MULTI-TURN REFLECTIVE MASKING ELICITS REASONING IN MASK DIFFUSION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

While reasoning on autoregressive (AR) models is often performed by chain-of-thought reasoning and reflection, their refinement of previous outputs still relies on fully sequential generation, even when only local edits are needed. In contrast, the masking mechanism in Mask Diffusion Models (MDMs) naturally supports explicit local edits on previous outputs, allowing selective refinement without discarding previous answers and generating another from scratch. While this property more closely aligns with how humans correct mistakes by iterative local refinement, standard MDM decoding does not allow previously revealed tokens to be actively revisited and re-masked for further revision. We propose **Reflective Masking** (RM), a lightweight post-training framework that equips MDMs with context-dependent iterative self-correction. RM enables a model to revisit and revise prior outputs based on the evolving context, providing an adaptive way to allocate additional inference computation to selective revision rather than forward generation alone. To exploit information from previous turns, we further introduce **History Reference**, a parameter-free mechanism that incorporates intermediate denoising states during revision. Our approach requires no architectural changes and is easily applicable to existing MDMs. Across diverse tasks including text generation, Sudoku, and image editing, **Reflective Masking** consistently outperforms standard masking-based baselines, suggesting that explicit in-place revision can serve as a useful primitive for iterative reasoning and self-correction in MDMs.

1 INTRODUCTION

While Large Language Models (LLMs) have been widely accepted for performing various reasoning tasks (Kojima et al., 2022; Wang et al., 2022; Lightman et al., 2023; Wei et al., 2022), recent studies have revealed that they still struggle in multi-turn and long-horizon settings (Laban et al., 2025): models tend to propagate previous errors and incorrect assumptions rather than revise them. A key limitation of their autoregressive (AR) paradigm is that local errors require regenerating the entire sequence, incurring unnecessary computation. Moreover, incorrect intermediate content persists in the context, occupying valuable capacity and contaminating subsequent reasoning.

In contrast, Mask Diffusion Models (MDMs) (Austin et al., 2021a; Sahoo et al., 2024; Nie et al., 2025; Xin et al., 2025) provide a fundamentally different generation mechanism that naturally supports localized revision. Through iterative masked updates, MDMs can keep the current context fixed while resampling uncertain tokens, avoiding the need to regenerate the entire sequence. This native remasking mechanism reveals a potential advantage of MDMs for iterative self-correction: if local errors can be revised in place, the model could maintain a cleaner intermediate context, reduce error propagation, and avoid unnecessary regeneration. In this sense, MDMs offer a natural foundation for localized self-correction that is difficult to realize in AR generation.

However, realizing this potential requires moving beyond the passive remasking behavior of existing MDM decoding. Standard MDM generation still follows an absorbing Markovian denoising process: once a token is confidently denoised, it is fixed. Existing remasking strategies focus on re-masking low-confidence tokens, but the model is unable to actively revisit and correct previously committed predictions. We argue that enabling such context-dependent revision is an important primitive for iterative self-correction in MDMs. To this end, the model must learn to identify unreliable predictions and actively revise them during generation. This aligns with recent theory that casts MDM reasoning

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

as a planner–predictor process (Svete & Sabharwal, 2025), a direction DiffusionGemma (Team et al., 2026) validates at scale, albeit at the cost of large-scale pretraining and task-specific fine-tuning. We therefore propose **Reflective Masking**, which instead elicits this behavior from an existing MDM **without extensive training**, formulating masking as an internal decision process driven by the model's uncertainty that turns adaptive position selection into an explicit, learnable revision policy.

In this work, we activate **Reflective Masking as a first-class capability** of MDMs by making it a central focus of post-training. We introduce a lightweight training paradigm that enables **self-initiated, context-aware revision** without architectural modifications, together with an effective data generation strategy that produces stable training signals aligned with the model's native output distribution. Our results show that this revision behavior can be reliably elicited through appropriate post-training. More broadly, RM provides a distinct way to use additional inference computation in MDMs: rather than allocating all computation to forward denoising, the model can spend extra steps selectively revisiting and correcting previously committed content. We therefore view Reflective Masking as a primitive for iterative state refinement and self-correction, which can support reasoning processes that require revising earlier decisions.

With multi-turn reflective masking, a key capability still missing relative to autoregressive reasoning is the ability to extract insights from the growing context of historical generations. To bridge this gap, we introduce **History Reference**, a parameter-free mechanism that enables MDMs to maintain a stateful view of their denoising trajectory by preserving intermediate decoding states. Unlike AR models, which encode history implicitly in the input context, History Reference allows MDMs to explicitly leverage past predictions. This mechanism introduces a temporal dimension orthogonal to the current text context, guiding reflective updates and improving consistency while reducing repeated errors during iterative refinement. Importantly, History Reference requires no additional learnable parameters or external memory, making it efficient and readily applicable to existing MDMs.

We evaluate our approach across a spectrum of generation tasks with varying levels of guidance, demonstrating consistent improvements and strong generality. We begin with image editing, where rich instructions specify both where and how to modify the input. We then consider Sudoku, a structured reasoning task that requires the model to identify and correct erroneous entries. Finally, we study text generation tasks, where supervision is minimal and no direct hints about the final answer are provided. Across all three settings, Reflective Masking consistently improves performance. For tasks that require autonomous exploration, History Reference proves particularly effective in stabilizing iterative revision and reducing repeated errors. Together, these results suggest that reflective masking provides a general mechanism for iterative self-correction through explicit revision in mask-based generation.

Our contributions can be summarized as follows:

- We identify **Reflective Masking** as a latent revision capability of MDMs and formulate generation as an iterative process in which previously committed predictions can be selectively revisited and refined. This provides a new primitive for context-dependent self-correction in MDMs.
- We propose a lightweight training paradigm, associated with an effective and scalable data pipeline, to activate Reflective Masking as an intrinsic skill, enabling context-aware and adaptive revision without architectural changes. We demonstrate its effectiveness across diverse downstream tasks and modalities, including text generation, Sudoku, and image editing, highlighting strong generality.
- We introduce **History Reference**, a parameter-free mechanism that enables MDMs to maintain a stateful view of their denoising trajectory by incorporating intermediate decoding history. It improves revision consistency and avoids repeating past errors during iterative refinement.

2 Reflective Masking

Reflective Masking (RM) treats every position at every denoising step as a per-position decision: *keep* the current token, *re-mask* it to `MASK` for re-prediction, or, if currently masked, *reveal* a new token. We present RM in the following parts: the inference decision rule that maps model probabilities to per-position keep / re-mask / reveal actions (§2.1); History Reference (HR), a parameter-free per-position history aggregation mechanism that summarizes the denoising trajectory into a history-aware embedding for the model (§2.2); the training objective that enables models' Reflective Masking capability and the recipe used to construct training inputs (§2.3).

Notations. Let $x^* \in \mathcal{V}^N$ denote the target sequence and $E \subseteq \{1, \ldots, N\}$ the set of editable positions (full sequence excluding the instruction prompt). For each $i \in E$, the position can take one of three values throughout the denoising trajectory: $\tilde{x}_i^{(t)} \in \{w_i, \texttt{MASK}, x_i^*\}$ where w_i is a task-specific wrong token. Non-edit positions $i \notin E$ stay at x_i^* at every step. We write T for the total number of denoising steps and $\bar{\mathcal{V}} := \mathcal{V} \cup \{\texttt{MASK}\}$ for the extended vocabulary.

2.1 Basic inference rules on MDMs with Reflective Masking enabled

Unlike the standard MDM inference rule, which follows an absorbing Markov process, we introduce Reflective Masking that allows the model to revisit and revise past decisions. To enable this, we define a new inference rule. At timestep $t \in \{0, 1, \ldots, T-1\}$ the model takes the current state $\tilde{x}^{(t)}$ as input and outputs a per-position categorical $p_\theta(\cdot \mid \tilde{x}^{(t)})_i$ over $\bar{\mathcal{V}}$. The next-step state is then determined per position by a deterministic rule that splits on whether the position is currently non-mask or masked. MASK is denoted as M:

$$\tilde{x}_i^{(t+1)} = \begin{cases} M & \text{if } \tilde{x}_i^{(t)} \neq M \text{ and } p_\theta(M \mid \tilde{x}^{(t)})_i > p_\theta(\tilde{x}_i^{(t)} \mid \tilde{x}^{(t)})_i \textit{ (RM),} \\ \tilde{x}_i^{(t)} & \text{if } \tilde{x}_i^{(t)} \neq M \text{ and } p_\theta(M \mid \tilde{x}^{(t)})_i \leq p_\theta(\tilde{x}_i^{(t)} \mid \tilde{x}^{(t)})_i \textit{ (Keep),} \\ \arg\max_{v \in \mathcal{V}} p_\theta(v \mid \tilde{x}^{(t)})_i & \text{if } \tilde{x}_i^{(t)} = M \textit{ (Reveal).} \end{cases} \quad (1)$$

The rule is intuitive. At a non-mask position, re-mask whenever the model assigns higher probability to MASK than to the current token, because the model is signaling that the non-mask token is wrong; otherwise keep it. At a MASK position, reveal the most likely vocabulary token. In general, at each iteration, the model can produce one of three actions per position: keep, re-mask, or reveal, all driven entirely by the model's per-position output distribution.

This basic rule as presented conditions only on the current state $\tilde{x}^{(t)}$. As a result, the model may produce the same state across different time steps, leading to a loop where identical states are repeatedly passed to subsequent steps. We address this in §2.2 by changing what the model sees as input, while leaving equation 1 unchanged.

2.2 Enhancing Reflective Masking with History Reference

To let the model condition on more than the current state, we maintain a per-position accumulated embedding that compresses the prefix $\tilde{x}_i^{(0:t)}$ into a single vector and feed it to the model. The accumulated embedding adds no learnable parameters and admits an $O(1)$ per-step update (Appendix C).

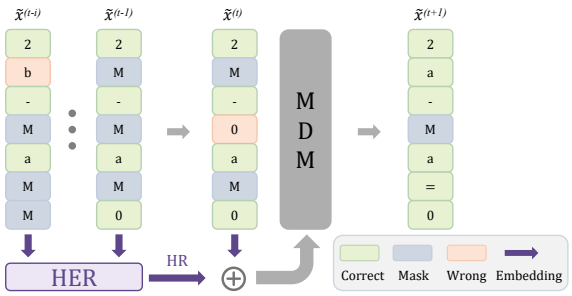


Figure 1: **Inference procedure with History Reference.** All states are embedded, and the historical states are further processed by HER. The embedding of current step $\tilde{x}^{(t)}$ is added to the history reference (HR) and fed into the model as the input, which predicts the next step $\tilde{x}^{(t+1)}$.

Setup. Let $e_i^{(k)} := \mathrm{wte}(\tilde{x}_i^{(k)})$ denote the model token embedding of position i at step k. Let R_Δ denote the *History Embedding Rotation (HER)* indexed by the distance $\Delta = k - t$ between a historical step k and the current step t, satisfying the standard rotation composition rules

$$R_0 = I, \qquad R_a R_b = R_{a+b}, \qquad R_{-\Delta} = R_\Delta^\top.$$

R_Δ is a parameter-free block-diagonal rotation that applies a distance to each two-dimensional block of an embedding. We instantiate it with the same two-dimensional sinusoidal blocks used by standard rotary encodings (Su et al., 2024).

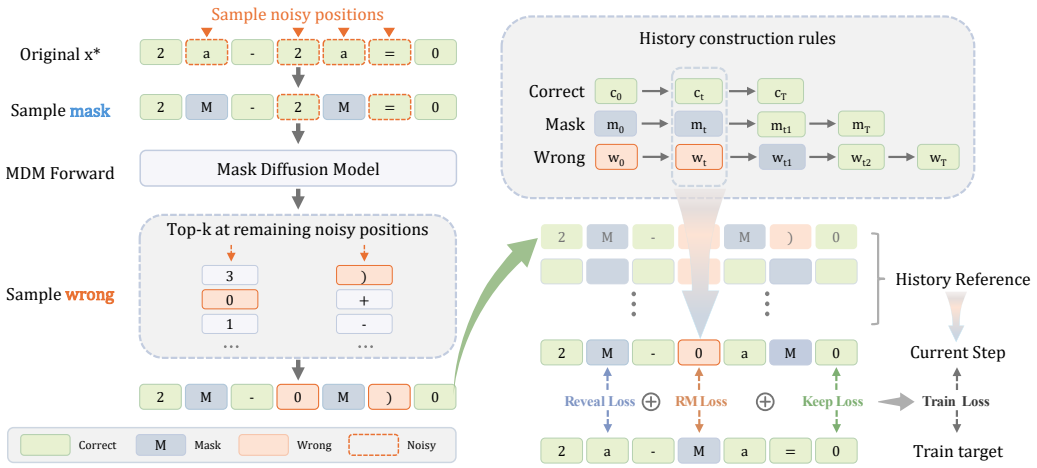


Figure 2: **Synthetic history data construction for Mask Diffusion Model training**. A noisy sequence is created from a clean sequence using mask and wrong-token corruption. Position-wise transition rules are then used to sample synthetic histories and define training targets.

History embedding. At iteration t we express every historical state in the current step's reference frame and accumulate them into a single vector:

$$a_i^{(t)} \;=\; \sum_{k=0}^{t} \gamma^{\,t-k}\, R_{k-t}\, e_i^{(k)}, \tag{2}$$

where $\gamma \in (0, 1]$ is a history-decay factor. Equivalently,

$$a_i^{(t)} \;=\; e_i^{(t)} + \gamma\, R_{-1}\, e_i^{(t-1)} + \gamma^2\, R_{-2}\, e_i^{(t-2)} + \cdots + \gamma^t\, R_{-t}\, e_i^{(0)}.$$

The current state contributes unrotated, while each past state is added with a lag-dependent rotation R_Δ and decay $\gamma^{|\Delta|}$. Two sequences that share the same current state are augmented by their histories, which provide additional conditioning signals. This historical context allows the model to reference past states when deciding whether to re-mask or keep a token, helping it avoid recurring errors and leverage useful cues from earlier versions.

2.3 Training paradigm for enabling Reflective Masking in MDMs

To enable the model to follow the inference rule of §2.1–§2.2, we design a training paradigm that aligns its per-position outputs with the correct next-step actions. Concretely, we train the model with per-position oracle labels, encouraging its output distribution to assign the highest probability to the desired next action at each position.

Oracle revision rule. For each $i \in E$ and any state $z_i \in \{w_i, \texttt{MASK}, x_i^*\}$, the optimal next-step action is the deterministic function

$$\tau(z_i, x_i^*) \;=\; \begin{cases} \texttt{MASK} & z_i \in \mathcal{V} \setminus \{x_i^*\} \quad \text{(re-mask wrong / initial token)} \\ x_i^* & z_i = \texttt{MASK} \quad \text{(reveal target)} \\ x_i^* & z_i = x_i^* \quad \text{(preserve target).} \end{cases} \tag{3}$$

A non-mask token that differs from the target should be re-masked, a MASK should be revealed to the target, and a correct non-mask token should be preserved. Non-edit positions are deterministically labelled x_i^* at every step.

Training data curation. As illustrated in Fig. 2, a training instance is constructed by simulating a synthetic trajectory, starting from an original clean sequence x^*. We first sample a subset of positions to corrupt and draw a timestep $t \sim \text{Uniform}\{0, \ldots, T-1\}$. At this timestep, the selected positions are split into two groups: one is replaced with MASK tokens, while the other is substituted with wrong

tokens. Wrong tokens are sampled from a corruption distribution $\nu(\cdot \mid x_i^*)$ over $\mathcal{V} \setminus \{x_i^*\}$. In practice, ν is chosen to better match inference-time errors, e.g., using top-k predictions from a frozen MDM backbone (excluding the ground-truth token) or task-specific token distributions. Appendix B.4 bounds the resulting shift of the training objective relative to a uniform proposal. This yields the current state $z = \tilde{x}^{(t)}$.

To approximate inference-time dynamics, we construct histories following position-wise transition rules (right side of Fig. 2). Each position evolves independently according to its current state: correct tokens remain unchanged; masked tokens transition to correct tokens at a sampled timestep t_1; and wrong tokens first transition to mask tokens at sampled timestep t_1 and then to correct tokens at later timestep t_2. These transitions are governed by per-position rules together with sampled transition timesteps within the T-step horizon, which jointly define the trajectory sampler.

Sampling these transition timesteps produces a set of reference states that mimic iterative refinement during inference. Based on the resulting state, training targets are defined per position: masked tokens are trained to predict the correct token, while wrong tokens are trained to predict the mask token. The overall objective combines reveal, mask, and keep losses. This construction defines a trajectory sampler that specifies the training input distribution, while remaining independent of model parameters θ, providing a stable target for optimization. The full procedure is given in Appendix D.

Training objective. The model receives $a^{(t)}$ and outputs per-position categorical distributions. We supervise each position using an oracle action $\tau(z_i, x_i^*)$ that depends on its current state: masked tokens are mapped to their correct token (reveal), wrong tokens are mapped to `MASK` (mask), and correct tokens are kept unchanged (keep).

We minimize the per-position cross-entropy against this oracle,

$$\mathcal{L}_{\text{train}}(\theta) \;=\; \mathbb{E}\left[\sum_{i \in E} -\log p_\theta\Big(\tau(z_i, x_i^*) \,\Big|\, a^{(t)}\Big)_i\right], \tag{4}$$

where the expectation is over the data distribution, the random trajectory and step sampling.

Equivalently, the objective decomposes into three components: a *reveal loss*, which predicts the correct token for masked positions; a *mask loss*, which predicts `MASK` for wrong positions; and a *keep loss*, which preserves correct tokens. This decomposition matches the training targets induced by the synthetic trajectory construction in Fig. 2.

Two properties of this objective justify its use. First, given a sufficiently expressive model, minimizing equation 4 drives the model's per-input output toward the conditional distribution of the oracle action τ, so the argmax of the trained model recovers the inference rule of equation 1. Second, our theoretical analysis shows that access to historical information in addition to the current state cannot increase the Bayes-optimal training risk. Our implementation realizes this idea through the parameter-free compressed representation $a^{(t)}$ introduced in Sec. 2.2. Please refer to Appendix B for the complete theoretical analysis.

3 Experiments

We evaluate our approach across three representative task categories that differ in the level of external guidance. These tasks are designed to systematically probe how Reflective Masking and History Reference contribute under varying conditions of supervision and exploration. At one end of the spectrum, we consider image editing, where rich instructions directly specify both the target regions and desired modifications. We then study Sudoku, a structured reasoning task where the model must identify and correct erroneous entries with limited guidance. Finally, we evaluate on text reasoning tasks such as mathematical reasoning and code synthesis, where no explicit hints about the final output are provided. This progression enables us to examine the roles of RM and HR, and to understand how they interact across tasks with increasing reasoning complexity and decreasing supervision. The training on these three tasks can be completed within **5 hours** on 2 NVIDIA H100 80GB GPUs, avoiding the **days or weeks** long training associated with architecture-modifying approaches.

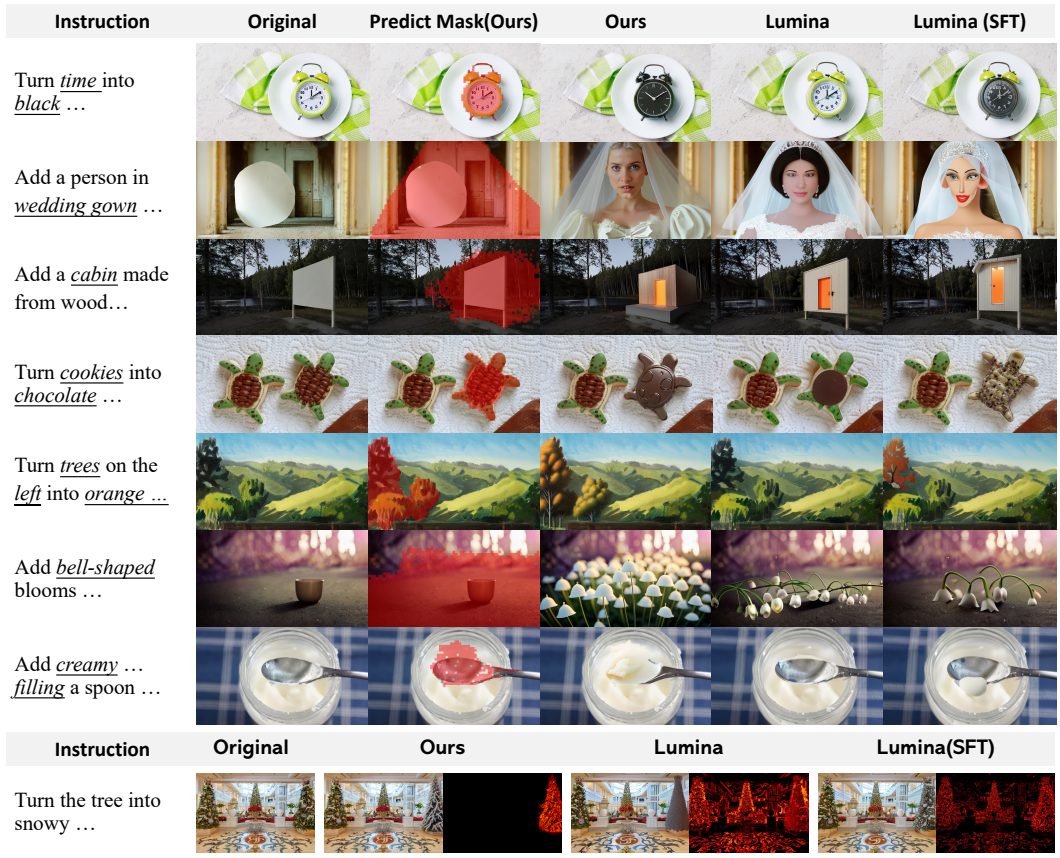


Figure 3: **Qualitative results on the image editing task.** The masks predicted by RM are highlighted in red. Heat maps at the bottom visualize pixel-wise differences between the edited images and the originals. Please refer to Appendix E for the complete editing prompts and more qualitative results.

3.1 Instruction-based in-place image editing

We begin with image editing (Brooks et al., 2023; Zhang et al., 2023; Sheynin et al., 2024; Hertz et al., 2022), a task characterized by strong external guidance from natural language instructions. Given an input image and an editing instruction, the model is required to localize the regions to be modified and generate corresponding updates that align with the semantic intent, while preserving the rest of the image. In this setting, the instruction provides explicit cues for both where and how to edit, leaving limited need for autonomous exploration. As a result, this task primarily evaluates whether Reflective Masking can enable precise, localized revisions based solely on instruction signals, without relying on iterative self-discovery or historical context.

Experiment settings. We use Lumina-DiMOO (Lumina) (Xin et al., 2025) as the base model. For training and evaluation, we sample 85,000 examples from ImgEdit (Ye et al., 2025b) as the training set and hold out another 1,700 examples as the test set. To ensure a fair comparison, our method and the vanilla SFT baseline are trained under the same experimental setup.

Table 1: Quantitative results on the image editing task

Method	Edit Localization		Background Preservation			Overall Editing Quality	
	Prec. ↑	Coverage ↑	MAE-RGB ↓	PSNR (dB) ↑	SSIM ↑	VQAScore ↑	User Study ↑
Lumina	65.81	41.68	12.497	23.092	0.6256	71.95	41.8
Lumina+SFT	71.84	48.42	11.035	23.900	0.6570	81.61	53.3
RM (Ours)	**99.73**	**73.02**	**3.613**	**34.759**	**0.9744**	**85.17**	**68.2**

Results. Fig. 3 and Tab. 1 present the qualitative and quantitative results on image editing. As shown in Fig. 3, our method accurately localizes the regions that require editing and masks them for subsequent generation. This provides the MDM with a cleaner current state, leading to higher-quality edits than the baselines. Moreover, since our method only regenerates the masked regions, the unmasked areas remain well aligned with the input image. This is further demonstrated by the heat maps in Fig. 3, where our changes are concentrated in the target edit regions, while the baselines introduce more globally distributed editing noise, leading to noticeable consistency issues that distort unaffected regions and even corrupt fine-grained details.

For quantitative evaluation, we consider three aspects of image editing quality: localization accuracy, background preservation, and overall editing quality. We evaluate them using Edit Precision and Coverage, MAE-RGB, PSNR, SSIM, VQAScore, and a 29-participant user study. Detailed metric definitions and the user-study protocol are provided in Appendix G. As shown in Tab. 1, our method outperforms baselines across all metrics.

3.2 SUDOKU REVISION

We next consider Sudoku (Palm et al., 2018; Wang et al., 2019), a structured reasoning task that requires the model to detect and correct errors in a partially incorrect grid. The input consists of Sudoku boards containing invalid entries, and the model must iteratively refine its predictions to produce a valid solution. Unlike image editing, this task requires the model to actively identify inconsistencies and explore possible corrections. At the same time, the search space is highly constrained due to the limited vocabulary and strict structural rules, making Sudoku a controlled environment for studying iterative revision behaviors. This setting provides a challenging testbed for Reflective Masking and allows us to evaluate how History Reference facilitates effective failure avoidance in the absence of explicit instructions about where errors occur or how revisions should be performed, thereby highlighting the importance of the HR mechanism for autonomous error localization and refinement.

Table 2: Quantitative results on Sudoku revision

Variant	Exact Accuracy(%) ↑	Valid Rate(%) ↑	Replay Mistake(%) ↓	Conflict Cells (Number / board) ↓
`RM w/o HR`	82.4	86.6	0.57	0.578
`RM + HR (history sum)`	91.4 (↑ 9.0)	91.8 (↑ 5.2)	0.07 (↓ 0.50)	0.300 (↓ 0.278)
`RM + HR + decay`	89.4 (↑ 7.0)	89.6 (↑ 3.0)	0.07 (↓ 0.50)	0.362 (↓ 0.216)
`RM + HR + decay + HER`	**93.4** (↑ 11.0)	**93.6** (↑ 7.0)	**0.03** (↓ 0.54)	**0.236** (↓ 0.342)

Experiment settings. We construct a lightweight MDM with four Transformer layers and 0.81M parameters. We construct test examples from solved 9×9 boards by corrupting a specified number of cells (randomly chosen from 4 to 20), replacing the original digits with incorrect values. The model is tasked with correcting the corrupted board through iterative re-masking and re-prediction.

Results. We evaluate four complementary metrics: Exact Accuracy, Valid Rate, Replay Mistake, and Conflict Cells, measuring exact recovery, Sudoku validity, repeated mistakes during revision, and remaining constraint violations, respectively. Definitions and motivation are provided in Appendix G.

As shown in Tab. 2, adding HR greatly reduces repeated mistakes and constraint conflicts compared with the variant without HR, suggesting that History Reference helps the model avoid revisiting the same erroneous predictions. However, introducing a decay factor alone still improves over the no-history baseline but performs worse than the HR-only variant, indicating that simply weakening historical signals is insufficient for effective correction. After further incorporating HER, which explicitly disentangles historical information, applying decay becomes beneficial. Our full method achieves the best performance across all metrics. These results suggest that properly structuring historical information, rather than merely attenuating it, is crucial for improving iterative error correction, leading to more accurate Sudoku solutions with fewer rule violations.

3.3 TEXT REASONING TASKS

Finally, we evaluate on text generation tasks, including mathematical problem solving and code generation, which require autonomous multi-step solution generation. Given only a problem description,

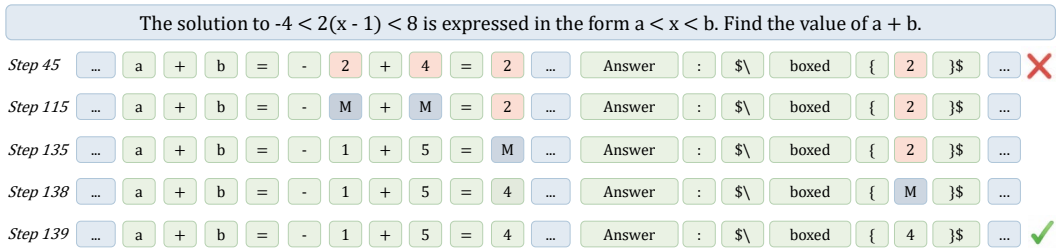


Figure 4: RM actively re-masks and corrects tokens during inference based on the evolving global context. As the preceding context is refined, the correction propagates to the final answer. Please refer to Appendix F for more results and representative revision trajectories.

the model must generate complete solutions without any direct hints about the final answer. These tasks present the most challenging setting, as the model needs to explore an unconstrained output space while maintaining logical consistency. In this regime, both Reflective Masking and History Reference become critical: RM enables iterative revision of intermediate outputs, while HR provides historical context that can help stabilize the revision trajectory and reduce repeated errors.

Table 3: Performance comparison across benchmarks. Δ denotes the improvement over Vanilla SFT

Benchmark	Category	LLaDA(%)	Vanilla SFT(%)	Ours(%)	Δ
MATH-500	Math	19.4	22.4	**24.8**	$\uparrow$ 2.4
MBPP	Code	28.0	30.6	**39.4**	$\uparrow$ 8.8
ARC-Challenge	MCQA	73.7	81.3	**86.1**	$\uparrow$ 4.8

Experiment settings. We use LLaDA (Nie et al., 2025) as the base model and implement it using the dLLM (Zhou et al., 2026) library. We evaluate our method on reasoning benchmarks including MATH (Lightman et al., 2023; Hendrycks et al., 2021; Lewkowycz et al., 2022), MBPP (Austin et al., 2021b), and ARC-Challenge (Clark et al., 2018). MATH and MBPP require multi-step reasoning and unconstrained generation, making them suitable for evaluating the revision capability introduced by RM. For each benchmark, we construct task-specific training data from the corresponding training split to equip the model with the revision capability of RM. For MBPP, we randomly use 30% of the data for training and reserve the remaining 70% for testing. In contrast, ARC-Challenge is evaluated in a standard forward-only multiple-choice setting, serving primarily to verify that introducing RM does not compromise the model's original inference capability on conventional generation tasks.

Main results. As shown in Tab. 3, RM improves over both LLaDA and Vanilla SFT on MATH-500 and MBPP. On ARC-Challenge, where reflective revision is disabled and inference remains forward-only, the post-trained model also retains strong conventional inference performance. We additionally observe consistent improvements across Minerva MATH subject categories, with aggregate accuracy increasing from 22.62% to 24.10%. The full subject-wise breakdown is provided in Appendix F. On MATH-500, we compare against RemeDi, the closest learned-remasking baseline. Using its released checkpoint with block size 512, matching our main evaluation setting, RemeDi achieves 24.4%, compared with 24.8% for our method. RemeDi introduces an additional four-block Unmasking Policy Stream, increasing the model size from 8B to 8.9B, and relies on substantially larger-scale Remask SFT and RL data. In contrast, our method requires no architectural modification and uses only 7.5K task-specific training examples. Thus, RM achieves competitive committed-token revision performance with a lighter adaptation setup. We further ablate HR under two wrong-token proposals: uniform sampling from all non-target tokens, and model-based top-k sampling from a frozen MDM's top-k predictions. HR improves MATH-500 accuracy from 21.6% to 24.8% with model-based top-k sampling and from 18.2% to 20.0% with uniform sampling. These results show that HR consistently benefits revision across different corruption distributions, while model-based wrong-token construction provides a stronger training signal.

We further observe consistent accuracy gains across different inference budgets with modest additional computation. Across four inference budgets, RM consistently improves MATH-500 accuracy by

2.2–2.6 percentage points while requiring only approximately 3–10% additional forward passes. More details are provided in Appendix F.

Revision behavior. Fig. 4 illustrates a representative revision trajectory. The model first re-masks an erroneous intermediate expression and subsequently revises dependent tokens as the surrounding context changes, eventually correcting the final answer. This example shows that RM can revise logic-critical tokens in the generated solution trace rather than only peripheral or low-confidence content. Additional trajectories are provided in Appendix F. We further audit sample-level outcome transitions on MATH-500. Across the evaluation set, RM produces 21 incorrect-to-correct transitions and 0 correct-to-incorrect transitions after reflective revision. Here, the transition audit compares the RM model's state before reflective revision with its final output, rather than Vanilla SFT. Thus, the additional revisions correct multiple initially failed solutions without causing sample-level regression.

4 Related Work

Mask diffusion models and revision task. MDMs (Nie et al., 2025; Ye et al., 2025a; Lou et al., 2023) generate sequences through iterative masked denoising and naturally permit previously generated tokens to be revisited (Gong et al., 2025; Xie et al., 2025). Existing methods, however, primarily use this capability within standard denoising rather than as an explicit mechanism for iterative self-correction. We build on this native property and make revision itself a learned model behavior. Prior work augments autoregressive generation with insertion-, edit-, and regeneration-based mechanisms (Stern et al., 2019; Gu et al., 2019; Guu et al., 2018; Malmi et al., 2019; Mallinson et al., 2020; 2022; Faltings et al., 2021). These methods enable revision but do not provide the same native in-place masked update mechanism available in MDMs.

Revision and re-masking in diffusion models. Recent work studies learned re-masking and self-correction in diffusion models. RemeDi (Huang et al., 2025) introduces a dedicated policy stream, while DiffusionGemma (Team et al., 2026) demonstrates parallel self-correction through large-scale pretraining and task-specific fine-tuning. Other approaches explore mixed noise schedules, predictor–corrector sampling, and per-step resampling (Von Rütte et al., 2025; Wang et al., 2025; Song et al., 2025). In contrast, RM activates context-dependent revision through lightweight post-training without architectural changes. A more detailed discussion of related work is provided in Appendix A.

5 Limitations and Future Work

While we evaluate RM on image editing, Sudoku revision, and text generation, these tasks still remain substantially simpler than the most challenging long-horizon reasoning problems typically studied in AR models, due to the limited reasoning capability of the current base mask diffusion models. In addition, our experiments are constrained by computational resources, and we do not investigate whether the RM capability can effectively transfer under significantly larger-scale training regimes. For future work, applying RM to stronger block diffusion models (Arriola et al., 2025; Cheng et al., 2025; Bie et al., 2025) presents an interesting direction; however, a direct application would only revise tokens within the currently generated block, limiting its effectiveness. Developing mechanisms for revising the full context beyond the current block may therefore be important for enabling more global self-correction and long-range reasoning.

6 Conclusion

In this work, we present Reflective Masking, a lightweight post-training framework that equips Mask Diffusion Models with context-dependent reflective revision. Instead of treating generation as a one-way denoising process, RM enables an MDM to iteratively revisit, re-mask, and refine its previous predictions based on the evolving context. We further introduce History Reference, a parameter-free mechanism that exposes intermediate denoising states to the model and helps stabilize multi-turn revision by reducing repeated errors. Experiments on image editing, Sudoku revision, and text generation suggest that Reflective Masking can serve as a useful primitive for iterative state refinement in reasoning tasks with diffusion-based generative models.

AI USE STATEMENT

In this work, generative AI tools were used only for minor language polishing. They were not used for research ideation, methodology design, experimental analysis, or the formulation of scientific claims. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Marianne Arriola, Aaron Gokaslan, Justin T Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. *arXiv preprint arXiv:2503.09573*, 2025.

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, 2021a.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021b.

Tiwei Bie, Maosong Cao, Kun Chen, Lun Du, Mingliang Gong, Zhuochen Gong, Yanmei Gu, Jiaqi Hu, Zenan Huang, Zhenzhong Lan, et al. Llada2. 0: Scaling up diffusion language models to 100b. *arXiv preprint arXiv:2512.15745*, 2025.

Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 18392–18402, 2023.

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.

Shuang Cheng, Yihan Bian, Dawei Liu, Linfeng Zhang, Qian Yao, Zhongbo Tian, Wenhai Wang, Qipeng Guo, Kai Chen, Biqing Qi, et al. Sdar: A synergistic diffusion-autoregression paradigm for scalable sequence generation. *arXiv preprint arXiv:2510.06303*, 2025.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Luc Devroye, László Györfi, and Gábor Lugosi. *A probabilistic theory of pattern recognition*, volume 31. Springer Science & Business Media, 2013.

Felix Faltings, Michel Galley, Gerold Hintz, Chris Brockett, Chris Quirk, Jianfeng Gao, and William B Dolan. Text editing by command. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5259–5274, 2021.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 6112–6121, 2019.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Jiawei Han, et al. Scaling diffusion language models via adaptation from autoregressive models. *arXiv preprint arXiv:2410.17891*, 2024.

Shansan Gong, Ruixiang Zhang, Huangjie Zheng, Jiatao Gu, Navdeep Jaitly, Lingpeng Kong, and Yizhe Zhang. Diffucoder: Understanding and improving masked diffusion models for code generation. *arXiv preprint arXiv:2506.20639*, 2025.

Jiatao Gu, Changhan Wang, and Junbo Zhao. Levenshtein transformer. *Advances in neural information processing systems*, 32, 2019.

Kelvin Guu, Tatsunori B Hashimoto, Yonatan Oren, and Percy Liang. Generating sentences by editing prototypes. *Transactions of the Association for Computational Linguistics*, 6:437–450, 2018.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022.

Zemin Huang, Yuhang Wang, Zhiyang Chen, and Guo-Jun Qi. Don't settle too early: Self-reflective remasking for diffusion language models. *arXiv preprint arXiv:2509.23653*, 2025.

Quan Huynh-Thu and Mohammed Ghanbari. Scope of validity of psnr in image/video quality assessment. *Electronics letters*, 44(13):800–801, 2008.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35: 22199–22213, 2022.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. Llms get lost in multi-turn conversation. *arXiv preprint arXiv:2505.06120*, 2025.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. *Advances in neural information processing systems*, 35: 4328–4343, 2022.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The twelfth international conference on learning representations*, 2023.

Zhiqiu Lin, Deepak Pathak, Baiqi Li, Jiayao Li, Xide Xia, Graham Neubig, Pengchuan Zhang, and Deva Ramanan. Evaluating text-to-visual generation with image-to-text generation. In *European Conference on Computer Vision*, pp. 366–384. Springer, 2024.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*, 2023.

Jonathan Mallinson, Aliaksei Severyn, Eric Malmi, and Guillermo Garrido. Felix: Flexible text editing through tagging and insertion. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1244–1255, 2020.

Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. Edit5: Semi-autoregressive text editing with t5 warm-start. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 2126–2138, 2022.

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. Encode, tag, realize: High-precision text editing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 5054–5065, 2019.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *arXiv preprint arXiv:2502.09992*, 2025.

Rasmus Palm, Ulrich Paquet, and Ole Winther. Recurrent relational networks. *Advances in neural information processing systems*, 31, 2018.

Subham S Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024.

Shelly Sheynin, Adam Polyak, Uriel Singer, Yuval Kirstain, Amit Zohar, Oron Ashual, Devi Parikh, and Yaniv Taigman. Emu edit: Precise image editing via recognition and generation tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8871–8879, 2024.

Yuxuan Song, Zheng Zhang, Cheng Luo, Pengyang Gao, Fan Xia, Hao Luo, Zheng Li, Yuehang Yang, Hongli Yu, Xingwei Qu, et al. Seed diffusion: A large-scale diffusion language model with high-speed inference. *arXiv preprint arXiv:2508.02193*, 2025.

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. Insertion transformer: Flexible sequence generation via insertion operations. In *International Conference on Machine Learning*, pp. 5976–5985. PMLR, 2019.

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

Anej Svete and Ashish Sabharwal. On the reasoning abilities of masked diffusion language models. *arXiv preprint arXiv:2510.13117*, 2025.

DiffusionGemma Team, Adrien Ali Taïga, James Assiene, Daniele Calandriello, Rahma Chaabouni, João Gante, Tamara von Glehn, Nate Keating, Chris Knutsen, Martin Kukla, et al. Diffusiongemma technical report. *arXiv preprint arXiv:2608.00146*, 2026.

Dimitri Von Rütte, Janis Fluri, Yuhui Ding, Antonio Orvieto, Bernhard Schölkopf, and Thomas Hofmann. Generalized interpolating discrete diffusion. *arXiv preprint arXiv:2503.04482*, 2025.

Guanghan Wang, Yair Schiff, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Remasking discrete diffusion models with inference-time scaling. *arXiv preprint arXiv:2503.00307*, 2025.

Po-Wei Wang, Priya Donti, Bryan Wilder, and Zico Kolter. Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. In *International Conference on Machine Learning*, pp. 6545–6554. PMLR, 2019.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Cort J Willmott and Kenji Matsuura. Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. *Climate research*, 30(1):79–82, 2005.

Zhihui Xie, Jiacheng Ye, Lin Zheng, Jiahui Gao, Jingwei Dong, Zirui Wu, Xueliang Zhao, Shansan Gong, Xin Jiang, Zhenguo Li, et al. Dream-coder 7b: An open diffusion language model for code. *arXiv preprint arXiv:2509.01142*, 2025.

Yi Xin, Qi Qin, Siqi Luo, Kaiwen Zhu, Juncheng Yan, Yan Tai, Jiayi Lei, Yuewen Cao, Keqi Wang, Yibin Wang, et al. Lumina-dimoo: An omni diffusion large language model for multi-modal generation and understanding. *arXiv preprint arXiv:2510.06308*, 2025.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025a.

Yang Ye, Xianyi He, Zongjian Li, Bin Lin, Shenghai Yuan, Zhiyuan Yan, Bohan Hou, and Li Yuan. Imgedit: A unified image editing dataset and benchmark. *arXiv preprint arXiv:2505.20275*, 2025b.

Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. Magicbrush: A manually annotated dataset for instruction-guided image editing. *Advances in Neural Information Processing Systems*, 36:31428–31449, 2023.

Zhanhui Zhou, Lingjie Chen, Hanghang Tong, and Dawn Song. dllm: Simple diffusion language modeling. *arXiv preprint arXiv:2602.22661*, 2026.

A Extended Related Work

Mask diffusion models. MDMs (Nie et al., 2025; Ye et al., 2025a; Lou et al., 2023) generate sequences by iteratively denoising masked inputs (Ghazvininejad et al., 2019; Chang et al., 2022; Li et al., 2022; Gong et al., 2024). Lumina-DiMOO (Xin et al., 2025) extends this to a multimodal setting. While these formulations naturally allow tokens to be revisited through masking (Gong et al., 2025; Xie et al., 2025), existing approaches primarily focus on one-shot denoising and do not exploit this capability for iterative revision.

Editing and revision in autoregressive models. Autoregressive models have also been extended to support revision and editing of generated content. Prior work explores editing capabilities by inserting special tokens or performing span-level regeneration, such as insertion-based generation (Stern et al., 2019; Gu et al., 2019), edit-based modeling (Guu et al., 2018), and controllable text editing frameworks (Malmi et al., 2019; Mallinson et al., 2020; 2022; Faltings et al., 2021). However, such mechanisms are not native to autoregressive generation, which fundamentally operates in a forward-only manner. As a result, revising prior outputs typically requires regenerating entire sequences, introducing additional decoding passes. This makes editing indirect and often inefficient, as the model cannot modify earlier decisions in place. In contrast, mask diffusion models naturally support in-place modification through masking, enabling direct and localized revision of generated content. This structural difference provides a more suitable foundation for iterative self-correction, where previous predictions can be selectively revisited and refined without restarting the generation process.

Revision and re-masking approaches. RemeDi (Huang et al., 2025) introduces self-reflective re-masking with a dual-stream architecture and achieves strong performance on from-scratch text generation, highlighting the importance of revising intermediate predictions in mask diffusion language models. However, RemeDi treats re-masking as an additional capability that must be explicitly learned through architectural modifications and auxiliary training objectives. This design increases both training and inference complexity, and limits its adaptability to existing MDM frameworks. DiffusionGemma (Team et al., 2026) similarly demonstrates the value of parallel self-correction at scale, but achieves this only through large-scale pretraining and task-specific fine-tuning. Like RemeDi, it is confined to text generation. Other works explore variants of re-masking, including mixed noise schedules (Von Rütte et al., 2025), predictor–corrector strategies (Wang et al., 2025), and per-step resampling schemes (Song et al., 2025). These approaches improve sampling efficiency or stability, but treat masking as an externally driven procedure rather than an intrinsic model behavior. In contrast, we view masking as a native capability of MDMs that can be activated, reframing revision as an internal decision process rather than an externally imposed operation.

B Theoretical Analysis

B.1 Bayes-Consistent Revision-Policy Learning

Under a fixed, θ-independent corruption proposal, cross-entropy training on per-position oracle revision labels recovers the conditional distribution of the optimal revision action at every position (Theorem 1). Theorem 7 converts model-side TV distance into excess 0–1 risk; Proposition 9 shows the History Reference conditioning cannot increase Bayes risk.

Notation. Let $\mathcal{V}$ denote the token vocabulary, $\bar{\mathcal{V}} := \mathcal{V} \cup \{\texttt{MASK}\}$, and $E \subseteq \{1, \dots, N\}$ the set of edit positions. A training sample is drawn as follows. Sample $(x^*, c) \sim \mathcal{D}$, then a rule $\rho_i \sim \pi_\rho$ over $\{\text{wrong}, \text{mask}\}$ independently at each $i \in E$. For $\rho_i = \text{wrong}$, draw a source token $w_i \sim \nu_{\text{wrong}}(\cdot \mid x_i^*)$ on $\mathcal{V} \setminus \{x_i^*\}$ and boundaries (β_i, μ_i) from a fixed θ-independent law $\mathcal{B}$ on $\{1 \le \beta < \mu \le T\}$; for $\rho_i = \text{mask}$, draw a single boundary $\mu_i \sim \text{Uniform}\{1, \dots, T\}$. The current-step index is $t \sim \text{Uniform}\{0, \dots, T-1\}$. Given these boundaries, the trajectory $\tilde{x}^{(0:T)}$ is per-position deterministic: $w_i \to \texttt{MASK} \to x_i^*$ at (β_i, μ_i) for wrong-rule positions, $\texttt{MASK} \to x_i^*$ at μ_i for mask-rule positions, and x_i^* throughout for non-edit positions $i \notin E$.

We write $z := \tilde{x}^{(t)}$ for the current state, $H := \tilde{x}^{(<t)}$ for the trajectory prefix, and $\phi(H, c) \in \mathbb{R}^d$ for a deterministic history feature map (the history component of the HR accumulated embedding $a^{(t)}$; see Appendix C). The accumulated embedding satisfies $a^{(t)} = e^{(t)} + \phi(H, c)$ with $e^{(t)} = \text{wte}(z)$.

Let $X := (z, \phi(H, c), c, E)$ denote the analyst-side conditioning. Our proofs condition on X, while the implementation passes only $(a^{(t)}, c)$ to the model and uses z, E to construct the supervision label. We write $p_\theta(\cdot \mid X)_i$ for the model's per-position categorical over $\bar{\mathcal{V}}$. By Proposition 9, the Bayes risk under $\sigma(X)$ lower-bounds the Bayes risk achievable from $\sigma(a^{(t)}, c)$, so Theorem 1's population minimum is a lower bound on the implementation's Bayes risk.

Oracle revision rule. For each $i \in E$, the deterministic oracle next-action is

$$\tau(z_i, x_i^*) \;=\; \begin{cases} \texttt{MASK} & z_i \in \mathcal{V} \setminus \{x_i^*\} \quad \text{(visible non-target token)} \\ x_i^* & z_i = \texttt{MASK} \quad \text{(reveal target)} \\ x_i^* & z_i = x_i^* \quad \text{(preserve target)} \end{cases} \tag{5}$$

This is the single optimal revision action at the current state, not a per-step transition kernel for the trajectory.

Training objective. The training loss is the per-position three-state cross-entropy summed over edit positions:

$$\mathcal{L}_{\text{train}}(\theta) \;=\; \mathbb{E}_{\mathcal{D}\otimes q}\left[\sum_{i\in E} -\log p_\theta\big(\tau(z_i, x_i^*) \,\big|\, X\big)_i\right], \tag{6}$$

where $\mathcal{D} \otimes q$ denotes the joint distribution generated by the sampling order above.

Conditional label distribution. For each $i \in E$, define

$$P_i^*(y \mid X) \;:=\; \Pr_{(x^*, w, \rho, \text{boundaries}, t)\sim(\mathcal{D}\otimes q)\mid X}\big[\tau(z_i, x_i^*) = y\big]. \tag{7}$$

This is the marginal distribution of the oracle label given the model's input. Since X does not include x^*, P_i^* is in general non-degenerate.

Theorem 1 (Population minimizer recovers the conditional label distribution)**.** *Assume the rich-family idealization: $\{p_\theta\}$ is large enough that for some θ, the model's per-position output $p_\theta(\cdot \mid X)_i$ matches $P_i^*(\cdot \mid X)$ jointly at every X in the support of $\mathcal{D} \otimes q$ and every $i \in E$. Then any population minimizer $\theta^* \in \arg\min_\theta \mathcal{L}_{\text{train}}(\theta)$ satisfies*

$$p_{\theta^*}(\cdot \mid X)_i \;=\; P_i^*(\cdot \mid X) \quad \textit{for } (\mathcal{D}\otimes q)\textit{-a.e. } X \textit{ and every } i \in E.$$

Proof. Decompose $\mathcal{L}_{\text{train}}$ point-wise in X via the tower property:

$$\mathcal{L}_{\text{train}}(\theta) \;=\; \mathbb{E}_X\left[\sum_{i\in E} \mathbb{E}_{Y_i\mid X}[-\log p_\theta(Y_i \mid X)_i]\right],$$

where $Y_i := \tau(z_i, x_i^*)$ has conditional distribution $P_i^*(\cdot \mid X)$. *Per-position decoupling.* At fixed X, the inner sum has terms depending on θ only through the per-position output distributions $\{p_\theta(\cdot \mid X)_i\}_{i\in E}$, which the rich-family hypothesis allows θ to realize independently. Hence minimizing the sum jointly over θ is equivalent to minimizing each per-position term independently in its corresponding categorical. *Per-position propriety.* By strict propriety of cross-entropy Gneiting & Raftery (2007), for any reference distribution r on $\bar{\mathcal{V}}$ the expected score $\mathbb{E}_{y\sim r}[-\log p(y)]$ is uniquely minimized over the categorical simplex at $p = r$. Apply this with $r = P_i^*(\cdot \mid X)$ to each position i; the unique per-position minimizer is $p_\theta(\cdot \mid X)_i = P_i^*(\cdot \mid X)$. The almost-everywhere conclusion follows because the support of $\mathcal{D} \otimes q$ has full measure. □

Remark 2 (well-specification). If $\{p_\theta\}$ is not rich enough to represent P_i^*, the conclusion holds in the function-class limit: the minimizer is the closest representable distribution in expected KL.

Remark 3 (loss reweighting). Theorem 1 is stated for the unweighted CE objective of Eq. equation 6. Adding any constant or X-measurable per-position weighting $w_i(X) > 0$ does not change the minimizer, since the per-X minimization scales by $w_i(X)$. Label-dependent weighting tilts the minimizer but is not used in our experiments.

The general form of Theorem 1's minimizer simplifies sharply at each of the two state types. Define $\alpha_i(X) := \Pr(x_i^* = z_i \mid X)$, the conditional probability that the visible token equals the gold target.

Proposition 4 (Visible-token minimizer is a binary calibrated correctness belief)**.** At any X with $z_i \neq \texttt{MASK}$,

$$P_i^*(\cdot \mid X) \;=\; \alpha_i(X) \cdot \delta_{z_i} + (1 - \alpha_i(X)) \cdot \delta_{\texttt{MASK}}.$$

Proof. Conditional on $z_i \neq \texttt{MASK}$, the value z_i is determined by X. The remaining randomness in $\tau(z_i, x_i^*)$ given X is the indicator $\mathbf{1}\{z_i = x_i^*\}$, which equals 1 with probability $\alpha_i(X)$ and 0 otherwise. By Eq. equation 5, $\tau = z_i$ in the first case and $\tau = \texttt{MASK}$ in the second. □

Interpretation. At visible-token positions, the population minimizer encodes a calibrated belief about whether the visible token is the gold target. The inference rule *mask iff* $p_\theta(\texttt{MASK} \mid X) > p_\theta(z_i \mid X)$ is the Bayes-optimal binary decision based on this belief, since the two probabilities are the atoms of P_i^*.

Remark 5 (degenerate case). If $\mathbf{1}\{z_i = x_i^*\}$ is X-measurable, then $\alpha_i(X) \in \{0, 1\}$ a.s. and P_i^* collapses to $\delta_{\texttt{MASK}}$ or δ_{z_i}. The main results do not require this.

Proposition 6 (Calibrated gold belief at MASK positions)**.** At any X with $z_i = \texttt{MASK}$,

$$P_i^*(y \mid X) \;=\; \Pr_{(x^*, w) \sim (\mathcal{D} \otimes q) \mid X}[x_i^* = y].$$

Proof. When $z_i = \texttt{MASK}$, Eq. equation 5 gives $\tau(z_i, x_i^*) = x_i^*$ tautologically, so the conditional distribution of τ given X equals the conditional distribution of x_i^* given X. □

Interpretation. At MASK positions the population minimizer becomes a conditional language model over the gold token — the same target as Nie et al. (2025)'s two-state model. Sampling from $p_\theta(\cdot \mid \texttt{MASK}, \phi(H, c), c)$ recovers the standard mask-prediction sampling step.

Summary. RM trains a single per-position categorical that carries two calibrated beliefs: a binary $\{z_i, \texttt{MASK}\}$ correctness belief at visible-token positions, and a full-vocabulary gold-token belief at MASK positions. The inference rule reads each off the appropriate atoms — the re-mask check at visible positions and the candidate reveal at MASK positions in Eq. 1.

B.2 Plug-in Excess-Risk Theorem

We analyze the deterministic argmax plug-in policy

$$\hat{a}_\theta(X)_i \;:=\; \arg\max_{v \in \bar{\mathcal{V}}} p_\theta(v \mid X)_i,$$

with ties broken by a fixed model-independent rule (e.g., lexicographic order on $\bar{\mathcal{V}}$); the Bayes-optimal classifier $\hat{a}^*(X)_i := \arg\max_v P_i^*(v \mid X)$ is defined under the same tie-breaking rule. Bounds below hold for any such fixed choice. The per-position 0–1 revision risk against the oracle is

$$R(\theta) \;:=\; \mathbb{E}_{(\mathcal{D} \otimes q)}\left[\frac{1}{|E|} \sum_{i \in E} \mathbf{1}[\hat{a}_\theta(X)_i \neq \tau(z_i, x_i^*)]\right],$$

and the Bayes risk is $R^* := \inf_p R(p)$. By Propositions 4–6, R^* has irreducible contributions from *both* state types: $1 - \max_y P_i^*(y \mid X)$ at MASK positions, and $\min(\alpha_i(X), 1 - \alpha_i(X))$ at visible-token positions. Both vanish only at degenerate X where P_i^* is a delta.

Theorem 7 (Plug-in excess Bayes risk)**.** *Under the setup above,*

$$R(\theta) - R^* \;\leq\; 2 \cdot \mathbb{E}_X\left[\frac{1}{|E|} \sum_{i \in E} \mathrm{TV}(p_\theta(\cdot \mid X)_i,\, P_i^*(\cdot \mid X))\right].$$

Proof. Fix X and $i \in E$; drop the i subscript on P^*. By definition $Y := \tau(z_i, x_i^*)$ has conditional distribution $P^*(\cdot \mid X)$, so $\Pr_{Y \sim P^*}[\hat{a} \neq Y \mid X] = 1 - P^*(\hat{a} \mid X)$ for any deterministic classifier $\hat{a}$ measurable in X.

Case $\hat{a}_\theta = \hat{a}^*$. Per-$X$ excess risk is zero; the bound is trivial.

Case $\hat{a}_\theta \neq \hat{a}^*$. Per-$X$ excess risk equals $P^*(\hat{a}^* \mid X) - P^*(\hat{a}_\theta \mid X)$. Decompose:

$$P^*(\hat{a}^*) - P^*(\hat{a}_\theta) = \underbrace{[P^*(\hat{a}^*) - p_\theta(\hat{a}^*)]}_{\text{(A)}} + \underbrace{[p_\theta(\hat{a}^*) - p_\theta(\hat{a}_\theta)]}_{\text{(B)}} + \underbrace{[p_\theta(\hat{a}_\theta) - P^*(\hat{a}_\theta)]}_{\text{(C)}}.$$

(B) ≤ 0 since $\hat{a}_\theta$ maximizes p_θ (under the fixed model-independent tie-break stated above). Bound each of (A) and (C) by absolute value: (A) $\leq |p_\theta(\hat{a}^*) - P^*(\hat{a}^*)|$ and (C) $\leq |p_\theta(\hat{a}_\theta) - P^*(\hat{a}_\theta)|$. Since $\hat{a}^* \neq \hat{a}_\theta$, these are two distinct nonnegative terms in the sum $\sum_v |p_\theta(v) - P^*(v)|$, hence (A) + (C) $\leq \sum_v |p_\theta(v) - P^*(v)| = 2 \cdot \mathrm{TV}(p_\theta, P^*)$. Take expectation over X and average over i. □

Remark 8 (on the factor of 2). This is the standard Devroye–Györfi–Lugosi excess-risk bound (Devroye et al., 2013, Theorem 2.2). The constant 2 is tight: with $P^* = \delta_1$ and $p_\theta = (1/2 - \epsilon, 1/2 + \epsilon)$, excess risk equals 1 while TV $\to 1/2$.

Combining Theorems 1 and 7. At the population minimum $p_\theta = P^*$, so $R(\theta^*) = R^*$: the CE-loss minimizer is Bayes-optimal under 0–1 revision risk. The Bayes risk R^* is non-zero by construction (Propositions 4–6); driving CE loss further does not — and should not — reduce R^*.

B.3 History Reference Information Bound

History Reference augments the conditioning: the model receives not just (z, c) but $(z, \phi(H, c), c)$. For a random variable (or tuple) W, write $\sigma(W)$ for the σ-algebra generated by W; a decision rule $\hat{a}$ is $\sigma(W)$-measurable exactly when $\hat{a}$ can be written as a deterministic function of W.

Proposition 9 (History monotonicity)**.** Let $\sigma_{\text{base}} := \sigma(z, c)$ and $\sigma_{\text{HR}} := \sigma(z, H, c)$. Then

$$\inf_{\hat{a}\ \sigma_{\text{HR}}\text{-meas.}} \mathbb{E}[\mathbf{1}[\hat{a} \neq \tau]] \ \leq \ \inf_{\hat{a}\ \sigma_{\text{base}}\text{-meas.}} \mathbb{E}[\mathbf{1}[\hat{a} \neq \tau]].$$

Proof. $\sigma_{\text{base}} \subseteq \sigma_{\text{HR}}$ since (z, c) is a sub-tuple of (z, H, c). Hence every σ_{base}-measurable estimator is also σ_{HR}-measurable, so the infimum over a larger family cannot exceed the infimum over a smaller one. □

The same argument applies to any history feature ϕ used by the model: $\sigma(z, \phi(H, c), c) \subseteq \sigma(z, H, c)$, so ϕ achieves a Bayes risk between the no-history baseline and the full trajectory. Whether the HR accumulated embedding strictly decreases Bayes risk relative to the no-history baseline is an empirical question answered in Section 3.

Information hierarchy. The data-processing argument gives a three-level hierarchy — current state only, current state plus the accumulated embedding, current state plus the full trajectory:

$$\sigma(z, c) \ \subseteq \ \sigma(z, \phi(H, c), c) \ \subseteq \ \sigma(z, H, c).$$

The history feature $\phi(H, c)$ is a compressed history signal; distinct trajectories generally produce distinct $\phi(H, c)$, but the mapping is many-to-one.

B.4 Top-k Variant: Distribution-Shift TV Bound

Theorem 1 assumes a fixed corruption ν. In practice we use a non-uniform θ-independent corruption $\nu^{(\text{prac})}$: a frozen pre-trained checkpoint's top-k distribution for text generation, a fixed wrong-digit distribution for Sudoku, and the source-image VQ tokens for image editing. In the image-edit case the VQ token is taken as w_i only when $w_i \neq x_i^*$; positions where the source already matches the target fall under the keep branch, keeping $\nu^{(\text{prac})}_{\text{wrong}}$ supported on $\mathcal{V} \setminus \{x_i^*\}$. The loss gap between ν_{unif} and $\nu^{(\text{prac})}$ decomposes in three layers.

Assumption 10 (Bounded log-likelihood)**.** Per-token log-probabilities are clipped: $-\log p_\theta(v \mid X)_i \leq B := -\log \varepsilon$ for every token v, position i, and input X. We use $\varepsilon = 10^{-8}$, giving $B \approx 18$.

Layer 1 (joint TV bound). For any test function f with $|f| \le B \cdot |E|$ and any two probability measures P, Q on the joint sample space,

$$\left|\mathbb{E}_P f - \mathbb{E}_Q f\right| \;\le\; 2B \cdot |E| \cdot \mathrm{TV}(P, Q).$$

Applied to $f = \sum_{i \in E} -\log p_\theta(\tau(z_i, x_i^*) \mid X)_i$, with $q^{(\mathrm{unif})}$ and $q^{(\mathrm{prac})}$ the joint proposals under ν_{unif} and $\nu^{(\mathrm{prac})}$ respectively:

$$\left|\mathcal{L}_{\mathrm{train}}^{(\mathrm{unif})}(\theta) - \mathcal{L}_{\mathrm{train}}^{(\mathrm{prac})}(\theta)\right| \;\le\; 2B \cdot |E| \cdot \mathbb{E}_{(x^*,c)\sim\mathcal{D}}\Big[\mathrm{TV}\big(q^{(\mathrm{unif})}(\cdot \mid x^*, c),\; q^{(\mathrm{prac})}(\cdot \mid x^*, c)\big)\Big].$$

Layer 2 (factorize the joint TV). The two proposals share the rule prior π_ρ, the boundary law $\mathcal{B}$, and the timestep distribution; they differ only in the conditional corruption at $\rho = \mathrm{wrong}$ positions. Define the rule-marginalized corruption $\tilde{\nu}_i(\cdot \mid x_i^*) := \pi_{\mathrm{wrong}} \cdot \nu_{\mathrm{wrong}}(\cdot \mid x_i^*) + \pi_{\mathrm{mask}} \cdot \delta_{\texttt{MASK}}$. The proposals agree on the π_{mask} branch, so by the mixture-TV identity $\mathrm{TV}(\alpha P_1 + (1-\alpha)P_2,\, \alpha Q_1 + (1-\alpha)P_2) = \alpha\,\mathrm{TV}(P_1, Q_1)$, the per-position TV between rule-marginalized corruptions equals $\pi_{\mathrm{wrong}} \cdot \mathrm{TV}(\nu_{\mathrm{wrong},i}^{(\mathrm{unif})}, \nu_{\mathrm{wrong},i}^{(\mathrm{prac})})$. Combining with the standard product-distribution TV inequality (independence across positions),

$$\mathrm{TV}\big(q^{(\mathrm{unif})},\, q^{(\mathrm{prac})}\big) \;\le\; \pi_{\mathrm{wrong}} \cdot \sum_{i \in E} \mathrm{TV}\big(\nu_{\mathrm{wrong},i}^{(\mathrm{unif})},\; \nu_{\mathrm{wrong},i}^{(\mathrm{prac})}\big).$$

Layer 3 (per-position bound, ready for analysis). Combining Layers 1 and 2:

$$\left|\mathcal{L}_{\mathrm{train}}^{(\mathrm{unif})}(\theta) - \mathcal{L}_{\mathrm{train}}^{(\mathrm{prac})}(\theta)\right| \;\le\; 2B \cdot |E| \cdot \pi_{\mathrm{wrong}} \cdot \mathbb{E}_{(x^*,c)\sim\mathcal{D}}\left[\sum_{i \in E} \mathrm{TV}\left(\nu_{\mathrm{wrong},i}^{(\mathrm{unif})},\; \nu_{\mathrm{wrong},i}^{(\mathrm{prac})}\right)\right]. \quad (8)$$

Only "wrong"-rule positions contribute; the mask-rule and non-edit branches contribute zero.

Implication. Theorem 1 still applies on the support of $q^{(\mathrm{prac})}$: the population minimizer recovers the conditional label distribution under $\nu^{(\mathrm{prac})}$ rather than under ν_{unif}. Since inference-time errors are not uniform, $\nu^{(\mathrm{prac})}$ is the more relevant covered region.

C Engineering Implementation

The relative-lag form of §2.2 admits a strict $O(1)$ per-step update of the accumulated embedding $a_i^{(t)}$ (also written acc_i).

Derivation of the recurrence. Starting from the closed form (Eq. equation 2)

$$a_i^{(t)} \;=\; \sum_{k=0}^{t} \gamma^{\,t-k}\, R_{k-t}\, e_i^{(k)}, \qquad e_i^{(k)} := \mathrm{wte}(\tilde{x}_i^{(k)}),$$

and the corresponding sum at step $t-1$,

$$a_i^{(t-1)} \;=\; \sum_{k=0}^{t-1} \gamma^{\,t-1-k}\, R_{k-(t-1)}\, e_i^{(k)},$$

left-multiplying $a_i^{(t-1)}$ by γR_{-1} and using the rotation composition rule $R_{-1} R_{k-(t-1)} = R_{k-t}$ gives

$$\gamma\, R_{-1}\, a_i^{(t-1)} \;=\; \sum_{k=0}^{t-1} \gamma^{\,t-k}\, R_{k-t}\, e_i^{(k)}.$$

Adding the current-step contribution $e_i^{(t)}$ (whose lag from the current step is zero, so $R_0 = I$) recovers $a_i^{(t)}$:

$$a_i^{(t)} \;=\; e_i^{(t)} \;+\; \gamma\, R_{-1}\, a_i^{(t-1)}.$$

This is the recurrence used in the implementation.

Memory layout. We maintain a single running tensor $a \in \mathbb{R}^{N \times d}$, where N is the sequence length and d the hidden width. The per-step update is one $N \times d$ block-diagonal rotation followed by an element-wise add; storing the full token-level trajectory would cost $O(T \times N \times d)$, while the recurrence collapses this to a single $O(N \times d)$ buffer.

Numerical stability. Each R_Δ is a block-diagonal orthogonal transform, so R_{-1} preserves norms: $\|R_{-1}\, a_i^{(t-1)}\| = \|a_i^{(t-1)}\|$. Combined with the decay factor $\gamma \in (0, 1]$, the running sum's norm grows at most linearly in t.

Contrast with an absolute-step formulation. A natural alternative assigns each historical token an absolute-step rotation in a fixed (step-zero) reference frame, $\sum_{k=0}^{t} \gamma^{t-k} R_k\, e_i^{(k)}$. This admits the same per-step recurrence but anchors the trajectory at the start of denoising rather than the current step. The relative-lag form of Eq. equation 2 instead anchors the phase at the model's most recent input, and is what we use in all reported experiments.

Loss reweighting. Training launchers may attach per-position weights w_i to the CE term. The math launcher uses constant weights; the Sudoku launcher emphasizes edit positions via an X-measurable edit-vs-non-edit weight. Both fall under the loss-reweighting remark following Theorem 1, so the population minimizer is unchanged.

Bounded-loss floor. We clip output probabilities at $\varepsilon = 10^{-8}$, giving $B = -\log \varepsilon \approx 18$ in Assumption 10.

D Training Algorithm

We give the complete per-sample procedure summarized in Section 2.3. Let $T = 6$ denote the trajectory length used in our experiments.

1. **Sample data.** Draw $(x^*, c) \sim \mathcal{D}$ (target sequence and task condition).
2. **Sample per-position rule.** For each $i \in E$, draw $\rho_i \sim \pi_\rho$ over $\{\text{wrong}, \text{mask}\}$ independently. Non-edit positions $i \notin E$ are deterministically labeled "keep"; their state is x_i^* at every t.
3. **Sample source token (wrong-rule positions only).** For each $i \in E$ with $\rho_i = \text{wrong}$, sample $w_i \sim \nu_{\text{wrong}}(\cdot \mid x_i^*)$ on $\mathcal{V} \setminus \{x_i^*\}$. The rigorous baseline of Appendix B.1 uses uniform sampling; the practical variant (Appendix B.4) uses a task-specific non-uniform $\nu^{(\text{prac})}$. Mask-rule positions ($\rho_i = \text{mask}$) need no source-token sample; they enter the trajectory directly at the `MASK` state.
4. **Sample boundaries.** For each $\rho_i = \text{wrong}$ position, draw $\beta_i \sim \text{Uniform}\{1, \ldots, T-1\}$, then $\mu_i \mid \beta_i \sim \text{Uniform}\{\beta_i + 1, \ldots, T\}$. For each $\rho_i = \text{mask}$ position, draw $\mu_i \sim \text{Uniform}\{1, \ldots, T\}$.
5. **Construct the trajectory $\tilde{x}^{(0:T)}$.** Per-position deterministic from the boundaries:
   - "wrong" rule: $\tilde{x}_i^{(t)} = w_i$ for $0 \le t < \beta_i$, `MASK` for $\beta_i \le t < \mu_i$, x_i^* for $\mu_i \le t \le T$.
   - "mask" rule: $\tilde{x}_i^{(t)} = \texttt{MASK}$ for $0 \le t < \mu_i$, x_i^* for $\mu_i \le t \le T$.
   - "keep" / non-edit: $\tilde{x}_i^{(t)} = x_i^*$ for all t.

   The wrong-rule ordering $w_i \to \texttt{MASK} \to x_i^*$ and the mask-rule ordering $\texttt{MASK} \to x_i^*$ both hold by construction; no post-processing is required.
6. **Sample current step.** Draw $t \sim \text{Uniform}\{0, \ldots, T-1\}$ and set $z = \tilde{x}^{(t)}$, $H = \tilde{x}^{(<t)}$.
7. **Construct labels.** For each $i \in E$, the per-position label is the oracle action $y_i = \tau(z_i, x_i^*)$ defined in Eq. equation 5: `MASK` if z_i is a visible non-target token, x_i^* if $z_i = \texttt{MASK}$ (reveal target), x_i^* if $z_i = x_i^*$ (preserve target).
8. **CFG dropout (image editing only).** Draw text_drop $\sim$ Bernoulli(0.10) and image_drop $\sim$ Bernoulli(0.10) independently. If text_drop, replace the user instruction

with its unconditional prefix. If image_drop, mark image-content positions to have their accumulated embedding overwritten by $\mathrm{wte}(\texttt{MASK})$ after RMSNorm in step 9. The generic algorithm (text generation, Sudoku) skips this step.

9. **Compute the accumulated embedding.** Accumulate via Eq. equation 2,

$$\mathrm{acc}_i \;=\; \textstyle\sum_{k=0}^{t} \gamma^{\,t-k}\, R_{k-t}\, \mathrm{wte}(\tilde{x}_i^{(k)}),$$

or by the recurrence $\mathrm{acc}_i^{(s)} = \mathrm{wte}(\tilde{x}_i^{(s)}) + \gamma R_{-1}\mathrm{acc}_i^{(s-1)}$ with $\mathrm{acc}_i^{(0)} = \mathrm{wte}(\tilde{x}_i^{(0)})$. Apply RMSNorm. For image editing, if image_drop from step 8 fires, overwrite acc at image-content positions with $\mathrm{wte}(\texttt{MASK})$. The model's input is (acc, c); the pipeline keeps z aside for label construction and for the analyst-side conditioning of Theorem 1.

10. **Compute the loss.** Cross-entropy of Eq. equation 4 against the per-position labels y_i from step 7.

Practical top-k proposal. For text generation, $\nu_{\mathrm{wrong}}^{(\mathrm{prac})}(\cdot \mid x_i^*)$ in step 3 is the truncated softmax of a frozen checkpoint θ_0's top-k output (default $k = 50$), restricted to $\mathcal{V} \setminus \{x_i^*\}$; sharper conditionals admit smaller k. We probe θ_0 once per training sample on x^* and cache the per-position top-k index sets, drawing each position's wrong token from its own conditional softmax. Since θ_0 is fixed, the proposal is independent of the live parameter θ. Appendix B.4 bounds the objective shift relative to the uniform baseline.

E Results gallery on image editing

Fig. 5 provides additional image editing examples spanning object replacement, attribute modification, object insertion, and localized scene editing. Each row shows the source image, the predicted mask, our edit, and Lumina / Lumina-SFT baselines. The brighter regions in pixel-wise difference heat maps indicate larger pixel change relative to the source. Across these examples, our edits concentrate within the predicted-mask region; the baselines exhibit pixel changes outside that region (visible as off-mask hot regions in the heat maps). For readability, figures use abbreviated editing instructions; Tab. 4 maps these prompts to the full prompts.

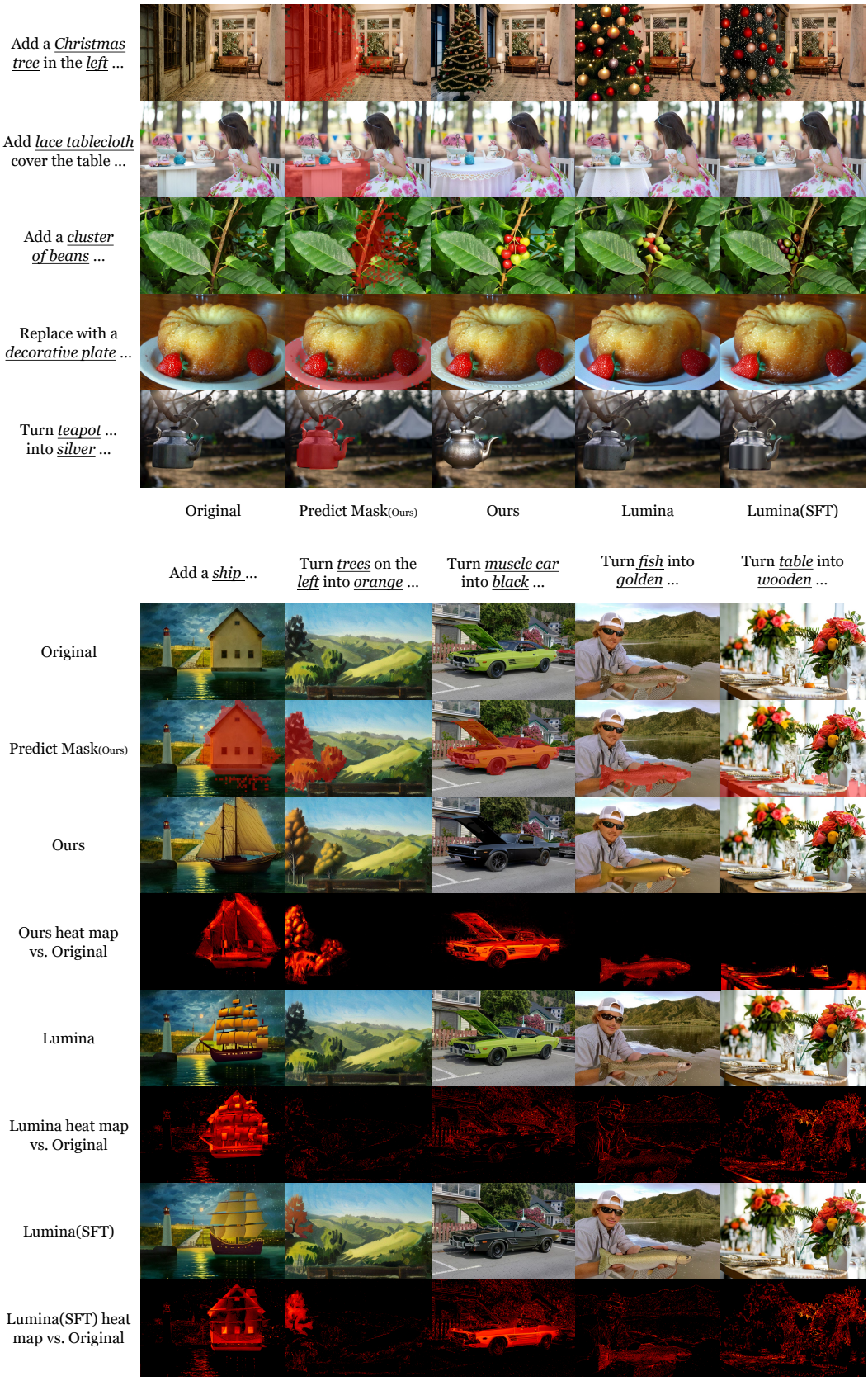


Figure 5: **Additional qualitative results on image editing.**

Table 4: Correspondence between the abbreviated editing instructions shown in the paper and the full editing instructions used in our experiments.

Abbreviated Editing Instruction	**Full Editing Instruction**
Turn time into black ...	Turn time positioned in the right area into black.
Add a person in wedding gown ...	Add a person in an elaborate white wedding gown with lace, veil, and tiara in the upper-middle of the image, occupying roughly half the height and most of the width.
Add a cabin made from wood ...	Add a modern, minimalist cabin made from light-colored wood with a single door glowing orange, located in the upper-right portion and taking up nearly a quarter of the image.
Turn cookies into chocolate ...	Turn cake positioned in the central area into chocolate.
Turn trees on the left into orange ...	Turn trees positioned on the left side into autumnal orange.
Add bell-shaped blooms	Add delicate, white bell-shaped blooms with slender green stems in the middle-lower portion of the image, occupying a large area spanning most of the width.
Add cream piled on the top ...	Add whipped cream piled on top of the French toast stack near the center-upper area of the image, covering about one sixth of the total image area.
Add creamy ... filling a spoon ...	Add a white, creamy substance, likely coconut oil, filling a spoon and partially spilling onto a jar in the center-right-middle of the image, occupying about one-fifth of the area.
Turn the tree into snowy ...	Turn tree positioned in the right area into snowy.
Add a Christmas tree in the left ...	Add a beautifully decorated Christmas tree covered with red, gold, and silver ornaments and sparkling white lights in the left foreground, occupying most of the left and lower-central area.
Add lace tablecloth cover the table ...	Add a small, round table covered with a lace tablecloth in the lower-left-middle of the image, occupying about one fourth of the area.
Add a cluster of beans ...	Add a cluster of ripening beans in various colors from green to red in the center-right of the image, occupying approximately one fourth of the area.
Replace with a decorative plate ...	Add a decorative white plate on a wooden table in the bottom-middle of the image, occupying about one third of the area.
Turn teapot into silver ...	Turn teapot positioned in the upper-central area into silver.
Add a ship ...	Add a large, fully-rigged sailing ship with golden sails and intricate rigging near the center-upper-right, occupying about one third of the image area.
Turn muscle car into black ...	Turn muscle car positioned in the central area into black.
Turn fish into golden ...	Turn fish positioned towards the lower central area into golden.
Turn table into wooden ...	Turn table positioned in the lower area into wooden.

F MORE RESULTS ON TEXT REASONING TASK

Minerva MATH subject breakdown. We further evaluate RM on the Minerva MATH benchmark (Lewkowycz et al., 2022) and report the performance across different subject categories in Tab. 5. Compared with Vanilla SFT, RM improves the aggregate accuracy from 22.62% to 24.10%. The improvements are observed across most subject categories, with particularly larger gains on Number Theory and Precalculus.

Table 5: Performance comparison on Minerva MATH task across different subject categories

Subject	Algebra	Counting & Prob.	Geometry	Intermediate Alg.	Num Theory	Prealgebra	Precalc	Aggregate
Vanilla SFT(%)	28.90	17.72	20.67	13.07	17.41	36.28	14.10	22.62
Ours RM (%)	**29.49**	**18.35**	**20.67**	**14.40**	**21.67**	**38.00**	**16.67**	**24.10**
Δ(%)	$\uparrow$ 0.59	$\uparrow$ 0.63	0	$\uparrow$ 1.33	$\uparrow$ 4.26	$\uparrow$ 1.72	$\uparrow$ 2.57	$\uparrow$ 1.48

Accuracy–computation trade-off. We further study how RM uses additional inference computation by varying the number of tokens revealed per denoising step. This auxiliary sweep uses block size 256; within each pair, RM-off and RM-on runs use the same backbone, checkpoint, reveal rate, and decoding settings. Tab. 6 shows that RM improves MATH-500 accuracy by 2.2–2.6 percentage points across all four evaluated inference budgets while requiring only a small number of additional forward passes. As the reveal rate increases, both variants become faster but less accurate, while RM consistently recovers part of the accuracy loss through learned revision.

Table 6: Accuracy–computation trade-off on MATH-500. Each cell reports RM-off $\rightarrow$ RM-on.

Reveal/step	Accuracy (%)	Forwards/query	Latency (s)
1	20.2 $\rightarrow$ 22.4	256 $\rightarrow$ 265.6	5.728 $\rightarrow$ 5.934
2	20.0 $\rightarrow$ 22.6	128 $\rightarrow$ 132.1	2.883 $\rightarrow$ 2.970
4	19.4 $\rightarrow$ 21.6	64 $\rightarrow$ 69.9	1.454 $\rightarrow$ 1.587
8	16.4 $\rightarrow$ 18.8	32 $\rightarrow$ 35.1	0.741 $\rightarrow$ 0.812

Additional qualitative results on text generation. Fig. 6 shows additional text revisions from our experiments. Cases 1 and 4: the model re-masks a token that is logically inconsistent with its surrounding context and re-predicts it. Cases 2 and 5: a single corrected token cascades downstream. After the earlier position is revised, the model also re-masks dependent positions further to the right and re-predicts them to be consistent with the corrected context. Case 3: format-level correction, where re-masking targets tokens whose surface form violates the answer template.

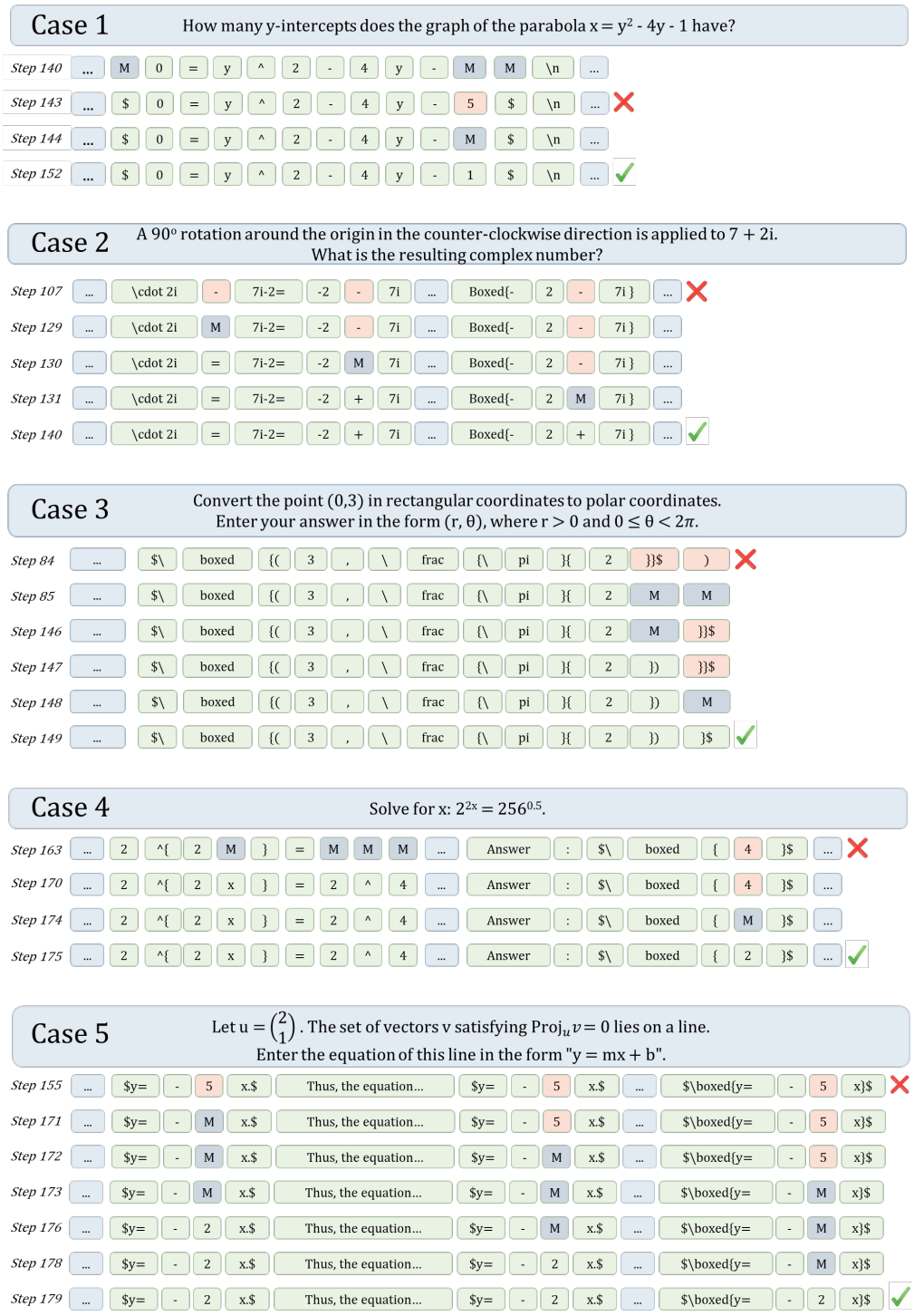


Figure 6: **Additional qualitative results on text generation.**

G EVALUATION DETAILS

Image Editing Evaluation. We evaluate image editing quality from three aspects: edit localization, background preservation, and overall editing quality. For localization, we report Edit *Precision*, which measures the percentage of changed pixels that fall inside the ground-truth target edit region,

while Edit *Coverage* measures the fraction of the target region that is modified. For background preservation, we use MAE-RGB (Willmott & Matsuura, 2005), PSNR (Huynh-Thu & Ghanbari, 2008), and SSIM (Wang et al., 2004) on the non-target regions between the edited and the input image. These metrics evaluate whether regions unrelated to the instruction remain unchanged. For overall editing quality, we use VQAScore (Lin et al., 2024) to evaluate instruction following and content preservation, together with a 29-participant user study described below for human preference.

User study details. We conducted a user study with 29 participants to evaluate the perceptual quality of image editing results. For the user study, we designed an online questionnaire based on the full prompts listed in Tab. 4 and their corresponding examples. Each question showed the original image, the editing instruction, and edited results from different methods in a randomized order. Participants were asked to independently rate each method according to how well the edited result followed the instructions while preserving unrelated regions of the image. The study only collected rating judgments and did not collect personally identifiable information.

Sudoku Evaluation. We evaluate four complementary metrics: Exact Accuracy, Valid Rate, Replay Mistake, and Conflict Cells. *Exact Accuracy* requires every cell in the final board to exactly match the ground-truth solution, while *Valid Rate* only requires the final board to satisfy Sudoku constraints. We include *Exact Accuracy* because our setting allows the model to re-mask and revise any position in the initial board. A low *Exact Accuracy* score would indicate that the model tends to discard key given clues and regenerate a generic valid solution. Taken together, these two metrics provide a more comprehensive evaluation of the correction capability of the RM method. *Replay Mistake* measures the fraction of re-masked erroneous cells that are later decoded back to their previous incorrect digit during the revision trajectory, and *Conflict Cells* reports the average number of Sudoku-rule-violating cells in each final board.