diffusion schedules, [35] and [34] designed diffusion models with optimizable diffusion schedules. In this subsection, we conduct systematic experiments on several datasets to test the robustness of our proposed GSDM against different diffusion schedules. As illustrated in Fig. 6(a), the state-of-the-art performance of our GSDM is immune to the choice of diffusion schedule and it frees us from tedious hyperparameter searching. When compared with GDSS, GSDM exhibits evidently better average performance with much smaller variance across various configurations.

Fig. 8. Experiments on synthetic datasets of which the eigenvalues' distributions are shown in Fig. 7. The y-axis denotes the MMD ($\downarrow$) of the adjacency matrices between the generated graphs to the test graphs .

VII. CONCLUSION AND FUTURE WORK

In this paper, we present a novel graph diffusion model named GSDM, which consists of diffusion methods on both graph node features and topologies through SDEs. Specifically, for graph topology diffusion, we design a novel spectral diffusion model in GSDM, to improve the accuracy of predicting score functions, and avoid the pitfalls of the conventional diffusion processes on graphs. Furthermore, we provide theoretical analysis to justify the advantages in effectiveness and efficiency of GSDM compared to the standard graph diffusion. To validate our proposed model, we conducted extensive experiments showing that our proposed GSDM outperforms the state-of-the-art baseline methods on both generic graph generation and molecule generation with significantly higher processing speed.

Note that in GSDM, we propose to perform a diffusion process on the eigenvalues of the graph adjacency matrix. We also present some ideas about performing diffusion on the eigenvectors of the graph adjacency matrix and discuss the potential issues of the eigenvector-diffusion paradigms in Appendix C, available online, and provide some preliminary empirical studies in Appendix E, available online. In the future, we will explore the possibility of developing an efficient algorithm to perform diffusion on eigenvectors for graph generation.

REFERENCES

[1] Y. Yang, Z. Feng, M. Song, and X. Wang, "Factorizable graph convolutional networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 20286–20296.

[2] H. Wang et al., "MCNE: An end-to-end framework for learning multiple conditional network representations of social network," in *Proc. Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 1064–1072.

[3] C. Zang and F. Wang, "Moflow: An invertible flow model for generating molecular graphs," in *Proc. Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 617–626.

[4] C. Shi, M. Xu, Z. Zhu, W. Zhang, M. Zhang, and J. Tang, "GraphAF: A flow-based autoregressive model for molecular graph generation," 2020, *arXiv: 2001.09382*.

[5] H. Lee, E. Hyung, and S. J. Hwang, "Rapid neural architecture search by learning to generate graphs from datasets," 2021, *arXiv:2107.00860*.

[6] H. Liu et al., "Interpretable deep generative recommendation models," *J. Mach. Learn. Res.*, vol. 22, pp. 202–1, 2021.

[7] H. Wang et al., "Learning graph representation with generative adversarial nets," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 8, pp. 3090–3103, Aug. 2019.

[8] M. Simonovsky and N. Komodakis, "GraphVAE: Towards generation of small graphs using variational autoencoders," in *Proc. Int. Conf. Artif. Neural Netw.*, 2018, pp. 412–422.

[9] J. You, R. Ying, X. Ren, W. Hamilton, and J. Leskovec, "GraphRNN: Generating realistic graphs with deep auto-regressive models," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5708–5717.

[10] J. Jo, S. Lee, and S. J. Hwang, "Score-based generative modeling of graphs via the system of stochastic differential equations," 2022, *arXiv:2202.02514*.

[11] Y. Song and S. Ermon, *Generative Modeling by Estimating Gradients of the Data Distribution*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[12] C. Niu, Y. Song, J. Song, S. Zhao, A. Grover, and S. Ermon, "Permutation invariant graph generation via score-based generative modeling," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 4474–4484.

[13] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," 2020, *arXiv: 2011.13456*.

[14] X. Guo and L. Zhao, "A systematic survey on deep generative models for graph generation," 2020, *arXiv: 2007.06686*.

[15] T. Ma, J. Chen, and C. Xiao, "Constrained generation of semantically valid graphs via regularizing variational autoencoders," in *Proc. Annl. Conf. Neural Inf. Process. Syst.*, 2018, pp. 7113–7124.

[16] N. De Cao and T. Kipf, "MolGAN: An implicit generative model for small molecular graphs," 2018, *arXiv: 1805.11973*.

[17] B. Jing, G. Corso, R. Berlinghieri, and T. Jaakkola, "Subspace diffusion generative models," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 274–289.

[18] F. Guth, S. Coste, V. D. Bortoli, and S. Mallat, "Wavelet score-based generative modeling," 2022, *arXiv:2208.05003*.

[19] B. Jing, G. Corso, J. Chang, R. Barzilay, and T. S. Jaakkola, "Torsional diffusion for molecular conformer generation," in *Proc. Annl. Conf. Neural Inf. Process. Syst.*, 2022.

[20] G. Corso, H. Stärk, B. Jing, R. Barzilay, and T. S. Jaakkola, "Diffdock: Diffusion steps, twists, and turns for molecular docking," in *Proc. Int. Conf. Learn. Representations*, 2023.

[21] K. E. Wu, K. K. Yang, R. van den Berg, J. Zou, A. X. Lu, and A. P. Amini, "Protein structure generation via folding diffusion," 2022, *arXiv:2209.15611*.

[22] B. D. Anderson, "Reverse-time diffusion equation models," *Stochastic Processes Appl.*, vol. 12, no. 3, pp. 313–326, 1982.

[23] P. Vincent, "A connection between score matching and denoising autoencoders," *Neural Comput.*, vol. 23, no. 7, pp. 1661–1674, 2011.

[24] C. Liu, L. Zhu, and M. Belkin, "Toward a theory of optimization for over-parameterized systems of non-linear equations: The lessons of deep learning," 2020, *arXiv:abs/ 2003.00307*.

[25] Y. Li, O. Vinyals, C. Dyer, R. Pascanu, and P. Battaglia, "Learning deep generative models of graphs," 2018, *arXiv: 1803.03324*.

[26] Y. Luo, K. Yan, and S. Ji, "GraphDF: A discrete flow model for molecular graph generation," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 7192–7203.

[27] J. Liu, A. Kumar, J. Ba, J. Kiros, and K. Swersky, "Graph normalizing flows," in *Proc. Annl. Conf. Neural Inf. Process. Syst.*, 2019.

[28] M. Liu, K. Yan, B. Oztekin, and S. Ji, "GraphEBM: Molecular graph generation with energy-based models," 2021, *arXiv:2102.00546*.

[29] R. Ramakrishnan, P. O. Dral, M. Rupp, and O. A. Von Lilienfeld, "Quantum chemistry structures and properties of 134 kilo molecules," *Sci. Data*, vol. 1, no. 1, pp. 1–7, 2014.

[30] J. J. Irwin, T. Sterling, M. M. Mysinger, E. S. Bolstad, and R. G. Coleman, "ZINC: A free tool to discover chemistry for biology," *J. Chem. Inf. Model.*, vol. 52, no. 7, pp. 1757–1768, 2012.

[31] G. Landrum et al., "RDKit: Open-source cheminformatics software," 2016. [Online]. Available: http://www.rdkit.org/

[32] K. Preuer, P. Renz, T. Unterthiner, S. Hochreiter, and G. Klambauer, "Fréchet chemnet distance: A metric for generative models for molecules in drug discovery," *J. Chem. Inf. Model.*, vol. 58, no. 9, pp. 1736–1741, 2018.

[33] F. Costa and K. De Grave, "Fast neighborhood subgraph pairwise distance kernel," in *Proc. Int. Conf. Mach. Learn.*, 2010, pp. 255–262.

[34] A. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," 2021, *arXiv:abs/2102.09672*.

[35] D. P. Kingma, T. Salimans, B. Poole, and J. Ho, "Variational diffusion models," 2021, *arXiv:abs/2107.00630*.

[36] H. Yuan, X. Gu, R. Lai, and Z. Wen, "Global optimization with orthogonality constraints via stochastic diffusion on manifold," *J. Sci. Comput.*, vol. 80, no. 2, pp. 1139–1170, 2019.

[37] V. D. Bortoli, E. Mathieu, M. J. Hutchinson, J. Thornton, Y. W. Teh, and A. Doucet, "Riemannian score-based generative modelling," in *Proc. Annl. Conf. Neural Inf. Process. Syst.*, 2022, pp. 2406–2422.

[38] T. Kynkäänniemi, T. Karras, S. Laine, J. Lehtinen, and T. Aila, "Improved precision and recall metric for assessing generative models," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 3927–3936.

[39] J. Ho and T. Salimans, "Classifier-free diffusion guidance," in *Proc. Annl. Conf. Neural Inf. Process. Syst.*, 2021.

[40] C. Morris, N. M. Kriege, F. Bause, K. Kersting, P. Mutzel, and M. Neumann, "TUDataset: A collection of benchmark datasets for learning with graphs," in *Proc. Annl. Conf. Neural Inf. Process. Syst.*, 2020.

Zhanfeng Mo received the bachelor's degree in statistics from the University of Science and Technology of China in 2020. He is currently working towards the PhD degree with the School of Computer Science and Engineering of Nanyang Technological University, Singapore. His main research interests include machine learning theory and algorithms.

Tianze Luo received the bachelor's (First Class Honours) and master's degrees from Nanyang Technological University, Singapore, in 2017 and 2019, respectively. He is currently working towards an Alibaba Talent Programme PhD degree with Alibaba-NTU Joint Research Institute, Nanyang Technological University.

Sinno Jialin Pan (Senior Member, IEEE) received the PhD degree in computer science from the Hong Kong University of Science and Technology (HKUST) in 2011. He is a professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong (CUHK), Hong Kong. Prior to joining NTU, he was a Provost's chair professor of computer science with NTU Singapore. He serves as an associate editor for *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *AIJ*, and *ACM Transactions on Intelligent Systems and Technology*.