REMASKING DISCRETE DIFFUSION MODELS WITH INFERENCE-TIME SCALING

Guanghan Wang[*†], **Yair Schiff**[†], **Subham Sekhar Sahoo**, **Volodymyr Kuleshov**
Department of Computer Science, Cornell University
{gw354,yzs2,sss284,vk379}@cornell.edu

ABSTRACT

Part of the success of diffusion models stems from their ability to perform iterative refinement, i.e., repeatedly correcting outputs during generation. However, modern masked discrete diffusion lacks this capability: when a token is generated, it cannot be updated again, even when it introduces an error. Here, we address this limitation by introducing the remasking diffusion model (ReMDM) sampler, a method that can be applied to pretrained masked diffusion models in a principled way and that is derived from a discrete diffusion model with a custom remasking backward process. Most interestingly, ReMDM endows discrete diffusion with a form of inference-time scaling. By increasing the number of sampling steps, ReMDM generates natural language outputs that approach the quality of autoregressive models, whereas when the computation budget is limited, ReMDM better maintains quality. ReMDM also improves sample quality of masked diffusion models for discretized images, and in scientific domains such as molecule design, ReMDM facilitates diffusion guidance and pushes the Pareto frontier of controllability relative to classical masking and uniform noise diffusion.

1 INTRODUCTION

Diffusion models have gained significant traction as algorithms for generating high-quality images and videos (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020). Part of the success of diffusion stems from its ability to perform iterative refinement—repeatedly modifying outputs and fixing errors over multiple steps of generation—which makes diffusion models especially effective at guided generation (Dhariwal & Nichol, 2021; Ho & Salimans, 2022), fast sampling (Song et al., 2020a; Salimans & Ho, 2022; Song et al., 2023), and supports inference-time scaling to improve sample quality Ma et al. (2025).

Discrete counterparts of diffusion models Austin et al. (2021) have also been steadily improving in quality on tasks such as language modeling and biological sequence design Lou et al. (2024); Sahoo et al. (2024); Schiff et al. (2024). However, modern discrete diffusion—especially the most performant kind that relies on masking (Ou et al., 2024; Shi et al., 2024; Sahoo et al., 2024)—lacks the fundamental ability to iteratively refine outputs. Once a discrete token is unmasked, it cannot be updated again, even if it introduces an error. The lack of error correction in turn sets limits on controllable generation, sampling speed, and sample quality.

Here, we address the inability of masked diffusion models to perform iterative refinement by introducing a new sampler that supports remasking during generation. Our method, the remasking diffusion model (ReMDM) sampler, is simple and allows users to directly specify the probability of remasking a token at each time step. We augment the sampler with components that range from nucleus sampling to remasking schedules, significantly boosting performance.

Our approach for deriving the sampler is rooted in probabilistic modeling—-we show that our method corresponds to ancestral sampling in a discrete diffusion model (also called ReMDM) characterized by a remasking backward process. The ReMDM model admits an objective that is a rescaled version of the MDLM objective (Sahoo et al., 2024)—this suggests using the ReMDM sampler on top of pretrained MDLMs, which we find to work well. We complement our analysis of the sampler by also

[*]Corresponding author. [†]Equal contribution. Project Page: https://remdm.github.io

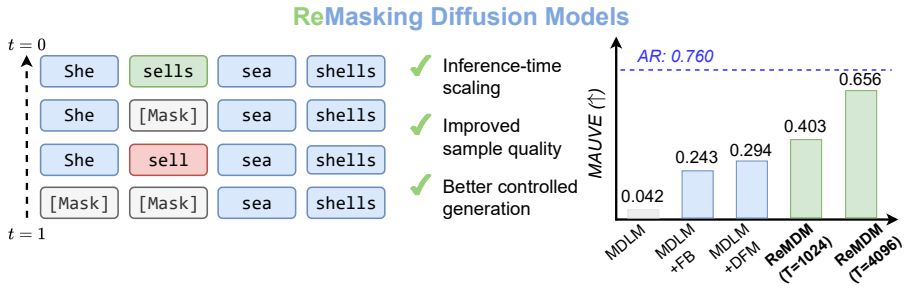


Figure 1: Our family of masked diffusion processes allow for more flexible generation with remasking of already decoded tokens. This improves sample quality and further closes the gap to AR models. *(Left)* An illustrative example of errors fixed by ReMDM. The first two tokens can be "They sell" or "She sells", but due to the independence of the parallelized decoding processes, "She sell" is decoded. Such mistakes can be corrected by remasking samplers. *(Right)* MAUVE scores on OpenWebText. MDLM is from Sahoo et al. (2024). FB and DFM denote the forward-backward Campbell et al. (2022) and discrete flow matching Gat et al. (2024) correctors, respectively.

showing that it can be interpreted as a predictor-corrector technique (Song et al., 2020b; Campbell et al., 2022).

Most interestingly, the ReMDM sampler endows discrete diffusion with a form of inference-time scaling. By increasing the number of denoising steps on language modeling tasks, ReMDM generates samples with significantly higher sample quality metrics than any previous diffusion model and almost matches the performance of an autoregressive (AR) model with the same architecture. Conversely, when we reduce sampling steps to increase speed, performance degrades less than other samplers on text and image generation. Lastly, ReMDM models are more amenable to guidance (Schiff et al., 2024)—in molecule design experiments, ReMDM pushes the Pareto frontier of novelty and the desired molecular property relative to alternatives relying on masking or uniform noise diffusion.

Contributions We summarize our contributions as follows:

1. We introduce the remasking diffusion model (ReMDM) sampler and several add-on components that bring performant iterative refinement via remasking to masked diffusion models.
2. We show that our method is a form of ancestral sampling in a probabilistic model whose ELBO is similar to that of classical masked diffusion. This analysis suggests using our sampler on top of pretrained models, which we find to work well.
3. Across the domains of natural language, discretized images, and molecule string representations, we demonstrate empirically that ReMDM endows masked diffusion with inference-time scaling that improves sample quality with more computation, and that also enhances controlled generation.

2 Background

Discrete Diffusion Models Diffusion models Sohl-Dickstein et al. (2015); Ho et al. (2020) are characterized by parametric models p_θ trained to reverse a fixed noising process q that maps clean data $\mathbf{x}$ to increasingly noisy latent variables $\mathbf{z}_t$, for $t \in [0, 1]$. Austin et al. (2021); Sahoo et al. (2024); Shi et al. (2024) extend this framework to discrete signals. Formally, we define $\mathbf{x} \in \{0, 1\}^{|V|} \subset \Delta^{|V|}$ as one-hot vectors, where $|V|$ is the vocabulary size and $\Delta^{|V|}$ is the simplex over this vocabulary. For finite-time processes, we let T be the number of time steps. For each time step $t(i) = \frac{i}{T} \in [0, 1]$ (i.e., for $i = 0, 1, \ldots, T$), each intermediate latent variable $\mathbf{z}_{t(i)} \in \{0, 1\}^{|V|}$ has the marginal:

$$q(\mathbf{z}_t \mid \mathbf{x}) = \mathrm{Cat}(\mathbf{z}_t; \alpha_t \mathbf{x} + (1 - \alpha_t)\boldsymbol{\pi}) \in \Delta^{|V|}, \tag{1}$$

where $\boldsymbol{\pi}$ is a user-specified prior and where we have dropped the explicit dependence of t on i. The predefined schedule $\alpha_t \in [0, 1]$ is monotonically decreasing in t. For sequences of L tokens, we denote clean / noisy sequences as $\mathbf{x}^{(1:L)}$ / $\mathbf{z}_t^{(1:L)}$ and each token $\ell \in \{1, \ldots, L\}$ as $\mathbf{x}^{(\ell)}$ / $\mathbf{z}_t^{(\ell)}$.

Masked Diffusion Language Models A special case of this formulation, known as 'masking' or 'absorbing state' discrete diffusion, sets the limiting distribution $\boldsymbol{\pi} = \boldsymbol{m}$, a one-hot vector centered on a special [MASK] token. Letting $s(i) = \frac{i-1}{T}$ be the time step directly preceding t, these posteriors take the form:

$$q(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x}) = \mathrm{Cat}\Big(\mathbf{z}_s; \frac{\alpha_s - \alpha_t}{1-\alpha_t}\mathbf{x} + \frac{1-\alpha_s}{1-\alpha_t}\mathbf{z}_t\Big). \tag{2}$$

Importantly, despite the recent success of MDLM Sahoo et al. (2024) and other absorbing state discrete diffusion models, these models suffer from a key drawback, which we call the **failure to remask** property. Namely, the posterior formula in (2) implies that any unmasked token $\mathbf{z}_t \neq \boldsymbol{m}$ must remain unchanged throughout the entire denoising process, for all $s \leq t$, which Sahoo et al. (2024) formalize as $q(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x}) = \mathrm{Cat}(\mathbf{z}_s; \mathbf{z}_t)$ if $\mathbf{z}_t \neq \boldsymbol{m}$. Once a token is decoded, this prediction is 'locked-in', a limitation that these diffusion models share with AR models.

As in Sahoo et al. (2024), a denoising neural network $\mathbf{x}_\theta$ is used for parameterizing $p_\theta(\mathbf{z}_s|\mathbf{z}_t) = q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}_\theta(\mathbf{z}_t))$ and trained using the following objective:

$$\mathcal{L} = \mathbb{E}_{\mathbf{z}_0 \sim q(\mathbf{z}_0|\mathbf{x})}\Big[-\log p_\theta(\mathbf{x} \mid \mathbf{z}_0)\Big] + \mathbb{E}_{t\in\{\frac{1}{T},\dots,1\}}\mathbb{E}_{\mathbf{z}_t\sim q(\mathbf{z}_t|\mathbf{x})}T\Big[\frac{\alpha_t - \alpha_s}{1-\alpha_t}\log(\mathbf{x}_\theta(\mathbf{z}_t)^\top \mathbf{x})\Big] \tag{3}$$

3 Remasking Diffusion Models

In this work, we alleviate the failure to remask limitation that affects state-of-the-art masked diffusion models. Our strategy is to design a probabilistic model similar to MDLM (Sahoo et al., 2024), but whose denoising posterior generalizes (2) and supports remasking.

Specifically, we will define a discrete diffusion model whose posterior for $\mathbf{z}_t \neq \boldsymbol{m}$ has the following form:

$$q(\mathbf{z}_s \mid \mathbf{z}_t = \mathbf{x}, \mathbf{x}) = (1-\sigma_t)\mathbf{x} + \sigma_t \boldsymbol{m}. \tag{4}$$

The parameter σ_t gives us the flexibility of remasking a decoded token during generation. Because of this property, we dub our method **ReM**asking **D**iffusion **M**odels (**ReMDM**).

3.1 ReMDM Forward Processes

We define the complete forward process of ReMDM as:

$$q_\sigma(\mathbf{z}_{0:1} \mid \mathbf{x}) = q_\sigma(\mathbf{z}_1 \mid \mathbf{x}) \prod_{i=1}^{T} q_\sigma(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x}) \tag{5}$$

where $q_\sigma(\mathbf{z}_1 \mid \mathbf{x}) = q(\mathbf{z}_1) = \boldsymbol{m}$ is the same limiting distribution as in masked diffusion. We construct the posteriors $q_\sigma(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x})$ as:

$$\begin{cases} \mathrm{Cat}(\mathbf{z}_s; (1-\sigma_t)\mathbf{x} + \sigma_t \boldsymbol{m}), & \mathbf{z}_t \neq \boldsymbol{m} \\ \mathrm{Cat}(\mathbf{z}_s; \frac{\alpha_s - (1-\sigma_t)\alpha_t}{1-\alpha_t}\mathbf{x} + \frac{1-\alpha_s-\sigma_t\alpha_t}{1-\alpha_t}\boldsymbol{m}), & \mathbf{z}_t = \boldsymbol{m}. \end{cases} \tag{6}$$

Observe how (6) maintains the remasking property laid out in (4). Additionally, our chosen form for $q(\mathbf{z}_s \mid \mathbf{z}_t = \mathbf{m}, \mathbf{x})$ ensures that the marginals for $q_\sigma(\mathbf{z}_t \mid \mathbf{x})$ are the same as in classical masked diffusion (e.g., Sahoo et al. (2024)). This property will yield an ELBO for ReMDM that is very similar to that of MDLM, and will help us argue for using the ReMDM sampling procedure on top of a pretrained MDLM model. Formally, we establish the following result (proof in Appendix A.1):

Theorem 3.1. *Given the posterior in (6), the marginal distribution $q_\sigma(\mathbf{z}_t \mid \mathbf{x})$ is the same as in (1).*

In Appendix A.2, we demonstrate that this ability to remask necessitates that ReMDM is a non-Markovian process (as in e.g., DDIM (Song et al., 2020a)).

Understanding the Role of σ_t To ensure that the posterior in (6) is a valid probability distribution, we place following constraints on σ_t (see derivation in Appendix A.3):

$$0 \leq \sigma_t \leq \min\Big\{1, \frac{1-\alpha_s}{\alpha_t}\Big\} =: \sigma_t^{max}. \tag{7}$$

Further examining (6), we see that as σ_t increases, we move mass away from $\mathbf{z}_t$. Importantly, this means that when $\mathbf{z}_t$ is unmasked, σ_t directly controls the probability of remasking. When $\sigma_t = 0$, we recover the posterior from MDLM in (2). Thus, ReMDM can be seen as a more general formulation that admits MDLM as a special case.

3.2 Negative Evidence Lower Bound

We define the joint distribution of the parameterized forward process p_θ as follows:

$$p_\theta(\mathbf{x})p_\theta(\mathbf{z}_{0:1} \mid \mathbf{x}) = p_\theta(\mathbf{z}_1)p_\theta(\mathbf{x} \mid \mathbf{z}_0)\prod_{i=1}^{T} p_\theta(\mathbf{z}_s \mid \mathbf{z}_t). \tag{8}$$

We define $p_\theta(\mathbf{z}_1) = \boldsymbol{\pi}$ and parameterize $p_\theta(\mathbf{z}_s \mid \mathbf{z}_t) = q(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x}_\theta(\mathbf{z}_t))$, where $\mathbf{x}_\theta(\mathbf{z}_t)$ is a denoising model. With this parameterization, the NELBO for ReMDM is (see Appendix A.4 for details):

$$\mathcal{L}^\sigma = \mathbb{E}_{\mathbf{z}_0 \sim q(\mathbf{z}_0|\mathbf{x})}\Big[-\log p_\theta(\mathbf{x} \mid \mathbf{z}_0)\Big] + \mathbb{E}_{t\in\{\frac{1}{T},\dots,1\}}\mathbb{E}_{\mathbf{z}_t\sim q(\mathbf{z}_t|\mathbf{x})} T\left[\frac{(1-\sigma_t)\alpha_t - \alpha_s}{1-\alpha_t}\log(\mathbf{x}_\theta^\top \mathbf{x})\right]. \tag{9}$$

Since $\alpha_t \leq \alpha_s$, each term in the second expectation increases monotonically in σ_t. When $\sigma_t = 0$, for all t, we recover the MDLM objective from (3). This implies that the MDLM NELBO produces a tighter bound compared to ReMDM. Furthermore, since the diffusion loss term in (9) is simply a reweighted version of that in (3), one can presumably reuse weights from a model trained with (3), i.e., using different σ_t at training and inference. Indeed, we find this to be a performant strategy in practice. Moreover, we find the performance (test perplexity) between models trained with the MDLM and the ReMDM objectives to be comparable (see Appendix E.1), further justifying our reuse of pretrained weights, with the added benefit of more flexible sampling unlocked by ReMDM.

4 ReMDM Samplers

Using ReMDM for sampling opens a large design space for choosing σ_t. In the following, we explore several practical design strategies for σ_t that significantly improve performance, and in Section 5 we describe further add-ons to the sampler (e.g., nucleus sampling) that also improve quality. See Appendix C for the high-level pseudocode for the ReMDM sampler and more detailed algorithms for implementing the schedules below.

4.1 Design Strategies for the Remasking Schedule σ_t

Here, we list several strategies for setting $\sigma_t \in [0, \sigma_t^{max}]$, which can either be employed separately or in conjunction with the strategies introduced in Section 4.2. Note that for sequences, we can assign a different $\sigma_t^{(\ell)}$ for each token. When we omit this superscript, this implies that the same σ_t is used for all tokens.

Max-Capped Schedule We can potentially reduce the maximum probability of remasking to a constant $\eta_{cap} \in [0, 1]$. Concretely, we let $\sigma_t = \min\{\eta_{cap}, \frac{1-\alpha_s}{\alpha_t}\}$, for all $t \in [0, 1]$. We denote this schedule as "ReMDM-cap."

Rescaled Schedule Alternatively, we can temper the chances of remasking by setting $\sigma_t = \eta_{rescale} \cdot \sigma_t^{max}$, with $\eta_{rescale} \in [0, 1]$ as a hyperparameter that controls this rescaling. We denote this schedule as "ReMDM-rescale."

Confidence-Based Schedule Apart from the two strategies above, we explore a further reweighing of σ_t which is based on the intuition that tokens of which the denoising model is less confident should be assigned a larger probability of remasking. Consider the ℓ-th token in a sequence of L latents at time t. For each $\ell \in \{1, \dots, L\}$, we store its decoding probability at the time τ at which it was last unmasked. Concretely, if $\mathbf{z}_t^{(\ell)} \neq \boldsymbol{m}$, then we define $\psi_t^{(\ell)} := \mathbf{x}_{\theta,\tau}^{(\ell)\top}\mathbf{z}_\tau^{(\ell)}$. If $\mathbf{z}_t = \boldsymbol{m}$, then $\psi_t^{(\ell)} := \infty$. Thus, $\psi_t^{(\ell)}$ serves as a 'confidence score' for unmasked tokens. We then compute $\sigma_t^{(\ell)} = \eta_{conf}^{(\ell)} \cdot \sigma_t$, where $\eta_{conf}^{(\ell)} = \frac{\exp(-\psi_t^{(\ell)})}{\sum_{l=1}^{L}\exp(-\psi_t^{(\ell')})}$. With this schedule, masked tokens are decoded

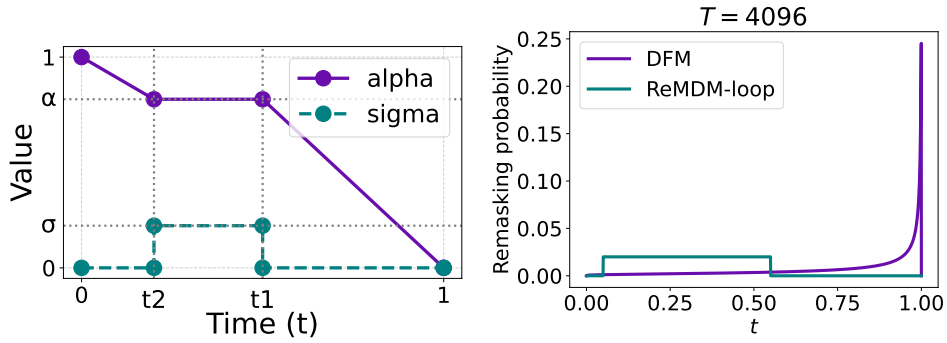
using the approximate posterior from MDLM, and the unmasked tokens are remasked negatively proportional to their confidence. We denote this schedule as "ReMDM-conf."

4.2 Design Strategies for 'Turning On' ReMDM

There may be certain periods of the generation process where remasking is more valuable and some when it is detrimental / can slow down sampling, e.g., at the beginning of sampling, one may wish to generate some tokens in a sequence using standard MDLM decoding, and only after generating a reasonable starting candidate, then spend some computational budget 'fixing mistakes' via the remasking posterior. We, therefore, propose two methods for optionally 'turning on/off' ReMDM sampling, which amount to the following modification of the σ_t schedules above:

$$\tilde{\sigma}_t = \begin{cases} \sigma_t, & \text{if } t \in [t_{on}, t_{off}), \text{with } t_{on} > t_{off} \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

Switch We choose some $t_{switch} \in (0, 1]$ and in (10) we have $[t_{on}, t_{off}) = [t_{switch}, 0)$. We denote this strategy as "ReMDM-switch."



(a) α_t and σ_t schedules in ReMDM-loop.

(b) ReMDM and DFM σ_t schedules for OWT.

Loop In this strategy, we set both $t_{on}, t_{off} \in (0, 1]$. Furthermore, in the range when ReMDM is activated, we modify the noise schedule to be constant, such that $\alpha_t = \alpha(t_{on})$. As shown in Figure 2a, this divides the sampling process into three phases. In the first phase, the model generates tokens without remasking ($\sigma_t = 0$, i.e., using MDLM). In the second phase, we hold α constant (i.e., $\alpha_s = \alpha_t$), and the model can 'correct potential mistakes' by remasking and predicting a fixed proportion of the generated tokens in a loop. Finally, in the third phase, we let the model predict any remaining unmasked tokens using the MDLM posterior. We denote this strategy as "ReMDM-loop." Note that we can also change the amount of computation spent in each of the three phases by taking non-fixed-width steps.

4.3 Comparison with Discrete Predictor-Corrector Samplers

Previous works, e.g., the forward-backward (FB; Campbell et al. (2024)) and discrete flow matching (DFM; Gat et al. (2024)) correctors, propose to tackle the failure to remask property with discrete predictor-corrector samplers, a special type of discrete diffusion samplers that decompose a single sampling step into one predictor step followed by a certain number of corrector steps that remediate possible mistakes without changing the marginals. Here, we demonstrate that these methods are either a special case or a reparameterization of ReMDM sampling by first noting the following result (see Appendix A.5 for the proof):

Theorem 4.1. *$q_\sigma(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x})$ from (6) is equivalent to an MDLM predictor step $q_{predictor}(\mathbf{z}_{s_p} \mid \mathbf{z}_t, \mathbf{x})$ followed by a corrector step $q_{corrector}(\mathbf{z}_{s_c} \mid \mathbf{z}_{s_p}, \mathbf{x})$ in the form of*

$$\begin{cases} \mathrm{Cat}(\mathbf{z}_{s_c}; (1-\sigma_t)\mathbf{x} + \sigma_t \boldsymbol{m}), & \mathbf{z}_{s_p} \neq \boldsymbol{m} \\ \mathrm{Cat}(\mathbf{z}_{s_c}; \frac{\sigma_t \alpha_s}{1-\alpha_s}\mathbf{x} + \frac{1-(1+\sigma_t)\alpha_s}{1-\alpha_s}\boldsymbol{m}), & \mathbf{z}_{s_p} = \boldsymbol{m}. \end{cases} \tag{11}$$

Since the only constraint on ReMDM's sampler is the bound on σ_t, ReMDM offers a general framework for the design of remasking samplers for discrete diffusion models. We now formalize the generality of ReMDM (proofs provided in Appendices A.6 & A.7).

Proposition 4.2. *The FB corrector Campbell et al. (2022) on MDLM is a special case of ReMDM where* $\sigma_t = \frac{\alpha_s - \alpha_t}{\alpha_t}$.

Proposition 4.3. *The DFM corrector Gat et al. (2024) on MDLM is a reparameterization of ReMDM where* $\sigma_t = \frac{\beta_t(\alpha_s - \alpha_t)}{\alpha_t}$. $\beta_t \in \mathbb{R}$ *denotes the corrector schecule.*

5 Experiments

5.1 ReMDM Improves Sample Quality

5.1.1 Text Generation

Experimental Setup We test the text generation capability of ReMDM with unconditional generation from models trained on OpenWebText (OWT; Gokaslan & Cohen (2019)). The OWT dataset was tokenized using the `gpt-2` tokenizer Radford et al. (2019) and sequences were wrapped to a max length of $L = 1024$. Our baselines consist of pretrained AR, SEDD Lou et al. (2024), and MDLM models with checkpoints taken from the implementation in Sahoo et al. (2024). We also compare to the MDLM pretrained models with the FB Campbell et al. (2022) and DFM Gat et al. (2024) correctors during inference. For our method, we also reuse the MDLM checkpoint and experiment with different ReMDM remasking schedules (see Appendix E.2.4 for the full set of results). We generate 5,000 samples from each model/sampler and evaluate methods using the MAUVE score Liu et al. (2021); Pillutla et al. (2021), as this metric balances sample quality and diversity. We also report the perplexity of the generated sequences under GPT-2 Large (Gen PPL.) to measure quality and average sequence entropy to reflect diversity of generated sequences as in Zheng et al. (2024).

Floating-Point Precision As indicated in Zheng et al. (2024), previous masked diffusion models Lou et al. (2024); Sahoo et al. (2024) report sampling with 32-bit floating point precision, which was found to significantly curb the diversity of generated samples. We therefore use 64-bit floating point for all text-sampling experiments.

Nucleus Sampling We observe that nucleus sampling, Holtzman et al. (2019) is critical for generating high-quality text sequences. We therefore use this strategy (with *top-p* = 0.9) for all models (except SEDD—it doesn't output logits).

Results In Table 1, we report results for the best ReMDM setting. For inference time scaling results ($T \geq 1024$), we use the max-capped schedule ($\eta_{cap} = 0.02$) in conjunction with the loop strategy ($t_{on} = 0.55, t_{off} = 0.05$, and $\alpha(t_{on}) = 0.9$ held constant in the ReMDM-loop). For faster sampling ($T < 1024$), we use the max-capped schedule ($\eta_{cap} = 0.04$) on its own. As shown in Table 1, ReMDM scales favorably with inference time compute, achieving a 15.62× MAUVE score compared to the masked diffusion models and a 2.23× MAUVE score compared to MDLM with corrector samplers (see Appendix E.2.2 for ablation study on ReMDM components). In contrast, masked diffusion models scale poorly with T and corrector sampler methods saturate when T is large. We also explore the scenario of faster sampling (i.e., $T < L = 1024$). Even with more limited computational budget, ReMDM is able to generate sequences with higher quality than the baselines.

5.1.2 Image Generation

Experimental Setup We use a pretrained MaskGiT model Chang et al. (2022) trained on ImageNet Deng et al. (2009) 256×256 samples. The images were patchified and flattened to a sequence of $L = 256$ and encoded according to a codebook with 1024 tokens Esser et al. (2021). See Chang et al. (2022) for more details about the model and training settings. We compare ReMDM's sampler to the original MaskGiT sampling and to MDLM. Although MaskGiT does not follow ancestral sampling, it is still restricted by the failure to remask property. For each sampler, we conditionally generate 50,000 images, with class labels randomly chosen from the 1,000 ImageNet categories, and we measure sample quality using Fréchet Inception Distance (FID; Heusel et al. (2017)) and Inception Score (IS; Salimans et al. (2016)). For all models, we explore the effect of scaling inference compute and tuning sampling temperature.

Table 1: ReMDM improves sample quality in inference-time scaling and faster sampling on OWT. ReMDM outperforms state-of-the-art masked diffusion models (SEDD; Lou et al. (2024), MDLM; Sahoo et al. (2024)) and masked diffusion models with corrector samplers such as Forward-Backward (FB; Campbell et al. (2022)) and Discrete Flow Matching (DFM; Gat et al. (2024)) corrector samplers. † indicates nucleus sampling. For each T, the best diffusion MAUVE score is **bolded**.

Method	MAUVE (↑)			Gen PPL. (↓)			Entropy (↑)		
Data	1.00			14.8			5.44		
AR *(T=1024)* †	0.760			12.1			5.22		
	T=1024	*T=2048*	*T=4096*	*T=1024*	*T=2048*	*T=4096*	*T=1024*	*T=2048*	*T=4096*
SEDD (absorb)	0.008	0.008	0.009	104.7	103.2	102.5	5.62	5.61	5.61
MDLM†	0.042	0.037	0.035	51.3	51.3	50.9	5.46	5.46	5.45
MDLM+FB†	0.133	0.197	0.243	33.8	28.6	22.8	5.35	5.28	5.18
MDLM+DFM†	0.254	0.294	0.269	21.7	21.0	20.7	5.20	5.19	5.17
ReMDM†	**0.403**	**0.610**	**0.656**	28.6	22.8	17.6	5.38	5.30	5.20
	T=128	*T=256*	*T=512*	*T=128*	*T=256*	*T=512*	*T=128*	*T=256*	*T=512*
SEDD (absorb)	0.007	0.007	0.008	119.2	110.1	107.2	5.65	5.63	5.62
MDLM†	0.015	0.023	0.031	61.5	55.8	53.0	5.52	5.49	5.48
MDLM+FB†	**0.064**	0.084	0.100	42.8	39.6	37.1	5.44	5.41	5.38
MDLM+DFM†	0.041	0.144	0.211	37.9	26.5	23.3	5.31	5.26	5.23
ReMDM†	0.057	**0.216**	**0.350**	42.5	30.5	21.1	5.43	5.34	5.21

Results We report the best results for each method in Table 2 (see Appendix E.3 for more details). For MaskGiT, we found best results using no temperature, and for MDLM and ReMDM, we use a temperature of 0.8. For our sampler, we report the ReMDM-rescale strategy with $\eta_{rescale} = 0.05$. Although for the smallest T decoding setting MaskGiT outperforms the other methods, we see that ReMDM has the best scaling, producing the highest quality images of any model at $T = 64$.

5.2 REMDM IMPROVES GUIDANCE

Experimental Setup We follow the setup from Schiff et al. (2024) to explore controlled small molecule generation. Specifically, we use the ∼133k molecules QM9 SMILES strings representation. (Ruddigkeit et al., 2012; Ramakrishnan et al., 2014; Weininger, 1988). Sequences were tokenized using a regular expression method Schwaller et al. (2019) and padded to a maximum length of $L = 32$. We use the D-CFG and D-CBG methods defined in Schiff et al. (2024) to conditionally generate molecules with higher ring counts (greater than 90th percentile in the original dataset). We vary the guidance strength $\gamma \in \{1, 2, 3, 4, 5\}$ and scale inference steps $T \in \{32, 64, 128\}$. Our baselines consist of an AR model guided using either CFG or the popular classifier-based FUDGE method Yang & Klein (2021), MDLM, and the uniform diffusion language model (UDLM) proposed in Schiff et al. (2024), which was also introduced to alleviate the no remasking limitation of absorbing state diffusion. For ReMDM sampling, we use pretrained MDLM models and explore various strategies defined in Section 4 (see Appendix E.4 for full search results). After generating 1,024 sequences, we use the RDKit library Landrum et al. (2013) to determine whether sequences are valid and compute novelty of the generated sequences (number of unique valid sequences that do not appear in the original QM9 dataset) and mean ring count of novel sequences.

Table 2: ReMDM produces the highest quality images on conditional discretized ImageNet generation. For each metric T pair, the best value is **bolded**.

Metric	Sampler	$T = 16$	$T = 32$	$T = 64$
FID (↓)	MaskGiT	**6.74**	**4.92**	4.85
	MDLM	7.88	5.37	4.69
	ReMDM	7.40	**4.92**	**4.45**
IS (↑)	MaskGiT	**155.32**	181.57	196.38
	MDLM	140.97	169.79	187.93
	ReMDM	145.27	**182.05**	**209.45**

Results In Figure 3, we display the trade-off that comes from increasing γ. We only visualize results for samples that have at least 50 novel sequences. For both CFG and CBG, ReMDM outperforms

AR and diffusion approaches, pushing the novelty-property maximization frontier beyond that of the baseline methods. Additionally, ReMDM scales favorably with more inference-time compute, seen by the curves for larger T dominating those for smaller T. For D-CBG, we found the best strategy to be a combination of the ReMDM-rescale (with $\eta_{rescale} = 0.9$) and ReMDM-conf schedules. For calculating confidence scores, we use the log-probabilities of the conditional denoising network. For D-CFG, we use the loop strategy, with $t_{on} = 0.25, t_{off} = 0.125$. In addition to these findings, in Appendix E.4.1, we present experimental results for maximizing a different property of interest, drug-likeness (QED; Bickerton et al. (2012)).

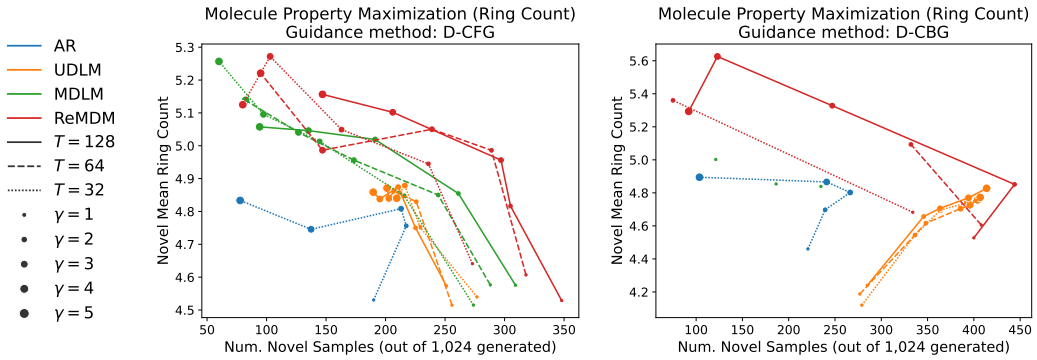


Figure 3: ReMDM improves steer-ability by extending the novelty-property maximization frontier. *(Left)* Discrete classifier-free guidance (D-CFG). *(Right)* Discrete classifier-based guidance (D-CBG) and FUDGE Yang & Klein (2021) for AR.

6 Related Works, Discussion, and Conclusion

Denoising Diffusion Implicit Models An analogy can be drawn between our work and DDIM Song et al. (2020a), where ReMDM is to MDLM for discrete signals as DDIM is to DDPM Ho et al. (2020) for continuous data. Of note, Song et al. (2020a) also present a way of adapting non-Markovian processes to discrete domains; however, they focus on uniform categorical as the limiting distribution. In Appendix B, we demonstrate how this can be adapted for absorbing state diffusion and derive an equivalence to our method.

Discrete Predictor-Corrector Samplers Continuous time Markov chain (CTMC) theory provides a framework for samplers that correct errors in the reverse process, extending the original predictor-corrector formulation for continuous data Song et al. (2020b) to discrete domains. In addition to the FB and DFM correctors discussed above, other correctors include: the Stein operator Zhao et al. (2024), and DPC Lezama et al. (2023). Unlike the plug-and-play methods of FB, DFM, and ReMDM, the Stein operator and DPC correctors require additional training.

We also conducted an analysis to find the reason of ReMDM's outperforming DFM. Unlike ReMDM, the best DFM schedule was not designed with a focus on remasking and when to utilize it. In Figure 2b, we depict the remasking schedule corresponding to results in Table 1 ($T = 4096$) for DFM and ReMDM(see full results in Appendix E.2.1). The DFM schedule shows a spike in the beginning and a sharp decay afterward. In contrast, the ReMDM-loop schedule is designed to provide non-trivial remasking probability after a good 'candidate' sequence has been generated by standard MDLM generation. Additionally, ReMDM possesses other theoretical advantages over DFM, namely the bound on σ_t and the NELBO analysis of the underlying probabilistic generative models that are not present in Gat et al. (2024).

Conclusion In this work, we present a novel family of absorbing state discrete diffusion samplers. Our method leverages the strong language modeling performance of this class of models by enabling the use of pretrained weights with the added benefit of more flexible sampling strategies that allow for remasking of predicted tokens. We demonstrate empirically that this leads to improved sample quality for unconditional generation and leads to better controlled generation. Our approach also unlocks an important inference-time compute scaling axis that is more limited for existing masked diffusion models.

REFERENCES

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993, 2021.

G Richard Bickerton, Gaia V Paolini, Jérémy Besnard, Sorel Muresan, and Andrew L Hopkins. Quantifying the chemical beauty of drugs. *Nature chemistry*, 4(2):90–98, 2012.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/jax-ml/jax`.

Andrew Campbell, Joe Benton, Valentin De Bortoli, Thomas Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. *Advances in Neural Information Processing Systems*, 35:28266–28279, 2022.

Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. *arXiv preprint arXiv:2402.04997*, 2024.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11315–11325, 2022.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12873–12883, 2021.

William Falcon and The PyTorch Lightning team. PyTorch Lightning, March 2019. URL `https://github.com/Lightning-AI/lightning`.

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky TQ Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. *arXiv preprint arXiv:2407.15595*, 2024.

Aaron Gokaslan and Vanya Cohen. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`, 2019.

Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL `https://doi.org/10.1038/s41586-020-2649-2`.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3): 90–95, 2007. doi: 10.1109/MCSE.2007.55.

Greg Landrum et al. Rdkit: A software suite for cheminformatics, computational chemistry, and predictive modeling. *Greg Landrum*, 8(31.10):5281, 2013.

Jose Lezama, Tim Salimans, Lu Jiang, Huiwen Chang, Jonathan Ho, and Irfan Essa. Discrete predictor-corrector diffusion models for image synthesis. In *The Eleventh International Conference on Learning Representations*, 2023.

Lang Liu, Krishna Pillutla, Sean Welleck, Sewoong Oh, Yejin Choi, and Zaid Harchaoui. Divergence frontiers for generative models: Sample complexity, quantization effects, and frontier integrals. *Advances in Neural Information Processing Systems*, 34:12930–12942, 2021.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Forty-first International Conference on Machine Learning*, 2024.

Nanye Ma, Shangyuan Tong, Haolin Jia, Hexiang Hu, Yu-Chuan Su, Mingda Zhang, Xuan Yang, Yandong Li, Tommi Jaakkola, Xuhui Jia, et al. Inference-time scaling for diffusion models beyond scaling denoising steps. *arXiv preprint arXiv:2501.09732*, 2025.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. *arXiv preprint arXiv:2406.03736*, 2024.

The pandas development team. pandas-dev/pandas: Pandas, February 2020. URL `https://doi.org/10.5281/zenodo.3509134`.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems*, 34:4816–4828, 2021.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

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

Lars Ruddigkeit, Ruud Van Deursen, Lorenz C Blum, and Jean-Louis Reymond. Enumeration of 166 billion organic small molecules in the chemical universe database gdb-17. *Journal of chemical information and modeling*, 52(11):2864–2875, 2012.

Subham Sekhar Sahoo, Marianne Arriola, Aaron Gokaslan, Edgar Mariano Marroquin, Alexander M Rush, Yair Schiff, Justin T Chiu, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL `https://openreview.net/forum?id=L4uaAR4ArM`.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016.

Yair Schiff, Subham Sekhar Sahoo, Hao Phung, Guanghan Wang, Sam Boshar, Hugo Dalla-torre, Bernardo P de Almeida, Alexander Rush, Thomas Pierrot, and Volodymyr Kuleshov. Simple guidance mechanisms for discrete diffusion models. *arXiv preprint arXiv:2412.10193*, 2024.

Philippe Schwaller, Teodoro Laino, Théophile Gaudin, Peter Bolgar, Christopher A Hunter, Costas Bekas, and Alpha A Lee. Molecular transformer: A model for uncertainty-calibrated chemical reaction prediction. *ACS central science*, 5(9):1572–1583, 2019.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K Titsias. Simplified and generalized masked diffusion for discrete data. *arXiv preprint arXiv:2406.04329*, 2024.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020a.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020b.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

Michael L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60): 3021, 2021. doi: 10.21105/joss.03021. URL `https://doi.org/10.21105/joss.03021`.

David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1):31–36, 1988.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

Omry Yadan. Hydra - a framework for elegantly configuring complex applications. Github, 2019. URL `https://github.com/facebookresearch/hydra`.

Kevin Yang and Dan Klein. Fudge: Controlled text generation with future discriminators. *arXiv preprint arXiv:2104.05218*, 2021.

Yixiu Zhao, Jiaxin Shi, Lester Mackey, and Scott Linderman. Informed correctors for discrete diffusion models. *arXiv preprint arXiv:2407.21243*, 2024.

Kaiwen Zheng, Yongxin Chen, Hanzi Mao, Ming-Yu Liu, Jun Zhu, and Qinsheng Zhang. Masked diffusion models are secretly time-agnostic masked models and exploit inaccurate categorical sampling. *arXiv preprint arXiv:2409.02908*, 2024.

A THEORETICAL RESULTS

A.1 PROOF OF THEOREM 3.1

Proof. We proceed by induction. The starting point $q(\mathbf{z}_1 \mid \mathbf{x}) = \mathrm{Cat}(\mathbf{z}_1; \alpha_1 \mathbf{x} + (1-\alpha_1)\boldsymbol{m})$ is true by design. For the induction hypothesis, assume that for any $t < 1$, $q(\mathbf{z}_t \mid \mathbf{x}) = \mathrm{Cat}(\mathbf{z}_t; \alpha_t \mathbf{x} + (1-\alpha_t)\boldsymbol{m})$. Recall that in absorbing state processes, given $\mathbf{x}$, for all $t \in [0,1]$, $\mathbf{z}_t \in \{\mathbf{x}, \boldsymbol{m}\}$. Then for the previous time step s, we have:

$$\begin{aligned} q(\mathbf{z}_s = \mathbf{x} \mid \mathbf{x}) &= q(\mathbf{z}_s = \mathbf{x} \mid \mathbf{z}_t = \mathbf{x}, \mathbf{x}) q(\mathbf{z}_t = \mathbf{x} \mid \mathbf{x}) + q(\mathbf{z}_s = \mathbf{x} \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) q(\mathbf{z}_t = \boldsymbol{m} \mid \mathbf{x}) \\ &= (1-\sigma_t)\alpha_t + \frac{\alpha_s - (1-\sigma_t)\alpha_t}{1-\alpha_t}(1-\alpha_t) = \alpha_s, \end{aligned} \tag{12}$$

and

$$\begin{aligned} q(\mathbf{z}_s = \boldsymbol{m} \mid \mathbf{x}) &= q(\mathbf{z}_s = \boldsymbol{m} \mid \mathbf{z}_t = \mathbf{x}, \mathbf{x}) q(\mathbf{z}_t = \mathbf{x} \mid \mathbf{x}) + q(\mathbf{z}_s = \boldsymbol{m} \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) q(\mathbf{z}_t = \boldsymbol{m} \mid \mathbf{x}) \\ &= \sigma_t \alpha_t + \frac{1 - \alpha_s - \sigma_t \alpha_t}{1-\alpha_t}(1-\alpha_t) = 1 - \alpha_s. \end{aligned} \tag{13}$$

Combining (12) and (13) yields $q(\mathbf{z}_s \mid \mathbf{x}) = \mathrm{Cat}(\mathbf{z}_s; \alpha_s \mathbf{x} + (1-\alpha_s)\boldsymbol{m})$. □

A.2 DERIVING THE NON-MARKOVIAN FORWARD STEP $q_\sigma(\mathbf{z}_t \mid \mathbf{z}_s, \mathbf{x})$

By Bayes' rule we have:

$$q_\sigma(\mathbf{z}_t \mid \mathbf{z}_s, \mathbf{x}) = \frac{q_\sigma(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x}) q_\sigma(\mathbf{z}_t \mid \mathbf{x})}{q_\sigma(\mathbf{z}_s \mid \mathbf{x})}.$$

From Theorem 3.1, we can replace the marginals $q_\sigma(\mathbf{z}_t \mid \mathbf{x})$ and $q_\sigma(\mathbf{z}_s \mid \mathbf{x})$ with those from (1).

To derive the form for $q_\sigma(\mathbf{z}_t \mid \mathbf{z}_s, \mathbf{x})$, we now look at the two possible values of $\mathbf{z}_s \neq \boldsymbol{m}$ and $\mathbf{z}_s = \boldsymbol{m}$.

Case 1a: $\mathbf{z}_s \neq \boldsymbol{m}, \mathbf{z}_t \neq \boldsymbol{m}$

$$q_\sigma(\mathbf{z}_t = \mathbf{x} \mid \mathbf{z}_s = \mathbf{x}, \mathbf{x}) = \frac{q_\sigma(\mathbf{z}_s = \mathbf{x} \mid \mathbf{z}_t = \mathbf{x}, \mathbf{x}) q_\sigma(\mathbf{z}_t = \mathbf{x} \mid \mathbf{x})}{q_\sigma(\mathbf{z}_s = \mathbf{x} \mid \mathbf{x})} = \frac{(1-\sigma_t)\alpha_t}{\alpha_s}. \tag{14}$$

Case 1b: $\mathbf{z}_s \neq \boldsymbol{m}, \mathbf{z}_t = \boldsymbol{m}$

$$\begin{aligned} q_\sigma(\mathbf{z}_t = \boldsymbol{m} \mid \mathbf{z}_s = \mathbf{x}, \mathbf{x}) &= \frac{q_\sigma(\mathbf{z}_s = \mathbf{x} \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) q_\sigma(\mathbf{z}_t = \boldsymbol{m} \mid \mathbf{x})}{q_\sigma(\mathbf{z}_s = \mathbf{x} \mid \mathbf{x})} \\ &= \frac{\left(\frac{\alpha_s - (1-\sigma_t)\alpha_t}{1-\alpha_t}\right)(1-\alpha_t)}{\alpha_s} \\ &= \frac{\alpha_s - (1-\sigma_t)\alpha_t}{\alpha_s}. \end{aligned} \tag{15}$$

Case 2a: $\mathbf{z}_s = \boldsymbol{m}, \mathbf{z}_t \neq \boldsymbol{m}$

$$q_\sigma(\mathbf{z}_t = \mathbf{x} \mid \mathbf{z}_s = \boldsymbol{m}, \mathbf{x}) = \frac{q_\sigma(\mathbf{z}_s = \boldsymbol{m} \mid \mathbf{z}_t = \mathbf{x}, \mathbf{x}) q_\sigma(\mathbf{z}_t = \mathbf{x} \mid \mathbf{x})}{q_\sigma(\mathbf{z}_s = \boldsymbol{m} \mid \mathbf{x})} = \frac{\sigma_t \alpha_t}{1-\alpha_s}. \tag{16}$$

Case 2b: $\mathbf{z}_s = \boldsymbol{m}, \mathbf{z}_t = \boldsymbol{m}$

$$\begin{aligned} q_\sigma(\mathbf{z}_t = \boldsymbol{m} \mid \mathbf{z}_s = \boldsymbol{m}, \mathbf{x}) &= \frac{q_\sigma(\mathbf{z}_s = \boldsymbol{m} \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) q_\sigma(\mathbf{z}_t = \mathbf{x} \mid \mathbf{x})}{q_\sigma(\mathbf{z}_s = \boldsymbol{m} \mid \mathbf{x})} \\ &= \frac{\left(\frac{1-\alpha_s - \sigma_t \alpha_t}{1-\alpha_t}\right)(1-\alpha_t)}{1-\alpha_s} \\ &= \frac{1-\alpha_s - \sigma_t \alpha_t}{1-\alpha_s}. \end{aligned} \tag{17}$$

Combining (14), (15), (16), and (17) yields the non-Markovian forward process:

$$q_\sigma(\mathbf{z}_t \mid \mathbf{z}_s, \mathbf{x}) = \begin{cases} \mathrm{Cat}(\mathbf{z}_t; \frac{(1-\sigma_t)\alpha_t}{\alpha_s}\mathbf{x} + \frac{\alpha_s - (1-\sigma_t)\alpha_t}{\alpha_s}\boldsymbol{m}), & \mathbf{z}_s \neq \boldsymbol{m} \\ \mathrm{Cat}(\mathbf{z}_t; \frac{\sigma_t \alpha_t}{1-\alpha_s}\mathbf{x} + \frac{1-\alpha_s - \sigma_t \alpha_t}{1-\alpha_s}\boldsymbol{m}), & \mathbf{z}_s = \boldsymbol{m} \end{cases}. \tag{18}$$

A.3 Deriving Bounds for σ_t

To ensure that we have defined a valid probability, in both cases of the posterior in (6), since $\mathbf{x}^\top \boldsymbol{m} = 0$, we require that the coefficients on both terms be within the range $[0, 1]$. Using this, we can derive the bounds for σ_t given in (7).

For case where $\mathbf{z}_t \neq \boldsymbol{m}$, both coefficients must satisfy:

$$0 \leq \sigma_t \leq 1. \tag{19}$$

For the case where $\mathbf{z}_t = \boldsymbol{m}$, ensuring that the coefficients are in the range $[0, 1]$ leads to the restriction that:

$$\frac{\alpha_t - \alpha_s}{\alpha_t} \leq \sigma_t \leq \frac{1 - \alpha_s}{\alpha_t}. \tag{20}$$

Since the noise schedule is monotonically decreasing in t, the lower bound in (20) is ≤ 0. Hence, combining (19) and (20) produces the bounds in (7).

A.4 Deriving the ReMDM NELBO

The variational objective or negative evidence lower bound (NELBO) for diffusion models has the following form Sohl-Dickstein et al. (2015):

$$\begin{aligned}
\mathcal{L} &= \mathbb{E}_{\mathbf{z}_{0:T} \sim q(\mathbf{z}_{0:1}|\mathbf{x})} \big[\log(q(\mathbf{z}_{0:1} \mid \mathbf{x}) \;/\; p_\theta(\mathbf{x}) p_\theta(\mathbf{z}_{0:1} \mid \mathbf{x})) \big] \\
&= \mathbb{E}_{\mathbf{z}_{0:1} \sim q(\mathbf{z}_{0:1}|\mathbf{x})} \Bigg[\underbrace{-\log p_\theta(\mathbf{x}|\mathbf{z}_0)}_{\mathcal{L}_{recon}} + \underbrace{D_{\mathrm{KL}}[q(\mathbf{z}_1|\mathbf{x}) \| p_\theta(\mathbf{z}_1)]}_{\mathcal{L}_{prior}} + \underbrace{\sum_{i=1}^{T} D_{\mathrm{KL}}[q(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x}) \| p_\theta(\mathbf{z}_s \mid \mathbf{z}_t)]}_{\mathcal{L}_{diffusion}} \Bigg],
\end{aligned} \tag{21}$$

where D_{KL} denotes the Kullback–Leibler divergence.

Starting from (21), we note that, in practice, we set $p_\theta(\mathbf{z}_1) = q(\mathbf{z}_1 \mid x) = q(\mathbf{z}_1) = \boldsymbol{\pi}$ which ensures $\mathcal{L}_{prior} = 0$. Additionally, the reconstruction loss $\mathcal{L}_{recon}$ is equivalent in both (21) and (9). We therefore turn our attention to the ReMDM diffusion loss term

$$\mathcal{L}^{\sigma}_{diffusion} = \sum_{j=1}^{T} D_{\mathrm{KL}}[q_\sigma(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x}) \| p_\theta(\mathbf{z}_s \mid \mathbf{z}_t)] \tag{22}$$

Additionally, we note that in Sahoo et al. (2024), MDLM is parameterized using the denoising $\mathbf{x}_\theta$ that is restricted in two ways. First the denoising model places zero probability mass on the [MASK] token, i.e., $\mathbf{x}_\theta^\top \boldsymbol{m} = 0$. Second the model 'carries-over' unmasked tokens, i.e., $\mathbf{x}_\theta^\top \mathbf{z}_t = 1$, if $\mathbf{z}_t \neq \boldsymbol{m}$. Below we assume that our denoising network follows a similar parameterization.

We break down each term in this summation by the two possible values $\mathbf{z}_t \neq \boldsymbol{m}$ and $\mathbf{z}_t = \boldsymbol{m}$.

Case 1: $\mathbf{z}_t \neq \boldsymbol{m}$

$$\begin{aligned}
&D_{\mathrm{KL}}[q_\sigma(\mathbf{z}_s \mid \mathbf{z}_t = \mathbf{x}, \mathbf{x}) \| p_\theta(\mathbf{z}_s \mid \mathbf{z}_t = \mathbf{x})] \\
&= q_\sigma(\mathbf{z}_s = \mathbf{x} \mid \mathbf{z}_t = \mathbf{x}, \mathbf{x}) \log \frac{q_\sigma(\mathbf{z}_s = \mathbf{x} \mid \mathbf{z}_t = \mathbf{x}, \mathbf{x})}{p_\theta(\mathbf{z}_s = \mathbf{x} \mid \mathbf{z}_t = \mathbf{x})} + q_\sigma(\mathbf{z}_s = \boldsymbol{m} \mid \mathbf{z}_t = \mathbf{x}, \mathbf{x}) \log \frac{q_\sigma(\mathbf{z}_s = \boldsymbol{m} \mid \mathbf{z}_t = \mathbf{x}, \mathbf{x})}{p_\theta(\mathbf{z}_s = \boldsymbol{m} \mid \mathbf{z}_t = \mathbf{x})} \\
&= (1 - \sigma_t) \log \frac{1 - \sigma_t}{1 - \sigma_t} + \sigma_t \log \frac{\sigma_t}{\sigma_t} = 0.
\end{aligned} \tag{23}$$

Case 2: $\mathbf{z}_t = \boldsymbol{m}$

$$\begin{aligned}
&D_{\mathrm{KL}}[q_\sigma(\mathbf{z}_s \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) \| p_\theta(\mathbf{z}_s \mid \mathbf{z}_t = \boldsymbol{m})] \\
&= q_\sigma(\mathbf{z}_s = \mathbf{x} \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) \log \frac{q_\sigma(\mathbf{z}_s = \mathbf{x} \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x})}{p_\theta(\mathbf{z}_s = \mathbf{x} \mid \mathbf{z}_t = \boldsymbol{m})} + q_\sigma(\mathbf{z}_s = \boldsymbol{m} \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) \log \frac{q_\sigma(\mathbf{z}_s = \boldsymbol{m} \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x})}{p_\theta(\mathbf{z}_s = \boldsymbol{m} \mid \mathbf{z}_t = \boldsymbol{m})} \\
&= \frac{\alpha_s - (1 - \sigma_t)\alpha_t}{1 - \alpha_t} \log \frac{\frac{\alpha_s - (1-\sigma_t)\alpha_t}{1-\alpha_t}}{\frac{\alpha_s - (1-\sigma_t)\alpha_t}{1-\alpha_t} \mathbf{x}^\top \mathbf{x}_\theta} + \frac{1 - \alpha_s - \sigma_t \alpha_t}{1 - \alpha_t} \log \frac{\frac{1-\alpha_s-\sigma_t\alpha_t}{1-\alpha_t}}{\frac{1-\alpha_s-\sigma_t\alpha_t}{1-\alpha_t}} \\
&= \frac{(1 - \sigma_t)\alpha_t - \alpha_s}{1 - \alpha_t} \log(\mathbf{x}^\top \mathbf{x}_\theta)
\end{aligned} \tag{24}$$

The carry-over unmasked tokens property of $\mathbf{x}_\theta$ implies that $\log(\mathbf{x}^\top \mathbf{x}_\theta) = \log(1) = 0$, which lets us write the two cases from (23) and (24) as a single expression:

$$D_{\mathrm{KL}}[q_\sigma(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x}) \| p_\theta(\mathbf{z}_s \mid \mathbf{z}_t)] = \frac{(1-\sigma_t)\alpha_t - \alpha_s}{1-\alpha_t} \log(\mathbf{x}^\top \mathbf{x}_\theta) \tag{25}$$

Then rewriting the summation in (22) as an expectation with t sampled uniformly from $\{\frac{1}{T}, \ldots, 1\}$ and plugging in (25), we recover the diffusion loss term in (9).

A.5 Proof of Theorem 4.1

Proof. Recall that the MDLM *predictor* step amounts to drawing a sample $\mathbf{z}_{s_p}$ from the posterior given in (2), which we can reformulate as:

$$q(\mathbf{z}_{s_p} \mid \mathbf{z}_t, \mathbf{x}) = \begin{cases} \mathrm{Cat}(\mathbf{z}_{s_p}; \mathbf{z}_t), & \mathbf{z}_t \neq \boldsymbol{m}, \\ \mathrm{Cat}(\mathbf{z}_{s_p}; \frac{\alpha_s - \alpha_t}{1-\alpha_t}\mathbf{x} + \frac{1-\alpha_s}{1-\alpha_t}\boldsymbol{m}), & \mathbf{z}_t = \boldsymbol{m}. \end{cases}$$

To prove the theorem statement, we must show that $q(\mathbf{z}_{s_c} \mid \mathbf{z}_t, \mathbf{x})$ is equivalent to the ReMDM posterior from (6). We begin by noting that conditioned on the predictor sample $\mathbf{z}_{s_p}$, the corrector sample $\mathbf{z}_{s_c}$ is independent of $\mathbf{z}_t$. That is

$$q(\mathbf{z}_{s_c} \mid \mathbf{z}_{s_p}, \mathbf{z}_t, \mathbf{x}) = q(\mathbf{z}_{s_c} \mid \mathbf{z}_{s_p}, \mathbf{x}) \tag{26}$$

We now look at the two cases of $\mathbf{z}_t \neq \boldsymbol{m}$ and $\mathbf{z}_t = \boldsymbol{m}$:

Case 1a: $\mathbf{z}_t \neq \boldsymbol{m}, \mathbf{z}_{s_c} = \mathbf{x}$

$$\begin{aligned} q(\mathbf{z}_{s_c} = \mathbf{x} \mid \mathbf{z}_t = \mathbf{x}, \mathbf{x}) &= \sum_{\mathbf{z}' \in \{\mathbf{x}, \boldsymbol{m}\}} q(\mathbf{z}_{s_c} = \mathbf{x} \mid \mathbf{z}_{s_p} = \mathbf{z}', \mathbf{z}_t = \mathbf{x}, \mathbf{x}) q(\mathbf{z}_{s_p} = \mathbf{z}' \mid \mathbf{z}_t = \mathbf{x}, \mathbf{x}) \\ &= q(\mathbf{z}_{s_c} = \mathbf{x} \mid \mathbf{z}_{s_p} = \mathbf{x}, \mathbf{x}) q(\mathbf{z}_{s_p} = \mathbf{x} \mid \mathbf{z}_t = \mathbf{x}, \mathbf{x}) + q(\mathbf{z}_{s_c} = \mathbf{x} \mid \mathbf{z}_{s_p} = \boldsymbol{m}, \mathbf{x}) q(\mathbf{z}_{s_p} = \mathbf{m} \mid \mathbf{z}_t = \mathbf{x}, \mathbf{x}) \\ &= q(\mathbf{z}_{s_c} = \mathbf{x} \mid \mathbf{z}_{s_p} = \mathbf{x}, \mathbf{x}) = (1 - \sigma_t). \end{aligned} \tag{27}$$

Case 1b: $\mathbf{z}_t \neq \boldsymbol{m}, \mathbf{z}_{s_c} = \boldsymbol{m}$ Using the same argument as in Case 1a, we have that

$$q(\mathbf{z}_{s_c} = \mathbf{m} \mid \mathbf{z}_t = \mathbf{x}, \mathbf{x}) = q(\mathbf{z}_{s_c} = \mathbf{m} \mid \mathbf{z}_{s_p} = \mathbf{x}, \mathbf{x}) = \sigma_t. \tag{28}$$

Case 2a: $\mathbf{z}_t = \boldsymbol{m}, \mathbf{z}_{s_c} = \mathbf{x}$

$$\begin{aligned} q(\mathbf{z}_{s_c} = \mathbf{x} \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) &= \sum_{\mathbf{z}' \in \{\mathbf{x}, \boldsymbol{m}\}} q(\mathbf{z}_{s_c} = \mathbf{x} \mid \mathbf{z}_{s_p} = \mathbf{z}', \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) q(\mathbf{z}_{s_p} = \mathbf{z}' \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) \\ &= q(\mathbf{z}_{s_c} = \mathbf{x} \mid \mathbf{z}_{s_p} = \mathbf{x}, \mathbf{x}) q(\mathbf{z}_{s_p} = \mathbf{x} \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) + q(\mathbf{z}_{s_c} = \mathbf{x} \mid \mathbf{z}_{s_p} = \boldsymbol{m}, \mathbf{x}) q(\mathbf{z}_{s_p} = \mathbf{m} \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) \\ &= (1 - \sigma_t)\Big(\frac{\alpha_s - \alpha_t}{1 - \alpha_t}\Big) + \Big(\frac{\sigma_t \alpha_s}{1 - \alpha_s}\Big)\Big(\frac{1 - \alpha_s}{1 - \alpha_t}\Big) \\ &= \frac{\alpha_s - (1 - \sigma_t)\alpha_t}{1 - \alpha_t}. \end{aligned} \tag{29}$$

Case 2b: $\mathbf{z}_t = \boldsymbol{m}, \mathbf{z}_{s_c} = \boldsymbol{m}$

$$\begin{aligned} q(\mathbf{z}_{s_c} = \boldsymbol{m} \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) &= \sum_{\mathbf{z}' \in \{\mathbf{x}, \boldsymbol{m}\}} q(\mathbf{z}_{s_c} = \mathbf{m} \mid \mathbf{z}_{s_p} = \mathbf{z}', \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) q(\mathbf{z}_{s_p} = \mathbf{z}' \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) \\ &= q(\mathbf{z}_{s_c} = \mathbf{m} \mid \mathbf{z}_{s_p} = \mathbf{x}, \mathbf{x}) q(\mathbf{z}_{s_p} = \mathbf{x} \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) + q(\mathbf{z}_{s_c} = \mathbf{m} \mid \mathbf{z}_{s_p} = \boldsymbol{m}, \mathbf{x}) q(\mathbf{z}_{s_p} = \mathbf{m} \mid \mathbf{z}_t = \boldsymbol{m}, \mathbf{x}) \\ &= \sigma_t\Big(\frac{\alpha_s - \alpha_t}{1 - \alpha_t}\Big) + \Big(\frac{1 - (1 + \sigma_t)\alpha_s}{1 - \alpha_s}\Big)\Big(\frac{1 - \alpha_s}{1 - \alpha_t}\Big) \\ &= \frac{1 - \alpha_s - \sigma_t \alpha_t}{1 - \alpha_t}. \end{aligned} \tag{30}$$

Combining (27), (28), (29), and (30) yields the desired result. □

A.6 PROOF OF PROPOSITION 4.2

Proof. The forward-backward corrector sampler Campbell et al. (2022) is derived using continuous time Markov chain theory. In particular, the rate matrix for the corrector sampler step $\mathbf{R}_{corrector}$ is the sum of the forward rate matrix $\mathbf{R}_t$ and the backward rate matrix $\tilde{\mathbf{R}}_t$, i.e. $\mathbf{R}_{corrector} = \mathbf{R}_t + \tilde{\mathbf{R}}_t$.

We first extract the forward rate matrix $\mathbf{R}_t$ of MDLM using the following discretization formula.

$$q(\mathbf{z}_t = \mathbf{y}' \mid \mathbf{z}_s = \mathbf{y}) = \delta_{\mathbf{y}',\mathbf{y}} + \mathbf{R}_t(\mathbf{y}, \mathbf{y}')\Delta t + o(\Delta t) \tag{31}$$

From Sahoo et al. (2024), we know that the forward step of MDLM $q(\mathbf{z}_t \mid \mathbf{z}_s) = \mathrm{Cat}(\mathbf{z}_t; \frac{\alpha_t}{\alpha_s}\mathbf{z}_s + (1 - \frac{\alpha_t}{\alpha_s})\boldsymbol{m})$. Let $\mathcal{V}$ denote the vocabulary set.

Case 1a: $\mathbf{z}_s = \mathbf{x} \in \mathcal{V} \setminus \boldsymbol{m}, \mathbf{z}_t = \mathbf{x}$

$$\begin{aligned} 1 + \mathbf{R}_t(\mathbf{x}, \mathbf{x})\Delta t + o(\Delta t) &= q(\mathbf{z}_t = \mathbf{x} \mid \mathbf{z}_s = \mathbf{x}) = \frac{\alpha_t}{\alpha_s} \\ \Rightarrow \mathbf{R}_t(\mathbf{x}, \mathbf{x}) &= \lim_{\Delta t \to 0} \frac{1}{\Delta t}(\frac{\alpha_t}{\alpha_s} - 1) = \frac{\alpha_t'}{\alpha_t} \end{aligned} \tag{32}$$

Case 1b: $\mathbf{z}_s = \mathbf{x} \in \mathcal{V} \setminus \boldsymbol{m}, \mathbf{z}_t = \boldsymbol{m}$

$$\begin{aligned} \mathbf{R}_t(\mathbf{x}, \boldsymbol{m})\Delta t + o(\Delta t) &= q(\mathbf{z}_t = \boldsymbol{m} \mid \mathbf{z}_s = \mathbf{x}) = 1 - \frac{\alpha_t}{\alpha_s} \\ \Rightarrow \mathbf{R}_t(\mathbf{x}, \boldsymbol{m}) &= \lim_{\Delta t \to 0} \frac{1}{\Delta t}(1 - \frac{\alpha_t}{\alpha_s}) = -\frac{\alpha_t'}{\alpha_t} \end{aligned} \tag{33}$$

Case 1c: $\mathbf{z}_s = \mathbf{x} \in \mathcal{V} \setminus \boldsymbol{m}, \mathbf{z}_t = \mathbf{y} \in \mathcal{V} \setminus \{\mathbf{x}, \boldsymbol{m}\}$

$$\mathbf{R}_t(\mathbf{x}, \mathbf{y})\Delta t + o(\Delta t) = q(\mathbf{z}_t = \mathbf{y} \mid \mathbf{z}_s = \mathbf{x}) = 0 \Rightarrow \mathbf{R}_t(\mathbf{x}, \mathbf{y}) = 0 \tag{34}$$

Case 1d: $\mathbf{z}_s = \boldsymbol{m}, \mathbf{z}_t = \boldsymbol{m}$

$$1 + \mathbf{R}_t(\boldsymbol{m}, \boldsymbol{m})\Delta t + o(\Delta t) = q(\mathbf{z}_t = \boldsymbol{m} \mid \mathbf{z}_s = \boldsymbol{m}) = 1 \Rightarrow \mathbf{R}_t(\boldsymbol{m}, \boldsymbol{m}) = 0 \tag{35}$$

Case 1e: $\mathbf{z}_s = \boldsymbol{m}, \mathbf{z}_t = \mathbf{x} \in \mathcal{V} \setminus \boldsymbol{m}$

$$\mathbf{R}_t(\boldsymbol{m}, \mathbf{x})\Delta t + o(\Delta t) = q(\mathbf{z}_t = \mathbf{x} \mid \mathbf{z}_s = \boldsymbol{m}) = 0 \Rightarrow \mathbf{R}_t(\boldsymbol{m}, \mathbf{x}) = 0 \tag{36}$$

Similarly, we can derive the backward rate matrix $\tilde{\mathbf{R}}_t$ using the following formula and MDLM posterior (2).

$$q(\mathbf{z}_s = \mathbf{y}' \mid \mathbf{z}_t = \mathbf{y}, \mathbf{x}) = \delta_{\mathbf{y}',\mathbf{y}} + \mathbf{R}_t(\mathbf{y}, \mathbf{y}')\Delta t + o(\Delta t) \tag{37}$$

Case 2a: $\mathbf{z}_t = \mathbf{x} \in \mathcal{V} \setminus \boldsymbol{m}, \mathbf{z}_s = \mathbf{x}$

$$1 + \tilde{\mathbf{R}}_t(\mathbf{x}, \mathbf{x})\Delta t + o(\Delta t) = q(\mathbf{z}_s = \mathbf{x} \mid \mathbf{z}_t = \mathbf{x}) = 1 \Rightarrow \tilde{\mathbf{R}}_t(\mathbf{x}, \mathbf{x}) = 0 \tag{38}$$

Case 2b: $\mathbf{z}_t = \mathbf{x} \in \mathcal{V} \setminus \boldsymbol{m}, \mathbf{z}_s = \mathbf{y} \in \mathcal{V} \setminus \mathbf{x}$

$$\tilde{\mathbf{R}}_t(\mathbf{x}, \mathbf{y})\Delta t + o(\Delta t) = q(\mathbf{z}_s = \mathbf{y} \mid \mathbf{z}_t = \mathbf{x}) = 0 \Rightarrow \tilde{\mathbf{R}}_t(\mathbf{x}, \mathbf{y}) = 0 \tag{39}$$

Case 2c: $\mathbf{z}_t = \boldsymbol{m}, \mathbf{z}_s = \mathbf{x}$

$$\begin{aligned} \tilde{\mathbf{R}}_t(\boldsymbol{m}, \mathbf{x})\Delta t + o(\Delta t) &= q(\mathbf{z}_s = \mathbf{x} \mid \mathbf{z}_t = \boldsymbol{m}) = \frac{\alpha_s - \alpha_t}{1 - \alpha_t} \\ \Rightarrow \tilde{\mathbf{R}}_t(\boldsymbol{m}, \mathbf{x}) &= \lim_{\Delta t \to 0} \frac{1}{\Delta t}\frac{\alpha_s - \alpha_t}{1 - \alpha_t} = -\frac{\alpha_t'}{1 - \alpha_t} \end{aligned} \tag{40}$$

Case 2d: $\mathbf{z}_t = \boldsymbol{m}, \mathbf{z}_s = \boldsymbol{m}$

$$1 + \tilde{\mathbf{R}}_t(\boldsymbol{m}, \boldsymbol{m})\Delta t + o(\Delta t) = q(\mathbf{z}_s = \boldsymbol{m} \mid \mathbf{z}_t = \boldsymbol{m}) = \frac{1-\alpha_s}{1-\alpha_t}$$
$$\Rightarrow \tilde{\mathbf{R}}_t(\boldsymbol{m}, \boldsymbol{m}) = \lim_{\Delta t \to 0} \frac{1}{\Delta t}\left(\frac{1-\alpha_s}{1-\alpha_t} - 1\right) = \frac{\alpha_t'}{1-\alpha_t} \tag{41}$$

Case 2e: $\mathbf{z}_t = \boldsymbol{m}, \mathbf{z}_s = \mathbf{y} \in \mathcal{V} \setminus \{\mathbf{x}, \boldsymbol{m}\}$

$$\tilde{\mathbf{R}}_t(\boldsymbol{m}, \mathbf{y})\Delta t + o(\Delta t) = q(\mathbf{z}_s = \boldsymbol{m} \mid \mathbf{z}_t = \mathbf{y}) = 0 \Rightarrow \tilde{\mathbf{R}}_t(\boldsymbol{m}, \boldsymbol{m}) = 0 \tag{42}$$

By adding $\mathbf{R}_t$ and $\tilde{\mathbf{R}}_t$, we get the rate matrix of the forward-backward corrector step. The next step is to discretize it. Concretely, we apply (37) to the forward-backward rate matrix.

$$q_{FB}(\mathbf{z}_{s_c} \mid \mathbf{z}_{s_p}, \mathbf{x}) = \begin{cases} \mathrm{Cat}(\mathbf{z}_{s_c}; \frac{2\alpha_t - \alpha_s}{\alpha_t}\mathbf{x} + \frac{\alpha_s - \alpha_t}{\alpha_t}\boldsymbol{m}), & \mathbf{z}_{s_p} \neq \boldsymbol{m} \\ \mathrm{Cat}(\mathbf{z}_{s_c}; \frac{\alpha_s - \alpha_t}{1-\alpha_t}\mathbf{x} + \frac{1-\alpha_s}{1-\alpha_t}\boldsymbol{m}), & \mathbf{z}_{s_p} = \boldsymbol{m}. \end{cases} \tag{43}$$

Note that (43) keeps the marginal at time t unchanged while (11) keeps the marginal at time s unchanged. In order to conduct a fair comparison, we rewrite the time t version of (11) as follows.

$$q_{ReMDM}^{corrector}(\mathbf{z}_{s_c} \mid \mathbf{z}_{s_p}, \mathbf{x}) = \begin{cases} \mathrm{Cat}(\mathbf{z}_{s_c}; (1-\sigma_t)\mathbf{x} + \sigma_t\boldsymbol{m}), & \mathbf{z}_{s_p} \neq \boldsymbol{m} \\ \mathrm{Cat}(\mathbf{z}_{s_c}; \frac{\sigma_t\alpha_t}{1-\alpha_t}\mathbf{x} + \frac{1-(1+\sigma_t)\alpha_t}{1-\alpha_t}\boldsymbol{m}), & \mathbf{z}_{s_p} = \boldsymbol{m}. \end{cases} \tag{44}$$

Comparing (43) and (44), we can find that (43) is a special case where $\sigma_t = \frac{\alpha_s - \alpha_t}{\alpha_t}$. □

A.7 Proof of Proposition 4.3

Proof. The DFM corrector sampler Gat et al. (2024) is derived from a generating velocity u_t^{corr} using the following transformation equation.

$$\mathbf{z}_{s_c} \sim \delta_{\mathbf{z}_t}(\cdot) + u_t^{corr}(\cdot, \mathbf{z}_t)\Delta t \tag{45}$$

The corrector generating velocity is defined as a weighted sum of the forward sampling generating velocity $\hat{u}_t$ and the backward sampling generating velocity $\check{u}_t$.

$$u_t^{corr}(\cdot, \mathbf{z}_t) = (1+\beta_t)\hat{u}_t(\cdot, \mathbf{z}_t) - \beta_t\check{u}_t(\cdot, \mathbf{z}_t) \tag{46}$$

The weighting coefficient $\beta_t \in \mathbb{R}$ is referred to as the corrector schedule and can be user-specified.

The forward and backward sampling generating velocities take the following forms:

$$\hat{u}_t(\cdot, \mathbf{z}_t) = -\frac{\alpha_t'}{1-\alpha_t}\left[p_{0|t}(\cdot \mid \mathbf{z}_t) - \delta_{\mathbf{z}_t}(\cdot)\right] \tag{47}$$

$$\check{u}_t(\cdot, \mathbf{z}_t) = -\frac{\alpha_t'}{\alpha_t}\left[\delta_{\mathbf{z}_t}(\cdot) - p_{1|t}(\cdot \mid \mathbf{z}_t)\right] \tag{48}$$

Note that our formulation is slightly different from the original presentation in Gat et al. (2024), since in our notation as t goes from 1 to 0, we move from noise to the target distribution, whereas in the flow matching literature this direction is reversed. Plugging (47) and (48) into (46), we derive the following form for u_t^{corr}:

$$u_t^{corr}(\cdot, \mathbf{z}_t) = \left[\frac{(1+\beta_t)\alpha_t'}{1-\alpha_t} + \frac{\beta_t\alpha_t'}{\alpha_t}\right]\delta_{\mathbf{z}_t}(\cdot) - \frac{(1+\beta_t)\alpha_t'}{1-\alpha_t}p_{0|t}(\cdot \mid \mathbf{z}_t) - \frac{\beta_t\alpha_t'}{\alpha_t}p_{1|t}(\cdot \mid \mathbf{z}_t) \tag{49}$$

By plugging (49) into (45), we have that

$$\begin{aligned}\mathbf{z}_{s_c} &\sim \big[1 + \frac{(1+\beta_t)\alpha_t'\Delta t}{1-\alpha_t} + \frac{\beta_t\alpha_t'\Delta t}{\alpha_t}\big]\delta_{\mathbf{z}_t}(\cdot) - \frac{(1+\beta_t)\alpha_t'\Delta t}{1-\alpha_t}p_{0|t}(\cdot \mid \mathbf{z}_t) - \frac{\beta_t\alpha_t'\Delta t}{\alpha_t}p_{1|t}(\cdot \mid \mathbf{z}_t) \\ &\sim \big[1 + \frac{(1+\beta_t)(\alpha_t-\alpha_s)}{1-\alpha_t} + \frac{\beta_t(\alpha_t-\alpha_s)}{\alpha_t}\big]\delta_{\mathbf{z}_t}(\cdot) - \frac{(1+\beta_t)(\alpha_t-\alpha_s)}{1-\alpha_t}p_{0|t}(\cdot \mid \mathbf{z}_t) - \frac{\beta_t(\alpha_t-\alpha_s)}{\alpha_t}p_{1|t}(\cdot \mid \mathbf{z}_t)\end{aligned} \tag{50}$$

We can rewrite this as

$$q_{DFM}(\mathbf{z}_{s_c} \mid \mathbf{z}_t, \mathbf{x}) = \begin{cases}\mathrm{Cat}(\mathbf{z}_{s_c}; (1 + \frac{\beta_t(\alpha_t-\alpha_s)}{\alpha_t})\mathbf{x} + \frac{\beta_t(\alpha_s-\alpha_t)}{\alpha_t}\boldsymbol{m}), & \mathbf{z}_t \neq \boldsymbol{m} \\ \mathrm{Cat}(\mathbf{z}_{s_c}; \frac{(1+\beta_t)(\alpha_s-\alpha_t)}{1-\alpha_t}\mathbf{x} + (1 + \frac{(1+\beta_t)(\alpha_t-\alpha_s)}{1-\alpha_t})\boldsymbol{m}), & \mathbf{z}_t = \boldsymbol{m}.\end{cases} \tag{51}$$

Comparing (51) and (6), we see that (51) is equivalent to (6) where $\sigma_t = \frac{\beta_t(\alpha_s-\alpha_t)}{\alpha_t}$.

□

B Comparison to DDIM Non-Markovian Processes

In DDIM Song et al. (2020a), the authors present non-Markovian forward processes for both continuous and discrete signals. For discrete data, although in DDIM the proposed method assumes a uniform categorical as the limiting distribution, here we demonstrate how one can adapt the proposed method in DDIM for absorbing state diffusion and derive an equivalence between ReMDM and this new DDIM process under a reparameterization of σ_t. For clarity, we use σ_t^{DDIM} to denote the parameter from DDIM and σ_t^{ReMDM} to denote the parameter used in our work.

In DDIM, the processes assuming a uniform categorical distribution as the limiting distribution is defined to have the following posteriors:

$$q(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x}) = (\alpha_s - \sigma_t\alpha_t)\mathbf{x} + \sigma_t^{DDIM}\mathbf{z}_t + (1 - \alpha_s - (1-\alpha_t)\sigma_t^{DDIM})|V|^{-1}\mathbf{1}, \tag{52}$$

where $\mathbf{1}$ is a column vector of ones and $|V|$ is the vocabulary size. We can apply the methodology from DDIM to absorbing state diffusion by replacing the limiting distribution $\mathbf{1}/|V|$ in (52) with $\boldsymbol{m}$, which produces:

$$\begin{aligned}q(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x}) &= (\alpha_s - \sigma_t\alpha_t)\mathbf{x} + \sigma_t^{DDIM}\mathbf{z}_t + (1 - \alpha_s - (1-\alpha_t)\sigma_t)\boldsymbol{m} \\ &= \begin{cases}(\alpha_s + \sigma_t^{DDIM}(1-\alpha_t))\mathbf{x} + ((1-\alpha_s) - (1-\alpha_t)\sigma_t^{DDIM})\boldsymbol{m} & \mathbf{z}_t \neq \boldsymbol{m} \\ (\alpha_s - \sigma_t^{DDIM}\alpha_t)\mathbf{x} + (1 - \alpha_s + \alpha_t\sigma_t^{DDIM})\boldsymbol{m} & \mathbf{z}_t = \boldsymbol{m}\end{cases}\end{aligned} \tag{53}$$

Comparing the $\mathbf{z}_t \neq \boldsymbol{m}$ case in (53) to that in the ReMDM posterior in (6), we can derive an equivalence between our proposed method and that from DDIM with the following reparameterization:

$$\sigma_t^{ReMDM} = 1 - \alpha_s - (1-\alpha_t)\sigma_t^{DDIM}. \tag{54}$$

Plugging this reparameterization into the $\mathbf{z}_t = \boldsymbol{m}$ case in (6) yields an equivalence to the $\mathbf{z}_t = \boldsymbol{m}$ case in (53) as well.

In addition to extending the discrete formulation from DDIM to absorbing state processes, we believe that our formulation is easier to analyze relative to the one when reparameterizing to use σ_t^{DDIM}, e.g., deriving bounds on σ_t is more straightforward for our work and it is more natural to explore the various design decisions defined in Section 4 using our formulation.

C ReMDM Sampler Algorithms

Below we present the high-level ReMDM sampling pseudocode in Algorithm 1, and algorithms for ReMDM-switch (Algorithm 2) and ReMDM-loop (Algorithm 3) both of which can optionally combine with the various strategies for setting σ_t described in Section 4.1.

Algorithm 1 Sampling with ReMDM.

`// Differences to standard MDLM sampling noted in` **`brown`**.
Input: pre-trained denoising network $\mathbf{x}_\theta$ (e.g., MDLM), number of timesteps T, noise schedule α_t, remasking schedule σ_t.
Initialize $\mathbf{z}_t = \boldsymbol{m}$.
for $i = T$ **to** 1 **do**
  $t = i/T, s = (i-1)/T$.
  Set α_t, α_s according to noise schedule.
  Set $\sigma_t \in [0, \sigma_t^{max}]$ according to remasking schedule.
  Compute approximate posterior:

$$p_\theta(\mathbf{z}_s \mid \mathbf{z}_t) = q_\sigma(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x} = \mathbf{x}_\theta(\mathbf{z}_t))$$
$$= \begin{cases} \mathrm{Cat}(\mathbf{z}_s; (1-\sigma_t)\mathbf{x}_\theta + \sigma_t \boldsymbol{m}), & \mathbf{z}_t \neq \boldsymbol{m} \\ \mathrm{Cat}(\mathbf{z}_s; \frac{\alpha_s - (1-\sigma_t)\alpha_t}{1-\alpha_t}\mathbf{x}_\theta + \frac{1-\alpha_s-\sigma_t\alpha_t}{1-\alpha_t}\boldsymbol{m}), & \mathbf{z}_t = \boldsymbol{m} \end{cases}$$

  Sample $\mathbf{z}_s \sim p_\theta$.
  Set $\mathbf{z}_t = \mathbf{z}_s$.
end for
Output: $\mathbf{z}_t$.

D Additional Experimental Details

D.1 OpenWebText

In this experiment, we reuse the pretrained AR, SEDD, and MDLM checkpoints released by Sahoo et al. (2024) where the diffusion models are trained using a log-linear schedule, i.e., $\alpha_t = 1 - t$. AR, SEDD, and MDLM share the same architecture: a Transformer-based model Vaswani (2017) that augments the diffusion transformer Peebles & Xie (2023) with rotary embeddings Su et al. (2024) and consists of 169M parameters. The neural network is comprised of 12 layers, 12 attention heads, and 768 hidden dimensions. Please see Sahoo et al. (2024) for the full model architecture and training details.

As in Sahoo et al. (2024), we use `gpt-2` tokenizer for our experiments. We use the same train-validation split as in Sahoo et al. (2024) (where the last 100k documents of OWT were designated as the validation set) and randomly select 5,000 samples from the validation set to serve as the 'reference' for MAUVE score computation. For the discrete flow matching corrector sampler, we follow Gat et al. (2024) and set the corrector schedule $\beta(t) = At^{0.25}(1-t)^{0.25}$ where $A = 10$ (see Appendix A.7 for the definition of corrector schedule). We empirically find that nucleus sampling performs better than the temperature technique proposed in Gat et al. (2024). We also use non-fixed-width time steps as in Gat et al. (2024).

For evaluation metrics, we report MAUVE scores, generative perplexity, and entropy. For MAUVE, we generate 5,000 samples for each model/sampler. We use the `gpt-2` tokenizer, GPT-2 Large Radford et al. (2019) as the embedding model, and the MAUVE scaling hyperparameter is set to 5. For generative perplexity, we use GPT-2 Large as the external model. As in Zheng et al. (2024), we also report the average sequence entropy as a diversity metric. Specifically, we compute the entropy for the number of tokens for each sequence before it is decoded by the tokenizer and then report the mean entropy value of 5,000 generated sequences.

D.2 ImageNet

In this experiment, we reuse the pre-trained MaskGiT model for all samplers Chang et al. (2022). The MaskGiT architecture is a Transformer model Vaswani (2017) that consists of 24 layers, 8 attention heads, 768 embedding dimension, and 3,072 hidden dimension. Similar to MDLM, this model implements the carry-over unmasked tokens property in the parameterization of $\mathbf{x}_\theta$. The full model architecture and training setup details are available in Chang et al. (2022).

Algorithm 2 Sampling with ReMDM-switch.

Input: pre-trained denoising network $\mathbf{x}_\theta$ (e.g., MDLM), number of timesteps T, number of tokens in a sequence L, noising schedule α_t, maximum value for remasking $\eta_{cap} \in [0, 1]$, rescale value for remasking $\eta_{rescale} \in [0, 1]$, boolean value for whether to use confidence strategy `use_conf`, time for 'switching on' ReMDM $t_{switch} \in (0, 1)$.
Initialize $\mathbf{z}_t^{(1:L)} = \{\boldsymbol{m}\}^L$.
Initialize $\psi_t^{(1:L)} = \{-\infty\}^L$.
for $i = T$ **to** 1 **do**
  $t = i/T, s = (i-1)/T$.
  Set α_t, α_s according to noise schedule.
  if $t \leq t_{switch}$ **then**
    $\sigma_t^{(\ell)} = \eta_{rescale} \cdot \min\{\eta_{cap}, (1-\alpha_s)/\alpha_t\}$ for all $\ell = \{1, \dots, L\}$.
    if `use_conf` **then**
      Compute $\eta_{conf}^{(\ell)} = \frac{\exp(-\psi^{(\ell)})}{\sum_{\ell'} \exp(-\psi^{(\ell')})}$ for all $\ell \in \{1, \dots, L\}$.
      $\sigma_t^{(\ell)} = \eta_{conf}^{(\ell)} \cdot \sigma_t^{(\ell)}$ for all $\ell \in \{1, \dots, L\}$.
    end if
  else
    $\sigma_t^{(1:L)} = \{0\}^L$.
  end if
  For all $\ell \in \{1, \dots, L\}$, compute approximate posterior:

$$p_\theta(\mathbf{z}_s^{(\ell)} \mid \mathbf{z}_t^{(1:L)}) = q_\sigma(\mathbf{z}_s^{(\ell)} \mid \mathbf{z}_t^{(1:L)}, \mathbf{x} = \mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t^{(1:L)}))$$
$$= \begin{cases} \mathrm{Cat}(\mathbf{z}_s; (1-\sigma_t^{(\ell)})\mathbf{x}_\theta^{(\ell)} + \sigma_t^{(\ell)}\boldsymbol{m}), & \mathbf{z}_t \neq \boldsymbol{m} \\ \mathrm{Cat}(\mathbf{z}_s; \frac{\alpha_s - (1-\sigma_t^{(\ell)})\alpha_t}{1-\alpha_t}\mathbf{x}_\theta^{(\ell)} + \frac{1-\alpha_s-\sigma_t^{(\ell)}\alpha_t}{1-\alpha_t}\boldsymbol{m}), & \mathbf{z}_t = \boldsymbol{m} \end{cases}$$

  Sample $\mathbf{z}_s^{(\ell)} \sim p_\theta$ for all $\ell \in \{1, \dots, L\}$.
  if `use_conf` **then**
    Store confidence scores $\psi_t^{(\ell)} = (\mathbf{z}_s^{(\ell)})^\top \mathbf{x}_\theta(\mathbf{z}_t^{(1:L)})^{(\ell)}$, for all newly decoded $\mathbf{z}_s^{(\ell)} \neq \mathbf{z}_t^{(\ell)} \neq \boldsymbol{m}$.
  end if
  Set $\mathbf{z}_t^{(1:L)} = \mathbf{z}_s^{(1:L)}$.
end for
Output: $\mathbf{z}_t^{(1:L)}$.

In Table 2, the MaskGiT sampler refers to the heuristic confidence-based decoding described in Chang et al. (2022). The MDLM sampler refers to using the outputs of the pre-trained MaskGiT denoising model, then applying the zero-mask parameterization and plugging $\mathbf{x}_\theta$ into the posterior from (2). The ReMDM sampler refers to using the same $\mathbf{x}_\theta$ as that used with the MDLM sampler, but with the posterior from (6). For ReMDM, we explore the max-capped (with $\eta_{cap} \in \{0.01, 0.02, 0.05\}$), rescaled (with $\eta \in \{0.01, 0.02, 0.05\}$), and confidence-based schedules described in Section 4.1. We do not combine these strategies, but rather test each separately. For all three samplers, we perform a sweep over softmax temperature scaling using a scale parameter $\tau \in \{0.6, 0.8, 1.0\}$. For a vector $\mathbf{y}$, with $\mathbf{y}_i$ denoting its ith component, softmax temperature scaling is implemented as follows:

$$\frac{\exp(\mathbf{y}_i/\tau)}{\sum_j \exp(\mathbf{y}_j/\tau)}.$$

For all samplers we use a log-linear schedule for α_t, i.e., $\alpha(t) = 1 - t$.

We use the code from Dhariwal & Nichol (2021) to compute FID and IS metrics.

D.3 QM9

For the guidance experiments on QM9, we follow the setup used in Schiff et al. (2024). The dataset was tokenized using a regular expression-based tokenizer Schwaller et al. (2019) with padded max sequence lengths of $L = 32$. Including special tokens, the tokenizer vocabulary size is $|V| = 40$. The

Algorithm 3 Sampling with ReMDM-loop.

Input: pre-trained denoising network $\mathbf{x}_\theta$ (e.g., MDLM), number of timesteps T, number of tokens in a sequence L, noising schedule α_t, maximum value for remasking $\eta_{cap} \in [0, 1]$, rescale value for remasking $\eta_{rescale} \in [0, 1]$, boolean value for whether to use confidence strategy `use_conf`, start time for ReMDM loop $t_{on} \in [0, 1)$, number of discrete time steps to spend prior to ReMDM loop $n_{phase_1} \in (0, T)$, number of discrete time steps to spend in ReMDM loop $n_{phase_2} \in (0, T - n_{phase_1})$.
Initialize $\mathbf{z}_t^{(1:L)} = \{\boldsymbol{m}\}^L$.
Initialize $\psi_t^{(1:t)} = \{-\infty\}^L$.
for $i = T$ **to** 1 **do**
  $t = i/T, s = (i-1)/T$.
  if $t > (T - n_{phase_1})/T$ **then**
    `// Phase 1`
    Rescale and shift time: $t = (t \cdot (1 - t_{on}) \cdot T/n_{phase_1}) + (T \cdot (t_{on} - 1)/n_{phase_1}) + 1$.
    Rescale and shift time: $s = (s \cdot (1 - t_{on}) \cdot T/n_{phase_1}) + (T \cdot (t_{on} - 1)/n_{phase_1}) + 1$.
    Set α_t, α_s according to noise schedule.
    $\sigma_t^{(1:L)} = \{0\}^L$.
  else if $t \geq (T - n_{phase_1} - n_{phase_2})/T$ **then**
    `// Phase 2: ReMDM loop`
    Set $\alpha_t = \alpha(t_{on}), \alpha_s = \alpha(t_{on})$.
    $\sigma_t^{(\ell)} = \eta_{rescale} \cdot \min\{\eta_{cap}, (1 - \alpha_s)/\alpha_t\}$ for all $\ell = \{1, \ldots, L\}$.
    if `use_conf` **then**
      Compute $\eta_{conf}^{(\ell)} = \frac{\exp(-\psi^{(\ell)})}{\sum_{\ell'} \exp(-\psi^{(\ell')})}$ for all $\ell \in \{1, \ldots, L\}$.
      $\sigma_t^{(\ell)} = \eta_{conf}^{(\ell)} \cdot \sigma_t^{(\ell)}$ for all $\ell \in \{1, \ldots, L\}$.
    end if
  else
    `// Phase 3`
    Rescale time: $t = t \cdot t_{on} \cdot T/(T - n_{phase_1} - n_{phase_2})$.
    Rescale time: $s = s \cdot t_{on} \cdot T/(T - n_{phase_1} - n_{phase_2})$.
    Set α_t, α_s according to noise schedule.
    $\sigma_t^{(1:L)} = \{0\}^L$.
  end if
  For all $\ell \in \{1, \ldots, L\}$, compute approximate posterior:

$$p_\theta(\mathbf{z}_s^{(\ell)} \mid \mathbf{z}_t^{(1:L)}) = q_\sigma(\mathbf{z}_s^{(\ell)} \mid \mathbf{z}_t^{(1:L)}, \mathbf{x} = \mathbf{x}_\theta^{(\ell)}(\mathbf{z}_t^{(1:L)}))$$
$$= \begin{cases} \mathrm{Cat}(\mathbf{z}_s; (1 - \sigma_t^{(\ell)})\mathbf{x}_\theta^{(\ell)} + \sigma_t^{(\ell)}\boldsymbol{m}), & \mathbf{z}_t \neq \boldsymbol{m} \\ \mathrm{Cat}(\mathbf{z}_s; \frac{\alpha_s - (1-\sigma_t^{(\ell)})\alpha_t}{1-\alpha_t}\mathbf{x}_\theta^{(\ell)} + \frac{1-\alpha_s-\sigma_t^{(\ell)}\alpha_t}{1-\alpha_t}\boldsymbol{m}), & \mathbf{z}_t = \boldsymbol{m} \end{cases}$$

  Sample $\mathbf{z}_s^{(\ell)} \sim p_\theta$ for all $\ell \in \{1, \ldots, L\}$.
  if `use_conf` **then**
    Store confidence scores $\psi_t^{(\ell)} = (\mathbf{z}_s^{(\ell)})^\top \mathbf{x}_\theta(\mathbf{z}_t^{(1:L)})^{(\ell)}$, for all newly decoded $\mathbf{z}_s^{(\ell)} \neq \mathbf{z}_t^{(\ell)} \neq \boldsymbol{m}$.
  end if
  Set $\mathbf{z}_t^{(1:L)} = \mathbf{z}_s^{(1:L)}$.
end for
Output: $\mathbf{z}_t^{(1:L)}$.

AR, MDLM, and UDLM models used for discrete CBG and CFG were taken from the implementation provide in Schiff et al. (2024). These models are based on a Transformer architecture (causally masked for AR) known as DiT Peebles & Xie (2023). See Schiff et al. (2024) for the full model and experimental setup details. Note that Schiff et al. (2024) provide a first-order approximation for D-CBG that avoids the required $\mathcal{O}(|V| \cdot L \cdot T)$ calls to the classifier p_ϕ, which comes from needing to evaluate every possible token replacement from a vocabulary of size $|V|$ at every position in a sequence of length L for each step of the sampling process of duration T. However, given the relatively shorter sequences and smaller vocabulary size used in the QM9 experiments and that Schiff

et al. (2024) found generally higher quality results *without* the first order approximation, we omit the first-order approximation from our experiments and instead use the full, un-approximated D-CBG implementation.

In the main text, we report results for experiments performed to maximize the ring count value of molecules. In Appendix E.4, we also present results for maximizing the property of drug-likeness (QED), which was also explored in Schiff et al. (2024). Guidance training and inference were performed on the molecular property of interest (ring count or QED) using a binarized label, where molecules with property value $\geq$ the 90th percentile in the dataset were labeled $y = 1$. The QED and ring count properties were extracted for each molecule in the QM9 dataset using the RDKit library Landrum et al. (2013).

For all the models and guidance mechanisms, we report results with varying guidance strength $\gamma \in \{1, 2, 3, 4, 5\}$ and timesteps $T \in \{32, 64, 128\}$. For the diffusion models, we generate using a log-linear schedule for α_t, i.e., $\alpha(t) = 1 - t$.

For evaluation, we generated 1,024 sequences with each model / guidance mechanism. We used RDKit to parse the generated sequences. Sequences that could not be parsed were deemed 'invalid'. Of the valid molecule strings, we remove duplicates and report the number of 'novel' sequences, which we define as valid and unique generated sequences that do not appear in the full QM9 dataset. We then use RDKit to compute QED / ring count of the novel generated sequences and report the mean value.

For ReMDM samplers, we reuse the pre-trained MDLM weights. We perform an extensive grid search for both properties, QED and ring count, and guidance mechanisms, D-CFG and D-CBG. Namely we explore the ReMDM-rescale schedule with $\eta_{rescale} \in \{0.1, 0.5, 0.9, 1.0\}$. We test sampling with and without the confidence-based schedule being used in conjunction with ReMDM-rescale. For each combination, we also try both the switch and loop strategies. For switch, we use $t_{switch} \in \{0.1, 0.5, 0.9\}$. For loop, we sweep over the tuples $(t_{on}, t_{off}) \in \{(0.5, 0.25), (0.25, 0.125), (0.1, 0.05)\}$. We use non-fixed width steps in ReMDM-loop, so that for each T, the loop starts at discrete step $i = T/2$ and ends at step $i = \lfloor (1 - t_{off}) \cdot T \rfloor$, and we 'rescale' time accordingly before and after the loop phases, as described in Algorithm 3.

E Additional Experimental Results

E.1 Training with MDLM v.s. ReMDM NELBO

In Table 3, we report QM9 dataset validation set perplexities for models trained with either the MDLM NELBO from (3) or the ReMDM NELBO from (9), with $\sigma_t = \min\{\eta_{cap}, (1 - \alpha_s)/\alpha_t\}$, for some $\eta_{cap} \in (0, 1]$. We follow the same model architecture and training settings defined in Schiff et al. (2024) for this dataset. Overall, we find that validation perplexities for this dataset are fairly consistent across models trained with either objective.

Table 3: Training with ReMDM and MDLM NELBO objectives leads to similar validation set perplexity values (for the QM9 dataset). In the top row, the model is trained using the continuous time formulation of the MDLM objective from Sahoo et al. (2024). In the bottom two rows, models are trained with discrete-time objectives. The first column, $\sigma_t = 0$, corresponds to the MDLM objective from (3) and the columns with varying η_{cap} correspond to training with the ReMDM objective from (9) with $\sigma_t = \min\{\eta_{cap}, (1 - \alpha_s)/\alpha_t\}$. We find that results are comparable across training settings.

	$\sigma_t = 0$	$\eta_{cap} = 0.5$	$\eta_{cap} = 1.0$
$T = \infty$	2.088	–	–
$T = 4096$	2.094	2.107	2.119
$T = 8192$	2.092	2.097	2.101

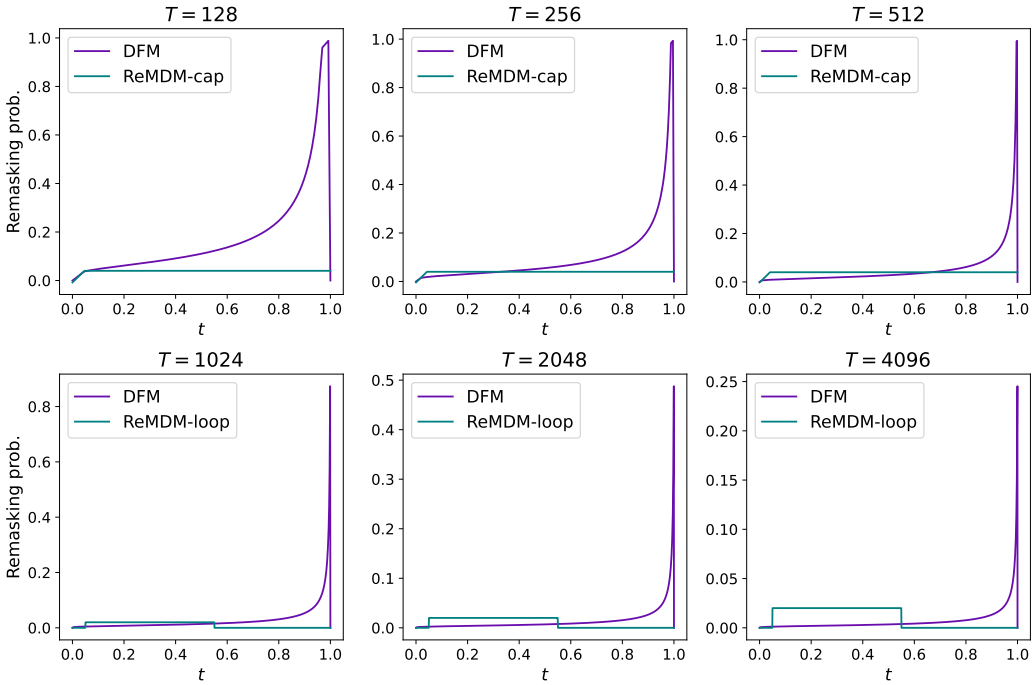


Figure 4: Remasking probability schedule of ReMDM and DFM corrector on OWT. Schedules used in experiments from Table 1 are plotted.

E.2 OpenWebText

E.2.1 ReMDM σ_t v.s. DFM σ_t

In Figure 4, we plot the remasking probability schedules of ReMDM and DFM correctors in the experiments reported in Table 1. Similarly to Figure 2b, the DFM schedules show a spike at first and a long tail afterward. Here, the best performing $\beta_t = At^{0.25}(1-t)^{0.25}$ schedule reported in Gat et al. (2024) is used. We also adjust the width of the time steps accordingly, as reported in Gat et al. (2024).

E.2.2 Ablation on ReMDM components

In Figure 5, we sequentially ablate each building block of ReMDM and report the MAUVE score of samples. Starting with MDLM, we see that each of our proposed sampling improvements increases the MAUVE score, with the largest improvement coming from remasking ability of ReMDM.

E.2.3 Results of Models without Nucleus Sampling

In Table 4, we present the sample quality of different models without nucleus sampling. Consistent with the results in Zheng et al. (2024), when using double-precision floating-point numbers during sampling, discrete diffusion models produce poor quality sequences, as reflected by the relatively higher generative perplexity values compared to Table 1. These results underscore the importance of nucleus sampling for discrete diffusion models in text generation.

E.2.4 Comparing Different ReMDM Schedules

In Table 5, we report the OpenWebText unconditional generation results for various ReMDM schedules. For ReMDM-conf, we set σ_t proportional to the upper bound σ_{max}. In the case of inference-time scaling ($T \geq 1024$), ReMDM-loop performs the best while in the case of faster sampling ($T < 1024$), ReMDM-cap is found to perform better than others. In both scenarios, ReMDM-rescale and ReMDM-conf do not perform as well as the other two schedules.

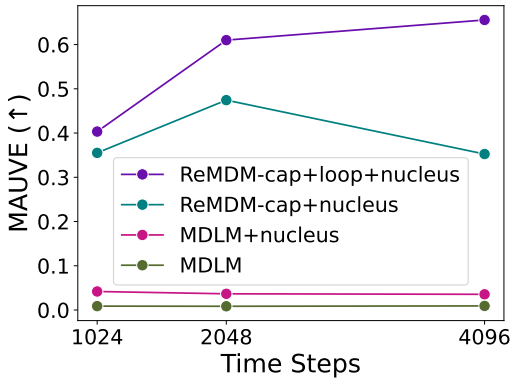


Figure 5: Effect of ReMDM components on OWT generation quality. Inference-time scaling with $T \in \{1024, 2048, 4096\}$.

Table 4: Sample quality of absorbing state discrete diffusion models without nucleus sampling.

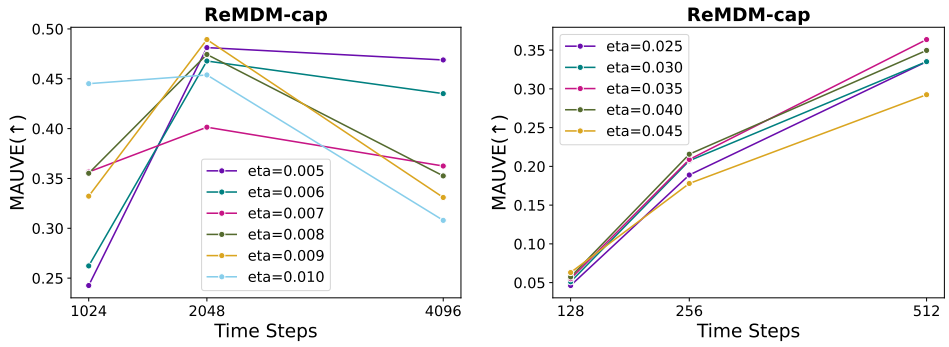
Method	MAUVE (↑)			Gen PPL. (↓)			Entropy (↑)		
AR *(T=1024)*	0.407			35.0			5.58		
	T=1024	*T=2048*	*T=4096*	*T=1024*	*T=2048*	*T=4096*	*T=1024*	*T=2048*	*T=4096*
SEDD	0.008	0.008	0.009	104.7	103.2	102.5	5.62	5.61	5.61
MDLM	0.009	0.008	0.009	104.8	104.4	104.1	5.63	5.63	5.63
MDLM+FB	0.009	0.011	0.010	102.6	101.9	100.9	5.65	5.65	5.64
MDLM+DFM	0.008	0.010	0.010	125.9	119.1	115.7	5.70	5.69	5.69
ReMDM	0.007	0.007	0.007	173.0	256.2	370.6	5.76	5.81	5.74
	T=128	*T=256*	*T=512*	*T=128*	*T=256*	*T=512*	*T=128*	*T=256*	*T=512*
SEDD	0.007	0.007	0.008	119.2	110.1	107.2	5.65	5.63	5.62
MDLM	0.007	0.008	0.008	121.4	111.8	107.2	5.67	5.65	5.64
MDLM+FB	0.007	0.009	0.010	112.8	106.4	104.4	5.67	5.66	5.65
MDLM+DFM	0.005	0.006	0.007	360.7	189.5	143.5	5.85	5.77	5.73
ReMDM	0.006	0.007	0.006	160.5	176.8	226.9	5.73	5.75	5.79

E.2.5 TUNING η_{cap} / $\eta_{rescale}$

In Figure 6, we plot MAUVE scores when using the ReMDM-cap schedule with different η_{cap} values and varying number of decoding steps T. As shown in Figure 6a, in the case of inference-time scaling ($T \geq 1024$), the MAUVE scores at $T = 4096$ tend to decrease as η_{cap} increases while the MAUVE scores at $T = 1024$ tend to increase with larger η_{cap}. We attribute this to the perplexity-entropy trade-off. In particular, when η_{cap} increases, the probability of remasking increases, improving perplexity, but also harming diversity. In the case of $T = 4096$, the models can generate higher quality sentences and the reduction in diversity outweighs marginal improvements in generative perplexity. For $T = 1024$, the samples are of relatively poorer quality and the improvement in perplexity plays a major role in driving up the MAUVE score. To achieve the best balance, we choose $\eta_{cap} = 0.008$ for the setting of inference-time scaling.

For faster sampling ($T < 1024$), in Figure 6b, with the exception of $\eta = 0.045$ we see a positive trend between increasing η_{cap} and improving MAUVE scores. In Tables 1 and 5, we report results using $\eta = 0.040$.

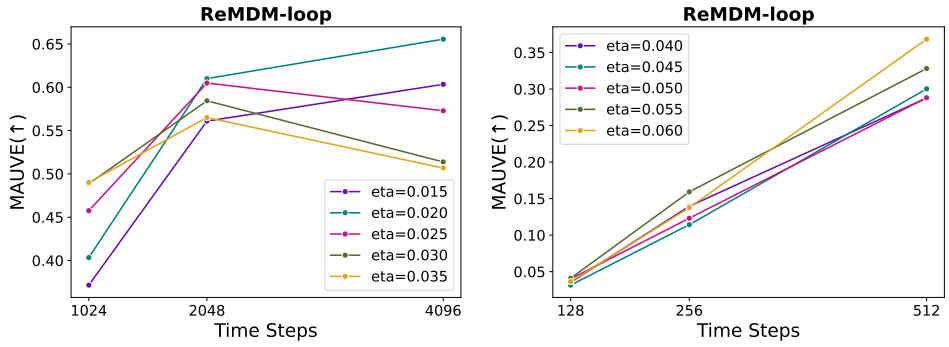
We also performed similar studies for schedules that combine ReMDM-cap with the ReMDM-loop strategy (Figure 7) and for ReMDM-rescale schedules (Figure 8). For both of these settings, we observe trends for increasing η_{cap} / $\eta_{rescale}$ that are similar to those described above. In Table 5, we report the following choice of hyperparameter for each of these settings. For ReMDM-loop combined with ReMDM-cap, in the inference-time scaling experiments ($T \geq 1024$), we report results using $\eta_{cap} = 0.020$. For the faster sampling experiments ($T < 1024$), we report results for $\eta_{cap} = 0.055$. For the ReMDM-rescale inference-time scaling experiments, we report results with $\eta_{rescale} = 0.015$ and for the faster sampling experiments, we use $\eta_{rescale} = 0.045$ in Table 1.



(a) MAUVE scores of ReMDM-cap inference-time scaling

(b) MAUVE scores of ReMDM-cap faster sampling

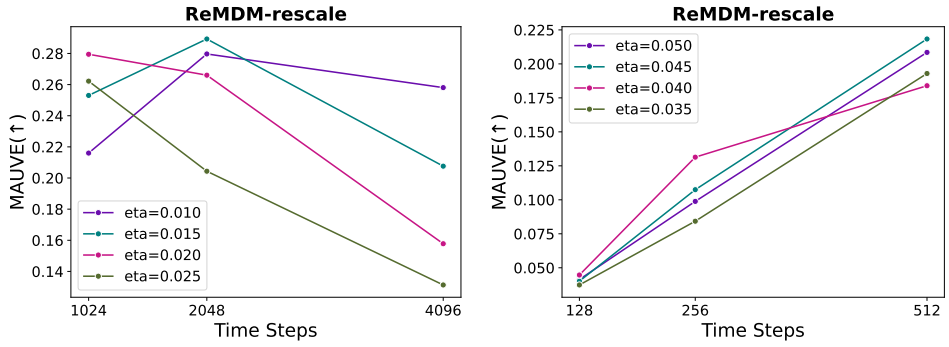
Figure 6: Impact of σ_t on ReMDM-cap's unconditional text generation quality on OpenWebText.



(a) MAUVE scores of ReMDM-loop inference-time scaling.

(b) MAUVE scores of ReMDM-loop faster sampling.

Figure 7: Impact of η_{cap} on unconditional text generation quality on OpenWebText using the ReMDM-loop strategy with ReMDM-cap.



(a) MAUVE scores of ReMDM-rescale inference-time scaling.

(b) MAUVE scores of ReMDM-rescale faster sampling.

Figure 8: Impact of $\eta_{rescale}$ on unconditional text generation quality on OpenWebText using the ReMDM-rescale schedule.

Table 5: Comparing sample quality on OWT for various ReMDM schedules. † indicates nucleus sampling. For each T, the best MAUVE score is **bolded**.

Method	MAUVE (↑)			Gen PPL. (↓)			Entropy (↑)		
	T=1024	*T=2048*	*T=4096*	*T=1024*	*T=2048*	*T=4096*	*T=1024*	*T=2048*	*T=4096*
ReMDM-cap†	0.355	0.474	0.353	27.7	20.2	14.4	5.32	5.21	5.06
ReMDM-loop†	**0.403**	**0.610**	**0.656**	28.6	22.8	17.6	5.38	5.30	5.20
ReMDM-rescale†	0.253	0.289	0.208	27.4	20.4	14.6	5.28	5.15	4.94
ReMDM-conf†	0.075	0.071	0.093	39.2	37.2	36.1	5.36	5.34	5.36
	T=128	*T=256*	*T=512*	*T=128*	*T=256*	*T=512*	*T=128*	*T=256*	*T=512*
ReMDM-cap†	**0.057**	**0.216**	**0.350**	42.5	30.5	21.1	5.43	5.34	5.21
ReMDM-loop†	0.041	0.159	0.328	51.5	38.1	29.3	5.52	5.45	5.38
ReMDM-rescale†	0.040	0.107	0.218	46.2	34.8	25.6	5.44	5.36	5.24
ReMDM-conf†	0.019	0.035	0.053	57.7	49.6	43.3	5.50	5.46	5.41

E.3 IMAGENET

In Table 6, we present the full results for varying the sampling temperature τ and the various ReMDM schedules and the their corresponding hyperparameters (η_{cap} for ReMDM-cap. and $\eta_{rescale}$ for ReMDM-rescale). As noted in Section 5.1.2, while both MDLM and ReMDM benefit from using a softmax temperature of $\tau = 0.8$, the MaskGiT sampler produces the best results with no temperature scaling (i.e., $\tau = 1$). Of note, we use FID to determine which setting constitutes the 'best' results for each sampler. With reduced temperature, the entropy of the softmax is reduced leading to less diverse samples. This is reflected in the trade-off of the FID vs. IS metrics across sampling temperatures, with IS penalizing a lack of diversity less than FID Heusel et al. (2017).

E.4 QM9

In Figures 10-25, we present the results from the extensive hyperparameter search we conducted across both properties, ring count (Figures 10-17) and QED (Figures 18-25), and guidance mechanisms D-CFG (Figures 10-13, 18-21) and D-CBG (Figures 14-17, 22-25).

E.4.1 DRUG-LIKENESS PROPERTY MAXIMIZATION

In Figure 9, we present D-CFG and D-CBG results for baseline and the 'best' ReMDM setting when maximizing the drug-likeness (QED) property. Similar to the results for maximizing the ring count property in Section 5.2, we find that ReMDM improves the novelty-property maximization frontier relative to MDLM when maximizing QED, although the benefits of ReMDM relative to MDLM are less pronounced for QED than for ring count.

For D-CFG, results reflect combining ReMDM-rescale ($\eta_{rescale} = 0.1$) with the confidence-based scheduler and switch strategy ($t_{switch} = 0.1$). For D-CBG, results reflect combining ReMDM-rescale ($\eta_{rescale} = 1.0$) with the confidence-based scheduler and switch strategy ($t_{switch} = 0.5$)

E.4.2 TUNING $\eta_{rescale}$

Larger $\eta_{rescale}$ is Better for Ring Count Maximization Across settings, we find that generally, with some exceptions in the settings where the confidence-based schedule is not used (where ReMDM performs comparatively worse than when confidence is used, as discussed below), we see a trend where large $\eta_{rescale}$ values lead to Pareto curves that are pushed further in the direction of more novelty and higher ring counts. Additionally, we find that for $\eta_{rescale} \geq 0.5$, the results are less sensitive to this parameter.

Inconclusive $\eta_{rescale}$ Results for QED Maximization For maximizing the QED property, the results are less conclusive. In the settings that use the confidence-based schedule, there is no clear $\eta_{rescale}$ that dominates the others. For settings that do not use the confidence-based schedule, we observe a trend where results improve with $\eta_{rescale} = 0.1$.

Table 6: Discretized ImageNet conditional generation grid search over softmax temperature τ and ReMDM hyperparameters. Values reflect FID / IS for varying T. For each sampler, the row corresponding to the hyperparameter setup that is reported in the main Table 2 is **bolded**.

		FID (↓)			*IS (↑)*		
Strategy	τ	$T = 16$	$T = 32$	$T = 64$	$T = 16$	$T = 32$	$T = 64$
MaskGiT	0.6	10.06	11.03	11.72	236.90	244.79	243.64
	0.8	6.66	7.09	7.75	205.91	225.08	233.87
	1.0	**6.74**	**4.92**	**4.85**	**155.32**	**181.57**	**196.38**
MDLM	0.6	6.99	7.67	8.43	214.55	233.59	242.28
	0.8	**7.88**	**5.37**	**4.69**	**140.97**	**169.79**	**187.93**
	1.0	26.02	18.18	13.96	64.01	82.54	96.30
ReMDM-cap, $\eta_{cap} = 0.01$	0.6	7.07	7.81	8.75	216.00	238.35	243.82
	0.8	7.71	5.20	4.56	141.31	172.66	194.77
	1.0	26.25	18.78	14.72	63.97	81.25	93.69
ReMDM-cap, $\eta_{cap} = 0.02$	0.6	7.10	7.94	9.01	217.12	239.50	244.98
	0.8	7.55	5.00	4.53	142.33	178.32	198.87
	1.0	26.86	19.50	15.85	63.06	79.46	91.68
ReMDM-cap, $\eta_{cap} = 0.05$	0.6	7.21	8.38	9.91	223.96	243.14	244.60
	0.8	7.24	4.83	4.52	147.20	184.21	211.06
	1.0	28.11	21.43	19.27	60.47	76.12	82.15
ReMDM-rescale, $\eta_{rescale} = 0.01$	0.6	7.09	7.78	8.66	215.52	237.82	244.96
	0.8	7.75	5.24	4.58	141.11	171.56	193.73
	1.0	26.09	18.48	14.29	64.45	82.26	95.69
ReMDM-rescale - $\eta_{rescale} = 0.02$	0.6	7.09	7.84	8.90	217.67	239.87	246.37
	0.8	7.64	5.07	4.48	142.50	176.57	197.35
	1.0	26.58	18.84	14.78	63.40	81.20	94.99
ReMDM-rescale - $\eta_{rescale} = 0.05$	0.6	7.19	8.22	9.65	223.05	242.99	250.68
	0.8	**7.40**	**4.92**	**4.45**	**145.27**	**182.05**	**209.45**
	1.0	27.39	19.99	16.69	62.09	78.82	90.58
ReMDM-conf	0.6	7.46	8.54	9.82	221.25	243.56	251.83
	0.8	6.91	5.16	5.35	150.73	189.51	212.66
	1.0	22.14	13.14	8.88	73.00	101.79	126.29

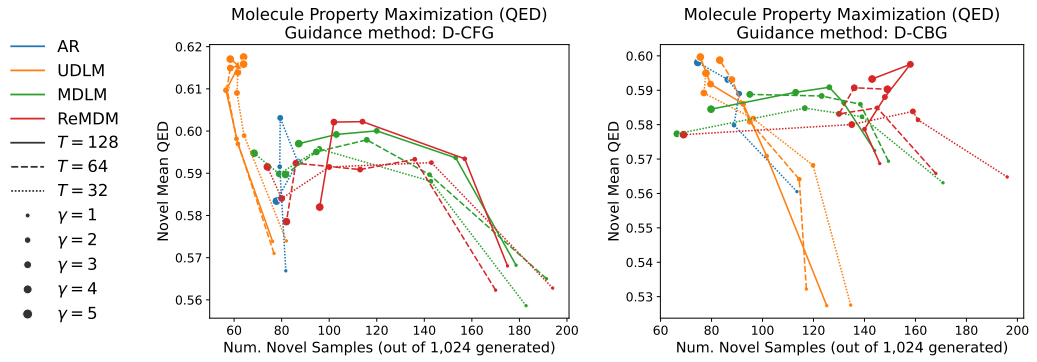


Figure 9: ReMDM improves steer-ability by extending the novelty-property maximization frontier. Controlled generation for drug-likeness (QED) maximization on QM9 dataset with varying inference compute T and guidance strength γ. *(Left)* Discrete classifier-free guidance (D-CFG). *(Right)* Discrete classifier-based guidance (D-CBG) and FUDGE for AR.

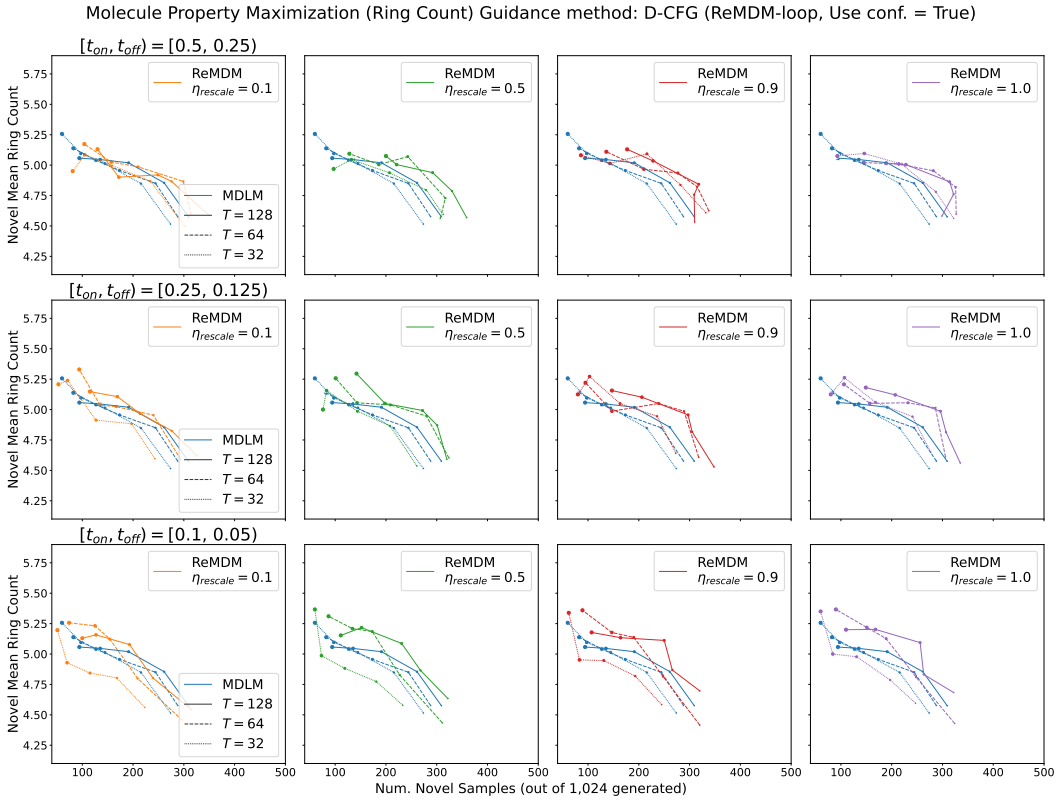


Figure 10: ReMDM-rescaled with **loop** and **confidence-based** schedules hyperparameter tuning for maximizing **ring count** using **D-CFG**. Larger marker sizes indicate larger γ values.

E.4.3 Ablating Use of Confidence-based Schedule

For maximizing the QED property, we observe a clear pattern where incorporating confidence-based schedules improves results. For ring count, this trend holds true as well, especially for larger values of $\eta_{rescale}$, where the confidence score potentially helps temper large remasking probabilities for relatively 'safe' tokens, but the trend is less pronounced overall than for QED experiments.

E.4.4 Tuning t_{switch} / Loop Parameters

For both QED and ring count maximization, when using D-CFG as the guidance mechanism, we observe a general trend that activating ReMDM, with either the switch or loop strategies, benefits from starting later in the decoding process, i.e., smaller t_{switch} / t_{on}. This trend also holds true for D-CBG in the QED experiments. A notable exception is when using D-CBG in the ring count experiments, where we see the opposite trend, i.e., activating ReMDM earlier (larger t_{switch} / t_{on}) improves results.

F Generated Samples

We visualize the generated sentences of MDLM ($T = 4096$) (Figure 26), MDLM+DFM ($T = 4096$) (Figure 27), and ReMDM ($T = 4096$) (Figure 28). We find that the MDLM samples contain many uncorrelated semantic fragments, with grammar errors appearing very often. MDLM+DFM can formulate the text around a special topic, but the internal logic is not fluent. In contrast, ReMDM is able to generate high quality, fluent English with a clear semantic topic.

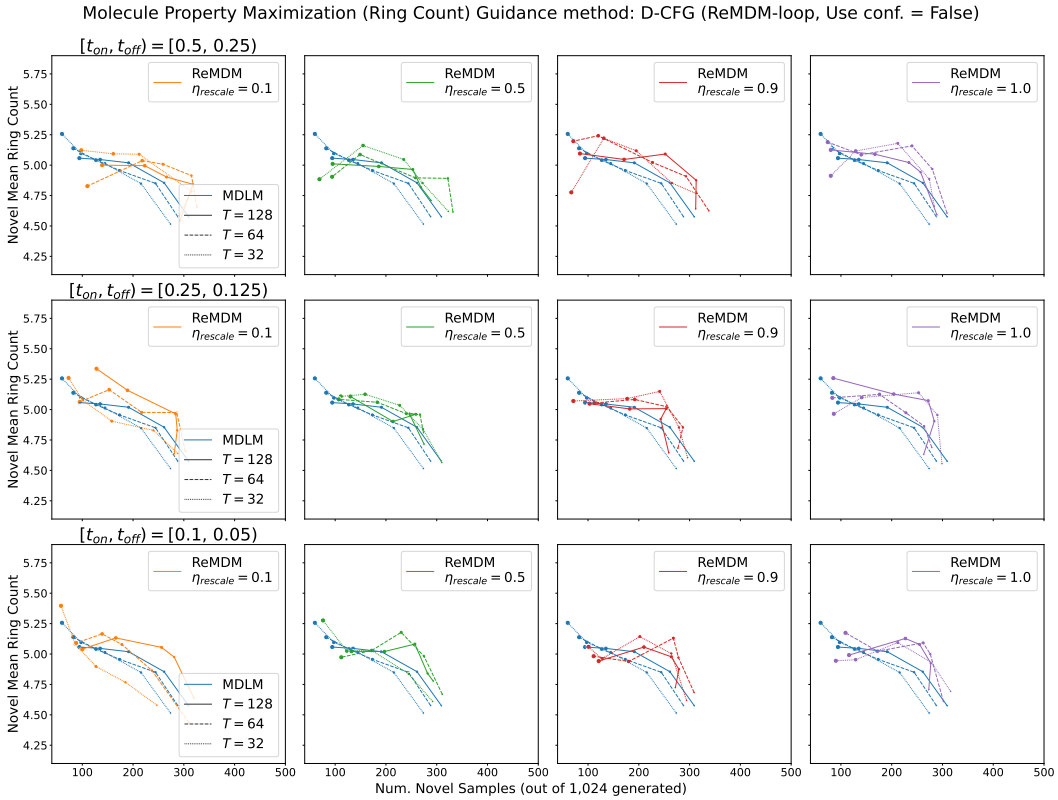


Figure 11: ReMDM-rescaled with **loop** and **without confidence-based** schedules hyperparameter tuning for maximizing **ring count** using **D-CFG**. Larger marker sizes indicate larger γ values.

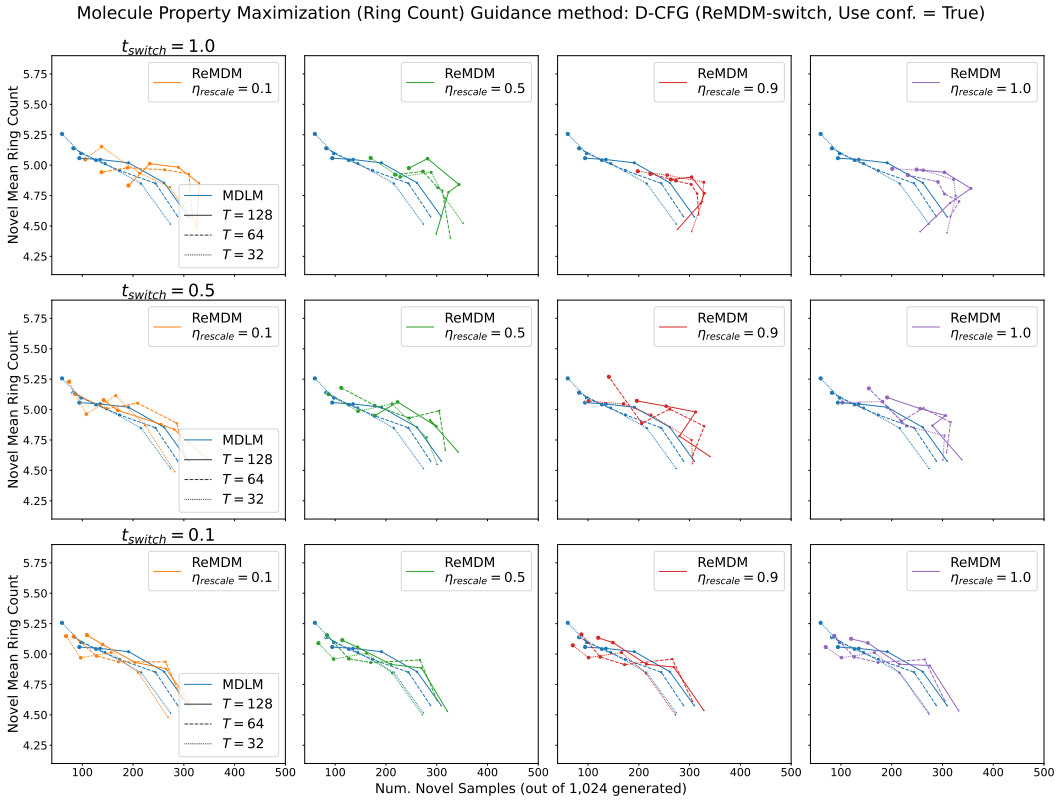


Figure 12: ReMDM-rescaled with **switch** and **confidence-based** schedules hyperparameter tuning for maximizing **ring count** using **D-CFG**. Larger marker sizes indicate larger γ values.

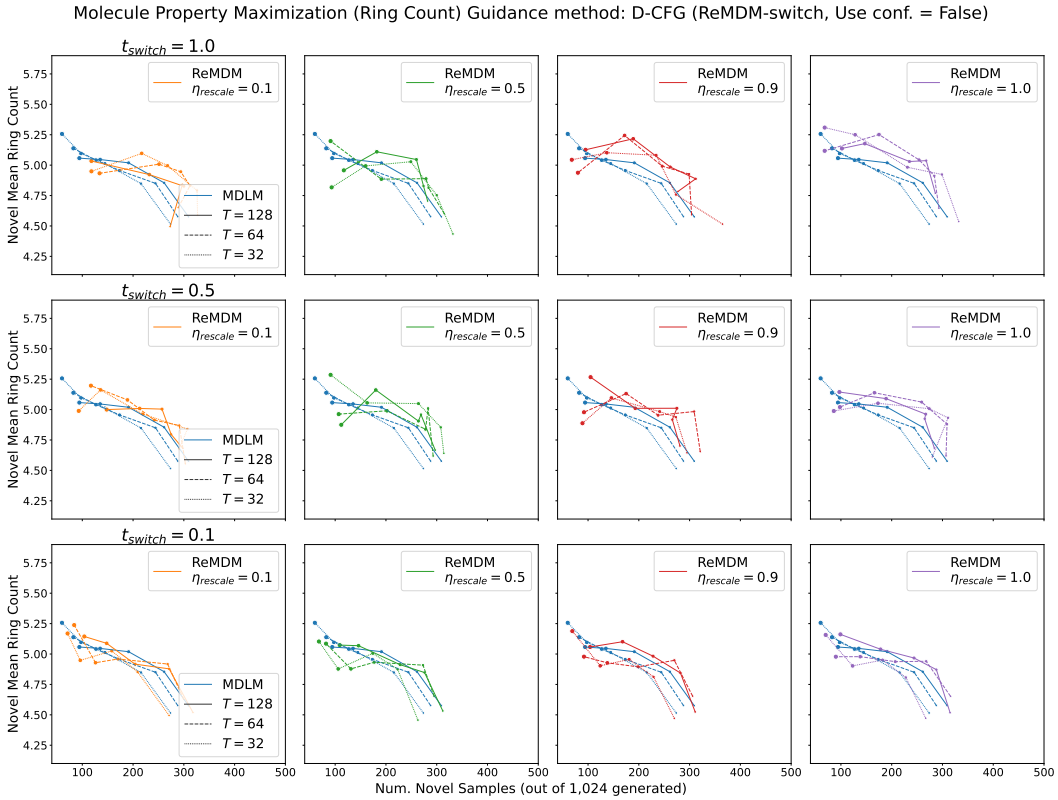


Figure 13: ReMDM-rescaled with **switch** and **without confidence-based** schedules hyperparameter tuning for maximizing **ring count** using **D-CFG**. Larger marker sizes indicate larger γ values.

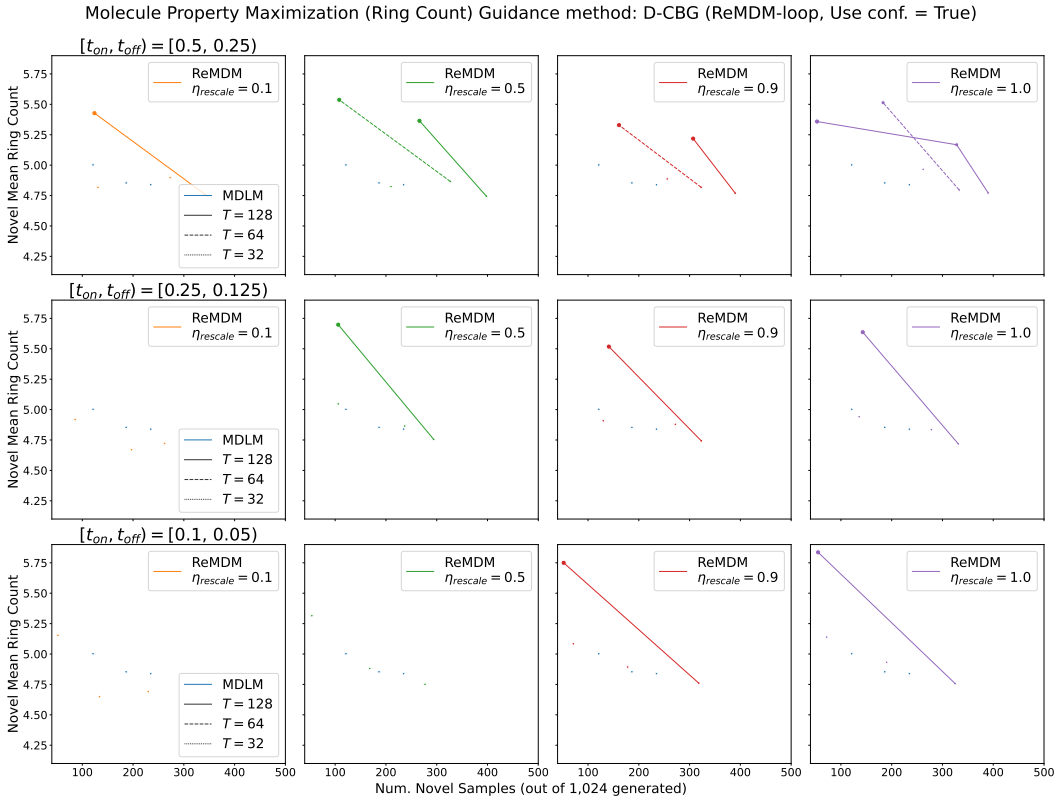


Figure 14: ReMDM-rescaled with **loop** and **confidence-based** schedules hyperparameter tuning for maximizing **ring count** using **D-CBG**. Larger marker sizes indicate larger γ values.

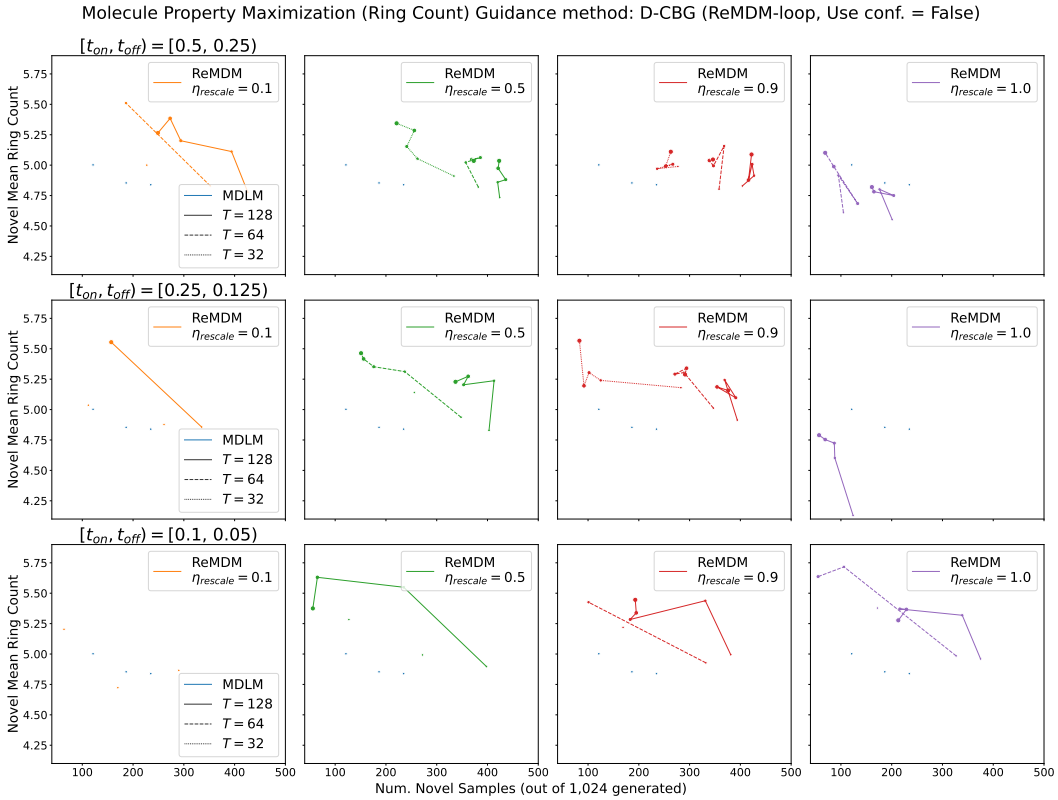


Figure 15: ReMDM-rescaled with **loop** and **without confidence-based** schedules hyperparameter tuning for maximizing **ring count** using **D-CBG**. Larger marker sizes indicate larger γ values.

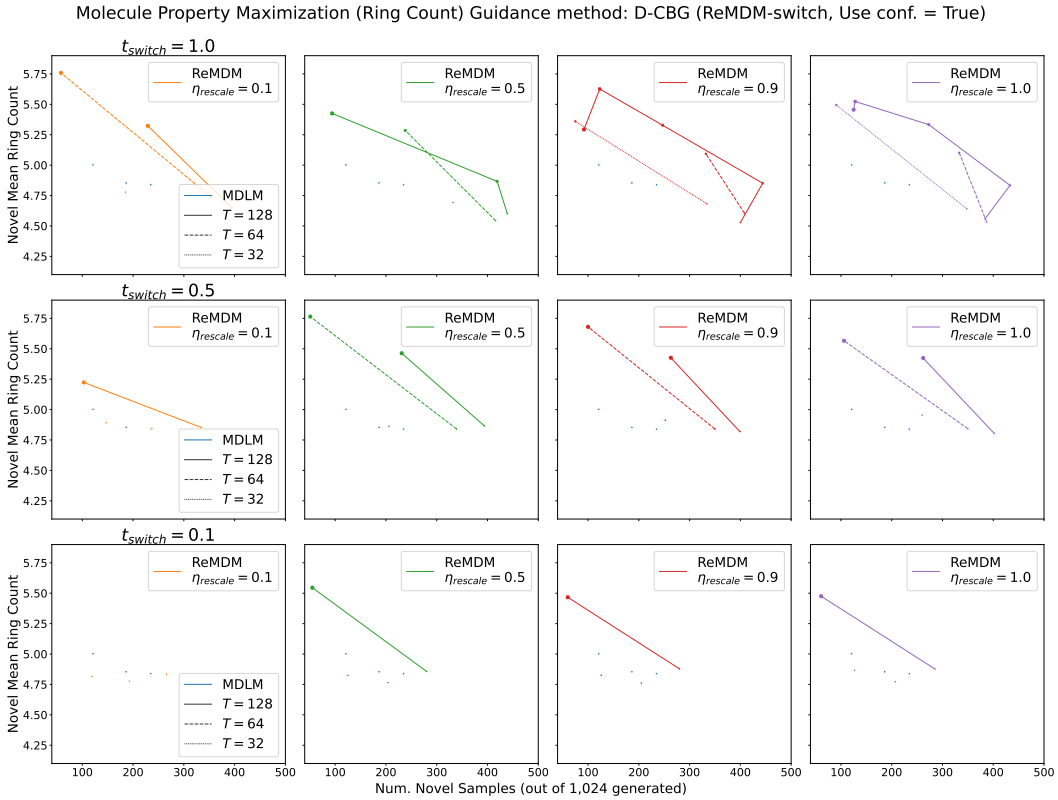


Figure 16: ReMDM-rescaled with **switch** and **confidence-based** schedules hyperparameter tuning for maximizing **ring count** using **D-CBG**. Larger marker sizes indicate larger γ values.

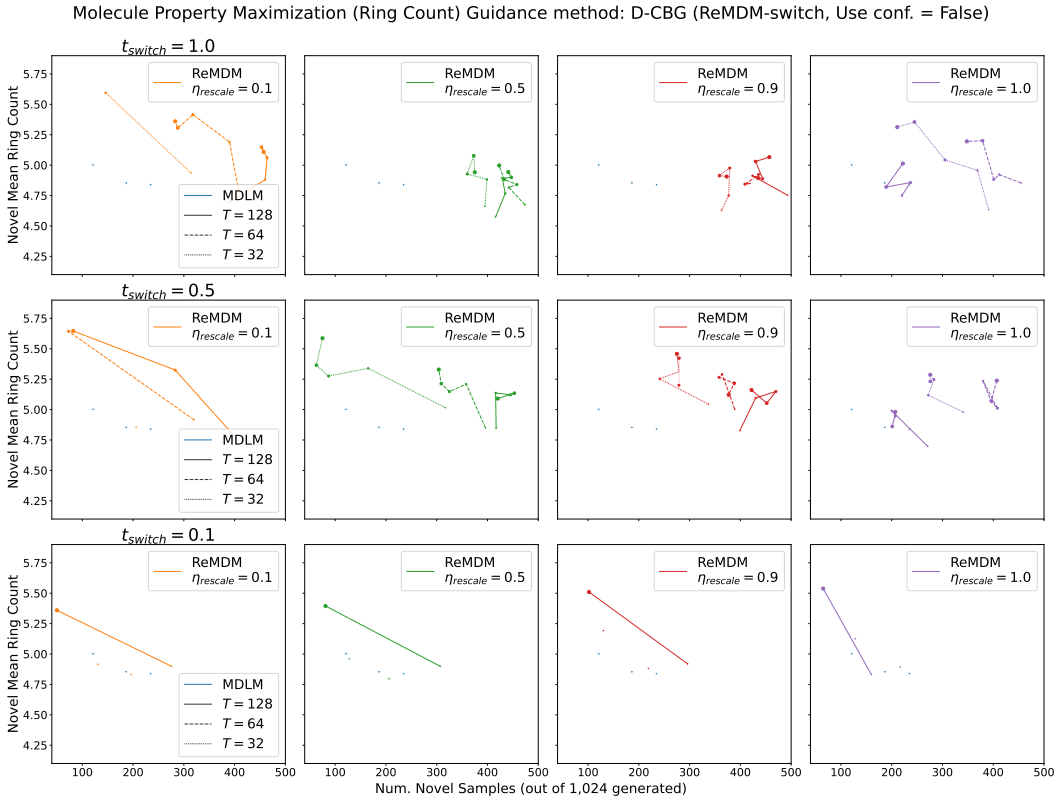


Figure 17: ReMDM-rescaled with **switch** and **without confidence-based** schedules hyperparameter tuning for maximizing **ring count** using **D-CBG**. Larger marker sizes indicate larger γ values.

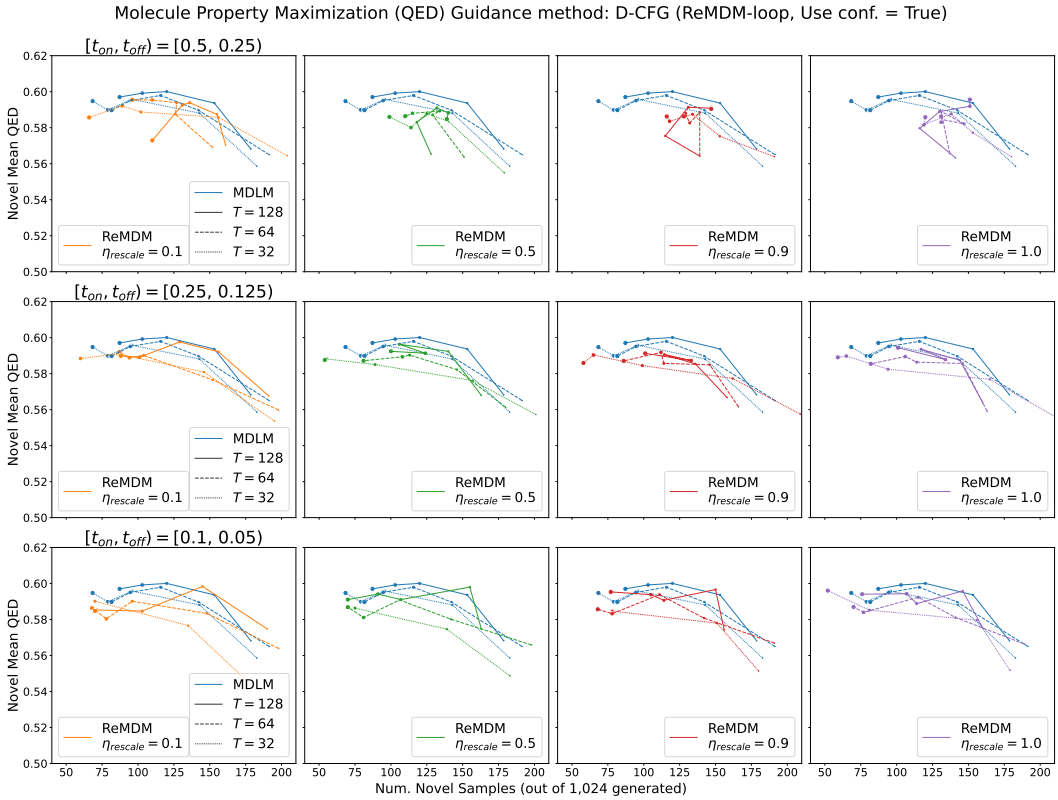


Figure 18: ReMDM-rescaled with **loop** and **confidence-based** schedules hyperparameter tuning for maximizing **drug-likeness (QED)** using **D-CFG**. Larger marker sizes indicate larger γ values.

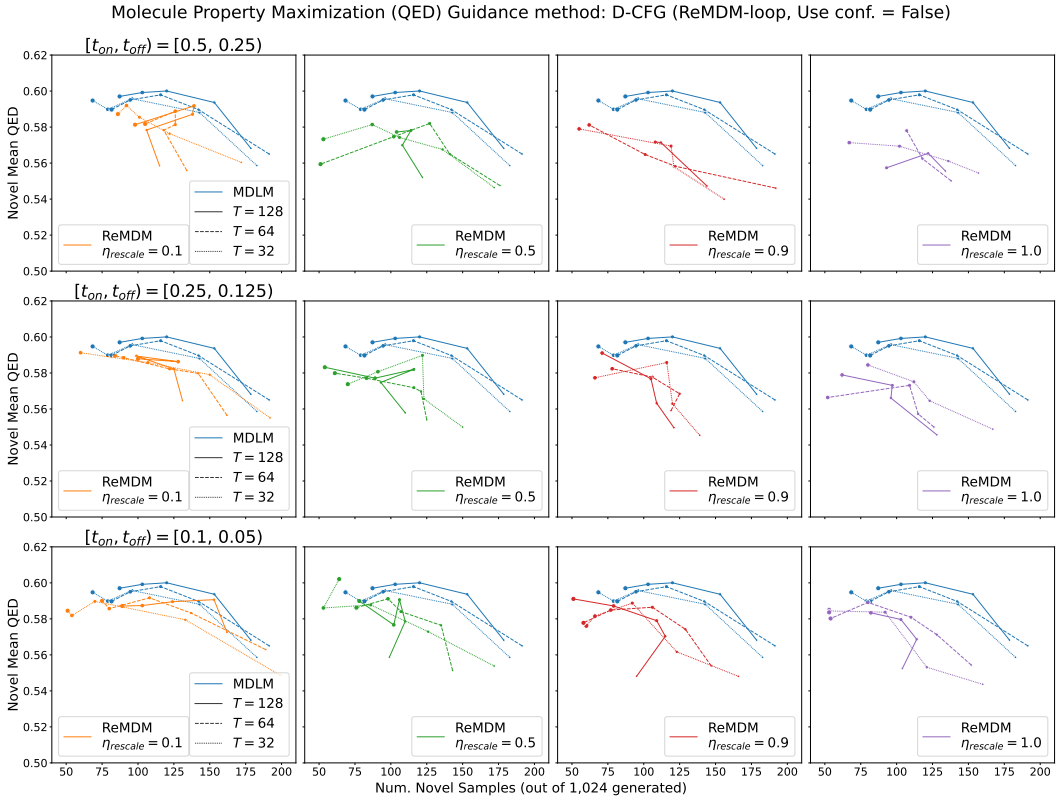


Figure 19: ReMDM-rescaled with **loop** and **without confidence-based** schedules hyperparameter tuning for maximizing **drug-likeness (QED)** using **D-CFG**. Larger marker sizes indicate larger γ values.

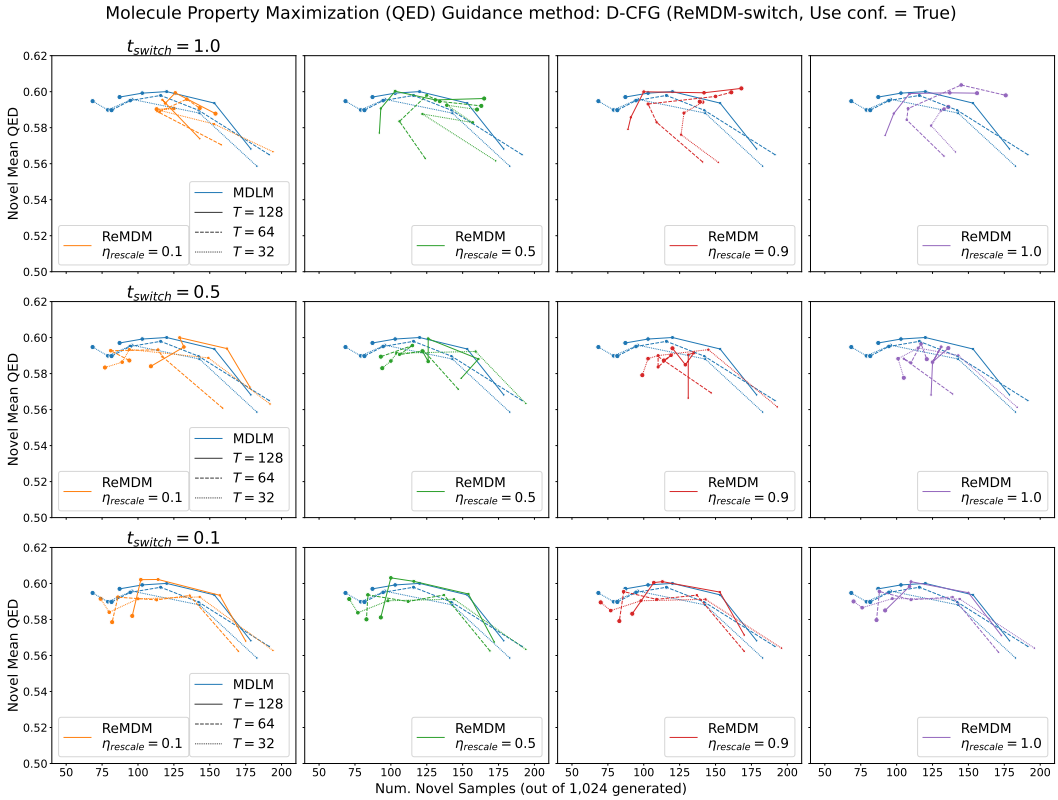


Figure 20: ReMDM-rescaled with **switch** and **confidence-based** schedules hyperparameter tuning for maximizing **drug-likeness (QED)** using **D-CFG**. Larger marker sizes indicate larger γ values.

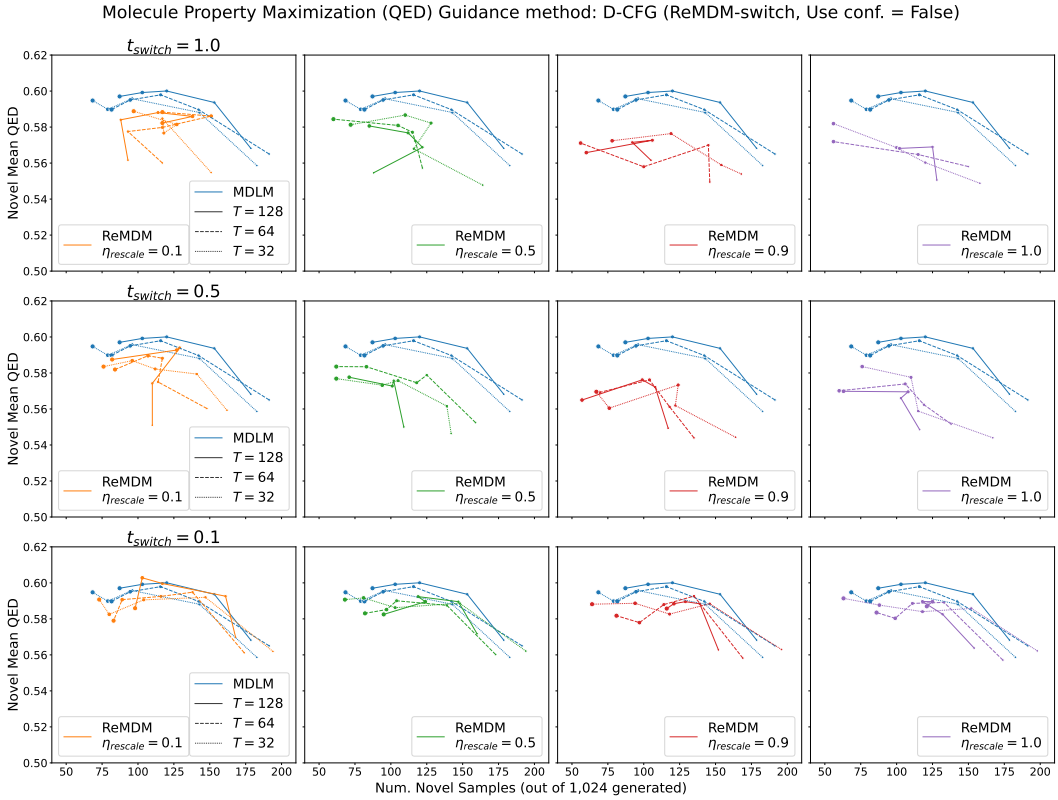


Figure 21: ReMDM-rescaled with **switch** and **without confidence-based** schedules hyperparameter tuning for maximizing **drug-likeness (QED)** using **D-CFG**. Larger marker sizes indicate larger γ values.

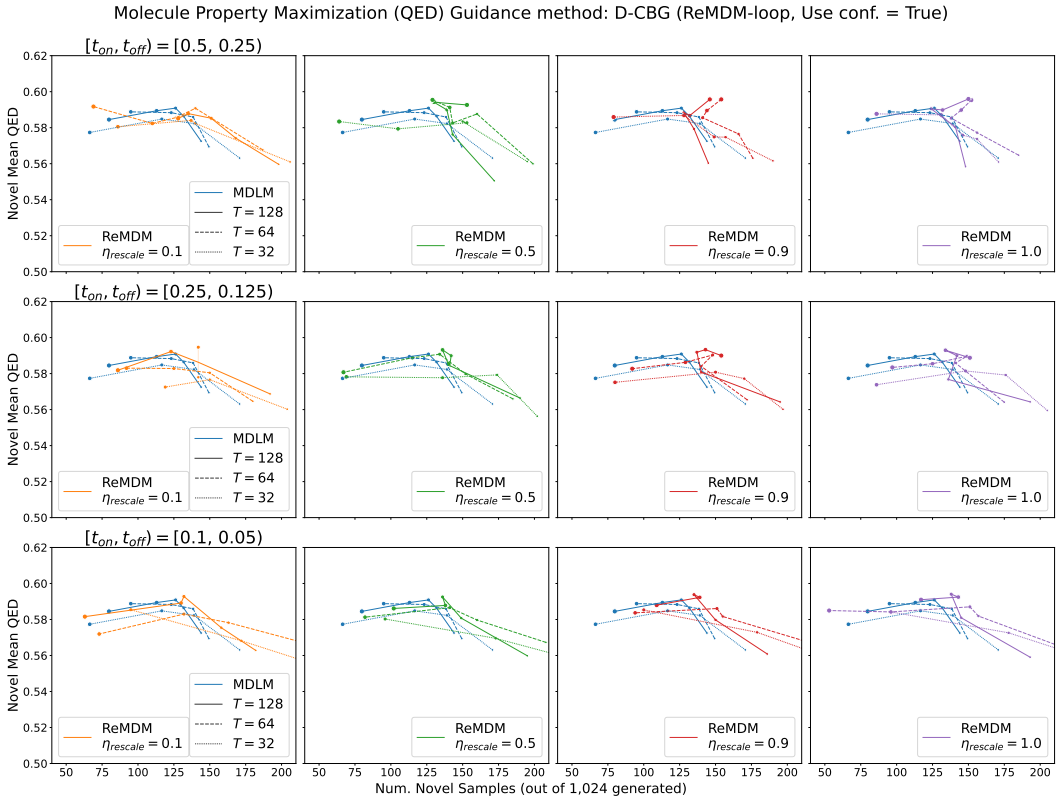


Figure 22: ReMDM-rescaled with **loop** and **confidence-based** schedules hyperparameter tuning for maximizing **drug-likeness (QED)** using **D-CBG**. Larger marker sizes indicate larger γ values.

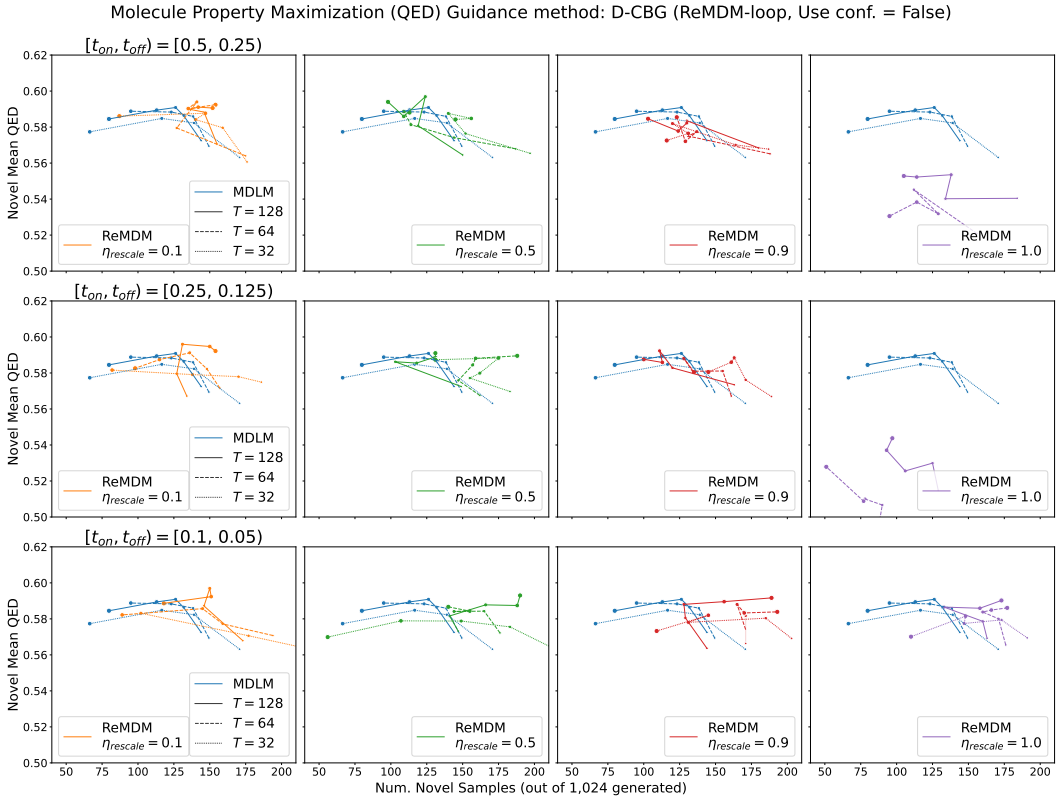


Figure 23: ReMDM-rescaled with **loop** and **without confidence-based** schedules hyperparameter tuning for maximizing **drug-likeness (QED)** using **D-CBG**. Larger marker sizes indicate larger γ values.

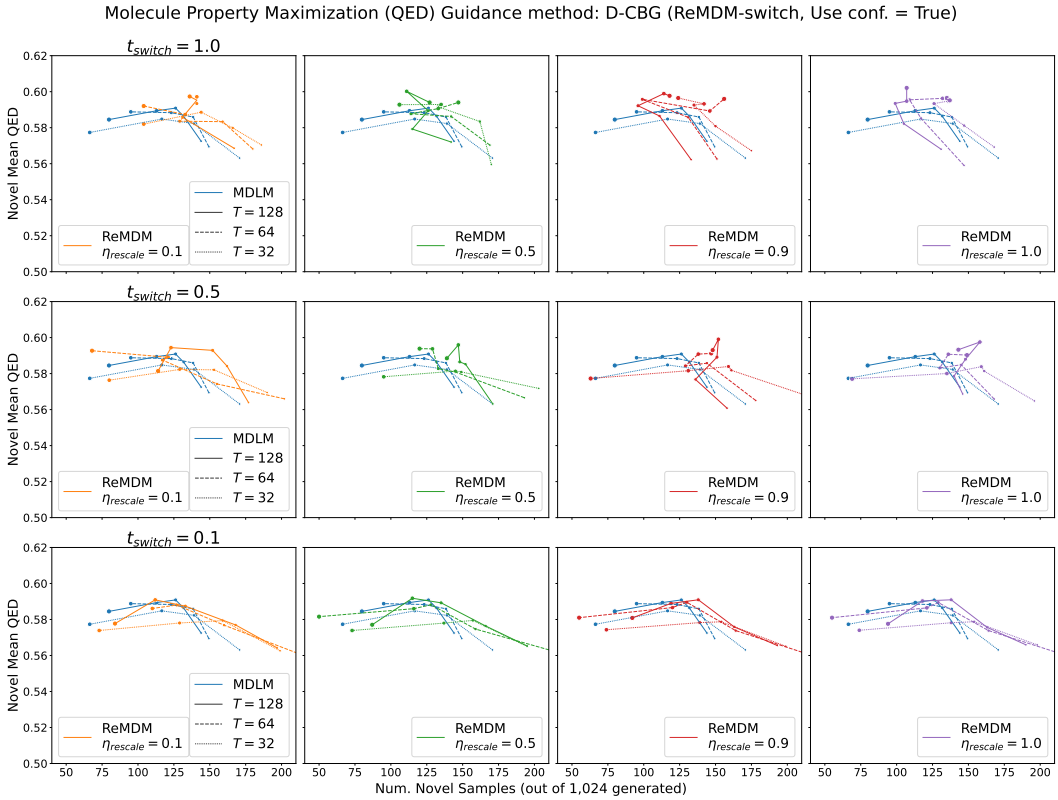


Figure 24: ReMDM-rescaled with **switch** and **confidence-based** schedules hyperparameter tuning for maximizing **drug-likeness (QED)** using **D-CBG**. Larger marker sizes indicate larger γ values.

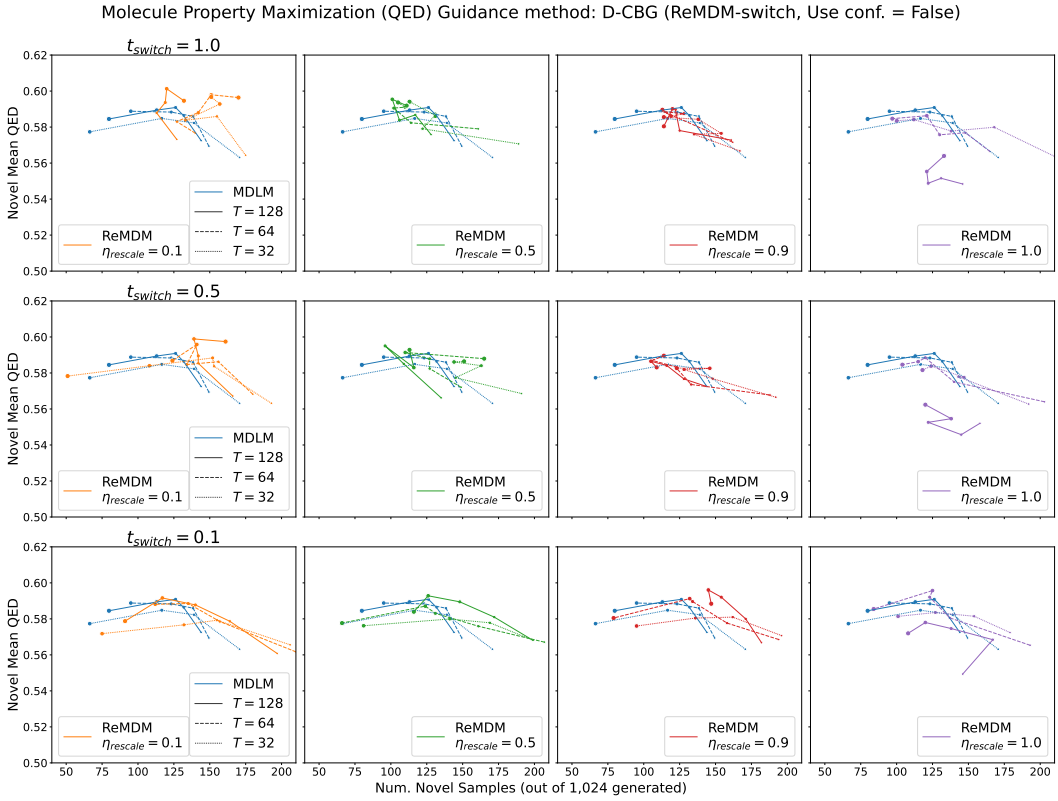


Figure 25: ReMDM-rescaled with **switch** and **without confidence-based** schedules hyperparameter tuning for maximizing **drug-likeness (QED)** using **D-CBG**. Larger marker sizes indicate larger γ values.

G ASSETS

In Table 7, we list the datasets (and corresponding licenses, when available) used in this work. In Table 8, we list the software packages (and corresponding licenses) used in this work.

Table 7: Datasets (and corresponding licenses) used in this work.

Dataset	License
ImageNet Deng et al. (2009)	ImageNet License
OpenWebText Gokaslan & Cohen (2019)	Creative Commons CC0 license ("no rights reserved")
QM9 Ruddigkeit et al. (2012); Ramakrishnan et al. (2014)	N/A

Table 8: Software (and corresponding license) used in this work.

Library	License
Guided Diffusion Dhariwal & Nichol (2021)	MIT
HuggingFace (Wolf et al., 2019)	Apache 2.0
Hydra (Yadan, 2019)	MIT
Jax (Bradbury et al., 2018)	Apache 2.0
NumPy (Harris et al., 2020)	NumPy license
MaskGiT Chang et al. (2022)	Apache 2.0
Matplotlib (Hunter, 2007)	Matplotib license
OmegaConf	BSD 3-Clause
Pandas (pandas development team, 2020)	BSD 3-Clause "New" or "Revised"
PyTorch (Paszke et al., 2019)	BSD-3 Clause
PyTorch Lightning (Falcon & The PyTorch Lightning team, 2019)	Apache 2.0
RDKit Landrum et al. (2013)	BSD 3-Clause "New" or "Revised"
Seaborn (Waskom, 2021)	BSD 3-Clause "New" or "Revised"

—endoftext— might be seen as a tough choice, it's smart to do so.
According to past reports, the SIM cards are only made available in the powerful manufacturer's range of carrier variants. And though the ZTE does its own case with these phones, this variant will not be disclosed.
We also know the color of the phone and what it also will cost remains to be seen. With the Carrington showdown, the new edition of the exclusive TVD features more pictures with Samsung.
—endoftext— Goodes
The Derby winner was named first-team of the Match on the BBC evening.
He feels his position as manager at Southall last season was a great honour and understands the surarious delight in earning that he can't complain about delivering an endless ovation.
The Newcastle coach knows how describes how people like their approach to games. As his biggest fans, he has criticised Paul Cruy and traded barbs with Chris Froome.
And sad to admit, he admits he did not get everything right. He just betrayed them in his intention and left an exasperation in others.
I've finished telling players that we'd get that, and they continued that attitude to how we didn't think we deserved the job ©Newcastle United Rovers manager Tony Wheeler
All this sound of being needed for interviews in various places.
"I'm quite pleased by the passion the fans have shown for me but believe me, I'm not the fan - I find the person out. It would be an extremely nice position to be in.
Illuminate. We'd like to think, not to get a player based on Newcastle's team when he does take charge.
"My head coach says, 'Listen, it's just too much to cry,' Tony Wheeler told the BBC on Tuesday about the injury he played in the Northern League final at Paroles.
"If the player is a part in the starting team, then I'm confident he would be put in there, but I also've finished telling players we thought we'd get the job. We had good options to go in the squad.
"We had, obviously, the people that were always willing for us to get the best player we could because as coaches we felt we had the best players, and we felt that no one had a choice.
"And the only thing [I] was just going on to say was that we were aware beforehand that it was an unusual situation for us to hook him up with the potential of [Meagan] Forster.
"We like his game and he likes ours, and if he's a good striker, he can go out there and play both for the team and for ourselves.
"Stephen [Biggish] Jackson, who's worked with here, obviously respects me with great respect and likes his opinions, but I think [Tony Wheeler] can say that it's satisfactory.
"We've got lots of many, and them combine them with Van Ferdinand. All three people we've been approached to, they need to be taken.
"Could something be done to them? Not be done."
United are certainly prepared to lose their star man for Tuesday's Rovers clash, with Borby to take charge of their trip to Dunferm on Saturday.
Although they have won five of their six games, they sit sixth in the Northern League, which includes the Parkhead and the two assistant charges but on Tuesday their only goal of the season is worthy of their reward.
United have more goals than Palace, scoring both goals this season despite the best form of Johnson as new coach
Winger Darius Boyd joined Derby after replacing Marlon Wickham in June and recently, he stayed for a year on loan until he joined Newcastle in defiance of Tony Wheeler.
Tom Williams, a future United captain, defender and striker, told BBC One Sports: "Being in Rovers is great for them.
"They develop their chemistry very well in the dugout and they try to stay in good relationships together.
"They're unabashed about that you can play with both of them now, so they've stayed loyal."
Brendan Rodgers, the current Newcastle United manager after Wheeler took over from Mick Jagger, and succeeded David Moyes, as the most paid Premier League manager in 1997 while the Scot was the highest paid Everton manager.
Leicester manager Leicester Leicester is yet to make terms with the joys of the job and has become so indebted he offered to forgive Newcastle United's owners for the cost of lost profits after buying a stake in the club earlier this year.
—endoftext— Sometimes, I enjoy having colleagues "embarrassed" to compete in sports, especially it comes to being a one-richen employee's career. I can't own the jobs of the colleagues that pressure us from Chinese and Australian competitive athletes and even to compete in the United States. They seem arrogant
—endoftext—

Figure 26: Sample from MDLM and $T = 4096$ diffusion steps. **—endoftext—** token is **bolded** to emphasize the boundary between paragraphs.

—endoftext— a component to renovating the park, but no one wants to involve others in it before facing a backlash.
"We are going to stand up and try to make a real statement on this," Thomas told reporters at a rally Tuesday night. "I would much rather have that there be a consensus not just to make a decision, but to have a very real discussion, and to begin to build a movement."
The idea behind the proposal is a call to the public, council and the youth advisory board, which was approved by residents on the first day of the meeting.
Thomas said the message will be important to Hamilton youth.
"It's going to be very much about a number of people," he said. "There will be arguments on both sides. It is not going to be all about the type of person that you are or the committee member that you are with a particular number of people."
Thomas said that the plan is really about an initiative to help the community do both of those things.
"This isn't going to be a positive thing, as it should be, but it's going to put people in position to make sure that our focus is not just about how to deal with poverty . . . and with homelessness. Our focus is on keeping the people in our community affordable and to create opportunities for their families.
"We have —believe it or not —we have increased community involvement in our city," Thomas said.
"I think it's obvious from all of the issues now that we'll consider resource-generating development and how to devribute those resources."
The renovated park is one of the 14 parks identified in the city's 2008 map of the park's 2-acre site, which was opened up for a community arts festival.
City staff started coming up with the project in late December, but the public has yet to come to terms.
For their part, residents sign the petition only to receive two differing opinions.
Although there are concerns that it would be an uphill battle that requires council approval, officials said Tuesday that the process could eventually take shape, with councillors likely to support it.
Hamilton city hall spokesman John Little told reporters that the city would have to address a petition from the public if there is one on site.
"We are not able to change the outcome because we are discussing it internally, so what we have has not made an official decision," Little said.
Thomas said staff wants to hear how the public feels about the project, and it must be incorporated in such a prompt fashion that the city is looking at more permanent changes.
"We implemented a 30-year plan, about 10 years ago, and we will continue it this year," Thomas said.
"We want to once again really emphasize community involvement. I mean, as we say we take this kind of step, we don't want to make separate efforts to change different things," he said.
Any plans would be discussed with community boards, Conley said.
Hamilton city councillor Steve Cowen said if residents sign the petition, "actions have to be done, knowing when these changes are needed and that we will have the right options to move forward."
Cowen said two renovations along with suggestions for revenue set by the Hulbert Company, which operates the Gerrard Elementary school theatre, will be included on city documents and website. They would set aside $2 million more than the result of Washtenaw Park Plaza renovation plans, sparked by the QEDA expansion that renovated the former Washtenaw Historic Restaurant and Bakery on Gerrard.
The company has also purchased Gerrard and turned it into a New American Food, Inc. restaurant, and brought the business before the city council in late 2015.
Hamilton city council has been having this conversation for more than a year, but it usually happens only now.
"It's really not, 'We need a meeting soon,' why we need to do this," Conley said. "It's just a forum where people can voice their views and inform council, and within the next year, hopefully council will do the right thing to do."
With John Elliott:
905-526-2431
mmelliott@thespec.com
—endoftext— Written by Steven B. Ollander
In the last five years, we have seen huge financial crises around the world, and not just economic ones. We've also seen rapid changes in economic systems, in central and peripheral countries.
All these systems are in the hands of banks, but there are other forces that have also been used to dump more money into other countries
—endoftext—

Figure 27: Sample from MDLM+DFM and $T = 4096$ diffusion steps. **—endoftext—** token is **bolded** to emphasize the boundary between paragraphs.

—endoftext— The Audi D3 is one of the fastest and most capable sports cars we've seen on the market. The retractable front wings were added to make the D3 R a more sporty car. The front wing design is painted yellow, which is a big surprise. It looks like the A4 R in a completely different color scheme.
While the front wings retract, the cooling system is a drag for sure. The front and rear seats are also light-weighted, which is also convenient for a racing driver. The car also features replaceable steering wheels for higher stability and better handling. D3 R is easy to drive and easy to use thanks to adaptive suspension technology.
Here are a few of the other features Audi had in mind:
Top speed is only four seconds higher, but more turbocharged engine power has been added to make it even faster.
By means, you reach 60 mph in five seconds with strong acceleration. The front wing design makes the engine a higher strength, making it just more powerful.
Speaking of power, D3 R is equipped with a 3.0 litre V8 with Adaptive Brake Control, which means you won't complain if you may have to move around it in wet conditions. The engine has been also upgraded, making a 204.26 horsepower.
This is a 2-liter V8, which means it can easily produce as much as 132 bhp.
By comparison, a 3.0 litre V8 produces 227.4 bhp which is a bit too much for sporty cars. If you want a sporty car, your only choice is to use this kind of engine in the guise of the D3 R.
Other innovative changes that create better handling in the car are the retractable rear wing and the front suspension. The forks, front and rear, are fold-down. The 17-inch adjustable steering wheel makes it easier for the driver. The car also sports a sport seats, which is quite different from the A4 R.
Behind the steering wheel, the sport driving mode has a full black screen touch option. You can move the car up and down by taping the button. The buttons can be mounted on the display, giving you more input of the screen. With the game on, you can walk to the steering wheel and then the interior button to start. The push button is an optional feature, similar to A4's push button.
The car also has a dedicated battery option that allows you to start the sport mode if the car is on the clock. In addition to these features, the car also has four different lights on the dashboard.
The car also has an Assist Compensate Brake function that lets you drive the car in a more controlled way by depending if the car is stopped. This is a nice feature if you are a more advanced driver.
D3 R comes with two different driving systems. The advanced driving system features a side-by-side aggressive driving mode that lets you drive the car more aggressively at low speeds or by braking at higher speeds. It also includes a complimentary steer-by-wire braking function.
When driving the car, you can take control of the headlights by simply turning on the headlights and turning on the gearbox to speed up. When you're in the park, you can press the manual park controller, and you'll see the power gauge on the steering wheel.
It produces 211.4 bhp which is higher than the power made by the first-generation sporty in cars like the R8 V6. The car is able to push the limits and create dynamic performance that can take on power and drag extremely quickly.
The D3 and A4 R come with four different modes of driving. The four modes are being labeled as Focus, Safety, and Defensive Proactive.
Additionally, there are four different lights on the dashboard. One is an LED display directly next to the other lights in the car. Another is a 17-inch display located behind the steering wheel. When you press it on the wheel, the display then shows the performance of the car again.
Compared to the Audi A4, you'll get better fuel economy, fewer accidents, and less noise. The D3 R is Audi's most balanced car, and we're happy to say that the D3 R stands as the single most overall balanced sports car on the global market.
PHOTOS by Sportscar365
Links:
Please note that the user name is the source for this article. Local or media images may be removed.
—endoftext— Written by Nick Ohr and James Chablofsky
In its 13-episode second season, the comedy The Good Wife blends action with the real world, bringing the story of a couple: Arianna (Garrett Rossi) and Michelle (Amy Braid) and Caitlin
—endoftext—

Figure 28: Sample from ReMDM and $T = 4096$ diffusion steps. **—endoftext—** token is **bolded** to emphasize the boundary between paragraphs.