PURE: Post-hoc Unlocking and REfinement for Discrete Diffusion Decoding

Yangryeol Park[1], **Kunhui Lee**[1], **Hanback Choi**[1], **Cheoneum Park**[2], **Donghyeon Jeon**[3], **Inho Kang**[3], **Seung-Hoon Na**[1*]
[1]UNIST [2]HBNU [3]NAVER Corporation
{yangreal, dkghszkfhs, hanback, nash}@unist.ac.kr
parkce@hanbat.ac.kr
{donghyeon.jeon, once.ihkang}@navercorp.com

Abstract

Masked diffusion language models (MDLMs) enable efficient parallel decoding but are limited by a monotonic unmasking policy, where committed tokens cannot be revised. While remasking-based methods mitigate early errors, they mainly intervene during generation. In this work, we study post-hoc refinement of a completed draft and find that naive correction often fails because of contextual lock-in, a phenomenon in which local error patterns become self-reinforcing. To address this, we propose PURE (Post-hoc Unlocking and REfinement), a training-free inference algorithm for two-phase decoding. PURE profiles confidence dynamics during drafting to identify unstable regions via an instability score (Δ_i), then unlocks them through deterministic window masking and stochastic leftward relaxation. On reasoning benchmarks, PURE substantially improves accuracy when applied to LLaDA-8B-Instruct, including a gain of +12.9 points over the baseline on GSM8K. These gains require only a small refinement budget, yielding a favorable compute-quality trade-off for discrete diffusion decoding.

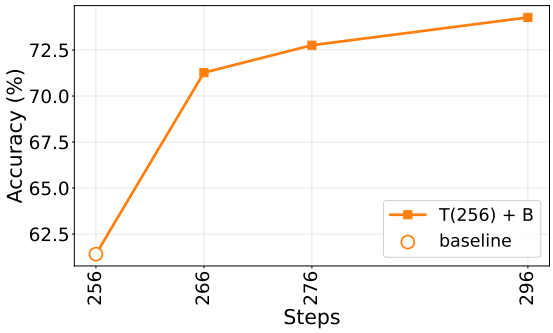

Figure 1: **Effect of refinement budget on GSM8K with LLaDA-8B-Instruct.** Starting from the standard k=1 draft (T=256), accuracy improves steadily as the post-hoc refinement budget B increases. Notably, even a small refinement budget (B=10) yields a substantial gain over the draft baseline. The x-axis shows the total number of denoising steps T+B; the baseline corresponds to B=0.

1 Introduction

Diffusion models have been highly successful in continuous domains such as image and video generation (Sohl-Dickstein et al., 2015; Song and Ermon, 2019; Ho et al., 2020; Rombach et al., 2022; Ho et al., 2022). Motivated by their multi-step generation procedure, recent work has adapted diffusion to language via masked diffusion language models (MDLMs), with rapidly improving performance (Austin et al., 2021; Hoogeboom et al., 2021; Sahoo et al., 2024; Nie et al., 2025).

Starting from a fully masked sequence and iteratively denoising it over multiple steps enables (i) bidirectional context usage (Ghazvininejad et al., 2019) and (ii) parallel decoding across positions (Yang et al., 2025).

However, despite the iterative nature of denoising, standard masked discrete diffusion decoding is often *irreversible*: it follows a monotonic unmasking policy where once a token is committed, it is not revised in later steps, even if it is found to be contextually inconsistent as more of the sequence is revealed. (Shi et al., 2024; Wang et al., 2025a) Consequently, early mistakes made under limited context (i.e., premature commitment) can become permanently embedded, and the risk grows as parallelism increases (i.e., when more tokens are committed per step) (Lavenant and Zanella, 2025), contributing to a quality–speed trade-off and limiting sample quality.

To alleviate this limitation, a growing line of work has proposed remasking-based sampling strategies that relax token commitment by allowing the model to revisit uncertain positions and refine them over multiple denoising steps (Wang et al., 2025a; Mounier and Idehpour, 2025b; Dong et al.,

*Corresponding author

Findings of the Association for Computational Linguistics: ACL 2026, pages 24653–24667

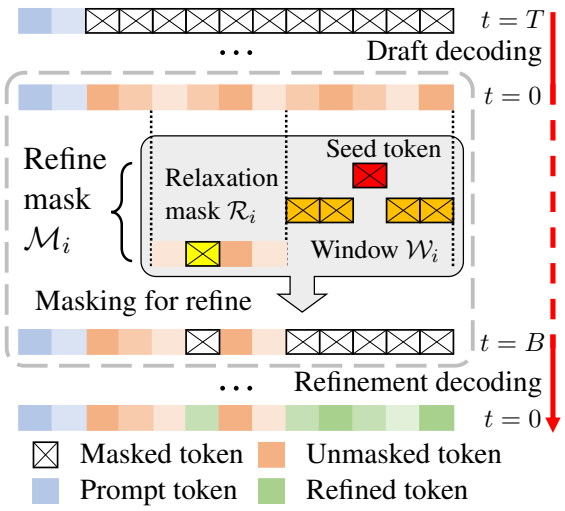

Figure 2: **Overview of post-hoc refinement (PURE).** After standard masked diffusion decoding ($t = T \rightarrow 0$), we select a seed token i and construct a refine mask $\mathcal{M}_i$ by remasking the seed token itself, its local window W_i, and a relaxation mask R_i over preceding context tokens. We then run an additional B-step refinement decoding ($t{=}B \rightarrow 0$) to obtain refined tokens (green).

2025; Hong et al., 2025).

Most existing approaches in this line primarily intervene during generation, i.e., they modify the sampling trajectory while the global context is still being formed and may remain partially inconsistent. Complementary to this, we ask a different question: *can we refine a completed draft after generation, without modifying the base model?* That is, we study post-hoc refinement for MDLM decoding: first generate a candidate sequence as a draft, then selectively remask targeted regions and refine them to correct errors. Figure 1 shows that even a small post-hoc refinement budget already yields a substantial gain over the draft baseline.

However, post-hoc refinement is not solved by simply remasking the erroneous token in isolation and re-sampling it. Through empirical analysis, we find that even with oracle-identified error locations, *pointwise correction* often reproduces the same mistake. We attribute this to a phenomenon we call **contextual lock-in**: once an incorrect token is generated, nearby tokens may co-adapt around it and form a locally self-consistent neighborhood that strongly biases regeneration toward the same error.

Moreover, we observe that errors exhibit **spatial locality**, tending to cluster within short spans rather than appearing uniformly at random. This suggests that effective post-hoc correction requires more than choosing which token to revisit; it also requires deciding how to disrupt the local neighborhood that sustains the error.

To address these challenges, we propose **PURE** (**P**ost-hoc **U**nlocking and **RE**finement), a training-free post-hoc sampling strategy designed to unlock locked-in error states with minimal modification to the standard decoding procedure. Figure 2 provides an overview of PURE and its two-phase procedure: draft decoding followed by targeted remasking and refinement. PURE appends a short, budget-efficient refinement chain to standard decoding:

$$x^T \xrightarrow{drafting} x^0 \xRightarrow{unlock} \hat{x}^B \xrightarrow{refining} \hat{x}^0.$$

We decode x^0, remask $\mathcal{M}$, and run B refinement steps to obtain $\hat{x}^0$.

PURE combines two mechanisms motivated by our analysis. First, **deterministic window masking** jointly remasks an error token and its neighbors, directly disrupting the local evidence that justifies the error. Second, **stochastic leftward relaxation** randomly remasks a small subset of preceding context tokens, relaxing prefix-induced constraints and opening alternative regeneration pathways. Together, these mechanisms effectively "unlock" erroneous local configurations and allow the model to regenerate a correct solution.

Contributions. Our contributions are summarized as follows:

- We identify *contextual lock-in* and *spatial locality* as key mechanisms that hinder post-hoc self-correction in masked discrete diffusion decoding, supported by oracle-style validity analyses.
- We introduce **PURE**, a novel training-free post-hoc inference algorithm that combines deterministic window masking and stochastic leftward relaxation to efficiently correct sampling errors.
- We demonstrate that PURE achieves consistent accuracy gains over standard baselines on reasoning benchmarks while remaining computationally efficient in practice.

2 Motivation and Preliminary Analysis

Before presenting our methodology, we investigate the fundamental dynamics of error correction in dis-

crete diffusion. We start by examining the feasibility of the most basic approach – **Pointwise Oracle Correction** – where we assume perfect knowledge of error positions and remask only those specific tokens.

2.1 The Phenomenon of Contextual Lock-in

We begin with a simple question: if we oracle-identify the erroneous positions and remask only those tokens, will the model correct them? Counter-intuitively, even with oracle-level error identification, the model often regenerates the same incorrect tokens. This indicates that the failure is not merely due to misidentifying error locations, but can persist even when the model is explicitly given a chance to revise them. We hypothesize that such errors become contextually self-reinforced: once an incorrect token is generated, neighboring tokens produced around it may form a locally coherent pattern that biases the model toward reselecting the same token upon re-decoding. We refer to this phenomenon as **contextual lock-in**, where a locally coherent but globally incorrect configuration becomes difficult to escape via pointwise resampling. In Section 2.3, we quantify how breaking this lock-in benefits from perturbing the local neighborhood and selectively relaxing constraints from the preceding context.

2.2 Oracle study setup

We conduct the oracle study on GSM1K, focusing on the 160 samples that are incorrect under the base LLaDA-8B-Instruct (Nie et al., 2025) with 256 generation tokens and 256 sampling steps. For each such sample, we define the oracle error-token set as a manually identified answer-flipping set: a small subset of generated token positions such that remasking and re-sampling only those positions can change the final answer from incorrect to correct. We fix the refinement budget to 10 steps with a uniform unmasking schedule. We found that increasing the number of steps (e.g., to 20 or 40 steps) yields only marginal differences and does not change the qualitative trends.

2.3 Empirical Justification: An Oracle Study

To quantify how to break this lock-in, we conduct a controlled experiment under an oracle setting. We measured the correction accuracy while varying two disruption strategies: the **window radius** (w) and the **relaxation rate** (ρ). Here, windowing remasks the error token together with its w left and w right neighbors, and ρ denotes the fraction of tokens randomly remasked in a fixed preceding-context region.

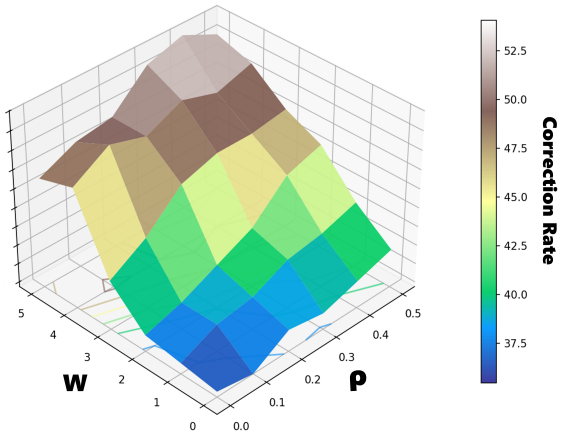

Figure 3: **Oracle correction rate** under varying **window radius** (w) and **relaxation rate** (ρ). The origin ($w = 0, \rho = 0$) corresponds to *Pointwise Oracle Correction* and yields the lowest correction rate, consistent with contextual lock-in. Increasing w and ρ both improve correction rate, and their combination achieves the highest performance, indicating complementary effects.

Figure 3 visualizes the results, providing empirical evidence for our design choices:

1. **Failure of Pointwise Oracle Correction ($w = 0, \rho = 0$):** The baseline strategy of masking only the error tokens yields the lowest performance. This provides evidence that without modifying the surrounding context, pointwise resampling often fails to escape the lock-in configuration.

2. **Role of Windowing ($w > 0$):** Increasing the window size w consistently improves correction accuracy. This suggests that the immediate local context plays a major role in sustaining the lock-in. By remasking neighbors along with the error token, we force the model to reconstruct local coherence, thereby relaxing rigid contextual constraints that may have favored the original error.

3. **Role of Stochastic Leftward Relaxation ($\rho > 0$):** Independently, increasing the relaxation rate ρ also boosts correction accuracy. This suggests that the lock-in is additionally reinforced by constraints inherited from the preceding context. Randomly remasking a fraction of tokens in the preceding context

relaxes these constraints, enabling alternative regeneration trajectories.

4. **Synergistic Effect:** The highest correction rate is achieved when both strategies are combined, indicating complementary effects. Together, windowing perturbs the immediate local neighborhood, while stochastic leftward relaxation relaxes constraints from the preceding context, improving the model's ability to revise locked-in errors.

Building upon these observations, we formally introduce PURE in Section 3.

2.4 An Information-Theoretic Perspective on Contextual Lock-in

The key intuition behind PURE is simple: to escape contextual lock-in, the model must become less certain about the token at the error position, so that re-sampling can explore alternative corrections rather than returning to the same locally supported token.

For an error position i, let $W_i(w) = \{j \mid 0 < |j - i| \leq w\}$ be the radius-w local window around i, and let $C_i = V \setminus (\{i\} \cup W_i(w))$ denote the remaining context. We define the uncertainty increase induced by removing the local window as

$$\begin{aligned} \Delta H_i(w) &:= H(X_i \mid X_{C_i}) - H(X_i \mid X_{C_i}, X_{W_i}) \\ &= I(X_i; X_{W_i} \mid X_{C_i}), \end{aligned} \tag{1}$$

where $I(X_i; X_{W_i} \mid X_{C_i})$ is the conditional mutual information, i.e., the additional information that the local window W_i provides about X_i beyond the remaining context C_i.

Eq. (1) admits a direct interpretation: if $\Delta H_i(w)$ is large, then the token at position i is strongly supported by its local neighborhood. Equivalently, removing the local window causes a large increase in uncertainty at the error position. In this sense, a large entropy gain is exactly the information-theoretic signature of strong local coupling.

This provides a formal view of contextual lock-in. When the current token at the error position is strongly anchored by its local neighborhood, masking only that token leaves most of the supporting local evidence intact, making re-sampling likely to return to the same locally self-consistent error. By contrast, window masking removes this support and creates room for alternative corrections. Stochastic leftward relaxation plays a complementary role by further weakening prefix-side evidence that may also support the same wrong local mode.

This view also yields a concrete prediction: structured window and leftward masking should increase entropy at error positions more than equal-budget random masking. We verify this pattern on incorrect GSM1K samples: the entropy increase grows with the window size w and the leftward masking rate ρ, whereas count-matched random controls remain close to zero. Detailed probe definitions and full results are provided in Appendix B.

3 Methodology

We propose Post-hoc Unlocking and REfinement for discrete diffusion decoding (PURE), a two-phase decoding procedure that improves discrete diffusion language models via targeted post-hoc re-decoding. The key idea is to treat the initial decoding result as a draft, then selectively remask a small set of suspicious tokens (and their local context) and refine them with a short additional denoising chain, rather than re-running the full trajectory. PURE consists of two sequential phases:

- **Phase 1: Drafting + Instability Profiling.** We run a standard discrete diffusion decoding process to generate an initial draft output. During this trajectory, we monitor token-level instability by tracking how each committed token's predictive confidence changes as the surrounding context evolves. Tokens that exhibit large confidence drops are flagged as refinement seeds, indicating regions that are likely to benefit from post-hoc revision.

- **Phase 2: Unlocking + Targeted Refinement.** Starting from the completed draft, we build a sparse refinement set $\mathcal{M}$ by expanding each seed to a local window and a small stochastic subset of preceding tokens. We remask only positions in $\mathcal{M}$ and run a short B-step targeted refinement. Over this roughly uniform schedule, each step unmasks about $|\mathcal{M}|/B$ positions in $\mathcal{M}$, while all positions outside $\mathcal{M}$ remain fixed to the draft.

3.1 Phase 1: Drafting + Instability Profiling

We first run the standard discrete diffusion decoder for T steps to obtain a draft x^0. At each step, the decoder outputs token-level predictive distributions for every sequence position conditioned on the current partially masked state x^t. Therefore, once a position i is first unmasked at step τ_i, we can continue

tracking the probability assigned to that committed token under later states as more context becomes available.

For each position i, let τ_i denote the step at which position i is first unmasked. We then measure how much the predictive probability of the committed token $x_i^{\tau_i}$ decreases under later states x^t, i.e., using $c_i^t = p_\theta(x_i^{\tau_i} \mid x^t)$ for $0 \le t \le \tau_i$. We define the instability score as the maximum post-commit probability drop:

$$\Delta_i = \max_{0 \le t \le \tau_i} \left(c_i^{\tau_i} - c_i^t \right). \tag{2}$$

A large Δ_i indicates that a token that initially looked confident becomes much less probable after more context is revealed.

We then select the top-q fraction of positions with the largest instability scores as refinement seeds:

$$\mathcal{S} = \text{Top-}q(\{\Delta_i\}_{i=1}^N). \tag{3}$$

3.2 Phase 2: Unlocking + Targeted Refinement

Phase 2 performs a short targeted refinement over a sparse refinement set $\mathcal{M}$. For each seed $i \in S$, we construct an unlock region by remasking the seed token itself, its radius-w local window $W_i(w) = \{j \mid 0 < |j - i| \le w\}$, and a stochastic subset of preceding tokens $R_i = \texttt{Ber}(\{j \mid i - L \le j < i - w\}, \rho)$. The refinement set is the union over all seeds:

$$\mathcal{M} = \bigcup_{i \in S} \left(\{i\} \cup W_i \cup R_i \right). \tag{4}$$

We initialize refinement by remasking only positions in $\mathcal{M}$:

$$\hat{x}^B = \texttt{Mask}(x^0, \mathcal{M}). \tag{5}$$

All positions outside $\mathcal{M}$ remain fixed to the draft throughout refinement.

We then run an additional B-step denoising chain over this partially masked state. The refinement schedule is defined over the still-masked positions in $\mathcal{M}$ rather than over the full sequence. At each step, we unmask roughly $|\mathcal{M}|/B$ currently masked positions in $\mathcal{M}$, prioritizing those with the highest model confidence under the current state, while leaving the remaining positions masked until later steps. Thus, Phase 2 is a targeted refinement over $\mathcal{M}$ rather than a second full decoding pass. For example, when $|\mathcal{M}| = 100$ and $B = 10$, PURE finalizes about 10 positions in $\mathcal{M}$ per step, with everything outside $\mathcal{M}$ unchanged.

Algorithm 1 PURE: Post-hoc Unlocking and REfinement

Require: Denoiser $p_\theta(\cdot \mid \cdot)$, draft budget T, refinement budget B, seed ratio q, window radius w, relaxation length L, relaxation rate ρ
Ensure: Refined output $\hat{x}^0$

Phase 1: Drafting + Instability Profiling
Run standard decoding to obtain trajectory $\{x^t\}_{t=T}^0$ and draft x^0
Compute instability scores $\{\Delta_i\}$ from $\{x^t\}$ and select seeds $\mathcal{S} \leftarrow \text{Top-}q(\{\Delta_i\})$
Phase 2: Unlocking
$\mathcal{M} \leftarrow \emptyset$
for $i \in \mathcal{S}$ **do**
    $W_i \leftarrow \{j \mid 0 < |j - i| \le w\}$
    $R_i \leftarrow \text{Ber}(\{j \mid i - L \le j < i - w\}, \rho)$
    $\mathcal{M} \leftarrow \mathcal{M} \cup (\{i\} \cup W_i \cup R_i)$
end for
Phase 2: Targeted Refinement
$\hat{x}^B \leftarrow \text{Mask}(x^0, \mathcal{M})$
$m \leftarrow \lceil |\mathcal{M}|/B \rceil$
for $t = B, B-1, \ldots, 1$ **do**
    Unmask up to m highest-confidence masked positions in $\mathcal{M}$
end for
return $\hat{x}^0$

4 Experiments

4.1 Experimental Setup

Hardware and models. All experiments are conducted on NVIDIA H200 GPUs. We use **LLaDA-8B-Instruct** (Nie et al., 2025) as the primary backbone and additionally report results on **LLaDA-1.5 (8B)** (Zhu et al., 2025).

Common inference regime and baselines. Unless otherwise noted, all methods are evaluated under a shared single-block masked diffusion regime: the maximum generation length and block length are both set to **256**. We focus on this setting to isolate PURE's post-hoc refinement effect from confounding factors introduced by block scheduling, KV caching, and inter-block conditioning. For Phase-1 draft generation, we consider LLaDA decoding with unmasking granularity $k \in \{1, 2, 4\}$, corresponding to 256, 128, and 64 denoising steps, respectively. We compare against WINO (Hong et al., 2025), RCR (He et al., 2025), ReMDM (Wang et al., 2025a), and Fast-dLLM (Wu et al., 2025b) under this shared regime. Exact baseline-specific settings are provided in Appendix F.

Post-hoc refinement (PURE). PURE introduces four hyperparameters: seed ratio q, window radius w, relaxation length L, and relaxation rate ρ.

Since **GSM8K** (Cobbe et al., 2021) does not

provide an official development split, we use **GSM1K** (Zhang et al., 2024) as a development set for hyperparameter selection and reuse the selected configuration for all benchmarks. Unless otherwise noted, we use the default setting q=0.2, w=3, L=15, and ρ=0.1. We report results under three refinement budgets, using $B \in \{10, 20, 40\}$ post-hoc steps, where each step denotes one additional refinement iteration applied after the Phase-1 draft generation.

Benchmarks. We evaluate on seven benchmarks spanning mathematical reasoning, scientific question answering, commonsense reasoning, and instruction following: **GSM8K** (Cobbe et al., 2021), **MATH** (Hendrycks et al., 2021), **ASDiv** (Miao et al., 2020), **SVAMP** (Patel et al., 2021), **GPQA-Diamond** (Rein et al., 2024), **CSQA** (Talmor et al., 2019), and **IFEval** (Zhou et al., 2023).

Compute budget. Our total step budget is the sum of Phase-1 decoding steps and Phase-2 refinement budgets B(steps). With the one-token-per-step Phase-1 schedule (256 steps for a 256-token draft), adding refinement steps corresponds to a total budget of $256+B$ steps. Unless otherwise stated, we report single-run accuracy; multi-seed averages are indicated in the relevant table captions.

4.2 Main Results

We first evaluate whether additional post-hoc refinement steps improve accuracy under fixed refinement budgets. Table 1 reports results across seven benchmarks as we vary the refinement budget $B \in \{10, 20, 40\}$ on top of the matched Phase-1 LLaDA draft baseline.

Overall, PURE substantially improves over the matched Phase-1 drafts, with the clearest and most consistent gains on mathematical reasoning benchmarks. On **GSM8K**, for example, PURE improves the k=1 draft from 61.41 to 71.70 with B=10 and further to 74.27 with B=40. We observe similar upward trends on **MATH** (25.68→31.33) and **ASDiv** (76.41→82.72). On **SVAMP**, the smallest budget (B=10) slightly underperforms the k=1 draft baseline (73.33→72.33), but larger budgets recover and reach 78.33 at B=40, matching the best overall result in the table.

On the remaining three benchmarks, results are mixed but still competitive. PURE achieves the best score on **IFEval** (46.39) and the second-best scores on **GPQA-Diamond** (26.77) and **CSQA** (80.26).

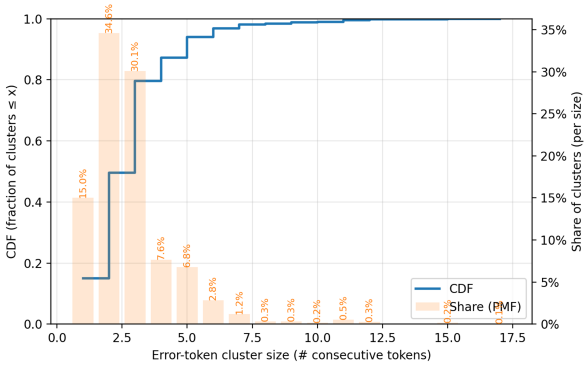

Figure 4: Error-token clustering on GSM1K incorrect samples. We compute clusters of *consecutive* error tokens and report the CDF (blue, left axis) and the per-size mass (orange bars, right axis). Errors frequently occur in short contiguous runs rather than isolated single tokens, motivating local window masking around detected errors.

Taken together, these results suggest that PURE offers a useful trade-off between additional inference compute and accuracy: a modest post-hoc refinement budget can often convert additional inference compute into higher accuracy. We observe a similar qualitative trend on **LLaDA-1.5** (Zhu et al., 2025). Full results are reported in Appendix Table 2.

4.3 Spatial Locality: Consecutive Error-Token Clusters

To understand why pointwise fixes can be insufficient, we analyze the *spatial structure* of errors in incorrect drafts on GSM1K. We group consecutive error tokens into clusters and measure the cluster-size distribution. Figure 4 shows that errors frequently occur in short contiguous spans rather than as isolated single tokens, supporting the use of deterministic window masking around detected errors.

4.4 Random Remasking is Not Enough

To rule out the hypothesis that PURE's gains arise from simply perturbing many tokens, we compare against a naive baseline that randomly remasks a fixed fraction r of tokens from the entire generated sequence before refinement. Figure 5 shows that sweeping large remasking ratios yields limited improvements, while PURE consistently achieves substantially higher accuracy for the same refinement budget. This indicates that effective post-hoc correction requires structured and targeted perturbations rather than uniformly random masking.

	GSM8K		MATH		ASDiv		SVAMP		GPQA-Diamond		CSQA		IFEval	
	Acc.	Steps	Acc.	Steps	Acc.	Steps	Acc.	Steps	Acc.	Steps	Acc.	Steps	Acc.	Steps
Draft generation (no refinement)														
LLaDA (k=1)	61.41	256	25.68	256	76.41	256	73.33	256	23.74	256	**80.51**	256	44.92	256
LLaDA (k=2)	56.41	128	25.08	128	69.72	128	73.00	128	23.74	128	72.48	128	44.54	128
LLaDA (k=4)	53.75	64	23.20	64	66.38	64	71.00	64	24.75	64	67.81	64	41.03	64
WINO	67.02	47	28.00	72	71.24	43	74.67	23	24.24	16	73.22	20	‡46.21	88
RCR	62.32	64	25.32	64	68.72	64	†77.67	64	23.23	64	56.51	64	18.85	64
ReMDM	57.16	384	25.04	384	72.89	384	**78.33**	384	24.75	384	72.67	384	43.07	384
Fast-dLLM	56.94	98	25.42	113	70.02	96	75.00	52	**27.78**	144	76.17	39	37.15	95
PURE on LLaDA (k=1)														
B=10	71.70 (+10.29p)	266	28.84 (+3.16p)	266	77.08 (+0.67p)	266	72.33 (-1.00p)	266	22.73 (-1.01p)	266	79.93 (-0.58p)	266	45.66 (+0.74p)	266
B=20	†72.76 (+11.35p)	276	29.81 (+4.13p)	276	†79.40 (+2.99p)	276	75.33 (+2.00p)	276	24.24 (+0.50p)	276	80.02 (-0.49p)	276	45.47 (+0.55p)	276
B=40	**74.27** (+12.86p)	296	**31.33** (+5.65p)	296	**82.72** (+6.31p)	296	**78.33** (+5.00p)	296	‡26.77 (+3.03p)	296	‡80.26 (-0.25p)	296	46.03 (+1.11p)	296
PURE on LLaDA (k=2)														
B=10	69.22 (+12.81p)	138	27.62 (+2.54p)	138	68.76 (-0.96p)	138	76.00 (+3.00p)	138	23.23 (-0.51p)	138	74.61 (+2.13p)	138	45.10 (+0.56p)	138
B=20	71.34 (+14.93p)	148	29.36 (+4.28p)	148	72.06 (+2.34p)	148	76.33 (+3.33p)	148	21.72 (-2.02p)	148	74.69 (+2.21p)	148	**46.39** (+1.85p)	148
B=40	72.55 (+16.14p)	168	†30.36 (+5.28p)	168	75.27 (+5.55p)	168	76.67 (+3.67p)	168	23.23 (-0.51p)	168	75.18 (+2.70p)	168	46.02 (+1.48p)	168
PURE on LLaDA (k=4)														
B=10	65.88 (+12.13p)	74	25.98 (+2.78p)	74	68.76 (+2.38p)	74	76.00 (+5.00p)	74	24.24 (-0.51p)	74	67.40 (-0.41p)	74	42.14 (+1.11p)	74
B=20	68.84 (+15.09p)	84	27.18 (+3.98p)	84	72.06 (+5.68p)	84	76.33 (+5.33p)	84	23.74 (-1.01p)	84	69.12 (+1.31p)	84	42.69 (+1.66p)	84
B=40	70.74 (+16.99p)	104	27.66 (+4.46p)	104	75.27 (+8.89p)	104	76.67 (+5.67p)	104	24.75 (+0.00p)	104	72.07 (+4.26p)	104	43.80 (+2.77p)	104

Table 1: Main results on seven benchmarks with the LLaDA-8B-Instruct backbone. Acc. denotes accuracy and Steps denotes the total number of denoising steps (NFE). For Phase-1 drafts, $k \in \{1, 2, 4\}$ corresponds to 256/128/64 denoising steps, respectively. PURE adds B post-hoc refinement steps on top of the matched Phase-1 draft baseline, so the total step budget is Phase-1 steps + B. "(+p)" denotes the absolute gain over the corresponding LLaDA draft at the same k. **Bold** and † indicate the best and second-best accuracy for each benchmark, respectively.

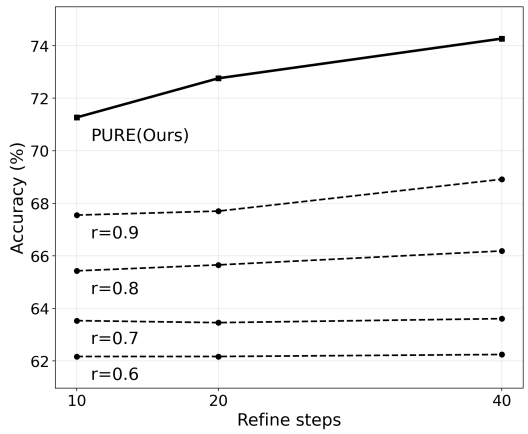

Figure 5: **PURE vs. naive random remasking.** We plot accuracy under increasing refinement steps for PURE and a baseline that performs refinement after randomly masking a fixed rate r of tokens from the entire generated sequence. PURE achieves substantially higher accuracy for the same refinement budget, suggesting that effective post-hoc correction requires more than uniformly random perturbations.

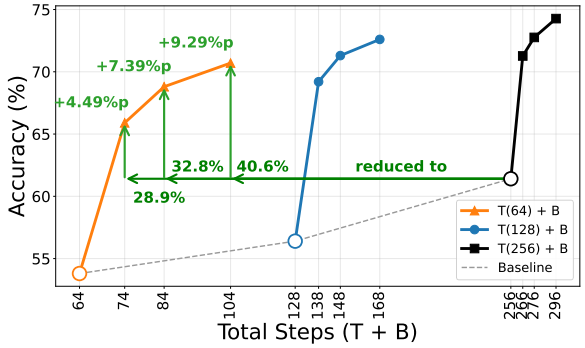

Figure 6: Accuracy vs. total decoding steps under different Phase-1 step granularities. We vary k (baseline steps $= 256/k$; shown in the legend) and apply PURE as post-hoc refinement. Open circles denote the corresponding baseline accuracy before refinement, and solid curves show accuracy after adding refinement steps. PURE recovers substantial accuracy even when the initial draft is generated with fewer steps, illustrating a favorable compute–quality trade-off.

4.5 Compute–Quality Trade-off via Faster Drafts

Finally, we study how PURE behaves when the Phase-1 draft is generated with fewer steps by increasing the unmasking granularity. Figure 6 reports accuracy as a function of the total decoding steps when we vary k (unmasking k tokens per step), which reduces the Phase-1 baseline steps to $256/k$. As expected, generating drafts with fewer steps lowers the baseline accuracy (open circles). However, PURE recovers a substantial portion of this lost accuracy as refinement steps are added, and achieves competitive performance even when the initial draft is produced under a smaller Phase-1 budget. These results suggest a practical operating point for inference: generate a fast initial draft and allocate the remaining compute budget to targeted post-hoc refinement.

5 Related Work

Masked discrete diffusion language models. Recent diffusion language models for text largely adopt masked discrete diffusion, where generation starts from a fully-masked sequence and iteratively predicts masked tokens. (Austin et al., 2021) MDLM establishes a strong masked diffusion baseline, and complementary objectives within discrete denoising have also been explored (Lou et al., 2024; Sahoo et al., 2024). Importantly, masked diffusion LMs have been scaled via large-scale pretraining and instruction tuning, and open diffusion LLMs continue to push quality and usability (Nie et al., 2025; Ye et al., 2025; Song et al., 2025; Bie et al., 2025; Zhu et al., 2025; Cheng et al., 2025).

Decoding policies and remasking-based refinement. In masked diffusion, inference-time policies—which tokens to unmask and when—critically determine the quality. (Kim et al., 2025). More broadly, diffusion models exhibit substantial variation in inference-time samplers and scheduling strategies, motivating a large design space for inference-time control (Li et al., 2025a; Wu et al., 2025a,b; Liu et al., 2025; Wang et al., 2025b; Huang et al., 2025a; Lee et al., 2025; Li et al., 2025b)

A line of work improves generation by allowing revision through remasking and refinement during sampling; for example, ReMDM introduces a remasking sampler that enables inference-time compute scaling (Wang et al., 2025a; Hong et al., 2025; Dong et al., 2025; Horvitz et al., 2025; Huang et al., 2025b; He et al., 2025; Mounier and Idehpour, 2025a; Labs et al., 2025). More broadly, refinement can also be organized as a two-stage fill-and-refine procedure (Tian et al., 2025).

6 Conclusion

We investigated why naive post-hoc correction often fails in masked discrete diffusion decoding, and identified two key factors: *contextual lock-in*, where local neighborhoods reconstruct the same error, and *spatial locality*, where errors cluster within short spans. Based on these findings, we proposed **PURE**, a training-free post-hoc refinement method that profiles token-level instability during drafting and performs targeted refinement through deterministic window masking and stochastic leftward relaxation. Across benchmarks, PURE improves over matched draft baselines with moderate refinement budgets, with the clearest gains on mathematical reasoning tasks, while adding only B post-hoc steps beyond the draft. These results suggest that lightweight, targeted post-hoc refinement can be an effective way to convert additional test-time compute into higher accuracy in masked discrete diffusion decoding.

7 Limitations

Our work has several limitations.

Limited evaluation scope. We evaluate PURE primarily on reasoning-focused benchmarks under a fixed output length. Appendix G adds a controlled creative-writing probe, but evidence for open-ended generation remains limited.

Dependence on decoding regime and confidence signals. PURE relies on instability signals such as `drop_conf` to select refinement seeds, and the reliability of these signals may depend on the model and inference setup. Our current formulation is developed for vanilla single-block masked diffusion decoding, where confidence can be tracked over the full sequence and targeted refinement is applied to a sparse subset of positions within the full draft. These conditions do not directly carry over to blockwise or KV-cached diffusion decoders, where signal availability and conditioning structure differ from the setting studied here. In preliminary experiments, a naive plug-in in a blockwise setting did not preserve gains, suggesting that extending PURE beyond the present setup may require a block- and cache-aware variant.

Failure modes requiring non-local corrections. PURE is designed for targeted refinement through limited remasking and a short refinement chain. It may therefore be less effective when correcting an error requires coordinated changes across distant positions rather than within a local neighborhood. Increasing the masking scope can partially address such cases, but overly aggressive remasking may also destabilize otherwise-correct context.

Additional inference-time cost. PURE improves quality by adding a second refinement phase on top of draft generation, and therefore still incurs extra inference-time cost even when the refinement budget is much smaller than the drafting budget. A practical question is not only whether refinement helps, but how to allocate refinement budget as a function of task difficulty, output length, and draft quality. More adaptive budget allocation strategies and length-aware refinement schedules would make the method more efficient in practice.

Acknowledgments

This work was supported by NAVER Corp. and the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea Government (MSIT) (No. RS-2020-II201336, Artificial Intelligence Graduate School Support (UNIST)).

References

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2021. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*.

Tiwei Bie, Maosong Cao, Kun Chen, Lun Du, Mingliang Gong, Zhuochen Gong, Yanmei Gu, Jiaqi Hu, Zenan Huang, Zhenzhong Lan, Chengxi Li, Chongxuan Li, Jianguo Li, Zehuan Li, Huabin Liu, Lin Liu, Guoshan Lu, Xiaocheng Lu, Yuxin Ma, and 12 others. 2025. Llada2.0: Scaling up diffusion language models to 100b. *Preprint*, arXiv:2512.15745.

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

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Yihong Dong, Zhaoyu Ma, Xue Jiang, Zhiyuan Fan, Jiaru Qian, Yongmin Li, Jianha Xiao, Zhi Jin, Rongyu Cao, Binhua Li, Fei Huang, Yongbin Li, and Ge Li. 2025. Saber: An efficient sampling with adaptive acceleration and backtracking enhanced remasking for diffusion language model. *Preprint*, arXiv:2510.18165.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Conference on Empirical Methods in Natural Language Processing*.

Haoyu He, Katrin Renz, Yong Cao, and Andreas Geiger. 2025. MDPO: Overcoming the training-inference divide of masked diffusion language models. *Preprint*, arXiv:2508.13148.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. 2022. Video diffusion models. In *Advances in Neural Information Processing Systems*, volume 35, pages 8633–8646. Curran Associates, Inc.

Feng Hong, Geng Yu, Yushi Ye, Haicheng Huang, Huangjie Zheng, Ya Zhang, Yanfeng Wang, and Jiangchao Yao. 2025. Wide-In, Narrow-Out: Revokable decoding for efficient and effective DLLM. In *NeurIPS 2025 Workshop on Efficient Reasoning*.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. 2021. Argmax flows and multinomial diffusion: Learning categorical distributions. In *Advances in Neural Information Processing Systems*.

Zachary Horvitz, Raghav Singhal, Hao Zou, Carles Domingo-Enrich, Zhou Yu, Rajesh Ranganath, and Kathleen McKeown. 2025. No Compute Left Behind: Rethinking reasoning and sampling with masked diffusion models. *Preprint*, arXiv:2510.19990.

Pengcheng Huang, Shuhao Liu, Zhenghao Liu, Yukun Yan, Shuo Wang, Zulong Chen, and Tong Xiao. 2025a. Pc-sampler: Position-aware calibration of decoding bias in masked diffusion models. *Preprint*, arXiv:2508.13021.

Zemin Huang, Yuhang Wang, Zhiyang Chen, and Guo-Jun Qi. 2025b. Don't Settle Too Early: Self-reflective remasking for diffusion language models. *Preprint*, arXiv:2509.23653.

Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham M. Kakade, and Sitan Chen. 2025. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. In *Forty-second International Conference on Machine Learning*.

Inception Labs, Samar Khanna, Siddhant Kharbanda, Shufan Li, Harshit Varma, Eric Wang, Sawyer Birnbaum, Ziyang Luo, Yanis Miraoui, Akash Palrecha, Stefano Ermon, Aditya Grover, and Volodymyr Kuleshov. 2025. Mercury: Ultra-fast language models based on diffusion. *Preprint*, arXiv:2506.17298.

Hugo Lavenant and Giacomo Zanella. 2025. Error bounds and optimal schedules for masked diffusions with factorized approximations. *Preprint*, arXiv:2510.25544.

Sanghyun Lee, Seungryong Kim, Jongho Park, and Dongmin Park. 2025. Lookahead unmasking elicits accurate decoding in diffusion language models. *Preprint*, arXiv:2511.05563.

Jinsong Li, Xiaoyi Dong, Yuhang Zang, Yuhang Cao, Jiaqi Wang, and Dahua Lin. 2025a. Beyond fixed: Training-free variable-length denoising for diffusion large language models. *Preprint*, arXiv:2508.00819.

Pengxiang Li, Yefan Zhou, Dilxat Muhtar, Lu Yin, Shilin Yan, Li Shen, Yi Liang, Soroush Vosoughi, and Shiwei Liu. 2025b. Diffusion language models know the answer before decoding. *Preprint*, arXiv:2508.19982.

Sulin Liu, Juno Nam, Andrew Campbell, Hannes Stark, Yilun Xu, Tommi Jaakkola, and Rafael Gomez-Bombarelli. 2025. Think while You Generate: Discrete diffusion with planned denoising. In *The Thirteenth International Conference on Learning Representations*.

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

Shen-yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A diverse corpus for evaluating and developing English math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 975–984, Online. Association for Computational Linguistics.

Nikita Mounier and Parsa Idehpour. 2025a. Review, Remask, Refine: Process-guided block diffusion for text generation. In *ICML 2025 Workshop on Methods and Opportunities at Small Scale*.

Nikita Mounier and Parsa Idehpour. 2025b. Review, remask, refine (r3): Process-guided block diffusion for text generation. *Preprint*, arXiv:2507.08018.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, JUN ZHOU, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2025. Large language diffusion models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695.

Subham Sekhar Sahoo, Marianne Arriola, Aaron Gokaslan, Edgar Mariano Marroquin, Alexander M Rush, Yair Schiff, Justin T Chiu, and Volodymyr Kuleshov. 2024. Simple and effective masked diffusion language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. 2024. Simplified and generalized masked diffusion for discrete data. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265, Lille, France. PMLR.

Yang Song and Stefano Ermon. 2019. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Yuxuan Song, Zheng Zhang, Cheng Luo, Pengyang Gao, Fan Xia, Hao Luo, Zheng Li, Yuehang Yang, Hongli Yu, Xingwei Qu, Yuwei Fu, Jing Su, Ge Zhang, Wenhao Huang, Mingxuan Wang, Lin Yan, Xiaoying Jia, Jingjing Liu, Wei-Ying Ma, and 3 others. 2025. Seed Diffusion: A large-scale diffusion language model with high-speed inference. *Preprint*, arXiv:2508.02193.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Runchu Tian, Junxia Cui, Xueqiang Xu, Feng Yao, and Jingbo Shang. 2025. Finish first, perfect later: Test-time token-level cross-validation for diffusion large language models. *Preprint*, arXiv:2510.05090.

Guanghan Wang, Yair Schiff, Subham Sahoo, and Volodymyr Kuleshov. 2025a. Remasking discrete diffusion models with inference-time scaling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Xu Wang, Chenkai Xu, Yijie Jin, Jiachun Jin, Hao Zhang, and Zhijie Deng. 2025b. Diffusion llms can do faster-than-ar inference via discrete diffusion forcing. *Preprint*, arXiv:2508.09192.

Chengyue Wu, Hao Zhang, Shuchen Xue, Shizhe Diao, Yonggan Fu, Zhijian Liu, Pavlo Molchanov, Ping Luo, Song Han, and Enze Xie. 2025a. Fast-dllm v2: Efficient block-diffusion llm. *Preprint*, arXiv:2509.26328.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. 2025b. Fast-dLLM: Training-free acceleration of diffusion LLM by enabling KV cache and parallel decoding. *Preprint*, arXiv:2505.22618.

Chenxiao Yang, Cai Zhou, David Wipf, and Zhiyuan Li. 2025. On powerful ways to generate: Autoregression, diffusion, and beyond. *Preprint*, arXiv:2510.06190.

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

Hugh Zhang, Jeff Da, Dean Lee, Vaughn Robinson, Catherine Wu, William Song, Tiffany Zhao, Pranav Vishnu Raja, Charlotte Zhuang, Dylan Z Slack, Qin Lyu, Sean M. Hendryx, Russell Kaplan, Michele Lunati, and Summer Yue. 2024. A careful examination of large language model performance on grade school arithmetic. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *Preprint*, arXiv:2311.07911.

Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Hu, Jun Zhou, Jianfei Chen, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2025. LLaDA 1.5: Variance-reduced preference optimization for large language diffusion models. *Preprint*, arXiv:2505.19223.

A Results on LLaDA-1.5

Table 2 reports results on LLaDA-1.5 under the same evaluation protocol as Table 1. Overall, PURE shows the same qualitative behavior as in the main results, with accuracy generally improving as the refinement budget increases. The gains are especially strong on math-oriented benchmarks such as GSM8K, MATH, and ASDiv, while the improvements are more mixed on some general-domain benchmarks such as CSQA and IFEval. These results suggest that PURE's post-hoc refinement benefits are not specific to LLaDA-8B-Instruct.

B Controlled Entropy Probes for Structured Masking

To support the interpretation in Sec. 2.4, we conduct controlled entropy probes on GSM1K incorrect samples. Our goal is to test whether structured window and leftward masking increase uncertainty at the error position more than random masking under the same masking budget.

Probe setup. For each incorrect draft, we keep all non-masked tokens fixed to their original values in the draft and measure the model predictive entropy at the error position under different masking patterns. Averaging these measurements over

	GSM8K		MATH		ASDiv		SVAMP		GPQA-Diamond		CSQA		IFEval	
	Acc.	Steps	Acc.	Steps	Acc.	Steps	Acc.	Steps	Acc.	Steps	Acc.	Steps	Acc.	Steps
Draft generation (no refinement)														
LLaDA-1.5 (k=1)	58.45	256	25.84	256	77.40	256	79.67	256	23.74	256	**81.41**	256	51.20	256
LLaDA-1.5 (k=2)	55.88	128	24.86	128	75.14	128	80.67	128	25.25	128	78.46	128	51.20	128
LLaDA-1.5 (k=4)	52.31	64	23.22	64	71.89	64	79.67	64	24.24	64	69.37	64	46.40	64
WINO	68.16	47	28.34	69	79.00	42	79.00	25	25.76	117	76.09	35	**53.42**	106
RCR	61.49	64	25.54	64	76.10	64	**83.33**	64	24.24	64	41.52	64	22.37	64
ReMDM	56.71	384	25.56	384	76.70	384	81.67	384	25.25	384	78.54	384	50.28	384
Fast-dLLM	56.86	71	24.46	81	77.18	65	78.33	36	**28.79**	120	77.72	53	45.66	110
PURE on LLaDA-1.5 (k=1)														
B=10	73.54 (+15.09p)	266	29.26 (+3.42p)	266	78.96 (+1.56p)	266	82.33 (+2.66p)	266	26.77 (+3.03p)	266	80.92 (-0.49p)	266	48.80 (-2.40p)	266
B=20	†75.51 (+17.06p)	276	†30.98 (+5.14p)	276	†80.95 (+3.55p)	276	82.33 (+2.66p)	276	**28.79** (+5.05p)	276	†81.08 (-0.33p)	276	51.02 (-0.18p)	276
B=40	**76.12** (+17.67p)	296	**31.84** (+6.00p)	296	**81.13** (+3.73p)	296	†83.00 (+3.33p)	296	†28.28 (+4.54p)	296	†81.08 (-0.33p)	296	†51.39 (+0.19p)	296
PURE on LLaDA-1.5 (k=2)														
B=10	71.04 (+15.16p)	138	28.26 (+3.40p)	138	76.62 (+1.48p)	138	82.67 (+2.00p)	138	25.25 (+0.00p)	138	78.13 (-0.33p)	138	45.66 (-5.54p)	138
B=20	73.24 (+17.36p)	148	29.68 (+4.82p)	148	78.74 (+3.60p)	148	**83.33** (+2.66p)	148	25.25 (+0.00p)	148	78.38 (-0.08p)	148	48.24 (-2.96p)	148
B=40	73.54 (+17.66p)	168	30.20 (+5.34p)	168	78.83 (+3.69p)	168	†83.00 (+2.33p)	168	25.76 (+0.51p)	168	78.46 (+0.00p)	168	50.46 (-0.74p)	168
PURE on LLaDA-1.5 (k=4)														
B=10	70.58 (+18.27p)	74	27.10 (+3.88p)	74	75.10 (+3.21p)	74	81.33 (+1.66p)	74	22.22 (-2.02p)	74	73.30 (+3.93p)	74	45.29 (-1.11p)	74
B=20	71.11 (+18.80p)	84	28.98 (+5.76p)	84	77.74 (+5.85p)	84	82.00 (+2.33p)	84	22.73 (-1.51p)	84	73.63 (+4.26p)	84	45.84 (-0.56p)	84
B=40	72.25 (+19.94p)	104	29.42 (+6.20p)	104	77.70 (+5.81p)	104	82.67 (+3.00p)	104	24.75 (+0.51p)	104	74.04 (+4.67p)	104	47.69 (+1.29p)	104

Table 2: Results on LLaDA-1.5 under the same evaluation protocol as Table 1. **Bold** and † indicate the best and second-best accuracy for each benchmark, respectively.

incorrect drafts provides an empirical estimate of the average uncertainty increase induced by structured masking. In all probes, we use the manually identified error position i and perform a one-step forward measurement at that position.

Masking conditions. We compare the following settings:

- **E1:** mask only the error token i.
- **E2 (window):** mask the error token and its local window, i.e., $\{i\} \cup W_i(w)$.
- **E2c (control):** mask the error token and the same number of additional tokens as in E2, but at random non-local positions.
- **E3 (leftward):** start from E2 and additionally mask a subset of preceding-context tokens using the same leftward masking rule as PURE with rate ρ.
- **E3c (control):** start from E2 and add the same number of extra masks as in E3, but at random positions.

Under this setup, E2-E1 measures the uncertainty increase induced by local window masking, while E3-E2 measures the additional uncertainty increase induced by leftward masking.

Window masking results. Table 3 reports the effect of varying the window radius w. As w increases, E2-E1 grows monotonically, meaning that masking a larger local window produces a larger uncertainty increase at the error position. By contrast, the count-matched random control E2c-E1 remains close to zero across all settings. This shows that the effect does not come from simply masking more tokens, but from removing the local context that supports the incorrect token.

window radius w	E2-E1	E2c-E1	E2-E2c
1	+0.01950	+0.00098	+0.01852
2	+0.03937	+0.00129	+0.03808
3	+0.06601	+0.00115	+0.06486
4	+0.09943	+0.00115	+0.09828

Table 3: Window sweep with count-matched control. E2-E1 is the uncertainty increase caused by masking the local window, E2c-E1 is the equal-budget random control, and E2-E2c is the gap between structured and random masking.

Leftward masking results. Table 4 reports the effect of varying the leftward masking rate ρ with $w = 3$ fixed. As ρ increases, E3-E2 also grows monotonically, indicating that masking prefix-side context further increases uncertainty at the error position. Again, the count-matched random control E3c-E2 remains near zero or slightly negative. This suggests that leftward masking helps not because it adds more noise, but because it specifically weakens prefix-side evidence that co-adapts with the same wrong local mode.

Takeaway. Together, these probes support the interpretation in Sec. 2.4: window masking and leftward masking increase uncertainty at the error position in a structured way, thereby weakening contextual lock-in and creating more room for alternative corrections during re-sampling.

leftward rate ρ	E3-E2	E3c-E2	E3-E3c
0.0	+0.00000	+0.00000	+0.00000
0.1	+0.00643	-0.00019	+0.00662
0.2	+0.01011	-0.00063	+0.01074
0.3	+0.01122	-0.00101	+0.01223

Table 4: Leftward sweep with count-matched control ($w = 3$ fixed). E3-E2 is the additional uncertainty increase caused by leftward masking, E3c-E2 is the equal-budget random control, and E3-E3c is the gap between structured and random masking.

Signal	**Avg Acc (%)**
Baseline	61.41
final_conf	66.19
random	67.20
init_conf	69.24
drop_margin	70.99
drop_conf	71.70

Table 5: Signal ablation results on GSM8K (average over 3 seeds).

C Ablation: Token-Selection Signals

Next, we ablate the token-selection signal used by PURE. Table 5 compares token-selection signals on GSM8K under the same refinement budget (B=10). Among the tested options, the confidence-drop signal used by PURE (`drop_conf`, corresponding to the instability score Δ_i in Sec. 3.1) performs best, reaching 71.70% accuracy (+10.29 over the Phase-1 baseline of 61.41%). It also consistently surpasses signals based on absolute confidence (e.g., `final_conf`) and random selection. These results suggest that relative confidence degradation across refinement is a stronger indicator of correctable regions than absolute uncertainty at a single snapshot.

D Hyperparameter Robustness

Robustness to hyperparameters. Figure 7 visualizes the accuracy landscape of our refinement procedure across three key hyperparameters: the window radius w, the relaxation rate ρ, and the seed ratio q. We report accuracy over (w, ρ) for three fixed values of q. Overall, the method exhibits stable performance across a broad region of the hyperparameter space, indicating that our gains do not rely on a narrowly tuned configuration.

Moderate context relaxation yields the best trade-off. Beyond robustness, the heatmaps reveal a consistent peak in a moderate regime (e.g., w=3, $q\approx$0.2, $\rho\approx$0.1), suggesting an optimal balance between escaping contextual lock-in and preserving global coherence. When the perturbation is too weak (small w and ρ), the model often reselects the same locally self-reinforced tokens, leading to limited correction. Conversely, overly aggressive configurations (large w and/or ρ combined with large q) can over-destabilize the context, degrading accuracy. This interaction highlights that w, ρ, and q jointly control the effective "context relaxation" strength, and that improvements arise from controlled relaxation rather than indiscriminate regeneration. Unless otherwise stated, we therefore use $(w, \rho, q) = (3, 0.1, 0.2)$ as the default setting in subsequent experiments.

Variant	(L, w)	Acc. (%)
Left	$(15, 3)$	71.70
Right	$(15, 3)$	71.32
Neighbor	$(9, 3)$	70.89

Table 6: Budget-matched relaxation variants (average over 3 seeds).

E Budget-matched relaxation variants

Setup. We compare three relaxation variants under a budget-matched remasking protocol: LEFT (stochastic leftward relaxation), RIGHT (stochastic rightward relaxation), and NEIGHBOR (symmetric local masking). All runs use the same default hyperparameters for our method; only the relaxation variant is varied. For LEFT and RIGHT, we set the relaxation length to L=15. For NEIGHBOR, we use a symmetric window radius w=3 and set L=9 to match the effective number of eligible remasking candidates (details below).

Budget matching. With w=3, tokens in the immediate w-neighborhood are already included in the fixed local remasking pool. Thus, for the directional variants (LEFT and RIGHT) with L=15, the number of additional eligible candidates beyond the fixed local window is $L_{\text{eff}} = L - w = 12$. To ensure a fair comparison, we set the NEIGHBOR parameter to L=9, so that its effective candidate count matches the same 12-candidate budget under our implementation. Equivalently, for NEIGHBOR we have $L_{\text{eff}}^{\text{NEIGHBOR}} = L + w = 12$ when w=3, hence L=9.

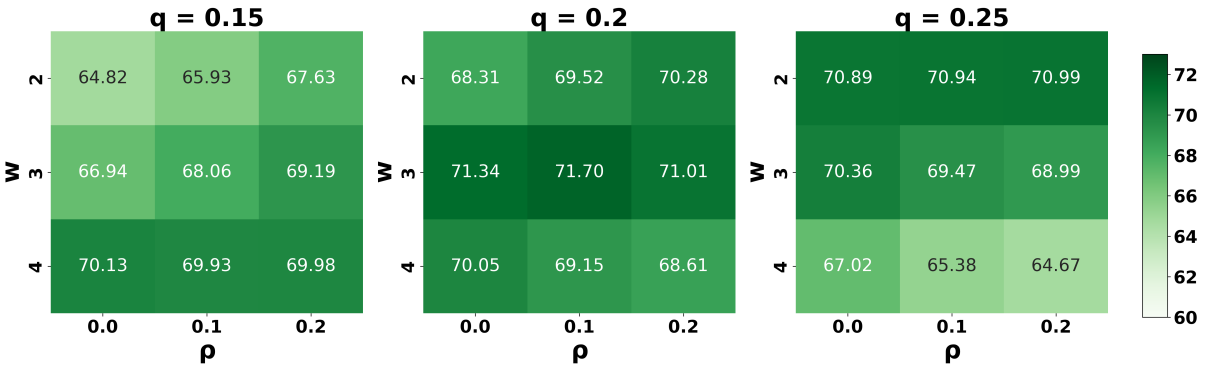

Figure 7: **Robust hyperparameter landscape and the role of moderate context relaxation.** Heatmaps report accuracy over the window radius w and stochastic leftward relaxation rate ρ for three fixed seed ratios q. Across slices, performance remains stable over a broad region and consistently peaks at *moderate* settings (e.g., w=3, $q\approx$0.2, $\rho\approx$0.1), indicating that partial context relaxation effectively mitigates contextual lock-in. In contrast, overly aggressive configurations (large w and/or large ρ with large q) can over-destabilize the context and degrade accuracy, revealing a clear trade-off between escape from lock-in and global coherence preservation.

F Baseline-specific inference settings

All baselines are evaluated under the shared single-block regime described in Sec. 4.1. Below we report only the settings specific to each baseline beyond this common setup.

WINO. We use WINO with drafting threshold $\tau_1 = 0.6$ and backward verification threshold $\tau_2 = 0.9$.

RCR. We use the MDPO implementation of diffusion decoding with RCR enabled. We set `steps` $= 64$ and use a linear schedule.

ReMDM. We use the paper's max-capped ReMDM-loop configuration for inference-time scaling, with $\eta_{\text{cap}} = 0.02$, `t_on` $= 0.55$, `t_off` $= 0.05$, and $\alpha(t_{\text{on}}) = 0.9$.

Fast-dLLM. For Fast-dLLM, we use the *factor*-based parallel decoding strategy with `factor` $= 1.0$.

G Controlled long-form constrained generation probe

We additionally evaluate long-form open-ended generation on a controlled creative-writing benchmark with 50 prompts designed to stress long-range discourse constraints. Each prompt includes hard constraints, such as mandatory phrase inclusion and an exact sentence in the final paragraph, along with soft global consistency constraints, such as a fixed protagonist role and a single-location setting. A representative prompt is shown below.

For this evaluation, we generate 1024-token outputs and use 100 Phase-2 refinement steps, scaling the refinement budget with output length for budget comparability. Hard-constraint satisfaction is evaluated automatically, while we use GPT-4o-mini to assess long-range consistency, fluency, engagement, contradiction, and soft-constraint satisfaction.

On this benchmark, PURE reduces the contradiction rate (0.26 vs. 0.30), slightly improves soft-constraint satisfaction (0.720 vs. 0.707), and slightly improves judged consistency (3.06 vs. 3.02), fluency (2.92 vs. 2.80), and engagement (2.52 vs. 2.50), while incurring a small drop in the strict hard-constraint pass rate (0.78 vs. 0.82). These results suggest that targeted refinement does not cause catastrophic degradation in long-form open-ended generation, although it introduces a modest trade-off in exact hard-constraint satisfaction.

Representative prompt from the long-form creative-writing set.

> **Genre:** Realism
>
> **Premise:** During a late-night shift at a small clinic, a receptionist keeps receiving calls that reference events that have not happened yet.
>
> **Hard constraints:**
> (1) Include all of the following phrases somewhere in the story: *sticky note*, *coin*

locker, and *paper cup*.
(2) In the last paragraph, include the exact sentence: *"I wrote it down anyway."*

Soft global consistency constraints:
(1) The protagonist remains a receptionist throughout.
(2) The setting stays within one building.
(3) The calls should admit two plausible interpretations without explicit explanation.

Style: First-person past tense with at least two lines of dialogue.