Graph Principal Flow Network for Conditional Graph Generation

Zhanfeng Mo*
Nanyang Technological University
Singapore
zhanfeng001@ntu.edu.sg

Tianze Luo*
Nanyang Technological University
Singapore
tianze002@ntu.edu.sg

Sinno Jialin Pan†
The Chinese University of Hong Kong
Hong Kong, China
sinnopan@cuhk.edu.hk

ABSTRACT

Conditional graph generation is crucial and challenging since the conditional distribution of graph topology and feature is complicated and the semantic information is hard to capture by the generative model. In this work, we propose a novel graph conditional generative model, Graph Principal Flow Network (GPrinFlowNet), which enables us to progressively generate high-quality graphs from low- to high-frequency components for a given graph label. We show that GPrinFlowNet follows a coarse-to-fine resolution generation curriculum, which enables it to capture subtle semantic information by generating intermediate graphs with high mutual information relative to the graph label. Extensive experiments and ablation studies showcase that our model achieves state-of-the-art performance compared to existing conditional graph generation models. The code is available on Github[1].

CCS CONCEPTS

• **Computing methodologies → Artificial intelligence**; *Graph generative models.*

KEYWORDS

Graph Generation; Deep Learning; Generative Flow Network.

ACM Reference Format:
Zhanfeng Mo, Tianze Luo, and Sinno Jialin Pan. 2024. Graph Principal Flow Network for Conditional Graph Generation. In *Proceedings of the ACM Web Conference 2024 (WWW '24), May 13–17, 2024, Singapore, Singapore.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3589334.3645570

1 INTRODUCTION

The task of conditional graph generation is crucial in various domains such as automatic compound discovery, drug design, and beyond [28, 48, 54, 57, 60]. It requires to generate graph data conditioned on a specific graph label, e.g. graph property, or category. In general, a graph with n nodes is defined as $\mathbf{G} \triangleq (\mathbf{A}, \mathbf{X}, y)$, where $\mathbf{A} \in \mathbb{R}^{n\times n}$ is the graph adjacency matrix, $\mathbf{X} \in \mathbb{R}^{n\times d}$ is the matrix of d-dimensional feature vectors of the nodes, and $y \in \mathbb{Z}$ is the graph label. Suppose the target graph distribution of interest is $\mathcal{G}$, the goal of unconditional generation is to generate plausible graph samples from the distribution $\mathcal{G}$ without any predefined criteria. For instance, this could involve generating grids, whether connected or disconnected, from the entire available population. On the other hand, given a label y, conditional generation aims to generate conditioned graph samples $\mathbf{G}|y$ from the conditional distribution $\mathcal{G}|y$. This might involve generating connected grids exclusively from a population of connected ones. To this end, we aim to learn a generative model g, such that the conditioned distribution $\mathcal{G}|y$ can be approximated well by the distribution of $g(\boldsymbol{\epsilon}_\mathbf{A}, \boldsymbol{\epsilon}_\mathbf{X}; y)$, where $\boldsymbol{\epsilon}_\mathbf{A}$ and $\boldsymbol{\epsilon}_\mathbf{X}$ are noisy matrices sampled from a given prior distribution (e.g. a Gaussian distribution).

While there is a considerable amount of work dedicated to unconditional graph generation [7, 10, 22, 33, 36, 40, 50, 67], the field of conditional graph generation is relatively understudied. The main challenge in conditional graph data generation arises from two aspects: Firstly, the conditional graph distribution $\mathcal{G}|y$ is highly complicated, as the relationship between graph features and topology varies significantly across different graph labels. Secondly, the dataset conditioned on a specific y typically consists of fewer data samples, leading to a higher demand for the effectiveness of the learning model due to data scarcity.

Indeed, unconditional generative models can be transformed into conditional models through two simple methods: 1) Train an unconditional model on the dataset fragment of each graph label. 2) Incorporate an extra graph label embedding module [16, 18]. However, existing unconditional models are inherently limited by their generation curriculum, i.e. the strategy of generating intermediate graph samples. In intuition, an ideal conditional generation curriculum is expected to satisfy the following principles:

1. **Easy-to-learn**: It should decompose the generation process into multiple intermediate steps that are easier to learn and execute.
2. **Semantic preservation**: Each intermediate state should strongly correlate with the given graph label, ensuring semantic information preservation throughout the generation process.

From this perspective, existing likelihood-based models [10, 36, 50] often encounter issues such as generation instability in single-shot models, and a loss of semantic information in multi-shot models due to substantial likelihood estimation errors, resulting in limited generation quality and increased computational overhead [22]. While diffusion-based models [7, 22, 33, 40] achieve state-of-the-art unconditional generation quality by progressively denoising corrupted graph data through reverse diffusion SDE, this process introduces Gaussian noise to each intermediate step, significantly impairing semantic preservation and leading to incorrectly generated samples.

Contributions. To address these limitations, we propose a novel conditional graph generative model, coined **G**raph **Prin**cipal **Flow** **Net**work (**GPrinFlowNet**). The key idea is that small eigenvalues

*Both authors contributed equally to this research.
†The author is also with Nanyang Technological University, Singapore.
[1]https://github.com/mzf666/gprinflownet

WWW '24, May 13–17, 2024, Singapore, Singapore
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0171-9/24/05
https://doi.org/10.1145/3589334.3645570

and their associated eigenvectors (low-frequency principal components) of the graph adjacency matrix are closely related to coarse-resolution graph structure and global graph topological properties [6, 49]. Thus, the low-frequency component is an ideal starting point for conditional graph generation, ensuring the model maintains label guidance even at early generation steps. Motivated by this, our GPrinFlowNet incorporates a low-to-high frequency generation curriculum, wherein the k-th intermediate generation state corresponds to the k smallest principal components of the desirable graph adjacency. To effectively follow the low-to-high frequency curriculum and sample from the intricate conditional distribution, we build our GPrinFlowNet upon the Generative Flow Network framework (GFlowNet) [4, 5], a recent advancement in generative modeling. Our low-to-high frequency curriculum in GPrinFlowNet, demonstrated by Figure 1, effectively preserves semantic information, resulting in accurate and high-quality conditional generation. The contributions of this paper are three-fold.

- Systematic analysis (Section 4.1) shows that preserving semantic information throughout the generation process improves the conditional generation performance. The low-to-high frequency curriculum illustrates a strong ability for semantic information preservation.
- We establish the Graph Principal Flow Network (GPrinFlowNet) in Section 4.2 to generate high-quality conditional graphs using the low-to-high frequency generation curriculum.
- Extensive experiments (Section 5) demonstrate that our GPrinFlowNet achieves state-of-the-art conditional graph generation performance across various real-world datasets.

2 RELATED WORK

Graph Generation. In recent years, there have been several advanced graph generation strategies proposed, as outlined in [34, 36, 40, 50, 59, 60] and [23]. Among these, GraphRNN [59] and GraphVAE [50] generate nodes and edges sequentially with validity checks, while GAN-based models [10], VAE-based models [36], flow-based models [60], score-based models [23, 40] generate the entire graph in an integrative way and exhibit high computational efficiency due to their node permutation-invariant property, and spectral-based model [33] generates the graph via spectral diffusion by reconstructing the graph eigenvalues through the score-based diffusion. Different from existing graph generation methods, our proposed GPrinFlowNet networks adopt a novel method for both generic graph and molecule generation based on GFlowNet. Compared to graph diffusion models such as GDSS [23], our model can generate graphs according to the conditions faster and more accurately.

Graph Curriculum Learning. Graph curriculum learning, inspired by human learning processes, organizes data samples from easy to hard to improve the performance of graph learning models [26, 51]. While various curricula have been proposed for improving node classification [14, 32, 52, 66], graph classification [1, 8, 15], link prediction [55, 58], curricula for generative models are relatively unexplored. GPrinFlowNet introduces a low-to-high frequency generation curriculum, enhancing conditional generation by preserving semantic information throughout the process.

Molecular Generation. Molecular generations are often coherent with graph generation methods, which aim at generating valid meaningful molecules with high efficiency and uniqueness. Molecules inherently adopt a graph-like structure with atoms as nodes interconnected by bonds, represented as edges, making them an optimal input for deep learning models. These molecular graphs are commonly characterized using three matrices: node feature matrix, edge feature matrix, and adjacency matrix. Initially stored in the SMILES format for ease of access, molecules are converted to molecular graphs using tools such as RDKit [25]. Earlier molecular generation approaches utilized sequence-based generative models, representing molecules as SMILES strings. However, these methods often face challenges from long dependency modeling and have issues with validity since SMILES strings do not guarantee correctness. As a result, recent studies have predominantly adopted graph representations for molecular structures. A variety of graph generative models have been introduced, employing methods like variational auto-encoders [31, 50], generative adversarial networks [2, 10], normalizing flows [35, 48], and graph diffusion models [19, 23]. Our proposed GPrinFlowNet can generate accurate graph representations for molecules that satisfy specific properties.

Generative Flow Networks. The Generative Flow Network (GFlowNet) is a recent advancement of generative models [4, 5, 63, 64] that enables us to model and sample from complicated and intractable compositional distributions in proportional to a given reward function, leading to applications on various domains. [12] proposed the DAG-GFlowNet as an alternative to MCMC for Bayesian networks inference; [43] applies GFlowNet to tackle the exploration problem in sparse reward environments; [62] proposed the MLE-GFlowNet to unify over many generative models; [20] applied GFlowNets on the biological sequence design; [39] adopted GFlowNets for molecule design and drug discovery. Our GPrinFlowNet borrows the philosophy of compositional generation and the trajectory balance supervision from GFlowNet.

3 PRELIMINARIES

In this paper, we consider generating undirected and weighted graphs denoted by $\mathbf{G} \triangleq (\mathbf{A}, \mathbf{X}, y)$. Here, $\mathbf{A} \in \mathbb{R}^{n\times n}$ is the adjacency matrix for n nodes with $\mathbf{A}[i,j]$ denoting the connection weight between nodes i and j; $\mathbf{X} \in \mathbb{R}^{n\times d}$ is the d-dimensional graph feature matrix; $y \in \mathbb{Z}$ is the graph label representing the graph category. In addition, we define $\mathbf{D} \triangleq \operatorname{diag}(d_1, ..., d_n) \in \mathbb{R}^{n\times n}$ with $d_i \triangleq \sum_j \mathbf{A}[i,j]$, $[n] \triangleq \{1, ..., n\}$ and $r(\mathbf{A})$ as the rank of $\mathbf{A}$.

Conditional Graph Generation. Let $\mathcal{G}$ be the target graph data distribution, and $\mathcal{G}|y$ denotes the conditional distribution. Given a specified graph label y, we aim to generate plausible samples from $\mathcal{G}|y$ using random noise drawn from a predefined prior π, i.e. $\hat{\mathbf{G}} \triangleq (\hat{\mathbf{A}}, \hat{\mathbf{X}}) = g_{\boldsymbol{\theta}}(\boldsymbol{\epsilon}_{\mathbf{A}}, \boldsymbol{\epsilon}_{\mathbf{X}}; y)$, where $g_{\boldsymbol{\theta}}$ is a generative model parameterized by $\boldsymbol{\theta}$, and $\boldsymbol{\epsilon}_{\mathbf{A}}$ and $\boldsymbol{\epsilon}_{\mathbf{X}}$ are random matrices drawn from π. A generative model enables multi-step generation via

$$\hat{\mathbf{G}}_t \triangleq (\hat{\mathbf{A}}_t, \hat{\mathbf{X}}_t) = g_{\boldsymbol{\theta}}(\hat{\mathbf{G}}_{t-1}; y),\ t = 1, ..., T, \tag{1}$$

where $\hat{\mathbf{G}}_0 \triangleq (\boldsymbol{\epsilon}_{\mathbf{A}}, \boldsymbol{\epsilon}_{\mathbf{X}})$ and $\hat{\mathbf{G}} \triangleq \hat{\mathbf{G}}_T$ is the final output.

Generation Curriculum. To improve generation quality and stability, a generation process usually involves a series of intermediate states $\mathbf{G}_0 \to \mathbf{G}_1 \to \cdots \to \mathbf{G}_T$, starting from a noisy priori $\mathbf{G}_0 \sim \pi$

and terminating at the final output $\mathbf{G}_T \sim \mathcal{G}$. Here, $\mathbf{A}_t$ and $\mathbf{X}_t$ denote the t-th intermediate graph adjacency and feature matrix. Each generated sample undergoes iterative refinement for T times before the final output. In this paper, we define "**generation curriculum**" as the trajectory comprising of all intermediate states $(\mathbf{G}_0, ..., \mathbf{G}_T)$. Noticeably, the generation curriculum defines the ideal generation strategy regardless of the implementation of the generative model. In practice, one can manipulate and improve the generation process of the model by aligning its intermediate generation with a desired generation curriculum. As one shall see later, our proposed low-to-high frequency generation curriculum preserves the semantic information well and it significantly improves the model performance.

Generative Flow Network. A desirable generative framework needs to be flexible to implement various generation curricula, and powerful to learn the complicated conditional distribution $\mathcal{G}|y$. Thus, we develop our framework upon Generative Flow Network (GFlowNet) [4, 5], an advanced technique for generating compositional discrete objects from complicated distributions. Given the target data space Ω and a reward function $R(\cdot) : \Omega \mapsto \mathbb{R}_+$, a GFlowNet aims to generate samples $\mathbf{x} \in \Omega$ with a probability proportional to $R(\mathbf{x})$. To this end, a GFlowNet generates a Markovian trajectory $\tau = (\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_T)$ following $\mathbf{x}_t \sim P_{F,\boldsymbol{\theta}}(\mathbf{x}_t|\mathbf{x}_{t-1})$, where $\mathbf{x}_T$ is the final output, and $P_{F,\boldsymbol{\theta}}(\cdot|\cdot) : \Omega \times \Omega \mapsto \mathbb{R}_+$ is termed the "**forward policy**", a learnable transition probability kernel. Starting with $\mathbf{x}_0$ drawn from priori π, at the t-th intermediate state, a GFlowNet accepts the previously constructed object $\mathbf{x}_{t-1}$ and adds a new building block sampled from $P_{F,\boldsymbol{\theta}}(\cdot|\mathbf{x}_{t-1})$ to produce $\mathbf{x}_t$. We take $\mathbf{x}_T$ as the final output when a termination action is sampled at step T. Therefore, the trajectory probability can be evaluated via forward decomposition

$$\hat{P}(\tau) = \prod_{t \in [T]} P_{F,\boldsymbol{\theta}}(\mathbf{x}_t|\mathbf{x}_{t-1})\pi(\mathbf{x}_0). \tag{2}$$

Note that a GFlowNet aims to learn a $P_{F,\boldsymbol{\theta}}$ to align the specified reward $R(\mathbf{x}_T)$ with the marginal probability $\hat{P}(\mathbf{x}_T)$. However, evaluating $\hat{P}(\mathbf{x}_T)$ requires an intractable integration of $\hat{P}(\tau)$ over all trajectories terminating at $\mathbf{x}_T$. To circumvent this integration, a GFlowNet incorporates an auxiliary "**backward policy**" denoted by $P_{B,\boldsymbol{\theta}}$ to supervise $\hat{P}(\tau)$ by minimizing the "**trajectory balance**" objective [37],

$$L_{\text{tb}}(\tau; \boldsymbol{\theta}) \triangleq \left(\log\left(\frac{\prod_{t\in[T]} P_{F,\boldsymbol{\theta}}(\mathbf{x}_t|\mathbf{x}_{t-1})\pi(\mathbf{x}_0)}{\frac{R(\mathbf{x}_T)}{Z_{\boldsymbol{\theta}}} \prod_{t\in[T]} P_{B,\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)}\right)\right)^2,$$

where $Z_{\boldsymbol{\theta}} \in \mathbb{R}_+$ is a trainable scalar such that $R(\cdot)/Z_{\boldsymbol{\theta}}$ is a normalized density function. In intuition, $P_{B,\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)$ represents the probability that $\mathbf{x}_t$ is constructed from $\mathbf{x}_{t-1}$ at the t-th intermediate state. Thus, minimizing L_{tb} is equivalent to matching the trajectory probability via either forward or backward Markovian decomposition. Notably, the supervision signal $R(\mathbf{x}_T)$ is only provided at the end of each trajectory. In general, a GFlowNet parameterized by $\boldsymbol{\theta}$ is represented by $(P_{F,\boldsymbol{\theta}}, P_{B,\boldsymbol{\theta}}, Z_{\boldsymbol{\theta}})$, where the forward and backward policies are implemented by two neural networks. During training, we calculate and minimize L_{tb} based on $P_{F,\boldsymbol{\theta}}$, $P_{B,\boldsymbol{\theta}}$, and $Z_{\boldsymbol{\theta}}$, while during sampling and evaluation, we only use $P_{F,\boldsymbol{\theta}}$ to generate plausible data $\mathbf{x}_T$.

4 CONDITIONAL GENERATION VIA GPRINFLOWNET

4.1 Low-to-High Frequency Generation

As discussed in Section 1, an ideal curriculum should be a composition of multiple easy-to-learn intermediate steps. This motivates us to decompose the graph adjacency generation process into multiple rank-increasing steps.

Definition 1 (Rank-increasing Curriculum). Following the notations defined in this paper, a generation curriculum $(\mathbf{G}_0, ..., \mathbf{G}_T)$ with $\mathbf{G}_t = (\mathbf{A}_t, \mathbf{X}_t)$, $t \in [T]$ is called "rank-increasing" if $r(\mathbf{A}_s) \leqslant r(\mathbf{A}_t)$ a.s. holds for any $s \leqslant t$.

Intuitively, the rank of graph adjacency is closely related to the essential information to be learned from it. Therefore, adopting a rank-increasing curriculum in graph adjacency generation adheres to the principle of "starting with simplicity and progressing gradually". As shown in Figure 1, diffusion-based models violate this principle, as they initiate the generation process from a full-rank dense Gaussian noise, which deviates from the naturally sparse and low-rank distribution of graph adjacency matrices.

To design an effective rank-increasing curriculum for conditional generation, it is essential to preserve the semantic information well throughout the entire generation process. Hence, each intermediate generation state can receive strong guidance from the graph label, even at the very early stage of the generation process, leading to improved generation accuracy. Motivated by the graph spectral theory [6, 61] that small frequency components are highly related to coarse global structure and graph topological property, we propose the low-to-high frequency generation curriculum to enable graph learning in a coarse-to-fine manner.

Definition 2 (Low-to-high Frequency Generation Curriculum). Following the notations defined in this paper, let π be a noisy prior, $(\mathbf{A}, \mathbf{X}) \sim \mathcal{G}$ be the observed data, let $\mathbf{A} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top$ be the eigen-decomposition of the graph adjacency matrix, where $\mathbf{U}$ is the eigenvector matrix and $\boldsymbol{\Lambda} = \text{diag}(\lambda_1, ..., \lambda_n)$ is the diagonal eigenvalue matrix, with $|\lambda_1| \leqslant |\lambda_2| \leqslant \cdots \leqslant |\lambda_n|$. Then, the low-to-high frequency generation curriculum is defined as $\tau = (\mathbf{G}_0, ..., \mathbf{G}_n)$ with $\mathbf{G}_T = (\mathbf{A}, \mathbf{X})$ and

$$\mathbf{G}_0 = (\boldsymbol{\epsilon}_\mathbf{A}, \boldsymbol{\epsilon}_\mathbf{X}) \sim \pi,\ \mathbf{G}_t = (\mathbf{A}_t, \mathbf{X}),\ 0 < t < n, \tag{3}$$

$$\mathbf{A}_t \triangleq \hat{\mathbf{A}}_t - \text{diag}(\hat{\mathbf{A}}_t),\ \hat{\mathbf{A}}_t \triangleq \mathbf{U}\ \text{diag}(\lambda_1, ..., \lambda_t, 0, ..., 0)\ \mathbf{U}^\top, \tag{4}$$

The intermediate $\mathbf{A}_t$ is reconstructed using the t lowest frequency components of the graph adjacency matrix $\mathbf{A}$, capturing the t-th slowest varying structural signals of the graph. A smaller t results in a coarser approximation of the full graph adjacency matrix. As t increases towards n, $\mathbf{A}_t$ increasingly recovers the full adjacency matrix. As depicted in Figure 1, this approach implements a coarse-to-fine strategy for adjacency generation, starting with a coarse, dense adjacency matrix and progressively refining it to full resolution. Figure 1 shows that the coarse, dense matrix $\mathbf{A}_t$ maintains a strong correlation with the graph label (e.g. graph properties), effectively preserving semantic information throughout the generation process.

We quantify the proficiency of the low-to-high frequency generation curriculum in semantic preservation on the Mutag, Enzymes,

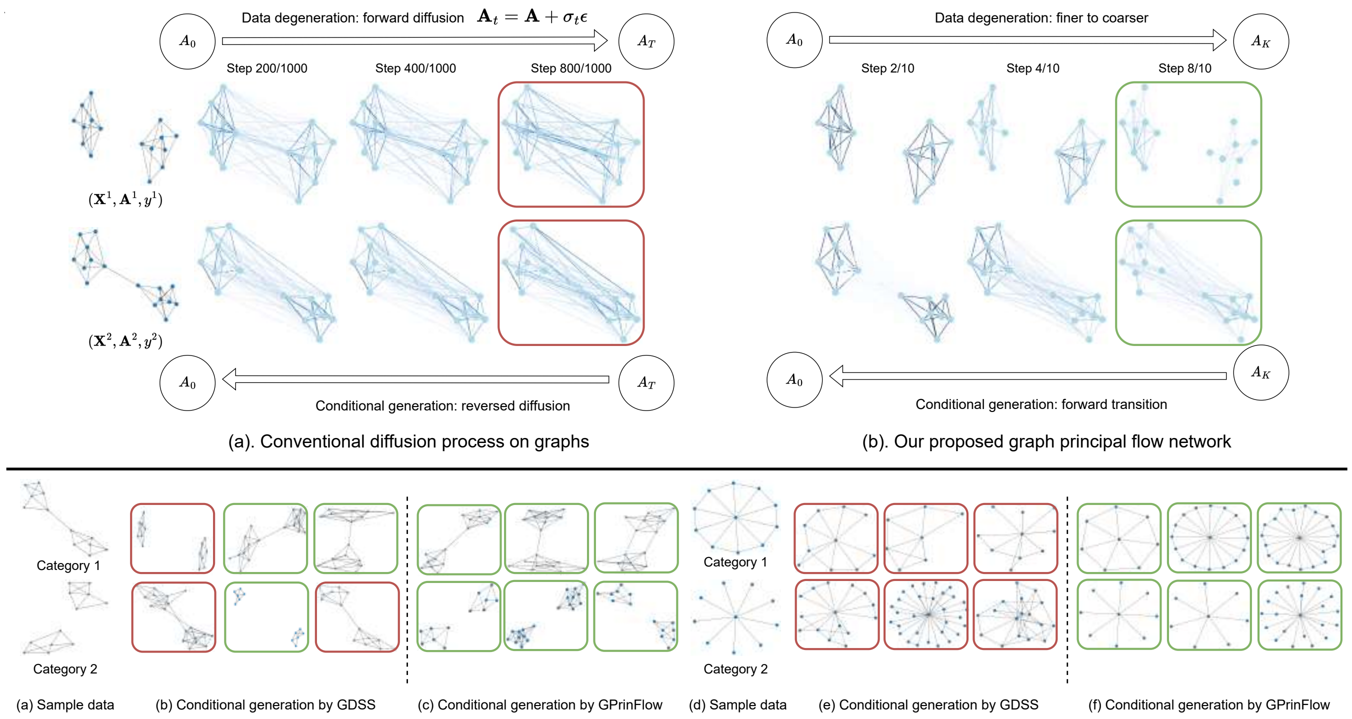
Data degeneration: forward diffusion $\mathbf{A}_t = \mathbf{A} + \sigma_t \epsilon$
A_0
A_T
Step 200/1000
Step 400/1000
Step 800/1000
$(\mathbf{X}^1, \mathbf{A}^1, y^1)$
$(\mathbf{X}^2, \mathbf{A}^2, y^2)$
Conditional generation: reversed diffusion
(a). Conventional diffusion process on graphs
Data degeneration: finer to coarser
A_K
Step 2/10
Step 4/10
Step 8/10
Conditional generation: forward transition
(b). Our proposed graph principal flow network
Category 1
Category 2
(a) Sample data
(b) Conditional generation by GDSS
(c) Conditional generation by GPrinFlow
(d) Sample data
(e) Conditional generation by GDSS
(f) Conditional generation by GPrinFlow

Figure 1: Visualization of the intermediate generation state of diffusion-based models (top left) and our GPrinFlowNet (top right) for two graph samples with different labels (y_1: disconnected, y_2: connected). The horizontal axis represents the generation steps. While the low-frequency graph component (green box) proficiently captures the subtle graph label semantic, the fully corrupted graph (red box) fails to distinguish the connectivity.

and IMDB-binary (IMDB-B) datasets [38]. In this paper, we evaluate semantic information preservation via average mutual information defined as follows.

Definition 3 (Average Mutual Information)**.** For any generation curriculum $\tau \triangleq (\mathbf{G}_0, ..., \mathbf{G}_T)$, the average mutual information of τ is defined as

$$\bar{I}(\tau) \triangleq \frac{1}{T} \sum_{t \in [T]} I(\mathbf{G}_t; y), \tag{5}$$

where $I(\cdot;\cdot)$ is the mutual information.

Intuitively, a high $\bar{I}(\tau)$ indicates that the correlation between graph label information remains to be strong throughout the entire generation process, implying that the generation curriculum τ effectively adheres to the specified condition, resulting in more accurate conditional generated samples.

We focus on graph adjacency and create multiple rank-increasing curriculum candidates. Each curriculum candidate is formed by adding a frequency component selected randomly from the remaining ones. For each dataset, we calculate the average mutual information for each curriculum following the implementation of MINE [3]. Finally, we train our GPrinFlowNet model on four representative curricula and we evaluate the corresponding performance. More details can be found in Appendix B.

Each circle point in Figure 2 denotes the estimated mutual information $\hat{I}(\mathbf{A}_t, y)$ between a particular intermediate adjacency matrix and the graph label. Figure 2 reveals that our low-to-high frequency curriculum achieves nearly the highest average mutual information

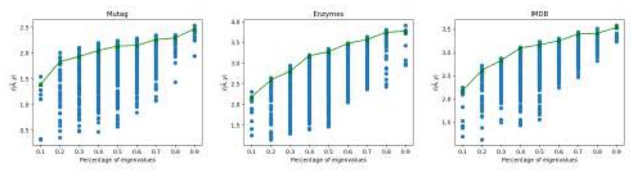

Figure 2: Estimated mutual information between different graph frequency components to the graph label. The x-axis is the percentage of eigenvalues used for low-resolution reconstruction, and the y-axis is the estimated value of $I(\hat{\mathbf{A}}_t, y)$. Each blue point in the figure represents a result of each frequency component. The green curve shows the mutual information of our proposed low-to-high frequency generation curriculum.

compared to other rank-increasing curricula. Additionally, as will be shown in Table 6, the proposed GPrinFlowNet model (in Section 4.2) demonstrates improved generation quality when supervised by a curriculum with higher average mutual information. These empirical results suggest that the low-to-high frequency curriculum is good at preserving strong graph label supervision signals throughout the generation process, leading to higher generation performance. Furthermore, the low-to-high frequency curriculum can be efficiently computed without the need to optimize the mutual information across all possible intermediate state combinations. Consequently, it turns out to be a desirable conditional generation curriculum, meeting both the criteria of being easy to learn and effectively preserving semantic principles.

4.2 Graph Principal Flow Network

Inspired by the analysis in Section 4.1, we establish a low-to-high frequency graph generation algorithm, coined **Graph Principal Flow Network**. Similar to the GFlowNet philosophy, at each shot, our GPrinFlowNet generates a graph sample via a Markovian trajectory, following the aforementioned low-to-high frequency curriculum.

Definition 4 (Graph Principal Flow Network)**.** Following the notations defined in Section 3, the Graph Principal Flow Network (GPrinFlowNet) is a generalized GFlowNet that generates samples via a low-to-high principal component curriculum. Let n be the target graph size, y be a specified graph label, and π be a noisy priori on $\mathbb{R}^{n\times n}\times\mathbb{R}^{n\times d}$, at each shot, GPrinFlowNet generates a Markovian trajectory $\hat{\tau}$, with

$$\hat{\tau} \triangleq (\hat{\mathbf{G}}_0, \hat{\mathbf{G}}_1|y, ..., \hat{\mathbf{G}}_n|y),\ \hat{\mathbf{G}}_0 \sim \pi, \tag{6}$$

$$\hat{\mathbf{G}}_t|y \sim P_{F,\boldsymbol{\theta}}(\cdot \mid \hat{\mathbf{G}}_{t-1}; y),\ t = 1, ..., n, \tag{7}$$

where $\hat{\mathbf{G}}_t|y = (\hat{\mathbf{A}}_t|y, \hat{\mathbf{X}}_t|y)$ is the t-th intermediate state, and $P_{F,\boldsymbol{\theta}}$ is the learnable forward policy determining the one-shot generation of the successive intermediate state. A GPrinFlowNet is equipped with a backward policy $P_{B,\boldsymbol{\theta}}$, modeling the probability of reversing a one-shot generation step. The forward and backward policies are required to satisfy the principal trajectory balance criterion, i.e.,

$$P_{F,\boldsymbol{\theta}}(\hat{\mathbf{G}}_t|\hat{\mathbf{G}}_{t-1}; y)\frac{R(\hat{\mathbf{G}}_{t-1})}{Z_{\boldsymbol{\theta},t-1}} = P_{B,\boldsymbol{\theta}}(\hat{\mathbf{G}}_{t-1}|\hat{\mathbf{G}}_t; y)\frac{R(\hat{\mathbf{G}}_t)}{Z_{\boldsymbol{\theta},t}}, \tag{8}$$

where $R(\hat{\mathbf{G}}_t) \triangleq \exp(-\alpha\|\hat{\mathbf{G}}_t - \mathbf{G}_t\|_{\mathrm{F}})$ is the energy function for $t = 0, ..., n$, $\alpha > 0$ is a temperature hyperparameter, $\{Z_{\boldsymbol{\theta},t-1}\}$ are learnable scalar normalizers, and $\tau = (\mathbf{G}_0, ..., \mathbf{G}_n)$ is the defined low-to-high frequency generation curriculum.

Our GPrinFlowNet differs from the standard GFlowNet in two key aspects. Firstly, each intermediate state of GPrinFlowNet resides within a continuous-valued space. Secondly, the principal trajectory balance criterion imposes supervision to every intermediate state, and it guides GPrinFlowNet to progressively generate conditional graph data from lower-frequency to higher-frequency components by aligning the i-th intermediate generated graph with the t-th granularity level.

Parameterization and Training. We present an effective parameterization and training objective for the implementation of GPrinFlowNet. Specifically, we set π as a standard Gaussian, and we employ a hierarchical parameterization for the forward policy

$$\hat{\mathbf{G}}_t \triangleq (\hat{\mathbf{A}}_t, \hat{\mathbf{X}}_t) \sim P^A_{F,\boldsymbol{\theta}}(\hat{\mathbf{A}}_t|\hat{\mathbf{G}}_{t-1}; y)P^X_{F,\boldsymbol{\theta}}(\hat{\mathbf{X}}_t|\hat{\mathbf{G}}_{t-1}; y).$$

We define $P^A_{F,\boldsymbol{\theta}}$ by $(\hat{\mathbf{A}}_t|\hat{\mathbf{A}}_{t-1}, y) \sim \hat{\lambda}_t\hat{\mathbf{u}}_t\hat{\mathbf{u}}_t^\top + \hat{\mathbf{A}}_{t-1}$, with $\hat{\mathbf{u}}_t$ sampled from the empirical eigenvector distribution, and

$$\hat{\lambda}_t \sim N(\mu^\lambda_{F,\boldsymbol{\theta}}(\hat{\mathbf{G}}_{t-1}, y), \sigma^\lambda_{F,\boldsymbol{\theta}}(\hat{\mathbf{G}}_{t-1}, y)),\ \hat{\lambda}_t \in \mathbb{R}. \tag{9}$$

Similarly, we parameterize $P^X_{F,\boldsymbol{\theta}}$ as a learnable Gaussian

$$(\hat{\mathbf{X}}_t|\hat{\mathbf{A}}_{t-1}, y) \sim N(\boldsymbol{\mu}^X_{F,\boldsymbol{\theta}}(\hat{\mathbf{G}}_{t-1}, y), \sigma^X_{F,\boldsymbol{\theta}}(\hat{\mathbf{G}}_{t-1}, y)\mathbf{I}). \tag{10}$$

Here, the means and variances $\mu^\lambda_{F,\boldsymbol{\theta}}$, $\sigma^\lambda_{F,\boldsymbol{\theta}}$, $\boldsymbol{\mu}^X_{F,\boldsymbol{\theta}}$, and $\sigma^X_{F,\boldsymbol{\theta}}$ are generated by 4 multi-layer Graph Convolutional Networks (GCN) [11] with compatible output dimensions. At each step, we generate a sample for the next graph adjacency by first sampling the succeeding frequency component and then adding it to the current resolution's graph adjacency. In practice, we can apply Gram–Schmidt orthonormalization to achieve orthonormal $\hat{\mathbf{u}}_t$. We parameterize the backward policy $P_{B,\boldsymbol{\theta}}$ in the same way as the forward policy. With the explicit expressions of the forward and backward policies, we train GPrinFlowNet by minimizing the principal trajectory balance objective

$$L(\hat{\tau}; \boldsymbol{\theta}) \triangleq \sum_{t=1}^{n}\left(\log\frac{\prod_{s=0}^{t-1} P_{F,\boldsymbol{\theta}}(\hat{\mathbf{G}}_{s+1}|\hat{\mathbf{G}}_s; y)\pi(\hat{\mathbf{G}}_0)}{\frac{R(\hat{\mathbf{G}}_t)}{Z_{\boldsymbol{\theta},t}}\prod_{s=0}^{t-1} P_{B,\boldsymbol{\theta}}(\hat{\mathbf{G}}_s|\hat{\mathbf{G}}_{s+1}; y)}\right)^2, \tag{11}$$

where the normalizers $\{Z_{i,\boldsymbol{\theta}}\}_{i=1}^n$ are trainable scalars. In practice, for each labeled sample $(\mathbf{X}, \mathbf{A}, y)$, we first perform the eigendecomposition of $\mathbf{A}$. Starting from the initial state $\hat{\mathbf{G}}_0 \sim \pi$, we leverage the forward policy $P_{F,\boldsymbol{\theta}}$ to generate a Markovian trajectory of graph samples $\hat{\tau} = (\hat{\mathbf{G}}_0, ..., \hat{\mathbf{G}}_n)$, such that $\hat{\mathbf{G}}_{t+1} \sim P_{F,\boldsymbol{\theta}}(\hat{\mathbf{G}}_{t+1}|\hat{\mathbf{G}}_t, y)$. Meanwhile, we calculate and store the forward and backward transition probabilities based on the hierarchical Gaussian parameterization. Finally, we calculate the principal trajectory balance objective and average it over a data batch, and we update the neural parameters $\boldsymbol{\theta}$ via stochastic gradient descent. The training process is detailed in Algorithm 1.

Conditional Generation with GPrinFlowNet. With a well-trained GPrinFlowNet, we can efficiently generate high-quality conditional graph data in at most n steps, substantially fewer than the steps required by diffusion-based models. The conditional generation process is detailed in Algorithm 2.

Algorithm 1 Training GPrinFlowNet

```
Input: labeled training data S, the forward and backward policy networks $P_{F,\theta}$ and $P_{B,\theta}$, learning rate $\beta > 0$.
Output: $P_{F,\theta}$, $P_{B,\theta}$, $\{Z_{\theta,t}\}_{t=1}^n$
while not converge do
    Sample data $(\mathbf{A}, \mathbf{X}, y) \sim \mathbf{S}$
    $(\mathbf{U}, \mathbf{\Lambda}) \leftarrow$ EigenDecomp$(\mathbf{A})$
    Initialize $\hat{\mathbf{G}}_0 \leftarrow (\mathbf{X}_0, \mathbf{0})$, $\mathbf{X}_0 \sim N(\mathbf{0}, \mathbf{I})$
    $L_\theta \leftarrow 0$, $L_F \leftarrow 0$, $L_B \leftarrow 0$
    for $t = 1$ to $n$ do
        $\hat{\mathbf{G}}_{t+1} \sim P_{F,\theta}(\cdot|\hat{\mathbf{G}}_t; y)$ {Forward transition}
        $L_F \leftarrow L_F + \log P_{F,\theta}(\hat{\mathbf{G}}_{t+1}|\hat{\mathbf{G}}_t; y)$
        $L_B \leftarrow L_B + \log P_{B,\theta}(\hat{\mathbf{G}}_t|\hat{\mathbf{G}}_{t+1}; y)$
        $L_\theta \leftarrow L_\theta + (L_F - L_B + \log Z_{\theta,t} - \log R(\hat{\mathbf{G}}_t))^2$
    end for
    $\theta \leftarrow \theta - \beta\nabla_\theta L_\theta$
end while
return $P_{F,\theta}$, $P_{B,\theta}$, $\{Z_{\theta,t}\}_{t=1}^n$
```

5 EXPERIMENTS

In this section, we systematically evaluate GPrinFlowNet's performance across various scenarios: conditional generic graph generation, conditional molecular generation, and unconditional generation. The experimental settings are detailed in Appendix A.

Algorithm 2 Conditional Generation with GPrinFlowNet

```
**Input:** A target label $y$, the forward and the backward policy networks $P_{F,\theta}$ and $P_{B,\theta}$, a temperature hyperparameter $\sigma$.
**Output:** A conditional graph data $(\hat{\mathbf{A}}, \hat{\mathbf{X}})$.
Initialize $\hat{\mathbf{G}}_0 \leftarrow (\mathbf{X}_0, \mathbf{0})$, $\mathbf{X}_0 \sim N(\mathbf{0}, \sigma^2\mathbf{I})$
**for** $t = 0$ **to** $n-1$ **do**
    $(\hat{\lambda}_{t+1}, \hat{\mathbf{u}}_{t+1}, \hat{\mathbf{X}}_{t+1}) \sim P_{F,\theta}(\cdot|\hat{\mathbf{G}}_t, y)$
    $\hat{\mathbf{A}}_{t+1} \leftarrow \hat{\lambda}_{t+1}\hat{\mathbf{u}}_{t+1}\hat{\mathbf{u}}_{t+1}^{\top} + \hat{\mathbf{A}}_t$
    $\hat{\mathbf{G}}_{t+1} \leftarrow (\hat{\mathbf{A}}_{t+1}, \hat{\mathbf{X}}_{t+1})$ {Forward transition}
**end for**
**return** $(\hat{\mathbf{A}}_n, \hat{\mathbf{X}}_n)$
```

5.1 Conditional Graph Generation

Baselines and datasets. We compare our method with the state-of-the-art graph generation methods, including graph diffusion methods such as GDSS [22], EDP-GNN [41]; VAE-based methods such as GraphVAE [50], CondGen [56]; auto-regressive models such as GraphAF [48], GraphDF [35], GraphRNN [59], CCGG [42]. Although these existing methods focus on unconditional generation, we modify and extend them for conditional generation by integrating a graph label embedding module, mirroring the approach we employed in GPrinFlowNet. We conduct experiments on the following datasets: AIDS [38] which contains 2 categories, Enzymes [47] which consists of 6 categories, and Synthie [13] which contains 4 categories for graph conditional generation. More details of the datasets and the evaluation metric are discussed in Appendix A.

Results and analysis. Following the graph generation evaluation setting used in [22], for each category, we adopt the same training versus test split ratio as [22]. We measure the maximum mean discrepancy (MMD) to compare the distributions of graph statistics between the same number of generated and test graphs under each category, including the degree, the clustering coefficient, and the number of occurrences of orbits with 4 nodes [22, 59]. We report the average of the degree, clustering coefficient, and the number of occurrences for each category in Table 1. We also report the mean MMD as our overall evaluation score under the Avg. column. As shown in Table 1, our proposed method achieves the best performance compared with the state-of-the-art graph generation baselines. Compared to GDSS, a leading graph generation method using diffusion, our model achieves significantly lower MMD scores of 2.4× and 3.0× on the Enzymes and Synthie datasets respectively, demonstrating the effectiveness of our model.

5.2 Conditional Molecular Generation

In addition to generic graph datasets, we further evaluate our GPrinFlowNet on conditional molecular generation on the well-known QM9 dataset [46]. Following previous studies [23, 35], all molecules are kekulized by the RDKit library [25] with hydrogen atoms removed. We label the molecules to 2 categories according to μ-dipole moment, 3 categories according to $\Delta\epsilon$-gap between ϵ_{LUMO} and ϵ_{HUMO}, and 2 categories according to α-isotropic polarizability. As shown in Figure 3, the labeling strategy is based on the histogram of the molecules with different properties. The split is according to the histogram of the corresponding molecular properties shown in

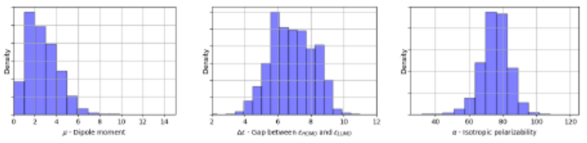

Figure 3: The histogram of molecules in QM9 dataset with specified properties: μ-dipole moment (left), gap between ϵ_{LUMO} and ϵ_{HUMO} (middle), and α-isotropic polarizability (right).

Figure 3. We show the details of molecular category assignments in Appendix A.

For each category, we evaluate the quality of 10,000 generated molecules with their validity, validity w/o check, Frechet ChemNet Distance (FCD) [45], Neighborhood Subgraph Pairwise Distance Kernel (NSPDK) MMD [9], uniqueness [22], and novelty [22]. FCD computes the distance between the test and the generated molecules using the activations of the penultimate layer of the ChemNet. NSPDK-MMD computes the MMD between the generated and the test 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 [23], we also include the "validity w/o correction" metric to evaluate the quality of molecular generation before the correction procedure. Unlike "validity" which measures the fraction of the valid molecules after the correction phase, it computes the fraction of the number of valid molecules without valency correction or edge resampling overall generated molecules.

Baselines. We evaluate our model against state-of-the-art molecular generation models, including auto-regressive methods, GraphAF [48] and GraphDF [35]. To ensure fair comparisons, as recommended by GDSS [23], we extend GraphAF and GraphDF to account for formal charges in the molecular generation, termed GraphAF+FC and GraphDF+FC respectively. Additionally, we also compare our GPrinflowNet with various representative flow-based and diffusion-based methods, including MoFlow [60], EDP-GNN [41], Graph-EBM [30], GDSS [40], and CDGS [19].

Results and analysis. We show the conditional generation results according to $\Delta\epsilon$ - ϵ_{LUMO} and ϵ_{HUMO} in Table 2 (left), α-isotropic polarizability in Table 2 (middle), and μ-dipole moment (right). GPrinFlowNet achieves the best performance under most of the metrics. The highest scores in NSPDK and FCD show that GPrinFlowNet can generate molecules with data distributions close to the real molecules in both the chemical semantic space and graph representation space. Especially, our model outperforms the state-of-the-art diffusion-based models, GDSS and CDGS, in most metrics. Our results verify that our proposed GPrinFlowNet is not only suitable for generic graph generation but also proficient in conditional molecular generation.

5.3 Unconditional Graph Generation

We further evaluate the unconditional graph generation performance of our GPrinFlowNet on the following synthetic datasets: 1)

Table 1: Generation results on the conditional graph generation datasets. 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.

	AIDS				Enzymes				Synthie			
	Real, $\lvert V\rvert \leq 95$, $\lvert C\rvert = 2$				Real, $\lvert V\rvert \leq 125$, $\lvert C\rvert = 6$				Synthetic, $\lvert V\rvert \leq 100$, $\lvert C\rvert = 4$			
	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Deg.↓	Clus.↓	Orbit↓	Avg.↓
GraphRNN [59]	0.241	0.143	0.034	0.139	0.086	0.294	0.307	0.229	0.247	0.285	0.419	0.317
GraphAF [48]	0.197	0.093	0.026	0.105	0.058	0.174	0.156	0.129	0.137	0.176	0.302	0.205
GraphDF [35]	0.184	0.085	0.031	0.101	0.062	0.196	0.204	0.154	1.681	1.265	0.258	1.068
GraphVAE [50]	0.358	0.284	0.127	0.256	1.249	0.687	0.381	0.772	1.554	1.074	0.232	0.953
GNF [29]	0.224	0.159	0.018	0.133	-	-	-	-	-	-	-	-
EDP-GNN [40]	0.127	0.082	0.024	0.077	0.067	0.241	0.225	0.177	0.148	0.185	0.347	0.226
GDSS [22]	0.062	0.049	0.022	0.044	0.038	0.158	0.132	0.109	0.114	0.126	0.269	0.169
CondGen [56]	0.138	0.115	0.032	0.095	0.065	0.184	0.213	0.154	0.151	0.162	0.295	0.202
CCGG [42]	0.097	0.074	0.035	0.068	0.043	0.125	0.117	0.095	0.107	0.159	0.236	0.167
DiGress [53]	0.060	0.048	0.021	0.043	0.033	0.146	0.085	0.088	0.109	0.097	0.158	0.121
GraphARM [24]	0.057	0.052	0.016	0.041	0.031	0.095	0.061	0.062	0.128	0.074	0.127	0.109
EB-GFN [65]	0.094	0.087	0.035	0.072	0.079	0.213	0.227	0.173	0.152	0.164	0.341	0.219
GPrinFlowNet (ours)	**0.046**	**0.031**	**0.012**	**0.029**	**0.027**	**0.062**	**0.046**	**0.045**	**0.048**	**0.042**	**0.079**	**0.056**

Table 2: Experiment results of conditional molecular generation on QM9 dataset. For each molecular sample, we assign three labels according to its chemical properties, including $\Delta\epsilon$-Gap between ϵ_{HOMO} and ϵ_{LUMO} (top), α-isotropic polarizability (middle), and the μ-dipole moment (bottom). The best results are highlighted in bold.

QM9	$\epsilon_{\text{HOMO}} - \epsilon_{\text{LUMO}}$			α-isotropic polarizability			μ-dipole moment		
Methods	VALID w/o check (%) ↑	NSPDK ↓	FCD ↓	VALID w/o check (%) ↑	NSPDK ↓	FCD ↓	VALID w/o check (%) ↑	NSPDK ↓	FCD ↓
GraphAF [48]	67.72	0.059	10.423	67.47	0.063	11.057	67.48	0.049	9.372
GraphAF + FC	74.37	0.053	10.536	74.17	0.055	11.147	74.72	0.053	9.248
GraphDF [35]	82.69	0.108	14.315	82.89	0.117	14.781	82.47	0.094	13.489
GraphDF+FC	93.74	0.121	14.846	93.48	0.134	14.482	93.31	0.114	13.476
MoFlow [60]	91.95	0.059	8.645	91.12	0.064	8.793	91.58	0.053	8.024
EDP-GNN [41]	47.30	0.032	5.642	47.74	0.037	5.884	47.72	0.030	5.081
GraphEBM [30]	8.13	0.096	10.404	8.03	0.104	10.527	8.91	0.087	9.970
GDSS [22]	95.20	0.028	5.417	95.58	0.029	5.863	95.76	0.022	5.047
CDGS [19]	99.41	0.021	3.326	99.44	0.023	3.741	99.17	0.017	3.024
GPrinFlowNet (Ours)	**99.72**	**0.012**	**2.798**	**99.74**	**0.013**	**2.925**	**99.79**	**0.011**	**2.627**

Community-small [59] ($12 \leq N \leq 20$): contains 100 small community graphs; 2) Enzymes [47] ($10 \leq N \leq 125$): contains 578 protein graphs which represent the protein tertiary structures of the enzymes from the BRENDA database; 3) Grid [59] ($100 \leq N \leq 400$): contains 100 standard 2D grid graphs. We compare our model against the aforementioned state-of-the-art unconditional generative models. As shown in Table 3, our GPrinFlowNet consistently achieves the best unconditional generation quality across various datasets, showing that semantic preservation is also beneficial for unconditional generation scenarios.

5.4 Generation Speed Analysis

In this section, we benchmark our GPrinFlowNet's graph generation speed against 4 baseline methods. Our GPrinFlowNet achieves superior efficiency for generating 100 samples on various datasets compared to other baselines, as demonstrated in Table 4. GPrinFlowNet's rapid generation stems from a fast-forward process, outperforming GDSS (based on graph Gaussian diffusion) by 26× to 58× due to its efficient mechanism and fewer required steps.

5.5 Ablation Studies

Ablation on intermediate supervision. We conduct ablation studies on how the intermediate supervision provided by the low-to-high frequency generation curriculum affects the performance of GPrinFlowNet. We control how frequently the intermediate supervision is computed in the principal trajectory balance objective. Results in Table 5 show that a stronger alignment between intermediate generation states and the low-to-high frequency generation

Table 3: Generation results on the unconditional graph datasets. 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.

	Community-small				Enzymes				Grid			
	Synthetic, $12 \leq \lvert V\rvert \leq 20$				Real, $10 \leq \lvert V\rvert \leq 125$				Synthetic, $100 \leq \lvert V\rvert \leq 400$			
	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Deg.↓	Clus.↓	Orbit↓	Avg.↓
DeepGMG [27]	0.220	0.950	0.400	0.523	-	-	-	-	-	-	-	-
GraphRNN [59]	0.080	0.120	0.040	0.080	0.017	**0.043**	0.021	0.043				
GraphAF [48]	0.18	0.200	0.020	0.133	1.669	1.283	0.266	1.073	-	-	-	-
GraphDF [35]	0.060	0.120	0.030	0.070	1.503	1.061	0.202	0.922	-	-	-	-
GraphVAE [50]	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 [29]	0.200	0.200	0.110	0.170	-	-	-	-	-	-	-	-
EDP-GNN [40]	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 [21]	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 [17]	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^2 [22]	0.045	0.086	0.007	0.046	0.026	0.102	0.009	0.046	0.111	0.005	0.070	0.062
GPrinFlowNet (ours)	**0.021**	**0.068**	**0.021**	**0.037**	**0.021**	0.088	**0.009**	**0.039**	**0.056**	0.042	**0.015**	**0.038**

Table 4: Graph generation speed comparison (in seconds) for generating 100 graphs under the methods' default setting.

Dataset	GraphAF	GraphDF	EDP-GNN	GDSS	**GPrinFlowNet (ours)**
Community-small	357	$2.47e^3$	368	72	**2.7**
Enzymes	596	$7.58e^3$	665	128	**10.2**
Grid	$5.83e^3$	$6.42e^4$	$7.58e^3$	$1.75e^3$	**30.89**

curriculum leads to higher generation quality. This alignment helps GPrinFlowNet effectively learn the distribution of the reconstructed graph adjacency matrix at various granularity levels.

Table 5: Ablation studies on the supervision scheme. We report the mean MMD over distributions of degree, clustering coefficients, and the number of orbits, for conditional graph generation.

Intermediate Supervision	AIDS	Enzymes	Synthie
No supervision	0.037	0.075	0.086
Supervision per 10 steps	0.035	0.061	0.072
Supervision per 5 steps	0.032	0.054	0.066
Supervision per 2 steps	0.030	0.049	0.058
Supervision in every step	**0.029**	**0.045**	**0.056**

Ablation on generation curricula. We implement and compare GPrinFlowNets trained with the five representative generation curricula proposed in Section 4.1, including the low-to-high frequency curriculum, high-to-low frequency curriculum, and the random curriculum. As shown in Table 6 and Figure 2, the random curriculum suffers from low average mutual information and the worst averaged generation performance as well. In contrast, our low-to-high frequency generation curriculum achieves high average mutual information, highlighting the importance of preserving semantic information for enhanced conditional graph generation. Notably, our low-to-high frequency curriculum does not require excessive calculation and optimization of average mutual information across all possible rank-increasing curricula.

Table 6: Ablation studies on the eigenvalue generation procedure. We report the mean MMD over distributions of degree, clustering coefficient, and the number of orbits.

Generation Curriculum	AIDS	Enzymes	Synthie
Random frequency curriculum	0.091	0.124	0.108
High-to-low frequency curriculum	0.047	0.068	0.075
Low-to-high frequency curriculum (ours)	**0.029**	**0.045**	**0.056**

6 CONCLUSION

In this paper, we propose the Graph Principal Flow Network (GPrinFlowNet), a conditional generation model that follows a progressive low-to-high frequency graph generation curriculum. Being a rank-increaing curriculum, GPrinFlowNet decomposes the complicated generation process into multiple easy-to-learn generation steps. Moreover, empirical experiments show that GPrinFlowNet is proficient at preserving the subtle yet crucial semantic features throughout a coarse-to-fine generation process. Thus, the GPrinFlowNet provides strong conditional guidance even at the early stage of generation process. Extensive experiments also show that our GPrinFlowNet achieves the state-of-the-art performance in conditional generation on both generic and molecular datasets.

ACKNOWLEDGEMENTS

Sinno J. Pan thanks the support of the Hong Kong Jockey Club Charities Trust to the JC STEM Lab of Integration of Machine Learning and Symbolic Reasoning and the Microsoft Research Asia collaborative research grant.

REFERENCES

[1] Usman Ahmed, Jerry Chun-Wei Lin, and Gautam Srivastava. 2022. Hyper-graph-based attention curriculum learning using a lexical algorithm for mental health. *Pattern Recogn. Lett.* 157, C (may 2022), 135–143. https://doi.org/10.1016/j.patrec.2022.03.018

[2] Rim Assouel, Mohamed Ahmed, Marwin H Segler, Amir Saffari, and Yoshua Bengio. 2018. Defactor: Differentiable edge factorization-based probabilistic graph generation. *arXiv preprint arXiv:1811.09766* (2018).

[3] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. 2018. Mutual information neural estimation. In *International conference on machine learning*. PMLR, 531–540.

[4] Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. 2021. Flow network based generative models for non-iterative diverse candidate generation. *Advances in Neural Information Processing Systems* 34 (2021), 27381–27394.

[5] Yoshua Bengio, Tristan Deleu, Edward J. Hu, Salem Lahlou, Mo Tiwari, and Emmanuel Bengio. 2021. GFlowNet Foundations. *CoRR* abs/2111.09266 (2021). arXiv:2111.09266 https://arxiv.org/abs/2111.09266

[6] Gene Cheung, Enrico Magli, Yuichi Tanaka, and Michael K Ng. 2018. Graph spectral image processing. *Proc. IEEE* 106, 5 (2018), 907–930.

[7] Hyuna Cho, Minjae Jeong, Sooyeon Jeon, Sungsoo Ahn, and Won Hwa Kim. 2023. Multi-resolution Spectral Coherence for Graph Generation with Score-based Diffusion. In *Thirty-seventh Conference on Neural Information Processing Systems*. https://openreview.net/forum?id=qUlpDjYnsp

[8] Guanyi Chu, Xiao Wang, Chuan Shi, and Xunqiang Jiang. 2021. CuCo: Graph Representation with Curriculum Contrastive Learning. In *International Joint Conference on Artificial Intelligence*. https://api.semanticscholar.org/CorpusID:235671564

[9] Fabrizio Costa and Kurt De Grave. 2010. Fast neighborhood subgraph pairwise distance kernel. In *ICML*.

[10] Nicola De Cao and Thomas Kipf. 2018. MolGAN: An implicit generative model for small molecular graphs. *arXiv preprint arXiv:1805.11973* (2018).

[11] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *Proceedings of the 30th International Conference on Neural Information Processing Systems* (Barcelona, Spain) *(NIPS'16)*. Curran Associates Inc., Red Hook, NY, USA, 3844–3852.

[12] Tristan Deleu, António Góis, Chris Chinenye Emezue, Mansi Rankawat, Simon Lacoste-Julien, Stefan Bauer, and Yoshua Bengio. 2022. Bayesian Structure Learning with Generative Flow Networks. In *The 38th Conference on Uncertainty in Artificial Intelligence*. https://openreview.net/forum?id=HElfed8j9g9

[13] Aasa Feragen, Niklas Kasenburg, Jens Petersen, Marleen de Bruijne, and Karsten Borgwardt. 2013. Scalable kernels for graphs with continuous attributes. *Advances in neural information processing systems* 26 (2013).

[14] Chen Gong. 2017. Exploring Commonality and Individuality for Multi-Modal Curriculum Learning. *Proceedings of the AAAI Conference on Artificial Intelligence* 31, 1 (Feb. 2017). https://doi.org/10.1609/aaai.v31i1.10774

[15] Yaowen Gu, Si Zheng, Zidu Xu, Qijin Yin, Liang Li, and Jiao Li. 2022. An efficient curriculum learning-based strategy for molecular graph learning. *Briefings in Bioinformatics* (04 2022). https://doi.org/10.1093/bib/bbac099

[16] Xiaojie Guo and Liang Zhao. 2020. A Systematic Survey on Deep Generative Models for Graph Generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45 (2020), 5370–5390. https://api.semanticscholar.org/CorpusID:220514457

[17] Florentin Guth, Simon Coste, Valentin De Bortoli, and Stéphane Mallat. 2022. Wavelet Score-Based Generative Modeling. *ArXiv* abs/2208.05003 (2022).

[18] Jonathan Ho and Tim Salimans. 2021. Classifier-Free Diffusion Guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*. https://openreview.net/forum?id=qw8AKxfYbI

[19] Han Huang, Leilei Sun, Bowen Du, and Weifeng Lv. 2023. Conditional diffusion based on discrete graph structures for molecular graph generation. *arXiv preprint arXiv:2301.00427* (2023).

[20] Moksh Jain, Emmanuel Bengio, Alex Hernandez-Garcia, Jarrid Rector-Brooks, Bonaventure FP Dossou, Chanakya Ajit Ekbote, Jie Fu, Tianyu Zhang, Michael Kilgour, Dinghuai Zhang, et al. 2022. Biological sequence design with gflownets. In *International Conference on Machine Learning*. PMLR, 9786–9801.

[21] Bowen Jing, Gabriele Corso, Renato Berlinghieri, and Tommi Jaakkola. 2022. Subspace Diffusion Generative Models. In *ECCV*.

[22] Jaehyeong Jo, Seul Lee, and Sung Ju Hwang. 2022. Score-based generative modeling of graphs via the system of stochastic differential equations. In *International Conference on Machine Learning*. PMLR, 10362–10383.

[23] Jaehyeong Jo, Seul Lee, and Sung Ju Hwang. 2022. Score-based Generative Modeling of Graphs via the System of Stochastic Differential Equations. *arXiv preprint arXiv:2202.02514* (2022).

[24] Lingkai Kong, Jiaming Cui, Haotian Sun, Yuchen Zhuang, B. Aditya Prakash, and Chao Zhang. 2023. Autoregressive Diffusion Model for Graph Generation. https://openreview.net/forum?id=98J48HZXxd5

[25] Greg Landrum et al. 2016. RDKit: Open-Source Cheminformatics Software. (2016). http://www.rdkit.org/

[26] Haoyang Li, Xin Wang, and Wenwu Zhu. 2023. Curriculum Graph Machine Learning: A Survey. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, Edith Elkind (Ed.). International Joint Conferences on Artificial Intelligence Organization, 6674–6682. https://doi.org/10.24963/ijcai.2023/748 Survey Track.

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

[28] Huafeng Liu, Liping Jing, Jingxuan Wen, Pengyu Xu, Jiaqi Wang, Jian Yu, and Michael K Ng. 2021. Interpretable Deep Generative Recommendation Models. *J. Mach. Learn. Res.* 22 (2021), 202–1.

[29] Jenny Liu, Aviral Kumar, Jimmy Ba, Jamie Kiros, and Kevin Swersky. 2019. Graph normalizing flows. In *NeurIPS*.

[30] Meng Liu, Keqiang Yan, Bora Oztekin, and Shuiwang Ji. 2021. GraphEBM: Molecular Graph Generation with Energy-Based Models. In *Energy Based Models Workshop - ICLR 2021*. https://openreview.net/forum?id=Gc51PtL_zYw

[31] Qi Liu, Miltiadis Allamanis, Marc Brockschmidt, and Alexander Gaunt. 2018. Constrained graph variational autoencoders for molecule design. *Advances in neural information processing systems* 31 (2018).

[32] Yue Liu, Xihong Yang, Sihang Zhou, Xinwang Liu, Zhen Wang, Ke Liang, Wenxuan Tu, Liang Li, Jingcan Duan, and Cancan Chen. 2023. Hard Sample Aware Network for Contrastive Deep Graph Clustering. In *Proc. of AAAI*.

[33] T. Luo, Z. Mo, and S. Pan. 2023. Fast Graph Generation via Spectral Diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 01 (dec 2023), 1–12. https://doi.org/10.1109/TPAMI.2023.3344758

[34] Tianze Luo, Zhanfeng Mo, and Sinno Jialin Pan. 2023. Fast graph generation via spectral diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023).

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

[36] Tengfei Ma, Jie Chen, and Cao Xiao. 2018. Constrained generation of semantically valid graphs via regularizing variational autoencoders. In *NeurIPS*.

[37] Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. 2022. Trajectory balance: Improved credit assignment in GFlowNets. In *Advances in Neural Information Processing Systems*, Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (Eds.). https://openreview.net/forum?id=5btWTw1vcw1

[38] Christopher Morris, Nils M Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. 2020. Tudataset: A collection of benchmark datasets for learning with graphs. *arXiv preprint arXiv:2007.08663* (2020).

[39] Andrei Cristian Nica, Moksh Jain, Emmanuel Bengio, Cheng-Hao Liu, Maksym Korablyov, Michael M Bronstein, and Yoshua Bengio. 2022. Evaluating generalization in gflownets for molecule design. In *ICLR2022 Machine Learning for Drug Discovery*.

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

[41] Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, and Stefano Ermon. 2020. Permutation invariant graph generation via score-based generative modeling. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 4474–4484.

[42] Yassaman Ommi, Matin Yousefabadi, Faezeh Faez, Amirmojtaba Sabour, Mahdieh Soleymani Baghshah, and Hamid R. Rabiee. 2022. CCGG: A Deep Autoregressive Model for Class-Conditional Graph Generation. In *Companion Proceedings of the Web Conference 2022* (Virtual Event, Lyon, France) *(WWW '22)*. Association for Computing Machinery, New York, NY, USA, 1092–1098. https://doi.org/10.1145/3487553.3524721

[43] Ling Pan, Dinghuai Zhang, Aaron Courville, Longbo Huang, and Yoshua Bengio. 2023. Generative Augmented Flow Networks. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=urF_CBK5XC0

[44] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. *PyTorch: an imperative style, high-performance deep learning library.* Curran Associates Inc., Red Hook, NY, USA.

[45] Kristina Preuer, Philipp Renz, Thomas Unterthiner, Sepp Hochreiter, and Gunter Klambauer. 2018. Fréchet ChemNet distance: a metric for generative models for molecules in drug discovery. *Journal of chemical information and modeling* 58, 9 (2018), 1736–1741.

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

[47] Ida Schomburg, Antje Chang, Christian Ebeling, Marion Gremse, Christian Heldt, Gregor Huhn, and Dietmar Schomburg. 2004. BRENDA, the enzyme database: updates and major new developments. *Nucleic acids research* 32, suppl_1 (2004),

D431–D433.

[48] Chence Shi, Minkai Xu, Zhaocheng Zhu, Weinan Zhang, Ming Zhang, and Jian Tang. 2020. Graphaf: a flow-based autoregressive model for molecular graph generation. *arXiv preprint arXiv:2001.09382* (2020).

[49] Jianbo Shi and Jitendra Malik. 1997. Normalized Cuts and Image Segmentation. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97) (CVPR '97)*. IEEE Computer Society, USA, 731.

[50] Martin Simonovsky and Nikos Komodakis. 2018. Graphvae: Towards generation of small graphs using variational autoencoders. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27*. Springer, 412–422.

[51] Petru Soviany, Radu Tudor Ionescu, Paolo Rota, and Nicu Sebe. 2022. Curriculum Learning: A Survey. *Int. J. Comput. Vision* 130, 6 (jun 2022), 1526–1565. https://doi.org/10.1007/s11263-022-01611-x

[52] Nidhi Vakil and Hadi Amiri. 2023. Curriculum Learning for Graph Neural Networks: A Multiview Competence-based Approach. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, Toronto, Canada, 7036–7051. https://doi.org/10.18653/v1/2023.acl-long.389

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

[54] Hao Wang, Tong Xu, Qi Liu, Defu Lian, Enhong Chen, Dongfang Du, Han Wu, and Wen Su. 2019. MCNE: an end-to-end framework for learning multiple conditional network representations of social network. In *KDD*. 1064–1072.

[55] Hui Wang, Kun Zhou, Xin Zhao, Jingyuan Wang, and Ji-Rong Wen. 2023. Curriculum Pre-training Heterogeneous Subgraph Transformer for Top-N Recommendation. *ACM Trans. Inf. Syst.* 41, 1, Article 19 (jan 2023), 28 pages. https://doi.org/10.1145/3528667

[56] Carl Yang, Peiye Zhuang, Wenhan Shi, Alan Luu, and Li Pan. 2019. Conditional structure generation through graph variational generative adversarial nets. In *NeurIPS*.

[57] Yiding Yang, Zunlei Feng, Mingli Song, and Xinchao Wang. 2020. Factorizable graph convolutional networks. In *NeurIPS*. 20286–20296.

[58] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (London, United Kingdom) *(KDD '18)*. Association for Computing Machinery, New York, NY, USA, 974–983. https://doi.org/10.1145/3219819.3219890

[59] Jiaxuan You, Rex Ying, Xiang Ren, William Hamilton, and Jure Leskovec. 2018. Graphrnn: Generating realistic graphs with deep auto-regressive models. In *ICML*. 5708–5717.

[60] Chengxi Zang and Fei Wang. 2020. MoFlow: an invertible flow model for generating molecular graphs. In *KDD*. 617–626.

[61] Hector Zenil, Narsis A Kiani, and Jesper Tegnér. 2015. Numerical investigation of graph spectra and information interpretability of eigenvalues. In *Bioinformatics and Biomedical Engineering: Third International Conference, IWBBIO 2015, Granada, Spain, April 15-17, 2015. Proceedings, Part II 3*. Springer, 395–405.

[62] Dinghuai Zhang, Ricky TQ Chen, Nikolay Malkin, and Yoshua Bengio. 2022. Unifying generative models with gflownets. *arXiv preprint arXiv:2209.02606* (2022).

[63] Dinghuai Zhang, Ricky T. Q. Chen, Nikolay Malkin, and Yoshua Bengio. 2022. Unifying Generative Models with GFlowNets. *ArXiv* abs/2209.02606 (2022).

[64] Dinghuai Zhang, Nikolay Malkin, Zhen Liu, Alexandra Volokhova, Aaron Courville, and Yoshua Bengio. 2022. Generative Flow Networks for Discrete Probabilistic Modeling. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (Eds.). PMLR, 26412–26428. https://proceedings.mlr.press/v162/zhang22v.html

[65] Dinghuai Zhang, Nikolay Malkin, Zhen Liu, Alexandra Volokhova, Aaron Courville, and Yoshua Bengio. 2022. Generative flow networks for discrete probabilistic modeling. In *International Conference on Machine Learning*. PMLR, 26412–26428.

[66] Dawei Zhou, Lecheng Zheng, Dongqi Fu, Jiawei Han, and Jingrui He. 2022. MentorGNN: Deriving Curriculum for Pre-Training GNNs. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (Atlanta, GA, USA) *(CIKM '22)*. Association for Computing Machinery, New York, NY, USA, 2721–2731. https://doi.org/10.1145/3511808.3557393

[67] Yanqiao Zhu, Yuanqi Du, Yinkai Wang, Yichen Xu, Jieyu Zhang, Qiang Liu, and Shu Wu. 2022. A survey on deep graph generation: Methods and applications. *arXiv preprint arXiv:2203.06714* (2022).

A EXPERIMENT DETAILS

In this work, all the experiments are conducted with PyTorch [44] on NVIDIA A100 40GB Tensor Core GPU. In general, we follow a similar graph generative model evaluation setting as introduced in [22, 59]. For each experiment, we first preprocess the dataset and split it into training and testing datasets. For generic graph datasets, e.g. AIDS, Enzymes, and Synthie, we inherit the original graph labels. For the QM9 dataset, we impose graph labels on each molecule according to their chemical properties. Specifically, we use α-isotropic polarizability, $\Delta\epsilon$-the gap between ϵ_{HOMO} and ϵ_{LOMO}, and μ-dipole moment as the category indicators. For isotropic polarizability, we categorize molecules with $\alpha \leq 78$ to the first category, and the ones with $\alpha > 78$ to the second category. For the gap between ϵ_{HOMO} and ϵ_{LOMO}, we label molecules with $\Delta\epsilon \leq 6$ the first category, molecules with $6 < \Delta\epsilon \leq 8$ as the second category, and molecules with $\Delta\epsilon > 8$ as the third category. For dipole moment, we set molecules with $\mu \leq 3$ to category one and $\mu > 3$ to the other category.

After training the graph generative model on the training data to convergence, we employ it to synthesize the same number of samples as in the testing dataset. Following [22], we assess the distribution discrepancy between the synthesized dataset and the evaluation datasets by computing the MMD w.r.t to multiple graph statistics. The performance of a generative model is evaluated by the average of the MMD scores of these graph statistics. A lower averaged MMD implies a smaller gap between the synthesized dataset and the evaluation dataset, indicating that the trained generative model has a higher generation quality. For generic graph datasets shown in Section 5.1 and Section 5.3, including Community-small, Enzymes, Grid, AIDS, and Synthie, we employ three graph statistics, including the degree, clustering coefficient, and orbit. For the QM9 molecular dataset shown in Section 5.2, while we assess the gap between the generated dataset and evaluation dataset via the FCD, NSPDK-MMD, we further evaluate the quality of the generated molecules based on its chemical validity, uniqueness, and novelty. For conditional generations, the MMD scores are first calculated for each conditioned dataset segment, i.e. the data with the sample graph labels. Then, the MMD scores are weighted and summed according to the marginal distribution of graph labels. At each experiment setting, we report the average of 3 different runs for 5 independently trained models. We exactly follow the dataset pre-processing and post-processing schemes in [22]. The model hyperparameters are shown in Table 7. We also report the full experimental results of conditional molecular generation in Table 8.

B DETAILS OF AVERAGE MUTUAL INFORMATION ESTIMATION OF GENERATION CURRICULA

Our goal is to estimate the average mutual information of different graph generation curricula w.r.t the graph label distribution. To this end, we estimate the mutual information $\hat{I}(\mathbf{A}_k, y)$ of each intermediate step $k \in [n]$ via Mutual Information Neural Estimation [3] and then we calculate the mean of each estimated score. To reduce excessive computation burden, did not traverse all the possible intermediate checkpoints. Instead, we conduct MINE for 10 intermediate steps, i.e. $k \in \{\lfloor 0.1n \rfloor, ..., \lfloor 0.9n \rfloor, n\}$. Specifically, for each graph data, we leave the node feature matrix untainted and we conduct eigendecomposition on its adjacency matrix $\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$, and calculate the k-th intermediate step as $\mathbf{A}_k = \mathbf{U}\ \text{diag}(\lambda_{(1)}, ..., \lambda_{(k)}, 0, ..., 0)\ \mathbf{U}^\top$. Then, we eliminate the self loops of $\mathbf{A}_k$ and we suppress the edges with values lower than 10^{-5}. For random curricula, $\lambda_{(1)}, ..., \lambda_{(n)}$ is a random permutation of the original eigenvalues, while it is sorted by magnitude accordingly for the "low-to-high" generation curricula. Moreover, we augment the graph label y with a dense embedding module with the same dimension as the node feature dimension. We first feed $(\mathbf{A}_k, \mathbf{X})$ into a 2-layer GCN with hidden dimension equal to the feature dimension. Then, we concatenate it with the dense label embedding, and feed it into the MINE estimator, which is a 4-layer-MLP with hidden dimension equals to the node feature dimension. All the activation functions are set to be ReLU. Following the hyperparameters in [3], we train the MINE estimator on the augmented dataset for 1000 epochs with batch size 128. We train the MINE estimator with an Adam optimizer with learning rate 0.0001. We report the largest mutual information estimation score along the MINE training procedure. We estimate the mutual information score for 30 randomly generated curricula in total. For each setting, the MINE training process is repeated 3 times and we report the average MINE score.

Table 7: Hyperparameters of GPrinFlowNet used in each experiment.

	Hyperparameter	Conditional Generation				Unconditional Generation		
		AIDS	Enzymes	Synthie	QM9	Community-small	Enzymes	Grid
$P_F^{\Lambda}, P_B^{\Lambda}$	Number of GCN layers	5	7	5	5	4	5	5
	Number of MLP layers	3	2	3	3	4	3	3
	Hidden dimension	16	32	16	16	32	32	32
	Label embedding dimension	16	32	16	16	32	32	32
$P_F^{\mathbf{X}}, P_B^{\mathbf{X}}$	Number of GCN layers	5	7	5	5	4	5	5
	Number of MLP layers	3	2	3	3	4	3	3
	Hidden dimension	16	32	16	16	32	32	32
	Label embedding dimension	16	32	16	16	32	32	32
Train	Epochs	5000	5000	5000	1000	5000	5000	5000
	Batch size	64	32	64	2048	128	32	8
	Exponential Model Averaging	0.999	0.999	0.999	0.999	0.999	0.999	0.999
	Optimizer	Adam	Adam	Adam	Adam	Adam	Adam	Adam
	Learning rate	0.01	0.01	0.01	0.005	0.01	0.01	0.01
	Weight decay	0.00001	0.0001	0.00001	0.0001	0.00001	0.0001	0.0001
	Gradient norm clipping	1	1	1	1	1	1	1

Table 8: Full experiment results of conditional molecular generation on QM9 dataset. For each molecular sample, we assign three class labels according to its $\Delta\epsilon$-Gap between ϵ_{HOMO} and ϵ_{LUMO} (top), α-isotropic polarizability (middle), and the μ-dipole moment (bottom). The best results in the first three metrics are highlighted in bold.

$\epsilon_{\text{HOMO}} - \epsilon_{\text{LUMO}}$	Method	VALID w/o check (%) ↑	NSPDK ↓	FCD ↓	VALID (%) ↑	UNIQUE (%) ↑	*NOVEL (%)*
Autoreg.	GraphAF	67.72	0.059	10.423	100.00	94.10	88.17
	GraphAF+FC	74.37	0.053	10.536	100.00	88.14	86.48
	GraphDF	82.69	0.108	14.315	100.00	97.31	98.11
	GraphDF+FC	93.74	0.121	14.846	100.00	98.79	98.20
One-shot	MoFlow	91.95	0.059	8.645	100.00	98.47	94.19
	EDP-GNN	47.30	0.032	5.642	100.00	99.69	87.82
	GraphEBM	8.13	0.096	10.404	100.00	97.61	96.27
	GDSS	95.20	0.028	5.417	100.00	98.48	86.94
	CDGS	99.41	0.021	3.326	100.00	96.79	69.73
	GPrinFlowNet (Ours)	**99.72**	**0.012**	**2.798**	100.00	98.87	94.71

α-isotropic polarizability	Method	VALID w/o check (%) ↑	NSPDK ↓	FCD ↓	VALID (%) ↑	UNIQUE (%) ↑	*NOVEL (%)*
Autoreg.	GraphAF	67.47	0.063	11.057	100.00	94.51	88.63
	GraphAF+FC	74.17	0.055	11.147	100.00	88.64	86.59
	GraphDF	82.89	0.117	14.781	100.00	97.62	98.10
	GraphDF+FC	93.48	0.134	14.482	100.00	98.58	98.54
One-shot	MoFlow	91.12	0.064	8.793	100.00	98.65	94.72
	EDP-GNN	47.74	0.037	5.884	100.00	99.25	86.58
	GraphEBM	8.03	0.104	10.527	100.00	97.90	97.01
	GDSS	95.58	0.029	5.863	100.00	98.46	86.27
	CDGS	99.44	0.023	3.741	100.00	96.83	69.62
	GPrinFlowNet (Ours)	**99.74**	**0.013**	**2.925**	100.00	98.85	94.72

μ-dipole moment	Method	VALID w/o check (%) ↑	NSPDK ↓	FCD ↓	VALID (%) ↑	UNIQUE (%) ↑	*NOVEL (%)*
Autoreg.	GraphAF	67.48	0.049	9.372	100.00	94.51	88.83
	GraphAF+FC	74.72	0.053	9.248	100.00	88.64	86.59
	GraphDF	82.47	0.094	13.489	100.00	97.62	98.10
	GraphDF+FC	93.31	0.114	13.476	100.00	98.58	98.54
One-shot	MoFlow	91.58	0.053	8.024	100.00	98.65	94.72
	EDP-GNN	47.72	0.030	5.081	100.00	99.25	86.58
	GraphEBM	8.91	0.087	9.970	100.00	97.90	97.01
	GDSS	95.76	0.022	5.047	100.00	98.46	86.27
	CDGS	99.17	0.017	3.024	100.00	96.83	69.62
	GPrinFlowNet (Ours)	**99.79**	**0.011**	**2.627**	100.00	98.64	93.75