

Critical Iterative Denoising

A Discrete Generative Model Applied to Graphs

Yoann Boget^{1,2} Alexandros Kalousis¹

Abstract

Discrete Diffusion and Flow Matching models have significantly advanced generative modeling for discrete structures, including graphs. However, the time dependencies in the noising process of these models lead to error accumulation and propagation during the backward process. This issue, particularly pronounced in mask diffusion, is a known limitation in sequence modeling and, as we demonstrate, also impacts discrete diffusion models for graphs.

To address this problem, we propose a novel framework called *Iterative Denoising*, which simplifies discrete diffusion and circumvents the issue by assuming conditional independence across time. Additionally, we enhance our model by incorporating a *Critic*, which during generation selectively retains or corrupts elements in an instance based on their likelihood under the data distribution.

Our empirical evaluations demonstrate that the proposed method significantly outperforms existing discrete diffusion baselines in graph generation tasks.

1. Introduction

Denoising models, such as Discrete Diffusion and Discrete Flow Matching, have significantly advanced generative modeling for discrete structures (Austin et al., 2021; Campbell et al., 2022; 2024; Gat et al., 2024), including graphs (Hae-feli et al., 2022; Vignac et al., 2023). Despite their success, these models suffer from an important limitation caused by the time dependency in their noising and denoising processes. Errors introduced early in the process accumulate

¹Geneva School for Business administration HES-SO
²University of Geneva. Correspondence to: Yoann Boget <yoann.boget@hesge>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Copyright 2025 by the author(s).

and propagate, degrading generative performance. This issue is particularly pronounced and apparent in mask diffusion, where the noising process progressively masks elements of an instance. This challenge has been identified in discrete sequence modeling, and solutions such as *corrector sampling* have been proposed to mitigate it.

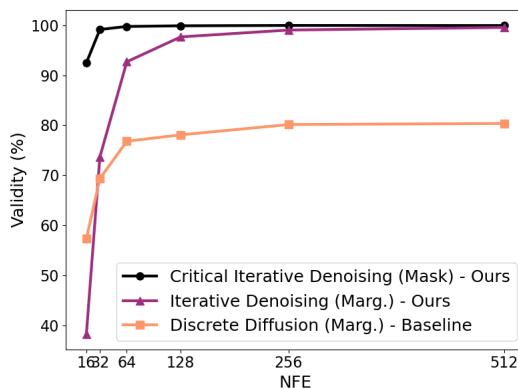


Figure 1: Validity rate (without correction) of generated molecules trained on Zinc250k as a function of the Number of Function Evaluations (NFE) for three models: Discrete Diffusion (baseline), Iterative Denoising (ours), and Critical Iterative Denoising (ours).

In graph modeling, the marginal distribution across node and edge attributes has proven to be a more suitable noise distribution than the mask distribution, as corroborated by our experiments. However, we show both theoretically and empirically that the issue of error propagation due to time dependencies also affects discrete diffusion and flow matching models in graph generation, where the noise corresponds to the marginal distributions.

To address this challenge, we introduce a novel framework called *Iterative Denoising*, which simplifies discrete diffusion by assuming conditionally independent noise across time. By removing the direct dependency on partially denoised instances from the previous step our framework facilitates error correction and substantially improves generative performance.

Furthermore, we show that our model can be interpreted as selectively corrupting certain elements while leaving others unmodified. Building on this insight, we introduce a *Critic*, which modulates the corruption probability of each element based on the element’s likelihood under the data distribution. We theoretically motivate our approach and we empirically show that the method further enhances models’ performances.

Our empirical evaluations highlight the effectiveness of the proposed framework, demonstrating superior performance over existing discrete diffusion baselines in graph generation tasks. Notably, our method tackles the challenge of generating valid molecular structures, achieving reliable performance for molecules. In Figure 1 we see that our method needs only a small number of denoising steps to reach almost 100% molecule validity, while the standard Discrete Diffusion tops at 80%.

Our code will be publicly released upon acceptance and is available to reviewers upon request.

2. Background and Related Works

Graph diffusion models are powerful methods for generating discrete graph structures, such as molecules. However, existing approaches, including Discrete Diffusion Models and Discrete Flow Matching (DFM), face challenges due to the temporal dependencies of the noising process. These dependencies introduce error propagation and accumulation, ultimately degrading generative performance. This section introduces core concepts, discusses these limitations, and motivates our approach.

2.1. Notation

We define a graph as a set of nodes and edges, denoted by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. A graph is represented by its adjacency matrix $\mathbf{A} \in [d_A + 1]^{n \times n}$, where $n = |\mathcal{V}|$, and $[d_A + 1] = \{1, \dots, d_A + 1\}$, with d_A representing the number of edge types, and the additional label corresponding to the absence of an edge. Node attributes, if present, are encoded as integers in an annotation vector $\mathbf{x} \in [d_X]^n$, where d_X is the number of node types.

To facilitate readability and simplify the method’s transferability to other discrete modalities, such as sequence modeling, we use z to denote a single element. In graph modeling, z may refer to either a node or an edge attribute when the formulation applies to both $x^{(i)}$ and $a^{(i,j)}$. The corresponding capital symbol Z represents the entire instance, such as a graph \mathcal{G} , \mathbf{z} . For simplicity and consistency with common practice, we use $p(x)$ to denote the Probability Mass Function (PMF) $P(X = x)$. The distribution of a data point element is represented via the Dirac delta distribution, denoted as $\delta_{z_1}(z)$, with all mass concentrated

on z_1 . We sometimes represent the univariate categorical distribution over the variable $p(z)$ as a vector \mathbf{z} , where the i -th component z_i denotes the probability that z belongs to the category indexed by i .

We refer to the *mask distribution* as the distribution where all probability mass is concentrated on an additional synthetic attribute labeled as `Mask`, denoted by $\delta_{\text{Mask}}(z)$. This distribution is sometimes referred to as the *absorbing-state* distribution (Austin et al., 2021). We define *mask diffusion* as the class of diffusion models that employ this distribution as noise.

We use q to refer to the noising distributions and p for the denoising ones.

2.2. Discrete Diffusion Models

In the forward process of diffusion models (Song & Ermon, 2019; Ho et al., 2020; Song et al., 2021), each element transitions independently from the data distribution $q_1(z) = \delta_{z_1}(z)$ at time $t = 1$ to a noise distribution $q_0(z)$ at time $t = 0$, which contains (or tends to contain) no information about the original data.

The forward noising process is expressed under a Markovian assumption as:

$$q_{t|1}(z | z_1) = \prod_{r \in \tau} q_{r|r+\Delta_t}(z | z_{r+\Delta_t}), \quad (1)$$

where $\tau = \{1 - \Delta_t, \dots, t + \Delta_t, t\}$.

The continuous case is described by taking $\lim_{\Delta_t \rightarrow 0} q_{t|1}(z | z_1)$, which transforms the product into a geometric integral. In discrete time, a forward diffusion step can be represented with transition probability matrices: $[\mathbf{Q}_t]_{i,j} = q(z_t = j | z_{t+\Delta_t} = i)$ (Vignac et al., 2023). A diffusion step then consists of sampling from:

$$q_{t-\Delta_t|t}(z | z_t) = \text{Cat}(z; \mathbf{z}_t^T \mathbf{Q}_{t-\Delta_t}).$$

Similarly, we have:

$$q_{t|1}(z | z_1) = \text{Cat}(z; \mathbf{z}_1^T \bar{\mathbf{Q}}_t),$$

where $\bar{\mathbf{Q}}_t = \prod_{r \in \tau} \mathbf{Q}_r$.

In this work, we focus on transition matrices of the form:

$$\mathbf{Q}_t = (1 - \beta_t)\mathbf{I} + \beta_t\mathbf{N}, \quad (2)$$

where \mathbf{I} is the identity matrix and \mathbf{N} represents the noise distribution q_0 . This formulation includes uniform noise ($\mathbf{N} = \frac{1}{d_Z} \mathbf{1}\mathbf{1}^T$), masking noise ($\mathbf{N} = \mathbf{1}\mathbf{e}_{\text{Mask}}^T$, where \mathbf{e}_{Mask} is the one-hot vector for the mask label), and marginal noise ($\mathbf{N} = \mathbf{1}\mathbf{m}^T$, where \mathbf{m} encodes the marginal distribution).

Under this formulation, $\bar{\mathbf{Q}}_t$ is computed efficiently as:

$$\bar{\mathbf{Q}}_t = \alpha_t \mathbf{I} + (1 - \alpha_t) \mathbf{N}, \quad (3)$$

where $\alpha_t = \prod_{r \in \tau} (1 - \beta_r)$.

By learning a backward process $p_{s|t}^\theta(z | Z_t)$ for $s = t + \Delta t$ and sampling iteratively from it, we generate data samples from the noise distribution.

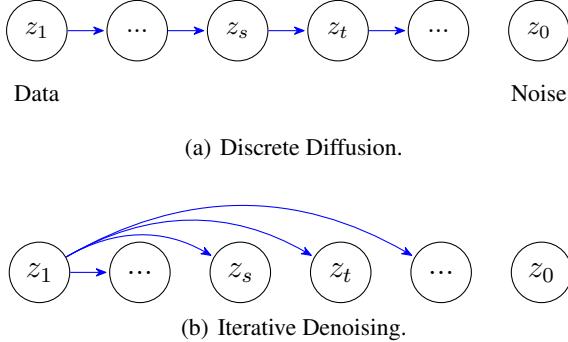


Figure 2: Noising Graphical Models in (a) Discrete Diffusion: $z_t \sim q_{t|s}(z | z_s)$; and (b) Iterative Denoising: $z_t \sim q_{t|1}(z | z_1)$

2.3. Temporal Dependencies and Compounding Error

At the beginning of the backward process, Z_t contains limited information, resulting in a high-entropy distribution $p_{s|t}^\theta(z | Z_t)$, from which the elements z_s of Z_s are independently sampled. At time s , the joint distribution from the backward process:

$$p_{s|t}^\theta(Z | Z_t) = \prod_{z \in Z} p_{s|t}^\theta(z | Z_t)$$

may deviate significantly from the forward distribution due to early-stage unmasking errors.

In the case of mask diffusion, unmasked elements cannot be masked again, as the conditional distribution collapses to a Dirac delta:

$$p_{s|t}^\theta(z | Z_t) = \delta_{z_t}(z), \quad \forall s \geq t \quad \text{if } z_t \neq \text{Mask}. \quad (4)$$

This leads to error accumulation, a phenomenon known as the *compounding denoising error* (Lezama et al., 2023).

The compounding denoising error is particularly pronounced in mask diffusion. It is somewhat less severe with uniform or marginal noise distributions, as all elements can change at each denoising step, allowing for corrections. However, due to the direct dependence of z_s on z_t , see graphical model of Discrete Diffusion in Figure 3 (a), the probability of correction is low, and the risk of error propagation and accumulation remains important. Specifically, Gat et al. (2024) have shown that a denoising step in discrete diffusion can be expressed as:

$$p_{t+\Delta_t|t}(z | Z_t) = \delta_{z_t}(z) + \Delta_t \frac{\dot{\alpha}_t}{1 - \alpha_t} [p_{1|t}(z | Z_t) - \delta_{z_t}(z)], \quad (5)$$

where $\dot{\alpha}_t$ is the α_t time-derivative. We interpret the second term on the right-hand side as the probability of modifying an element in a denoising step of size Δ_t . Notably, the scaling factor $\Delta_t \frac{\dot{\alpha}_t}{1 - \alpha_t}$ imposes a strong constraint on the probability of modifying an element. Consequently, the *compounding denoising error* issue also affects other common noise distributions, such as the uniform and marginal distributions.

To address this issue, works have adopted a discrete version of *Predictor-Corrector* sampling (Campbell et al. (2024); Lezama et al. (2023)). In the discrete case, predictor-corrector sampling methods are equivalent to simultaneously performing forward and backward steps. Recently, Gat et al. (2024) generalized the *corrector sampling* strategy, integrating it in the Flow Matching Framework, and Zhao et al. (2024) proposed informed correctors. Our approach circumvents the need for such correctors or additional sampling steps.

2.4. Motivation

We argue that the strong dependence on previous time steps in existing models hinders the generative performance of discrete diffusion models for graph generation. These models still struggle to consistently generate graphs with desired structural properties, such as valid molecular graphs or planar graphs.

In this work, we propose a simple yet effective framework that overcomes compounding denoising errors by removing temporal dependencies in the noising process. Our Iterative Denoising approach (Sec. 3) assumes conditionally independent noise distributions over time, alleviating the problem of error accumulation.

Additionally, we introduce a Critic-Guided Sampling procedure (Sec. 4), which prioritizes renoising elements with lower probabilities under the data distribution, further improving generative performance, particularly in molecular graph generation.

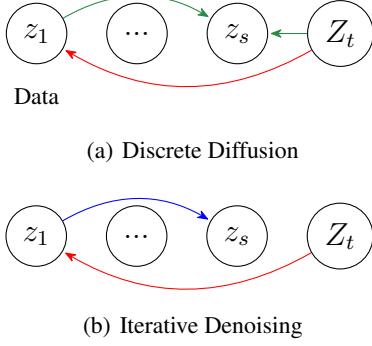


Figure 3: Denoising Graphical Model for a single denoising step in (a) Discrete Diffusion: $\hat{z}_s \sim p_{s|t}(z | Z_t) = \sum_{z_1} p_{s|1,t}(z|z_1, z_t) p_{1|t}(z_1|Z_t)$; and in (b) Iterative Denoising $\hat{z}_s \sim p_{s|t}(z | Z_t) = \sum_{z_1} q_{s|1}(z|z_1) p_{1|t}(z_1|Z_t)$.

3. The Iterative Denoising Model

In this section, we introduce Iterative Denoising (ID), a simple yet effective generative framework for modeling discrete-state graphs. This approach simplifies existing denoising models, such as DDM and DFM, while enhancing generative performance. ID serves as the foundation for Critical Iterative Denoising (CID) which further improves the generative performance.

Similar to DDM, ID consists of both noising and denoising operations. The generative sampling procedure involves iteratively applying small denoising steps, enabling effective graph generation.

Notably, if we have access to a pre-trained DDM or DFM denoiser, i.e., a parameterized model predicting $p_{1|t}^\theta(z|Z_t)$, the ID method is training free. In the following, we present the method in details.

3.1. Noising

We define a noising process that independently acts on each element of an instance as follows:

$$q_{t|1}(z|z_1) = \alpha_t \delta_{z_1}(z) + (1 - \alpha_t) q_0(z), \quad (6)$$

where $q_0(z)$ represents the noise distribution, $\delta_{z_1}(z)$ is the data distribution, and α_t is a non-decreasing scheduling parameter in $[0, 1]$. The scheduler α_t , which defines the noise level, depends on a continuous time parameter, which also takes values in $[0, 1]$. Note that we adopt the notation from DFM, but Equation 6 is equivalent to the $q_{t|1}(z|z_1) = \text{Cat}(z; z_1^T \bar{Q}_t)$ from DDM seen in Section 2.2.

Unlike standard diffusion models, where noise distributions are computed via a Markov chain (Eq. 1), our noisy distributions are directly parameterized as a mixture of the

noise and the data distribution. Thus we do not assume any time dependency in the noising process other than the dependency on the original data point z_1 . This assumption introduces a significant simplification over DDM, leading to beneficial implications for the ID denoising process. In figure 2 we juxtapose the graphical models of our method and DDM at the noising phase. More formally we assume conditional independence of the noisy distributions across time given z_1 :

Assumption 3.1. The noisy distributions are conditionally independent such that:

$$q_{t|1}(z|z_1) = q_{t|1}(z|z_s, z_1) \quad \forall t \neq s. \quad (7)$$

As a result, our noising procedure is not a diffusion process. Under this assumption, the denoising phase does not suffer from error propagation and accumulation caused by the temporal dependencies inherent in standard DDM (see Section 3.2).

Despite this assumption, we note that the distribution at time t in our method is identical to that of discrete diffusion, as formalized in the following proposition:

Proposition 3.2. Given a noise distribution, a data point, and a scheduler, the distribution $q_{t|1}^{ID}(z|z_1)$ defined by our noising procedure (Equation 6) is equal to the distribution $q_{t|1}^{DDM}(z|z_1)$ obtained by the discrete forward process distribution defined in (Equation 1).

Proof. See Appendix A.1 \square

The proposition follows from the fact that our noising procedure corresponds to the close-form expression (Eq. 3) used in discrete diffusion to sample efficiently from $q_{t|1}(z|z_1)$. However, we make no assumption that the noisy distribution results from a diffusion process.

3.2. Denoising

Denoising progressively refines noisy instances by learning the distribution $p_{s|t}(z|Z_t)$, $s = t + \Delta t$. By leveraging Assumption 3.1, the denoising process is significantly simplified compared to discrete diffusion, as formalized in the following proposition:

Proposition 3.3. Given the noising process defined in Equation 6 and Assumption 3.1, the denoising process is expressed as:

$$p_{s|t}(z|Z_t) = \alpha_s p_{1|t}(z|Z_t) + (1 - \alpha_s) q_0(z), \quad (8)$$

Proof. See Appendix A.3. \square

We can factorize the denoising process as:

$$p_{s|t}(z|Z_t) = \sum_{z_1} q_{s|1}(z|z_1) p_{1|t}(z_1|Z_t). \quad (9)$$

We thus interpret the denoising as a two-step process: 1. Predict a clean instance from the noisy data, $p_{1|t}(z_1|Z_t)$, 2. Re-noise the predicted clean instance, $q_{s|1}(z|z_1)$. The main difference from the standard DDM model is that z_s in ID directly depends only on the predicted clean instance, unlike the DDM model in which it directly depends on both the predicted clean instance and z_t . In Figure 3 we juxtapose the denoising graphical models of DDM and ID. The fact that z_s is generated only from the predicted clean instance acts as a barrier to error propagation. This factorization will also prove useful in Section 4.

At inference, we define the Number of Function Evaluations (NFE), denoted as T , as a hyperparameter. The time step size is set to $\Delta_T = 1/T$. The whole denoising process consists of iterating T times the denoising step defined in Equation 8.

3.2.1. CORRECTOR SAMPLING INTERPRETATION

The denoising process can also be viewed through the lens of discrete diffusion corrector sampling. Defining a discrete diffusion corrector sampling step as applying $k\Delta_t$ backward steps, and $(k-1)\Delta_t$ forward steps, and noticing that the maximal corrector step is such that $1-k\Delta_t=1$, we state the following proposition:

Proposition 3.4. *An Iterative Denoising step, as described in Equation 8, is equivalent to a discrete diffusion corrector sampling step with maximal corrector step size.*

Proof. See Appendix A.2. □

Consequently, the denoising process inherits the properties of the backward process in discrete diffusion; by iteratively sampling $p_{s|t}(z|z_t)$ with sufficiently small Δ_t , the denoising process gradually transitions from the noise distribution to the data distribution.

3.3. Parametrization and Learning

Similar to discrete diffusion, the conditional probability $p_{1|t}(z_1|Z_t)$ is intractable. We model this distribution by parametrizing a neural network $f_\theta(Z_t, \alpha_t)$ called the denoiser. We use Graph Neural Networks (GNNs) to enforce equivariance to node permutations, and thereby preserving the essential structural invariance of graphs.

Since the training objective is identical to that of DDM and DFM, we can adopt any training criterion proposed for these models to train $f_\theta(Z_t, \alpha_t)$. Similar to Digress (Vignac et al., 2023), we minimize the weighted negative log-likelihood in our experiments:

$$\begin{aligned} \mathcal{L} = & \mathbb{E}_{\mathcal{G} \sim p_{\text{data}}, t \sim \mathcal{U}(0,1), \mathcal{G}_t \sim (q_{t|1}(x_1^{(i)}|x_t^{(i)}), q_{t|1}(e_t^{(i,j)}|e_1^{(i,j)}))} \\ & \left[\gamma \sum_{x_1^{(i)}} \left[-\log(p_\theta(x_1^{(i)}|\mathcal{G}_t)) \right] \right. \\ & \left. + (1-\gamma) \sum_{e_1^{(i,j)}} \left[-\log(p_\theta(e_1^{(i,j)}|\mathcal{G}_t)) \right] \right], \end{aligned} \quad (10)$$

where γ is a weighting factor between nodes and edges. We use $\gamma = n/(n+m)$. We can leverage any pre-trained DDM model for inference in the Iterative Denoising (ID) framework.

4. Critical Denoising

This section introduces *Critical Iterative Denoising* (CID), a method designed to improve *Iterative Denoising*.

As discussed in Section 3, Mask ID can be viewed as a sequence of denoising steps comprising: (1) unmasking elements of an instance based on the denoiser’s prediction and (2) re-masking a random subset of the instance’s elements.

However, in Mask ID, all elements are re-masked with the same probability. This approach is suboptimal, as not all elements are equally likely under the data distribution after the unmasking step.

CID addresses this limitation by dynamically adjusting the re-noising probability during sampling. Specifically, denoised elements that are overrepresented under the data distribution are re-noised with a higher probability, while elements that are underrepresented are more likely to remain uncorrupted. Our objective here is twofold: (1) Reducing error propagation by resampling elements that are likely out of the data distribution, and (2) Accelerating inference by decreasing the number of function evaluations required during sampling.

Figure 1 illustrates that CID achieves a state-of-the-art validity rate on Zinc250k with only a few tens of denoising steps.

4.1. Preliminaries

Before proceeding to the description of the method, we will recast the definition/computation of the noise distributions by introducing a Bernoulli random variable a_t the mean of which is given by the α_t scheduling parameter in Equation 6; a_t indicate whether an element has been corrupted ($a_t = 0$) or not ($a_t = 1$). As with z , we use the capital letter A to denote the set of indicators for all elements in an instance. The resulting formulation allows us to inter-

pret the noising/noising process as selectively corrupting some elements while leaving others untouched. Through the random variables a_t , we can identify which are the z_t elements that have been noised. This identification would otherwise be impossible for any noise distribution sharing support with the data distribution (i.e., all practical noise distributions except the mask distribution).

Thanks to the random variable $a_t \sim p_{\alpha_t}(a) = \text{Bernoulli}(a; \alpha_t)$, we rewrite Equation 6 as:

$$q_t(z|z_1) = p_{\alpha_t}(a)\delta_{z_1}(z_t) + (1 - p_{\alpha_t}(a))q_0(z). \quad (11)$$

Similarly, we rewrite the denoising equation (Eq. 8) as:

$$p_{s|t}(z|Z_t) = p_{\alpha_t}(a)\delta_{z_t}(z) + (1 - p_{\alpha_t}(a))p_{1|t}^\theta(z|Z_t, A_t) \quad (12)$$

Intuitively, when the element is not corrupted ($a_t = 1$), it remains unchanged ($\delta_{z_t}(z)$); when $a_t = 0$ we denoise it using the denoising distribution $p_{1|t}^\theta(z|Z_t, A_t)$. We simplify notation as follows $p_{\text{data}}(z) := \delta_{z_t}(z)$ and $p_{\text{pred}}(z) := p_{1|t}^\theta(z|Z_t, A_t)$.

In principle, our method supports any noise distribution and produces pairs (z_t, a_t) , which indicate whether the element z_t was sampled from the noise distribution or corresponds to the original clean element. However, denoising using the pair (z_t, a_t) reduces denoising models with any noise distribution (e.g., marginal, uniform) to a mask denoising model.

To see this, we observe that the corruption indicator a_t acts as a mask, with $a_t = 0$ indicating a masked element. Defining the set of uncorrupted elements as $Z_t^a = \{z_t \in Z_t \mid a_t = 1\}$, and its complement as $Z_t^{\bar{a}}$, the masked denoiser can be expressed as $p_{1|t}^\theta(z \mid Z_t^a, A_t)$. Since $Z_t^{\bar{a}}$ carries no information about Z_1 , it follows that, for any noise distribution, $p_{1|t}^\theta(z \mid Z_t, A_t) = p_{1|t}^\theta(z \mid Z_t^a, A_t)$. As Critical Iterative Denoising (CID) explicitly requires A_t , and any noise distribution under this configuration collapses to the Mask Iterative Denoising (Mask ID) model, we always use Mask ID within CID.

4.2. Iterative Denoising with a Critic

We now describe the critic used to predict the corruption state of the graph elements during the denoising phase. Let $\hat{z}_{1|t} \sim p_{1|t}(z|Z_t, A_t)$ denote an element after the denoising step at time t . CID trains a Critic C that models $\hat{\alpha}_t = p_\phi(a|\hat{Z}_{1|t})$, which predicts the probability that \hat{z} comes from p_{data} or p_{pred} . The Critic is trained to minimize the negative log-likelihood of the $a_t^{(i)}$ corruption indicators produced at the noising phase (one per element and time step):

$$\mathcal{L}_\phi = -\mathbb{E}_{t, \hat{Z}_{1|t}} \sum_i \log p_\phi(a_t^{(i)} | \hat{Z}_{1|t}), \quad (13)$$

During inference, $\hat{\alpha}_t$ parametrizes the Bernoulli distribution $p_{\hat{\alpha}_t}(a_t) = \text{Bernoulli}(a; \hat{\alpha}_t)$ which determines the noising probability $1 - \hat{\alpha}_t$. It is important to remind that α_t is the scheduler's noise rate at time step t . So we interpret the output of the critic as an adaptive scheduling parameter setting the level of noise per element.

In the following theorem we characterise the optimal critic:

Theorem 4.1. *The optimal Critic C^* is:*

$$C^*(\hat{z}_{1|t}) = \frac{\alpha_t p_{\text{data}}(\hat{z}_{1|t})}{\alpha_t p_{\text{data}}(\hat{z}_{1|t}) + (1 - \alpha_t)p_{\text{pred}}(\hat{z}_{1|t})} \quad (14)$$

Proof. See Appendix A □

From Theorem 4.1, two lemmas follow:

Lemma 4.2. *If $p_{\text{data}}(\hat{z}_{1|t}) = p_{\text{pred}}(\hat{z}_{1|t})$, the optimal $\hat{\alpha}_t^*$ coincide with the true α_t , that is:*

$$p_{\text{data}}(\hat{z}_{1|t}) = p_{\text{pred}}(\hat{z}_{1|t}) \implies \hat{\alpha}_t^* = \alpha_t \quad (15)$$

Proof. The lemma follows directly from Theorem 4.1. □

Lemma 4.3. *If $p_{\text{data}}(\hat{z}_{1|t}) > p_{\text{pred}}(\hat{z}_{1|t})$, the optimal Critic's noising probability $\hat{\alpha}_t^*$ is smaller than the schedule's noising rate α_t , that is:*

$$p_{\text{data}}(\hat{z}_{1|t}) > p_{\text{pred}}(\hat{z}_{1|t}) \implies \alpha_t > \hat{\alpha}_t^*. \quad (16)$$

Conversely, if $p_{\text{data}}(\hat{z}_{1|t}) < p_{\text{pred}}(\hat{z}_{1|t})$, the optimal Critic's noising probability $\hat{\alpha}_t^$ is larger than the schedule's noising rate α_t :*

$$p_{\text{data}}(\hat{z}_{1|t}) < p_{\text{pred}}(\hat{z}_{1|t}) \implies \alpha_t < \hat{\alpha}_t^*. \quad (17)$$

Proof. See Appendix A.5 □

Additionally, if $p_{\text{pred}}(\hat{z}_{1|t}) = 0$ the element will not be masked ($\alpha_t = 0$). Conversely, if $p_{\text{data}}(\hat{z}_{1|t}) = 0$, the element will be masked with probability $\alpha_t = 1$.

Thanks to Theorem 4.1 and the following lemmas, the Critic steers the denoised element toward the data distribution, by re-noising with higher probability elements that are overrepresented under the data distribution, and leaving uncorrupted elements that are underrepresented.

4.3. Implementation and Sampling

Since $p_{\alpha_t}(a)$ does not depends on $\hat{Z}_{1|t}$, $\mathbb{E}_{\hat{Z}_{1|t}} p(a|\hat{Z}_{1|t}) = p_{\alpha_t}(a) = \alpha_t$, we actually cast the Critic as a predictor of the residual logit with respect to the true α_t :

$$p_\phi(a|\hat{Z}_{1|t}) = \sigma(f_\phi(\hat{Z}_{1|t}, \alpha_t) + \sigma^{-1}(\alpha_t)), \quad (18)$$

where σ is the sigmoid function and $f_\phi(\hat{Z}_{1|t}, \alpha_t)$ is a GNN. The Critic operates over the denoiser's outputs; we train it

post hoc with a fixed denoiser. We can leverage any denoiser implementing mask diffusion and no retraining is needed.

During inference, we are interested in $\hat{a}_{t+\Delta_t}$ rather than \hat{a}_t . We therefore use the approximation $\hat{a}_{t+\Delta_t} \approx \sigma(f_\phi(\hat{Z}_{1|t}, \alpha_t) + \sigma^{-1}(\alpha_{t+\Delta_t}))$.

In summary a denoising step involves: (1) sampling $\hat{Z}_{1|t}$ from the Denoiser, getting $\hat{\alpha}_{t+\Delta_t}$ from the Critic, masking back using Equation 6, replacing α_t with $\hat{\alpha}_{t+\Delta_t}$.

5. Evaluation

We evaluate our model on molecular and synthetic graph datasets.

For molecular data, we use the Qm9 and Zinc250k datasets. The Qm9 dataset contains 133,885 molecules with up to 9 atoms of 4 types, while the Zinc250k dataset consists of 250,000 molecular graphs with up to 38 heavy atoms of 9 types. For generic graphs, we run experiments on the Planar and the Stochastic Block Model (SBM) datasets. Both contain 200 unattributed graphs, with 64 nodes and up to 200 nodes, respectively. Visualizations of generated molecules and graphs are available in Appendix F.

Our objective is to evaluate our proposed method in direct comparison with Discrete Diffusion Models (DDM). To ensure a rigorous comparison, we train two denoisers, one using the marginal distribution and the other using the mask distribution. We use these denoisers to compare the two sampling procedures: Discrete Diffusion and Iterative Denoising. Additionally, we train a Critic associated with the model using the mask distribution. Specifically, the Marginal DDM and Marginal ID share the same denoiser, while the Mask DDM, Mask ID, and Mask CID also share the same denoiser (architecture and parameters).

This setup ensures a controlled and fair ablation study, isolating the impact of our iterative denoising approach while maintaining identical model architectures and training conditions across all comparisons. We provide the technical experimental details and implementation in Appendix D.

We also provide baselines results from various diffusion models; GDSS (Jo et al., 2022) is a continuous diffusion model, DruM (Jo et al., 2024) is a diffusion bridge model, and DiGress (Vignac et al., 2023) is a Discrete Diffusion model. DiGress uses the marginal distribution as noise. It differs from our Marginal DDM only by the network architecture, and hyperparameters. Results are taken from Jo et al. (2024). Importantly, these baseline results were obtained using 1000 function evaluation steps during generation, whereas we use 500 steps in our experiments.

5.1. Molecule Generation

We report the Frechet ChemNet Distance (FCD) (Preuer et al., 2018), which measures the similarity between generated molecules and real molecules in chemical space, as well as the Neighborhood Subgraph Pairwise Distance Kernel (NSPDK - NPK) (Costa & Grave, 2010), which evaluates the similarity of their graph structures. Additionally, we report the proportion of chemically valid molecules (validity) without any post-generation correction or resampling. The results are presented in Table 1. We highlight the best

Table 1: Results on Qm9 and Zinc250k datasets.

| MODEL | QM9 | | | ZINC250K | | |
|-----------|-----------------|------------------|------------------|-----------------|------------------|------------------|
| | VAL. \uparrow | NPK \downarrow | FCD \downarrow | VAL. \uparrow | NPK \downarrow | FCD \downarrow |
| GDSS | 95.72 | 3.3 | 2.90 | 97.01 | 19.5 | 14.65 |
| DRUM | 99.69 | 0.2 | 0.11 | 98.65 | 1.5 | 2.25 |
| DiGRESS | 98.19 | 0.3 | 0.10 | 94.99 | 2.1 | 3.48 |
| MARG. DDM | 95.73 | 1.92 | 1.09 | 80.40 | 12.96 | 8.50 |
| MASK DDM | 48.38 | 14.75 | 3.76 | 8.96 | 78.63 | 24.98 |
| MARG. ID | 99.67 | 1.04 | 0.50 | 99.50 | 2.06 | 2.01 |
| MASK ID | 96.43 | 1.40 | 1.80 | 93.85 | 11.08 | 9.05 |
| MASK CID | 99.92 | 1.40 | 1.76 | 99.97 | 2.26 | 3.46 |

results among our experiments. We also highlight the best baseline if better than our models. We generate 10,000 samples and compare them with 10,000 test samples. Results are averaged over 5 sampling runs. Standard deviations and additional metrics (e.g., uniqueness and novelty) are included in Appendix E.

We observe that our Iterative Denoising (ID) models systematically outperform their Discrete Diffusion (DDM) counterparts using the same denoiser on both datasets and this with both noise distributions, i.e. marginal and mask. Furthermore, the results confirm our analysis of compounding denoising error, particularly affecting mask-based discrete diffusion, which performs poorly across both datasets. ID and CID address the compounding error successfully. Our experiments confirm that the marginal distribution is a more effective noise distribution in graph modeling compared to the mask distribution. Even if the compounding denoising error is less pronounced with the marginal distribution, we still observe that our *Iterative Denoising* method outperforms consistently the marginal prior discrete diffusion baselines (DDM, Marg. DDM) by a large margin. Finally, we observe that the Critic further improves the generative performance. Notably, our Critical Iterative Denoising method reaches a high-level validity rate, with only a few invalid graphs, specifically on Zinc250k invalid molecules are 50 less frequent in our model than in the best baseline model (DruM).

5.2. Generic graphs

For generic graphs, we follow the evaluation procedure established by Martinkus et al. (2022) and used by Jo et al. (2022) and Jo et al. (2024), using an 80/20 train-test split, with 20% of the training data reserved for validation.

Table 2: Results on Planar and SBM datasets

| MODEL | PLANAR | | SBM | |
|-----------|---------------------|-------------------|---------------------|-------------------|
| | SPECT. \downarrow | V.U.N. \uparrow | SPECT. \downarrow | V.U.N. \uparrow |
| GDSS | 37.0 | 0 | 12.8 | 5 |
| DRUM | 6.2 | 90 | 5.0 | 85 |
| DiGRESS | 10.6 | 75 | 40.0 | 74 |
| MARG. DDM | 83.57 | 0.0 | 11.82 | 0.0 |
| MASK DDM | 84.44 | 0.0 | 11.38 | 0.0 |
| MARG. ID | 7.62 | 91.3 | 5.93 | 63.5 |
| MASK ID | 8.72 | 67.0 | 15.05 | 17.5 |
| MASK CID | 6.40 | 66.0 | 11.94 | 19.0 |

We evaluate graph similarity using the Spectral Maximum Mean Discrepancy (Spect.), which compares the spectra of the graph Laplacian between generated and real graphs. Unlike other metrics that rely on local structural properties, such as node degrees or clustering coefficients, the Spectral MMD captures graph structures at any level. Additionally, we report the rate of Valid, Unique, and Novel graphs (V.U.N.). For the Planar dataset, a valid graph must be both planar and connected. For the Stochastic Block Model dataset, validity indicates that the generated graph is likely to follow the block model distribution used to generate the training data (i.e., an intra-community edge density of 0.3 and an inter-community edge density of 0.005). Uniqueness represent the fraction of unique graphs among the generated samples, and novelty the fraction of unique graphs that do not appear in the training set. In practice, all generated graphs in our experiments were both unique and novel, meaning that the V.U.N. metric actually reports the validity rate. We present our results in Table 2. We report the average over 5 sampling comparing the test set and a generative sample batch of same size. Standard deviation and complementary results including MMDs based on degree, clustering coefficient, and orbits counts on Appendix E.

The analysis of general graphs aligns with our findings on molecular graphs, further reinforcing our conclusions. Our Iterative Denoising models consistently outperform their Discrete Diffusion counterparts, demonstrating their effectiveness across different graph types. Results also confirm that mask-based models are not well suited for graph generation. In particular, on the SBM dataset, their results are barely significant due to their performances .

While the Critic improves the spectral MMD on the Planar dataset, we observe that its contribution is less effective

for general graphs compared to molecular graphs. We hypothesize that this is due to deviations from the data distribution being less localized in generic graphs, making it more difficult to discriminate between graph elements. With the marginal distribution, our Iterative Denoising model presents results that are significantly better than the Discrete Diffusion baseline.

5.3. Ablation

We assess the impact of the Number of Function Evaluations (NFE)—i.e., the number of denoising steps during sampling—on model performance. Specifically, we investigate how the Critic influences the required NFE.

We generate graphs with varying NFEs in $\{16, 32, 64, 128, 256, 512\}$ and evaluate performance on the Zinc250k dataset for Marginal DDM, Marginal ID, and Mask CID. The validity rate is reported in Figure 1, while additional figures for other metrics and full numerical results are provided in Appendix E due to space limitation.

Our findings indicate that the Critic significantly reduces the required NFEs to achieve a high validity rate, reaching over 99% validity in just 32 steps. Furthermore, in the lowest-NFE regime (16 steps), Mask CID consistently outperforms all other models across all metrics, demonstrating its effectiveness in reducing the number of denoising steps.

6. Conclusion

Discrete diffusion models suffer from the Compounding Denoising Error issue, which leads to error accumulation during sampling and negatively affects their performance. In this paper, we address this issue by removing the time dependency in the noising process. The resulting method does not rely on a diffusion process. We further improve upon our method, and introduce a Critic, which steers the distribution of generated elements towards the data distribution, improving sample quality.

Our experimental results demonstrate the effectiveness of our method. Our Iterative Denoising model consistently outperforms its corresponding denoising model. Moreover, our Critic-based model systematically improves generative performance over the corresponding Mask Iterative Denoising model without the Critic.

Additionally, we show that Critical Iterative Denoising significantly reduces the Number of Function Evaluations (NFE) required for sampling, making the generation process more efficient. While this work focuses on graph modeling, future research should explore the applicability of our approach to other structured data domains.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

- Austin, J., Johnson, D. D., Ho, J., Tarlow, D., and van den Berg, R. Structured denoising diffusion models in discrete state-spaces. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 17981–17993. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/958c530554f78bcd8e97125b70e6973d-Paper.pdf.
- Bergmeister, A., Martinkus, K., Perraudin, N., and Wattenhofer, R. Efficient and scalable graph generation through iterative local expansion. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=2XkTz7gdpc>.
- Boget, Y., Gregorova, M., and Kalousis, A. Discrete graph auto-encoder. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL <https://openreview.net/forum?id=bZ80b0wb9d>.
- Boget, Y., Strasser, P., and Kalousis, A. Hierarchical equivariant graph generation, 2025. URL <https://openreview.net/forum?id=uEqOYXtn7f>.
- Campbell, A., Benton, J., De Bortoli, V., Rainforth, T., Deligiannidis, G., and Doucet, A. A continuous time framework for discrete denoising models. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 28266–28279. Curran Associates, Inc., 2022.
- Campbell, A., Yim, J., Barzilay, R., Rainforth, T., and Jaakkola, T. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design, 2024. URL <https://arxiv.org/abs/2402.04997>.
- Chen, X., He, J., Han, X., and Liu, L. Efficient and degree-guided graph generation via discrete diffusion modeling. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 4585–4610. PMLR, 23–29 Jul 2023. URL <https://proceedings.mlr.press/v202/chen23k.html>.
- Costa, F. and Grave, K. D. Fast neighborhood subgraph pairwise distance kernel. In Fürnkranz, J. and Joachims, T. (eds.), *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, June 21-24, 2010, Haifa, Israel, pp. 255–262. Omnipress, 2010. URL <https://icml.cc/Conferences/2010/papers/347.pdf>.
- Eijkelboom, F., Bartosh, G., Naesseth, C. A., Welling, M., and van de Meent, J.-W. Variational flow matching for graph generation, 2024. URL <https://arxiv.org/abs/2406.04843>.
- Gat, I., Remez, T., Shaul, N., Kreuk, F., Chen, R. T. Q., Synnaeve, G., Adi, Y., and Lipman, Y. Discrete flow matching. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL <https://openreview.net/forum?id=GTDKo3Sv9p>.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.
- Goyal, N., Jain, H. V., and Ranu, S. Graphgen: A scalable approach to domain-agnostic labeled graph generation. In Huang, Y., King, I., Liu, T., and van Steen, M. (eds.), *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pp. 1253–1263. ACM / IW3C2, 2020. doi: 10.1145/3366423.3380201. URL <https://doi.org/10.1145/3366423.3380201>.
- Gómez-Bombarelli, R., Wei, J. N., Duvenaud, D., Hernández-Lobato, J. M., Sánchez-Lengeling, B., Sheberla, D., Aguilera-Iparraguirre, J., Hirzel, T. D., Adams, R. P., and Aspuru-Guzik, A. Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. *ACS Central Science*, 4(2):268–276, February 2018. ISSN 2374-7943. URL <https://doi.org/10.1021/acscentsci.7b00572>. Publisher: American Chemical Society.
- Haefeli, K. K., Martinkus, K., Perraudin, N., and Wattenhofer, R. Diffusion models for graphs benefit from discrete state spaces. In *The First Learning on Graphs Conference*, 2022. URL <https://openreview.net/forum?id=CtsKBwhTMKg>.
- Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc.,

2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.
- Irwin, J. J., Sterling, T., Mysinger, M. M., Bolstad, E. S., and Coleman, R. G. Zinc: A free tool to discover chemistry for biology. *Journal of Chemical Information and Modeling*, 52(7):1757–1768, 2012. doi: 10.1021/ci3001277. URL <https://doi.org/10.1021/ci3001277>. PMID: 22587354.
- Jin, W., Barzilay, R., and Jaakkola, T. Junction tree variational autoencoder for molecular graph generation. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2323–2332. PMLR, 10–15 Jul 2018. URL <https://proceedings.mlr.press/v80/jin18a.html>.
- Jin, W., Barzilay, R., and Jaakkola, T. Hierarchical Generation of Molecular Graphs using Structural Motifs. In *37th International Conference on Machine Learning, ICML 2020*, volume PartF16814, pp. 4789–4798, 2020. ISBN 9781713821120. URL <https://github.com/wengong-jin/hgraph2graph>.
- Jo, J., Lee, S., and Hwang, S. J. Score-based Generative Modeling of Graphs via the System of Stochastic Differential Equations. *Proceedings of the 39th International Conference on Machine Learning*, 162:10362–10383, 2022. URL <https://github.com/harryjo97/GDSS>. [http://arxiv.org/abs/2202.02514](https://arxiv.org/abs/2202.02514).
- Jo, J., Kim, D., and Hwang, S. J. Graph generation with diffusion mixture. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 22371–22405. PMLR, 21–27 Jul 2024. URL <https://proceedings.mlr.press/v235/jo24b.html>.
- Karami, M. Higen: Hierarchical graph generative networks. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=KNvubydSB5>.
- Kong, L., Cui, J., Sun, H., Zhuang, Y., Prakash, B. A., and Zhang, C. Autoregressive diffusion model for graph generation. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 17391–17408. PMLR, 23–29 Jul 2023. URL <https://proceedings.mlr.press/v202/kong23b.html>.
- Krawczuk, I., Abranches, P., Loukas, A., and Cevher, V. {GG}-{gan}: A geometric graph generative adversarial network, 2021. URL <https://openreview.net/forum?id=qiAxL3Xqxlo>.
- Kusner, M. J., Paige, B., and Hernández-Lobato, J. M. Grammar Variational Autoencoder. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 1945–1954. PMLR, July 2017. URL <https://proceedings.mlr.press/v70/kusner17a.html>. ISSN: 2640-3498.
- Lezama, J., Salimans, T., Jiang, L., Chang, H., Ho, J., and Essa, I. Discrete predictor-corrector diffusion models for image synthesis. In *The Eleventh International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=VM8batVBWvg>.
- Liao, R., Li, Y., Song, Y., Wang, S., Hamilton, W., Duvenaud, D. K., Urtasun, R., and Zemel, R. Efficient Graph Generation with Graph Recurrent Attention Networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.
- Liu, J., Kumar, A., Ba, J., Kiros, J., and Swersky, K. Graph normalizing flows. In *Advances in Neural Information Processing Systems*, volume 32, 2019.
- Luo, Y., Yan, K., and Ji, S. GraphDF: A Discrete Flow Model for Molecular Graph Generation. *Proceedings of the 38th International Conference on Machine Learning*, 139:7192–7203, 2021. URL [http://arxiv.org/abs/2102.01189](https://arxiv.org/abs/2102.01189).
- Madhawa, K., Ishiguro, K., Nakago, K., and Abe, M. Graph-nvp: An invertible flow model for generating molecular graphs, 2019.
- Martinkus, K., Loukas, A., Perraudeau, N., and Wattenhofer, R. SPECTRE: Spectral conditioning helps to overcome the expressivity limits of one-shot graph generators. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 15159–15179. PMLR, 17–23 Jul 2022. URL <https://proceedings.mlr.press/v162/martinkus22a.html>.
- Nguyen, V. K., Boget, Y., Lavda, F., and Kalousis, A. GLAD: Improving latent graph generative modeling with simple quantization. In *ICML 2024 Workshop on Structured Probabilistic Inference & Generative Modeling*, 2024. URL <https://openreview.net/forum?id=aY1gdSolIV>.

- Nichol, A. Q. and Dhariwal, P. Improved denoising diffusion probabilistic models. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8162–8171. PMLR, 18–24 Jul 2021. URL <https://proceedings.mlr.press/v139/nichol21a.html>.
- Preuer, K., Renz, P., Unterthiner, T., Hochreiter, S., and Klambauer, G. Fréchet chemnet distance: A metric for generative models for molecules in drug discovery. *Journal of Chemical Information and Modeling*, 58(9):1736–1741, 2018. doi: 10.1021/acs.jcim.8b00234. PMID: 30118593.
- Qin, Y., Madeira, M., Thanou, D., and Frossard, P. Defog: Discrete flow matching for graph generation, 2024a. URL <https://arxiv.org/abs/2410.04263>.
- Qin, Y., Vignac, C., and Frossard, P. Sparse training of discrete diffusion models for graph generation, 2024b. URL <https://openreview.net/forum?id=oTRekADULK>.
- Shi, C., Xu, M., Zhu, Z., Zhang, W., Zhang, M., and Tang, J. Graphaf: a flow-based autoregressive model for molecular graph generation. In *International Conference on Learning Representations*, 2020. URL <https://openreview.net/forum?id=S1esMkHYPr>.
- Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. In Wallach, H., Larochelle, H., Beygelzimer, A., d’Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/3001ef257407d5a371a96dc947c7d93-Paper.pdf.
- Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=PxTIG12RRHS>.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Transformer: Attention is all you need. *Advances in Neural Information Processing Systems 30*, pp. 5998–6008, 2017. ISSN 10495258.
- Vignac, C., Krawczuk, I., Siraudin, A., Wang, B., Cevher, V., and Frossard, P. Digress: Discrete denoising diffusion for graph generation. In *The Eleventh International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=UaAD-Nu86WX>.
- Wu, Z., Ramsundar, B., Evan N Feinberg and, J. G., Geniesse, C., Pappu, A. S., Leswing, K., and Pande, V. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 2017. URL <https://pmc.ncbi.nlm.nih.gov/articles/PMC5868307/>.
- Xu, Z., Qiu, R., Chen, Y., Chen, H., Fan, X., Pan, M., Zeng, Z., Das, M., and Tong, H. Discrete-state continuous-time diffusion for graph generation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL <https://openreview.net/forum?id=YkSKZEhIYt>.
- Yang, C., Zhuang, P., Shi, W., Luu, A., and Li, P. Conditional Structure Generation through Graph Variational Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, volume 32, 2019.
- You, J., Ying, R., Ren, X., Hamilton, W., and Leskovec, J. GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 5708–5717. PMLR, July 2018. URL <https://proceedings.mlr.press/v80/you18a.html>. ISSN: 2640-3498.
- Zang, C. and Wang, F. MoFlow: An Invertible Flow Model for Generating Molecular Graphs. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 10:617–626, aug 2020. doi: 10.1145/3394486.3403104. URL <https://dl.acm.org/doi/10.1145/3394486.3403104>.
- Zhao, Y., Shi, J., Mackey, L., and Linderman, S. Informed correctors for discrete diffusion models, 2024. URL <https://arxiv.org/abs/2407.21243>.

A. Proofs

A.1. Proposition 3.2: Noising and Diffusion Forward Process

Given a noise distribution, a data point, and a scheduler, the distribution $q_{t|1}^{ID}(z|z_1)$ defined by our noising procedure (Equation 6) is equal to the distribution $q_{t|1}^{DDM}(z|z_1)$ obtained by the discrete forward process distribution defined in (Equation 1).

Proof. We adopt here the notation of Austin et al. (2021), where the probabilities and probability transition are written as matrices, and we start with the discrete time case.

We have:

$$q_{t|t+\Delta_t} = \mathbf{x} \mathbf{Q}_t \quad (19)$$

where \mathbf{Q}_t is a transition probability matrix. In our case,

$$\mathbf{Q}_t = (1 - \beta_t) \mathbf{I} + \beta \mathbf{A}, \quad (20)$$

where \mathbf{A} is an idempotent matrix, representing the noise distribution, the mask distribution being $\mathbf{A}_{\text{mask}} \mathbf{1} e_m^T$, where e_m is the one-hot vector indicating the mask and the marginal distribution being $\mathbf{A}_{\text{marg}} \mathbf{1} m^T$, with m representing the marginal distribution.

Using the fact that $\mathbf{A}^2 = \mathbf{A}$, and denoting $\alpha'_t = 1 - \beta$, we observe that:

$$\prod_i^{\tau} \mathbf{Q}_i = \left(\prod_i^{\tau} \alpha'_i \right) \mathbf{I} + \left(1 - \left(\prod_i^{\tau} \alpha'_i \right) \right) \mathbf{A}. \quad (21)$$

With $\alpha_t = \prod_i^{\tau} \alpha'_i$, we get: $q_t(z|z_1) = \alpha_t \delta_{z_1}(z) + (1 - \alpha_t) q_0(z)$.

For the continuous case, we remark that, α_t being smooth, we can turn the product into geometric integral. \square

A.2. Proposition 3.4

An *Iterative Denoising* step, described in Equation 8, is equivalent to a corrector sampling discrete diffusion step with a maximal corrector step.

Proof. Corrector sampling in discrete diffusion consists of applying k backward (denoising) steps, i.e., predicting $p_{t+k\Delta_t|t}(z|z_t)$ and $k-1$ forward (noising step), $q_{t|t+k\Delta_t}(z|z_{t+(k-1)\Delta_t})$. We call maximal corrector step a corrector sampling step such that $1 - k\Delta_t = t$. In this case, a corrector sampling step becomes:

$$\sum_z p_{1|t}(z|Z_t) q_{t+\Delta_t|1}(z|z_1) = p(z_{t+\Delta_t}|z_t) \quad (22)$$

$$= \alpha_{t+\Delta_t} p_{1|t}(z|z_t) + (1 - \alpha_{t+\Delta_t}) q_0(z), \quad (23)$$

where the second equality follows from Proposition A.3. \square

A.3. Denoising

$$\begin{aligned}
 p(z_s|z_t) &= \sum_z p_{s|1}(z|z_t, z_1) p_{1|t}^\theta(z|Z_t) \\
 &= \sum_z p_{s|1}(z|x_1) p_{1|t}^\theta(z|Z_t) \\
 &= \sum_z p_{s|1}(z|x_1) p_{1|t}^\theta(z|Z_t) \\
 &= \sum_z \sum_{a \in \{0,1\}} p_{s|1}(z|x_1, a_s) p_s(a) p_{1|t}^\theta(z|Z_t) \\
 &= \sum_z (q_1(z)\alpha_s + q_0(z)(1-\alpha_s)) p_{1|t}^\theta(z|Z_t) \\
 &= \sum_z (p_{1|t}^\theta(z|Z_t)q_1(z)\alpha_s) + \sum_z p_{1|t}^\theta(z|Z_t)q_0(z)(1-\alpha_s)) \\
 &= (p_{1|t}^\theta(z|Z_t)\alpha_s) + q_0(z)(1-\alpha_s)) \sum_z p_{1|t}^\theta(z|Z_t) \\
 &= \alpha_s p_{1|t}(z|z_t) + (1-\alpha_s)q_0(z)
 \end{aligned} \tag{24}$$

A.4. Proposition 4.1: Optimal Critic

The optimal Critic is:

$$C^*(\hat{z}_{1|t}) = \frac{\alpha_t p_{data}(\hat{z}_{1|t})}{\alpha_t p_{data}(\hat{z}_{1|t}) + (1-\alpha_t) p_{pred}(\hat{z}_{1|t})} \tag{25}$$

Proof. The proof is strongly inspired by Goodfellow et al. (2014).

$$\begin{aligned}
 \mathcal{L}_\phi &= -\mathbb{E}_{t, \hat{Z}_{1|t}} p_\phi(a|\hat{Z}_{1|t}) \\
 &= -\alpha_t \mathbb{E}_{\hat{Z}_{1|t} \sim p_{data}} p_\phi(a|\hat{Z}_{1|t}) \\
 &\quad - (1-\alpha_t) \mathbb{E}_{\hat{Z}_{1|t} \sim p_{pred}} p_\phi(a|\hat{Z}_{1|t})
 \end{aligned} \tag{26}$$

From Equation 13, we have:

$$\begin{aligned}
 \mathbb{E}_{\hat{Z}_{1|t}} p_\phi(a|\hat{Z}_{1|t}) &= \alpha_t \mathbb{E}_{\hat{Z}_{1|t} \sim p_{data}} p_\phi(a|\hat{Z}_{1|t}) + (1-\alpha_t) \mathbb{E}_{\hat{Z}_{1|t} \sim p_{pred}} p_\phi(a|\hat{Z}_{1|t}) \\
 &= \sum_{z \in \mathcal{Z}} \alpha_t p_{data}(\hat{z}_{1|t}) p_\phi(a|\hat{Z}_{1|t}) + (1-\alpha_t) p_{pred}(\hat{z}_{1|t}) p_\phi(a|\hat{Z}_{1|t})
 \end{aligned} \tag{27}$$

For any $(u, v) \in \mathbb{R}^2 \setminus \{(0, 0)\}$ and $y \in [0, 1]$, the function $f(y) = u \log(y) + v \log(1-y)$ reaches its maximum at $\frac{u}{u+v}$.

Hence, Equation 27 reaches its maximum at:

$$C^*(\hat{z}_{1|t}) = \frac{\alpha_t p_{data}(\hat{z}_{1|t})}{\alpha_t p_{data}(\hat{z}_{1|t}) + (1-\alpha_t) p_{pred}(\hat{z}_{1|t})} \tag{28}$$

□

A.5. Lemma 4.3: Noising Rate

If $p_{data}(\hat{z}_{1|t}) > p_{pred}(\hat{z}_{1|t})$, the optimal critic's noising rate $\hat{\beta}_t^*$ is smaller than the schedule's noising rate β_t , that is:

$$p_{data}(\hat{z}_{1|t}) > p_{pred}(\hat{z}_{1|t}) \implies \alpha_t < \hat{\alpha}_t, \tag{29}$$

and conversely

$$p_{data}(\hat{z}_{1|t}) < p_{pred}(\hat{z}_{1|t}) \implies \alpha_t > \hat{\alpha}_t. \quad (30)$$

Proof. From Theorem 4.1, we have:

$$\hat{\alpha}_t = \frac{\alpha_t p_{data}(\hat{z}_{1|t})}{\alpha_t p_{data}(\hat{z}_{1|t}) + (1 - \alpha_t) p_{pred}(\hat{z}_{1|t})} \quad (31)$$

$$\iff \hat{\alpha}_t^{-1} \alpha_t p_{data}(\hat{z}_{1|t}) = \alpha_t p_{data}(\hat{z}_{1|t}) + (1 - \alpha_t) p_{pred}(\hat{z}_{1|t}) \quad (32)$$

$$\iff \hat{\alpha}_t^{-1} p_{data}(\hat{z}_{1|t}) = p_{data}(\hat{z}_{1|t}) + (\alpha_t^{-1} - 1) p_{pred}(\hat{z}_{1|t}) \quad (33)$$

$$\iff \frac{p_{data}(\hat{z}_{1|t})}{p_{pred}(\hat{z}_{1|t})} = \frac{\alpha_t^{-1} - 1}{\hat{\alpha}_t^{-1} - 1} \quad (34)$$

Hence:

$$p_{data}(\hat{z}_{1|t}) > p_{pred}(\hat{z}_{1|t}) \implies \frac{\alpha_t^{-1} - 1}{\hat{\alpha}_t^{-1} - 1} > 1 \implies \alpha_t < \hat{\alpha}_t \quad (35)$$

$$p_{data}(\hat{z}_{1|t}) < p_{pred}(\hat{z}_{1|t}) \implies \alpha_t > \hat{\alpha}_t \quad (36)$$

□

B. Generative Graph Modeling: Related Works

One of the main challenges in graph modeling arises from the multiple possible representations of a single graph, due to the $n!$ possible node permutations. This complexity has led to two main approaches: sequential models and equivariant models.

Sequential models generate graphs auto-regressively, progressively adding nodes, edges, or subgraphs (You et al., 2018; Shi et al., 2020; Luo et al., 2021; Liao et al., 2019; Kong et al., 2023; Gómez-Bombarelli et al., 2018; Kusner et al., 2017; Goyal et al., 2020; Jin et al., 2018; 2020). Conversely, *Equivariant models* tackle the node permutation problem by ensuring a unique computational graph regardless of the specific instantiation of the graph structure. These models have been developed under various generative frameworks, including GANs (Krawczuk et al., 2021; Martinkus et al., 2022), normalizing flows (Madhawa et al., 2019; Zang & Wang, 2020; Liu et al., 2019), and vector-quantized auto-encoders (Boget et al., 2024; Nguyen et al., 2024).

Recently, equivariant denoising models have employed continuous denoising techniques, such as score-based models (Yang et al., 2019; Jo et al., 2022), diffusion bridges (Jo et al., 2024), and flow matching (Eijkelboom et al., 2024). However, these continuous relaxations disrupt the discrete structure of graphs during the noising process. Discrete diffusion models in discrete time (Haefeli et al., 2022; Vignac et al., 2023) and continuous time (Xu et al., 2024) have achieved significant improvements in generating small graphs. Additionally, Qin et al. (2024a) recently adapted the discrete flow matching framework for graph modeling.

In parallel, several approaches address scalability challenges in graph generation (Qin et al., 2024b; Chen et al., 2023; Karami, 2024; Bergmeister et al., 2024; Boget et al., 2025). Notably, Qin et al. (2024b) and Boget et al. (2025) propose methods that enable any equivariant model to scale to large graphs. While these methods could extend our iterative denoising framework to larger graphs, we leave this exploration for future research.

C. Models

C.1. GNNs architecture

Our denoisers are Graph Neural Network, inspired by the general, powerful, scalable (GPS) graph Transformer.

We implement a single layer as:

$$\tilde{\mathbf{X}}^{(l)}, \tilde{\mathbf{E}}^{(l)} = \text{MPNN}(\mathbf{X}^{(l)}, \mathbf{E}^{(l)}), \quad (37)$$

$$\mathbf{X}^{(l+1)} = \text{MultiheadAttention}(\tilde{\mathbf{X}}^{(l)} + \mathbf{X}^{(l)}) + \tilde{\mathbf{X}}^{(l)} \quad (38)$$

$$\mathbf{E}^{(l+1)} = \tilde{\mathbf{E}}^{(l)} + \mathbf{E}^{(l)} \quad (39)$$

where, $\mathbf{X}^{(l)}$ and $\mathbf{E}^{(l)}$ are the node and edge hidden representations after the l^{th} layer. The *Multihead Attention* layer is the classical multi-head attention layer from [Vaswani et al. \(2017\)](#), and MPNN is a Message-Passing Neural Network layer described hereafter.

The MPNN operates on each node and edge representations as follow:

$$\mathbf{h}_{i,j}^l = \text{ReLU}(\mathbf{W}_{src}^l \mathbf{x}_i^l + \mathbf{W}_{trg}^l \mathbf{x}_j^l + \mathbf{W}_{edge}^l \mathbf{e}_{i,j}^l) \quad (40)$$

$$\mathbf{e}_{i,j}^{l+1} = \text{LayerNorm}(f_{\text{edge}}(\mathbf{h}_{i,j}^l)) \quad (41)$$

$$\mathbf{x}_i^{l+1} = \text{LayerNorm} \left(\mathbf{x}_i^l + \sum_{j \in \mathcal{N}(i)} f_{\text{node}}(\mathbf{h}_{i,j}^l) \right), \quad (42)$$

where \mathbf{W}_{src}^l , \mathbf{W}_{trg}^l , and \mathbf{W}_{edge}^l are matrices of parameters and f_{node} , and f_{edge} are small neural networks.

The node hidden representation, i.e., the \mathbf{x}_i 's and the hidden representation of f_{node} are of dimensions d_h , an hyperparameter (see Table 3). The edge hidden representation, i.e., the $\mathbf{e}_{i,j}$'s, $\mathbf{h}_{i,j}^l$'s, and the hidden representation of f_{edge} are of dimensions $d_h/4$.

Inputs and Outputs In the input, we concatenate the node attributes, extra features, and time step as node features, copying graph-level information (e.g., time step or graph size) to each node. The node and edge input vectors are then projected to their respective hidden dimensions, d_h for nodes and $d_h/4$ for edges.

Similarly, the outputs of the final layer are projected to their respective dimensions, d_x for nodes and d_e for edges (or to a scalar in the case of the *Critic*). To enforce edge symmetry, we compute $\mathbf{e}_{i,j} = \frac{\mathbf{e}_{i,j} + \mathbf{e}_{j,i}}{2}$. Finally, we ensure the outputs can be interpreted as probabilities by applying either a softmax or sigmoid function, as appropriate.

C.2. hyperparameters

As explained here above, the node hidden representation has size d_h and the edge representation $d_h/4$. We use $d_h = 64$ with the Qm9 dataset and $d_h = 256$ with all the other datasets.

Table 3: Hyperparameters

| | |
|---------------------------|--|
| MPNN layers | 4 |
| Layers in MLPs | 3 |
| Diffusion steps | 500 |
| Learning rate | 0.0002 |
| Optimizer | Adam |
| Betas parameters for Adam | (0.9, 0.999) |
| Scheduler | cosine (Nichol & Dhariwal, 2021) |

C.3. Extra Features

Following a common practice ([Vignac et al., 2023](#); [Qin et al., 2024a](#); [Boget et al., 2025](#)), we enhance the graph representation with synthetic extra node features. We use the following extra features: eigen features, graph size, molecular features (for molecular datasets such as Qm9 and Zinc250k), and cycle information (for the Planar datasets). All these features are concatenated to the input node attributes.

Spectral features We use the eigenvectors associated with the k lowest eigenvalues of the graph Laplacian. Additionally, we concatenate the corresponding k lowest eigenvalues to each node.

Graph size encoding The graph size is encoded as the ratio between the size of the current graph and the largest graph in the dataset, n/n_{\max} . This value is concatenated to all nodes in the graph.

Molecular features For molecular datasets, we use the charge and valency of each atom as additional features.

Cycles Following Vignac et al. (2023), we count the number of cycles of size 3, 4, and 5 that each node is part of, and use these counts as features.

D. Evaluation

D.1. Molecular Benchmark

For molecular graphs, we adopt the evaluation procedure followed by Jo et al. (2024), from which we took the baseline model results, and which was originally established in Jo et al. (2022).

Datasets The QM9 dataset (Wu et al., 2017) consists of 133,885 organic molecules with up to 9 heavy atoms, including carbon (C), oxygen (O), nitrogen (N), and fluorine (F). In contrast, the Zinc250k dataset (Irwin et al., 2012) contains 250,000 molecules with up to 38 atoms spanning 9 element types: C, O, N, F, phosphorus (P), sulfur (S), chlorine (Cl), bromine (Br), and iodine (I). Both datasets are divided into a test set (25,000 molecules), a validation set (25,000 molecules), and a training set (the remaining molecules).

For our experiments, we preprocess the datasets following standard procedures (Jo et al., 2022; 2024). Molecules are kekulized using RDKit, and explicit hydrogen atoms are removed from the QM9 and Zinc250k datasets.

We evaluate the models using three metrics:

1. Validity: The percentage of chemically valid molecules among the generated samples, determined using RDKit’s Sanitize function without applying post-hoc corrections, such as valency adjustments or edge resampling.
2. Fréchet ChemNet Distance (FCD) (Preuer et al., 2018): Measures the distance between feature distributions of generated molecules and test set molecules, using ChemNet to capture their chemical properties.
3. Neighborhood Subgraph Pairwise Distance Kernel (NSPDK) MMD (Costa & Grave, 2010): Assesses the quality of graph structures by computing the maximum mean discrepancy (MMD) between the generated molecular graphs and those from the test set.

D.2. Generic Graphs

We assess the quality of the generated graphs using three benchmark datasets from Martinkus et al. (2022).

Planar Graph Dataset: This dataset consists of 200 synthetic planar graphs, each containing 64 nodes. A generated graph is considered valid if it is connected and planar.

Stochastic Block Model (SBM) Dataset: This dataset consists of 200 synthetic graphs generated using the Stochastic Block Model (SBM). The number of communities is randomly sampled between 2 and 5, and the number of nodes per community is sampled between 20 and 40. The edge connection probabilities are set as follows:

- Intra-community edges: 0.3 (probability of an edge existing within the same community).
- Inter-community edges: 0.005 (probability of an edge existing between different communities).

A generated graph is considered valid if it satisfies the statistical test introduced in (Martinkus et al., 2022), which assess it corresponds to to the SBM structure.

We adopt the evaluation framework proposed by (Liao et al., 2019), utilizing total variation (TV) distance to measure the Maximum Mean Discrepancy (MMD). This approach is significantly more computationally efficient than using the Earth Mover’s Distance (EMD) kernel, particularly for large graphs.

Additionally, we employ the V.U.N. metric as introduced by Martinkus et al. (2022), which evaluates the proportion of valid, unique, and novel graphs among the generated samples. A graph is considered valid if it satisfies the dataset-specific structural properties described earlier.

E. Complementary Experimental Results

Results on Qm9 As additional metrics, we report uniqueness, defined as the fraction of unique molecules among the generated samples, and novelty, the fraction of unique molecules that are not present in the training dataset. Furthermore, all models achieve 100% validity with valency correction.

Table 4: Results on Qm9

| AVERAGE | VALID | UNIQUE | NOVEL | NSPDK | FCD |
|--------------|----------------|----------------|----------------|------------------|-----------------|
| MARGINAL DDM | 95, 73 ± 0, 24 | 97, 52 ± 0, 16 | 79, 55 ± 0, 28 | 1, 922 ± 0, 054 | 1, 090 ± 0, 035 |
| MARGINAL ID | 99, 67 ± 0, 06 | 95, 66 ± 0, 28 | 72, 57 ± 0, 52 | 1, 041 ± 0, 049 | 0, 504 ± 0, 010 |
| MASK DDM | 48, 38 ± 0, 47 | 78, 47 ± 0, 65 | 75, 21 ± 0, 88 | 14, 750 ± 0, 509 | 3, 760 ± 0, 035 |
| MASK ID | 96, 43 ± 0, 16 | 98, 02 ± 0, 14 | 86, 79 ± 0, 31 | 1, 395 ± 0, 015 | 1, 797 ± 0, 027 |
| MASK CID | 99, 92 ± 0, 02 | 96, 93 ± 0, 13 | 80, 74 ± 0, 26 | 1, 402 ± 0, 034 | 1, 757 ± 0, 035 |

Results on Zinc250k As additional metrics, we report uniqueness, defined as the fraction of unique molecules among the generated samples, and novelty, the fraction of unique molecules that are not present in the training dataset. Furthermore, all models achieve 100% validity with valency correction.

Table 5: Results on Zinc250k

| | VALID | UNIQUE | NOVEL | NSPDK | FCD |
|--------------|----------------|-----------------|-----------------|----------------|----------------|
| MARGINAL DDM | 80, 40 ± 0, 31 | 95, 93 ± 0, 02 | 99, 98 ± 0, 03 | 12, 96 ± 0, 25 | 8, 50 ± 0, 09 |
| MARGINAL ID | 99, 50 ± 0, 06 | 99, 84 ± 0, 02 | 99, 97 ± 0, 00 | 2, 06 ± 0, 05 | 2, 01 ± 0, 01 |
| MASK DDM | 8, 96 ± 0, 36 | 99, 37 ± 0, 00 | 100, 00 ± 0, 24 | 78, 63 ± 2, 41 | 24, 98 ± 0, 24 |
| MASK ID | 93, 85 ± 0, 25 | 100, 00 ± 0, 01 | 100, 00 ± 0, 02 | 11, 08 ± 0, 17 | 9, 05 ± 0, 16 |
| MASK CID | 99, 97 ± 0, 01 | 99, 98 ± 0, 01 | 99, 98 ± 0, 01 | 2, 26 ± 0, 09 | 3, 46 ± 0, 01 |

Results on Planar We provide additionally the results using the degree, clustering and orbit MMD.

Table 6: Results on Planar

| | SPECTRAL | V.U.N. | DEGREE | CLUSTERING | ORBIT |
|--------------|----------------|--------------|----------------|------------------|-------------------|
| MARGINAL DDM | 83, 57 ± 2, 56 | 0, 0 ± 0, 0 | 53, 83 ± 0, 80 | 300, 32 ± 3, 15 | 1441, 45 ± 83, 57 |
| MARGINAL ID | 7, 62 ± 1, 34 | 91, 3 ± 4, 1 | 5, 93 ± 1, 26 | 163, 40 ± 31, 86 | 19, 08 ± 4, 14 |
| MASK DDM | 84, 44 ± 2, 72 | 0, 0 ± 0, 0 | 57, 07 ± 2, 15 | 297, 75 ± 2, 50 | 1397, 94 ± 34, 17 |
| MASK ID | 8, 72 ± 1, 37 | 67, 0 ± 6, 5 | 2, 30 ± 0, 67 | 78, 48 ± 13, 12 | 11, 68 ± 4, 78 |
| MASK CID | 6, 40 ± 1, 20 | 66, 0 ± 5, 5 | 2, 11 ± 0, 62 | 85, 86 ± 10, 74 | 14, 18 ± 5, 54 |

Results on SBM We provide additionally the results using the degree, clustering and orbit MMD.

Critical Iterative Denoising

Table 7: Results on SBM

| | SPECTRAL | V.U.N. | DEGREE | CLUSTERING | ORBIT |
|--------------|------------------|----------------|-------------------|------------------|--------------------|
| MARGINAL DDM | $11,82 \pm 0,85$ | $0,0 \pm 0,0$ | $0,96 \pm 0,73$ | $85,66 \pm 6,26$ | $72,37 \pm 5,51$ |
| MARGINAL ID | $5,93 \pm 1,18$ | $63,5 \pm 3,7$ | $11,54 \pm 2,72$ | $51,41 \pm 1,49$ | $123,14 \pm 5,35$ |
| MASK DDM | $11,38 \pm 1,04$ | $0,0 \pm 0,0$ | $3,81 \pm 1,79$ | $83,13 \pm 2,27$ | $123,00 \pm 3,73$ |
| MASK ID | $15,05 \pm 4,29$ | $17,5 \pm 5,7$ | $70,12 \pm 21,99$ | $55,63 \pm 0,66$ | $127,05 \pm 17,95$ |
| MASK CID | $11,94 \pm 2,71$ | $19,0 \pm 4,9$ | $43,13 \pm 12,31$ | $55,06 \pm 2,29$ | $92,80 \pm 16,96$ |

NFE Ablation on Zinc250 We present here addition figures and tables ablating the effect of the NFE.

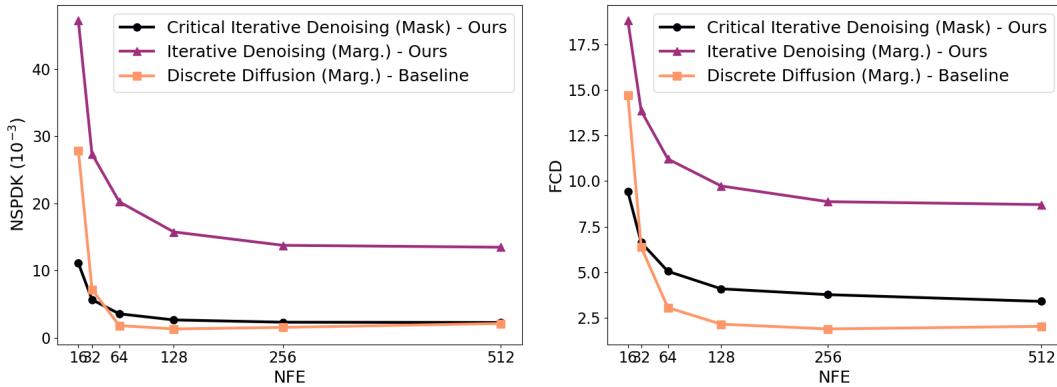


Figure 4: NSPDK and FCD as a function of the Number of Function Evaluations (NFE) for three models: Discrete Diffusion (baseline), Iterative Denoising (ours), and Critical Iterative Denoising (ours).

Table 8: Validity vs NFE

| NFE | 16 | 32 | 64 | 128 | 256 | 512 |
|-----------|-------|-------|-------|-------|-------|-------|
| MASK ID | 36,34 | 56,60 | 70,94 | 81,78 | 89,37 | 93,57 |
| MASK DDM | 9,35 | 8,84 | 9,05 | 9,14 | 9,06 | 8,49 |
| MASK CID | 92,51 | 99,17 | 99,78 | 99,91 | 99,98 | 99,96 |
| MARG. DDM | 57,31 | 69,33 | 76,80 | 78,09 | 80,15 | 80,37 |
| MARG. ID | 38,25 | 73,66 | 92,71 | 97,69 | 99,06 | 99,58 |

Table 9: NSPDK vs NFE

| NFE | 16 | 32 | 64 | 128 | 256 | 512 |
|-----------|-------|-------|-------|-------|-------|-------|
| MASK ID | 30,05 | 24,90 | 20,03 | 15,68 | 13,10 | 10,49 |
| MASK DDM | 78,27 | 79,66 | 77,94 | 79,59 | 79,11 | 77,97 |
| MASK CID | 11,10 | 5,66 | 3,55 | 2,64 | 2,30 | 2,26 |
| MARG. DDM | 47,22 | 27,35 | 20,23 | 15,74 | 13,75 | 13,46 |
| MARG. ID | 27,86 | 7,18 | 1,80 | 1,30 | 1,53 | 2,11 |

Critical Iterative Denoising

Table 10: FCD vs NFE

| NFE | 16 | 32 | 64 | 128 | 256 | 512 |
|-----------|-------|-------|-------|-------|-------|-------|
| MARG, DDM | 18,81 | 13,85 | 11,21 | 9,73 | 8,87 | 8,71 |
| MARG ID | 14,70 | 6,37 | 3,06 | 2,15 | 1,89 | 2,03 |
| MASK DDM | 25,23 | 25,35 | 24,73 | 25,09 | 24,96 | 24,69 |
| MASK ID | 15,80 | 14,16 | 12,43 | 10,80 | 9,80 | 9,01 |
| MASK CID | 9,42 | 6,63 | 5,05 | 4,09 | 3,77 | 3,40 |

F. Visualizations

F.1. Molecular graphs

Figure 5: Qm9

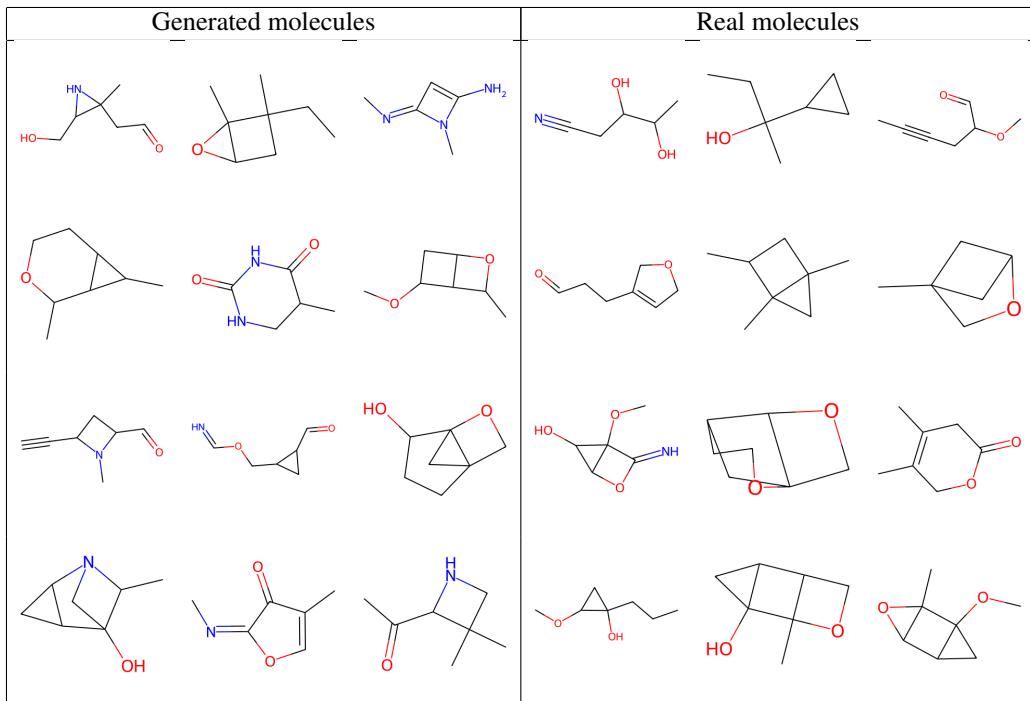
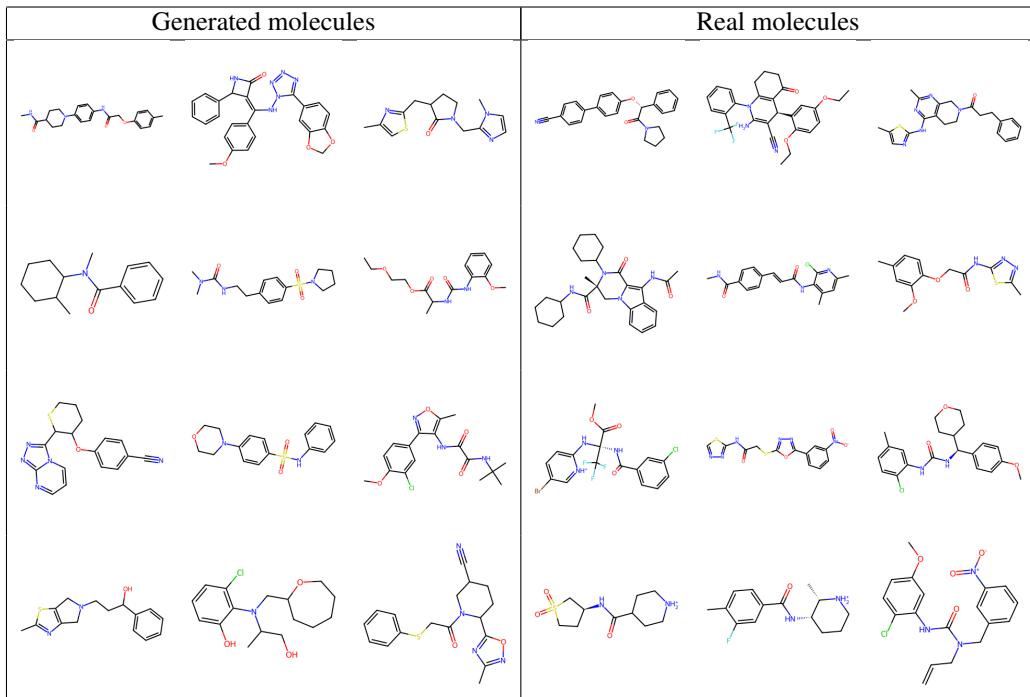


Figure 6: Zinc



F.2. Generic Graphs

Figure 7: Planar

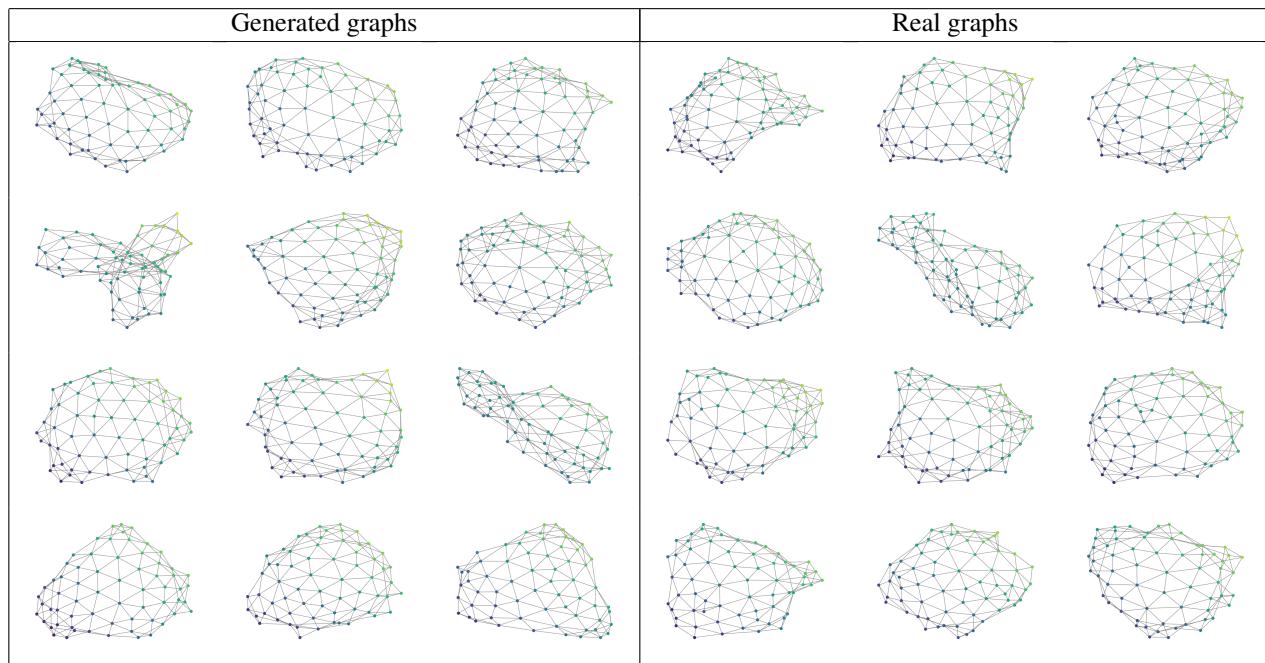
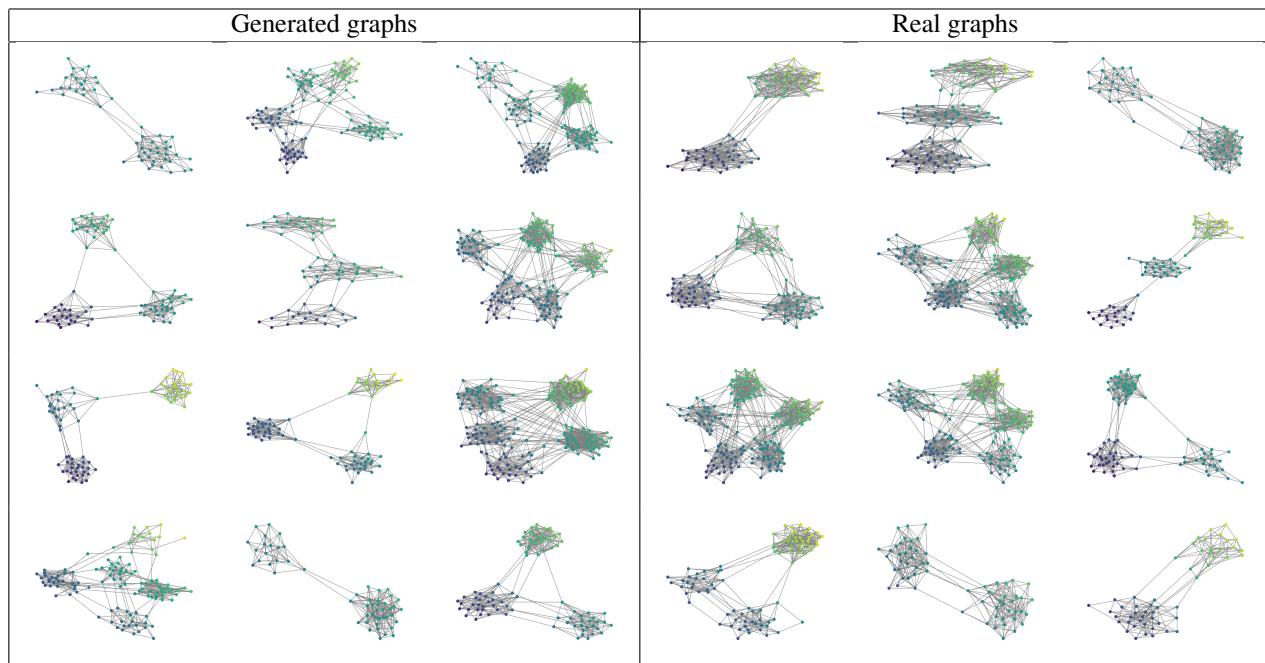


Figure 8: SBM



All generated graphs comes from the Marginal Iterative Denoising model.