DPRM: A Plug-in Token-Ordering Module for Diffusion Language Models

Anonymous Authors[1]

Abstract

Diffusion language models generate without a fixed left-to-right order, leaving token ordering as a central algorithmic choice. Existing systems mainly use random masking or confidence-driven ordering, which respectively suffer from train–test mismatch and myopic exploration. We introduce **DPRM** (Doob h-transform Process Reward Model), a plug-in token-ordering module that keeps the host architecture, denoising objective and supervision unchanged, and modifies only the ordering policy. DPRM starts from confidence-driven ordering and gradually shifts to process-reward-guided ordering through online estimates. We characterize the exact DPRM policy as a reward-tilted Gibbs reveal law, prove $O(1/N)$ convergence of its stage-wise Soft-BoN approximation, show that the online bucketized controller tracks the exact DPRM score at empirical-Bernstein rates, and establish a sample-complexity advantage under tractable optimization assumptions. We evaluate DPRM as a matched plug-in intervention across *diffusion pretraining, reasoning post-training, test-time scaling, multimodal protein diffusion, single-cell gene-expression diffusion, molecular drug-design diffusion, and DNA regulatory-sequence diffusion*. DPRM improves pretraining, post-training, test-time scaling and single-cell masked diffusion, with strong gains on harder reasoning subsets. In protein, molecular and DNA generation, ordering-aware variants improve selected structural, fragment-constrained or reward-specific metrics, though not every quality metric. These results identify token ordering as a fundamental control axis in diffusion language models and adjacent masked discrete diffusion systems. An anonymized code snapshot is available at: `https://anonymous.4open.scienc e/r/DPRM-DLLM-CBCA/README.md`.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the FoGen Workshop at ICML 2026. Do not distribute.

1. Introduction

Large language models (LLMs) have become the dominant paradigm for modern AI, with autoregressive scaling driving strong performance across language understanding, generation, coding, and reasoning. Yet the autoregressive factorization itself is not equally well matched to every domain. In scientific settings such as proteins, the target object is governed by strong global dependencies rather than a single natural left-to-right order, and recent protein language modeling work explicitly argues that proteins should not be treated as simple linear strings. At the same time, diffusion language models (DLMs) and diffusion large language models (DLLMs) have rapidly emerged as a viable alternative: recent results show that large-scale diffusion language models can approach or match autoregressive baselines on general language tasks, support flexible generation orders, and extend naturally to domains such as reasoning and protein modeling. (Brown et al., 2020; Nie et al., 2025; Sahoo et al., 2024; Ye et al., 2025; Wang et al., 2024b; 2025b; Hsieh et al., 2025)

This flexibility, however, exposes a new algorithmic question that autoregressive models largely hide: *how should a diffusion language model order its tokens?* Recent work has shown that token ordering is not a minor implementation detail. Kim et al. (2025) demonstrate that inference-time adaptive ordering can substantially improve masked diffusion models even when training follows the standard masking setup. Kim et al. (2026) go further and argue that training-time and test-time orders should be aligned, replacing random masking with teacher-forced progressive masking; nevertheless, their practical policy remains confidence-driven. Confidence-centric heuristics also appear beyond training, for example in test-time scaling systems such as Prism (Bai et al., 2026), which preserve or commit high-confidence tokens while selectively revising lower-confidence ones. This reliance on confidence is appealing because it is cheap and effective, but recent diffusion-language-model studies such as Fang et al. (2026) also show its limits: confidence-based decoding can be myopic, suppress useful exploration, and create a quality–exploration trade-off, especially when globally valuable trajectories are not locally the most confident

Table 1. Seven host settings used to test DPRM as a plug-in token-ordering module.

Variant	Host	Stage	Domain	Reference
DPRM-PUMA	PUMA	Pretraining	Language Reasoning	Kim et al. (2026)
DPRM-DMPO	DMPO	Post-training	Language Reasoning	Zhu et al. (2025c)
DPRM-Prism	Prism	Test-time scaling	Language Reasoning	Bai et al. (2026)
DPRM-DPLM	DPLM-2 Bit	Generative modeling	Protein inverse folding	Wang et al. (2025b); Hsieh et al. (2025)
DPRM-DCM	DCM	Generative modeling	Single-cell RNA expression	Bhattacharya et al. (2026)
DPRM-GenMol	GenMol V2	Generative modeling	Molecular drug design	Lee et al. (2025)
DPRM-SDPO	SDPO	Reward optimization	DNA regulatory sequence design	Wang et al. (2025a)

ones.

This raises the central question of the paper: *is there a token-ordering policy that improves on confidence-driven ordering for diffusion language models?* Our answer is yes. We draw inspiration from Bu et al. (2025b), which modeled Chain-of-Thought as Markov process and interpret post-training as reweighting over stochastic reasoning trajectories, and address exploration dilemma by additional process reward. That viewpoint is especially natural for DLLMs, because its token ordering is itself an explicit sequential Markov process over partially revealed states. This suggests that ordering should depend not only on local confidence.

Motivated by this view, we introduce **DPRM** (Doob h-transform Process Reward Model), a plug-in token-ordering module for diffusion language models. DPRM does not redesign the host model, loss, or data pipeline. Instead, it upgrades only the ordering rule. In principle, once a terminal reward oracle is specified, the exact DPRM controller is given by a Doob-style process-reward guidance to the host proposal distribution. In practice, this exact conditional expectation is expensive to evaluate at every candidate token. Our solution is therefore an online approximation that begins with confidence-driven progressive ordering and gradually shifts toward reward-aware DPRM guidance as bucket-level reward estimates become reliable. The design principle is simple: preserve the efficiency of confidence when confidence is informative, then transition to reward-aware ordering when confidence alone becomes too myopic.

We provide theoretical support for this design at three levels. First, the stagewise Soft-BoN approximation converges to the exact DPRM target at rate $O(1/N)$ in terminal KL. Second, the online bucketized controller tracks the exact DPRM score at empirical-Bernstein rates, up to bias from bucket coarsening, warmup, and nonstationarity. Third, under tractable stagewise optimization assumptions, DPRM enjoys a sample-complexity advantage over both random and confidence-only ordering.

We evaluate DPRM as a matched plug-in intervention in seven settings, spanning natural-language pretraining, reasoning post-training, test-time scaling, protein diffusion, single-cell diffusion, molecular diffusion, and DNA reward optimization. In all cases, we keep the host model, objective, and data pipeline fixed as much as possible, and modify only the ordering controller. The results show both consistent gains and meaningful trade-offs. In natural-language tasks, DPRM-PUMA improves the GSM8K validation mean from 29.34 to 34.27 ($+16.8\%$); DMPO-DPRM improves MATH Hard from 44.3 to 47.9 ($+8.1\%$) and Countdown Hard from 29.6 to 33.4 ($+12.8\%$); and DPRM-Prism improves GSM8K voted accuracy from 82.41 to 83.85 ($+1.44$ points). In scientific domains, ordering changes the performance frontier: DPLM ordering variants reduce forward-folding RMSD from 35.47 to 29.43 (-17.0%) and improve TM-score from 0.3071 to 0.3321 ($+8.1\%$); DPRM-DCM improves token recovery from 63.97% to 75.92% and zero-expression accuracy from 78.39% to 99.90%; DPRM-GenMol improves linker validity from 0.142 to 0.429 and scaffold quality from 0.429 to 0.712; and DPRM-SDPO raises HepG2 scores while preserving more ATAC and k-mer quality than confidence-only progressive ordering. These findings support the central claim of the paper: token ordering is a reusable control axis for diffusion language models and adjacent masked discrete diffusion systems, sometimes yielding direct accuracy gains and sometimes exposing controllable domain-specific trade-offs. Additional Related Work is defered to Appendix A.

2. DPRM as a Plug-in Ordering Module

2.1. A Generic View of Token Ordering in Diffusion Models

Let a clean token sequence be $x = (x_1, \dots, x_L) \in [V]^L$, where V is the vocabulary and L is the sequence length; in our language settings, one may equivalently write $x = (q, o)$ for a prompt–response pair. A partially observed diffusion state is denoted by

$$z = (z_1, \dots, z_L) \in ([V] \cup \{\texttt{[MASK]}\})^L,$$

where each coordinate is either a visible token or the mask symbol. We use $t \in \{0, \dots, T\}$ for the denoising stage or phase index, and write the full host state as $s = (x, z, t)$. In multimodal or variable-length hosts, the same notation should be read as the current partially observed token array together with its denoising stage; the presentation below

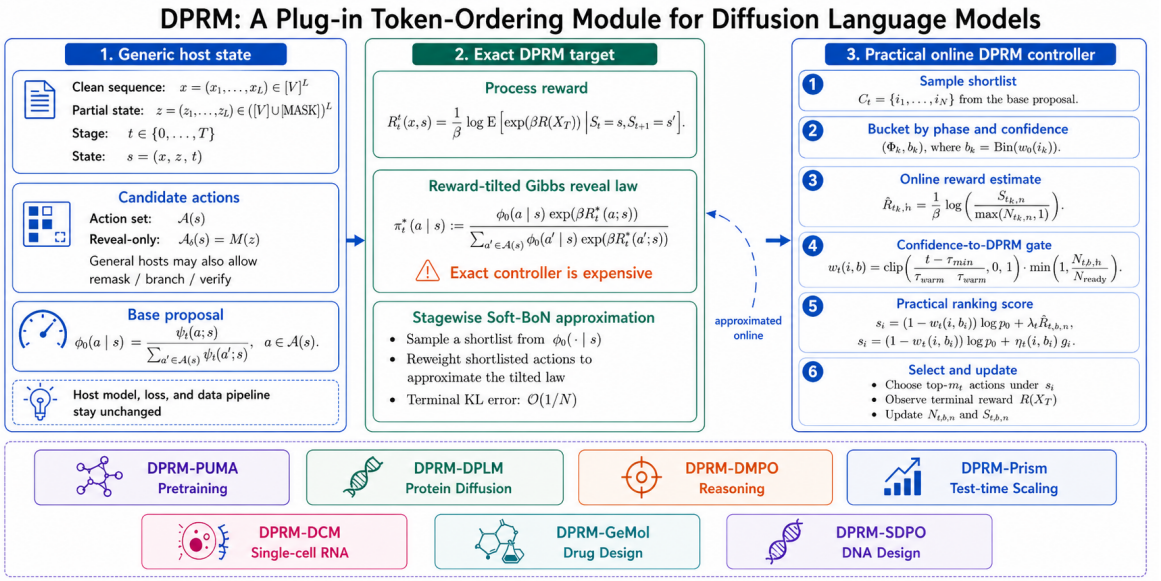

Figure 1. DPRM as a plug-in token-ordering module. The host provides a local proposal over candidate actions, and DPRM replaces only the ordering rule. In practice, DPRM starts from confidence-based warmup and then uses shortlist-based Soft-BoN reweighting together with an online reward estimate to approximate the exact tilted reveal law.

does not depend on the exact modality.

Many diffusion algorithms in this paper share the same ordering problem. At a host state $s = (x, z, t)$, let $\mathcal{I}(s)$ be the set of candidate items whose status may be updated next. In a reveal-only host, $\mathcal{I}(s)$ is simply the set of currently masked positions. In hosts that also allow remasking, branching, or verification, $\mathcal{I}(s)$ may additionally include already visible positions or partial hypotheses. The host then associates each $i \in \mathcal{I}(s)$ with a host-specific local action, such as reveal, keep, remask, branch on, or verify that item.

The generic computation has three steps:

1. compute a base proposal score $\psi_i(s)$ for each $i \in \mathcal{I}(s)$;
2. rank candidate items using an ordering score;
3. choose an update budget $m(s)$ and apply the host-specific action to the top-$m(s)$ items.

Different host algorithms differ mainly in the choice of the base proposal $\psi_i(s)$ and in the host action attached to each item i. Random masking uses a uniform proposal. Random decoding corresponds to $\psi_i(\cdot) = 1$. Confidence-based decoding uses $\psi_i(s) = p_i(s)$, where $p_i(s)$ is the model's current confidence for the update associated with item i. In reveal-only settings, we later write $M(z) \subseteq [L]$ for the masked coordinates and recover the simpler notation $\mathcal{I}(s) = M(z)$.

We now introduce DPRM as a generic plug-in controller for this template.

2.2. Progressive online DPRM

We propose **DPRM** (Doob h-transform Process Reward Model), a generic ordering module that targets a reward-tilted update law. The name comes from the following construction. Let $(S_0, S_1, \ldots, S_T)$ be the host Markov chain under the base controller, where S_t is the partially observed state at stage t, and let X_T be the completed terminal sample. For a terminal reward $R(X_T)$, define the positive space–time harmonic function

$$h_t(s) := \mathbb{E}_{q_0}[\exp(\beta R(X_T)) \mid S_t = s],$$

where q_0 denotes the base transition law. The classical Doob h-transform (Doob, 1957; Léonard, 2014; Chen et al., 2021) tilts the base transition from s to the next state by the ratio h_{t+1}/h_t. Thus high-reward future continuations receive larger transition probability. DPRM applies this idea to token-ordering: it changes the local ordering policy while leaving the host architecture, denoising objective, and data pipeline unchanged.

For a current state s, let $\mathcal{A}(s)$ be the set of candidate local actions. In reveal-only settings, an action is simply a candidate position i, so one may write $a = i$. If action $a \in \mathcal{A}(s)$ is applied at state s, denote the resulting next state by s^a. The normalized base proposal is

$$q_0(a \mid s) := \frac{\psi(a;s)}{\sum_{a' \in \mathcal{A}(s)} \psi(a';s)}, \qquad a \in \mathcal{A}(s), \quad (1)$$

where $\psi(a;s)$ is the host's incumbent local score.

Algorithm 1 Generic DPRM ordering module

Require: host state s_t, candidate-item set $\mathcal{I}(s_t)$, base proposal $\psi_i(s_t)$, phase ϕ_t, update budget m_t, shortlist size N_t, bucket statistics $\{N_{\phi,b}, S_{\phi,b}\}$

```
sample a shortlist C_t = {i_1, ..., i_{N_t}} from the base proposal over I(s_t)
for each shortlisted candidate i ∈ C_t do
    compute bucket index b_i ← Bin(ψ_i(s_t))
    estimate process reward R̂_{φ_t,b_i} ← (1/β) log( S_{φ_t,b_i} / max(N_{φ_t,b_i}, 1) )
    compute gate η_{t,i} ← η_t(φ_t, b_i) by (7)
    set confidence score u_i ← log ψ_i(s_t)
    set DPRM score g_i ← log ψ_i(s_t) + β R̂_{φ_t,b_i}
    combine s_i ← (1 − η_{t,i}) u_i + η_{t,i} g_i
end for
rank the shortlist C_t by s_i
let Â_t(s_t) be the top-m_t candidates in C_t, and apply the corresponding host actions
observe the terminal reward R(X_T) of the completed trajectory generated from this decision
for each i ∈ Â_t(s_t) do
    update N_{φ_t,b_i} ← N_{φ_t,b_i} + 1 and S_{φ_t,b_i} ← S_{φ_t,b_i} + exp(β R(X_T))
end for
```

The exact DPRM process reward is the log-moment future reward after taking action a:

$$R_t^\star(a;s) := \frac{1}{\beta}\log \mathbb{E}[\exp(\beta R(X_T)) \mid S_t = s,\ S_{t+1} = s^a]\,. \tag{2}$$

Equivalently, $\exp(\beta R_t^\star(a;s)) = h_{t+1}(s^a)$. The Doob-transformed local update law is therefore

$$\pi_t^\star(a \mid s) := \frac{q_0(a \mid s)\exp(\beta R_t^\star(a;s))}{\sum_{a' \in \mathcal{A}(s)} q_0(a' \mid s)\exp(\beta R_t^\star(a';s))}. \tag{3}$$

This is the exact reward-tilted Gibbs law targeted by DPRM (Doob, 1957; Léonard, 2014; Chen et al., 2021). It is also expensive: evaluating $R_t^\star$ requires future reward information, and normalizing $\pi_t^\star$ over all candidate actions can be costly.

We therefore use an online, bucketized approximation. A *phase* is a coarse label for where the host is in its reveal or denoising process, for example early, middle, or late. A *confidence bucket* is a coarse bin of the base score $\psi_i(s_t)$, so candidates with similar confidence share statistics. The pair (ϕ, b) defines a small decision cell: decisions made at a similar stage and with similar confidence. For each cell, $N_{\phi,b}$ records how many previous selected events contributed to it, and $S_{\phi,b}$ records their accumulated exponentiated terminal rewards. A *shortlist* is a small subset of candidates sampled from the base proposal before reranking; it avoids evaluating the guidance on every candidate item.

Stage 1: confidence-based train–test aligned progressive sampling. Following PUMA (Kim et al., 2026), we train on inference-like masked states through teacher-forced progressive unmasking. The reveal order is the model's current confidence order. Thus train-time teacher forcing and test-time decoding are aligned from the start: both use the same confidence-based order and therefore traverse the same type of masked states. This removes the exponential waste of random masking and, in the low-entropy regime, inherits the efficiency guarantees of confidence-based decoding (Cai & Li, 2026).

Stage 2: online DPRM guidance with sampled Soft-BoN. After warmup, we gradually replace pure confidence with the Doob-inspired guidance term. As in Stage 1, train-time teacher forcing and test-time decoding remain aligned: at any point in training, both use the same current ordering rule. The rule itself changes, not the alignment principle. Because exact evaluation of (3) is expensive, we approximate it stagewise. We first draw a shortlist from the base proposal $q_0(\cdot \mid s)$, then apply Soft-BoN-style reweighting inside that shortlist using an online bucketized estimator of the process reward.

Each past selected event j contributes its phase ϕ_j, confidence bucket b_j, and terminal reward R_j obtained after completing the resulting trajectory. Define

$$\mathcal{H}_{\phi,b} := \{j : \phi_j = \phi,\ b_j = b\}.$$

For phase ϕ and bucket b, we maintain

$$S_{\phi,b} = \sum_{j \in \mathcal{H}_{\phi,b}} \exp(\beta R_j), \qquad N_{\phi,b} = |\mathcal{H}_{\phi,b}|. \tag{4}$$

This gives the estimator

$$\widehat{R}_{\phi,b} = \frac{1}{\beta}\log\left(\frac{S_{\phi,b}}{\max(N_{\phi,b},1)}\right). \tag{5}$$

For a candidate position i in phase ϕ and bucket b_i, the DPRM-corrected score is

$$g_i = \log \psi_i(s) + \beta \widehat{R}_{\phi,b_i}. \tag{6}$$

The reward guidance is controlled by β through the log-moment estimator, while the gate below determines when the estimate is trusted enough to affect ordering.

We interpolate between confidence and DPRM through

$$\eta_t(\phi,b) = \underbrace{\mathrm{clip}\left(\frac{t - T_{\mathrm{warm}}}{T_{\mathrm{switch}} - T_{\mathrm{warm}}}, 0, 1\right)}_{\text{global transition}} \cdot \underbrace{\min\left(\frac{N_{\phi,b}}{N_{\mathrm{ready}}}, 1\right)}_{\text{bucket readiness}}, \tag{7}$$

so the practical ranking score is

$$s_i = (1 - \eta_t(\phi, b_i)) \log \psi_i(s) + \eta_t(\phi, b_i)\, g_i. \quad (8)$$

Algorithm 1 summarizes the practical controller. It uses the same current ordering rule during teacher-forced training and test-time decoding, thereby preserving train–test alignment while gradually moving from confidence ordering to reward-guided ordering.

3. Theoretical Guarantee

Throughout, let $x = (q, o)$ denote a clean prompt–response pair, let z denote a partially masked state, and let $M(z)$ denote its masked coordinates. The results in this section are host-agnostic: the host enters only through the clean target law ν_*, the terminal reward, the base proposal, and the uncertainty score.

The theory makes four points. First, Proposition 3.1 shows that teacher-forced progressive masking improves train–test alignment without changing the population optimum, and can yield a large statistical advantage over random masking. Second, Theorem 3.2 proved the online bucketized score tracks the exact DPRM score at the standard empirical-Bernstein rate, up to a bundled bias term. Third, Theorem 3.3 reveals that the exact DPRM controller induces a reward-tilted Gibbs reveal law, and its stagewise Soft-BoN approximation converges at rate $O(1/N)$ in terminal KL. Finally, Theorem 3.5 proves the sample complexity gap between DPRM, confidence-driven order, and random counterpart under traditional but plausible assumptions.

Teacher-forced alignment, target preservation, and statistical motivation. Fix a prompt distribution ν_Q and a clean conditional target $p_*(o \mid q)$. Let

$$\nu_*(q, o) = \nu_Q(q)\, p_*(o \mid q) \quad (9)$$

be the induced clean-sample law.

A conditional law $q_\pi(z \mid q, o)$ is *admissible* if it is induced by a finite-horizon teacher-forced reveal chain that starts from the fully masked state outside the prompt, ends at (q, o), and at each step first chooses a reveal set measurable with respect to the current visible state and time, then fills those coordinates with their ground-truth tokens from o. For any admissible q_π, define the target-weighted denoising risk

$$\mathcal{L}_\pi(\theta) = \mathbb{E}_{(q,o)\sim\nu_*}\Big[\mathbb{E}_{z\sim q_\pi(\cdot|q,o)} \textstyle\sum_{i\in M(z)} -\log p_\theta(o_i \mid z, q)\Big]. \quad (10)$$

Also define the weighted masked-state occupancy measure

$$\bar\rho_\pi(q, z, i) := \mathbb{P}_{\nu_*,\, q_\pi}(Q = q, Z = z, i \in M(Z)). \quad (11)$$

Adapting Proposition 1–3 of Kim et al. (2026), we obtain the following informal picture.

Proposition 3.1 (Informal consequences of teacher-forced progressive masking in Proposition 1–3 of Kim et al. (2026))**.** *Suppose the reveal policy depends only on the current visible masked state and time. Then, informally:*

1. ***Teacher-forced alignment.*** *The masked-state marginals induced by the teacher-forced chain agree with those of idealized posterior-based inference run under the same reveal policy.*
2. ***Population minimizer preservation.*** *Under the population objective (10), replacing random masking with admissible teacher-forced progressive masking does not change the Bayes-optimal conditional denoiser; it changes only the occupancy measure over masked states.*
3. ***Statistical separation.*** *There exist latent-variable families for which random masking requires exponentially many samples in the latent dimension, whereas teacher-forced oracle trajectories require only linearly many informative states.*

This is the conceptual starting point for DPRM. Teacher forcing improves alignment, but by itself it does not change the population optimum. The remaining question is therefore how to choose a better reveal order inside the aligned family. Formal statements and proofs are deferred to Appendix C.1.

Online bucketized DPRM approximate the exact controller. The exact DPRM controller uses the process reward $R_t^\star(i; s)$ for every candidate position $i \in M(z)$, which is not available online. Our practical controller replaces these candidate-wise rewards by bucket-level estimates over phase and confidence. Its practical and exact scores are

$$\widehat{g}_t(i; s) := \log \psi_i(s) + \beta\, \eta_t(\phi, b_i(s))\, \widehat{R}_{\phi, b_i(s), t},$$
$$g_t^\star(i; s) := \log \psi_i(s) + \beta R_t^\star(i; s).$$

Also write

$$N_{\min,t}(s) := \min_{i\in M(z)} N_{\phi,b_i(s),t},$$
$$\eta_t(s) := \max_{i\in M(z)} \eta_t(\phi, b_i(s)).$$

Let $\mathcal{N}$ denote the total number of bucket-time events covered by the uniform concentration argument.

Theorem 3.2 (Informal)**.** *Under bounded rewards and mild regularity assumptions on the bucket abstraction and bucket means, there is a nonnegative term* $\mathrm{Bias}_t(s; \delta)$ *such that, with high probability, uniformly over reachable states,*

$$\sup_{i\in M(z)} |\widehat{g}_t(i; s) - g_t^\star(i; s)| = O\Big(\beta\eta_t(s)\sqrt{\frac{\log(\mathcal{N}/\delta)}{N_{\min,t}(s)}} + \beta\eta_t(s)\frac{\log(\mathcal{N}/\delta)}{N_{\min,t}(s)} + \mathrm{Bias}_t(s; \delta)\Big).$$

Consequently, if $\widehat{A}_t(s)$ is the reveal set chosen by the practical controller and $A_t^\star(s)$ is the exact DPRM top-$m(s)$ set under $g_t^\star$, then

$$\sum_{i\in A_t^\star(s)} g_t^\star(i;s) - \sum_{i\in \widehat{A}_t(s)} g_t^\star(i;s) = O\Big(m(s)\Big[\beta\eta_t(s)\sqrt{\frac{\log(\mathcal{N}/\delta)}{N_{\min,t}(s)}} + \beta\eta_t(s)\frac{\log(\mathcal{N}/\delta)}{N_{\min,t}(s)} + \mathrm{Bias}_t(s;\delta)\Big]\Big).$$

This is the second approximation layer. The theorem above controls the additional error from estimating the required process rewards online. The leading terms are the standard empirical-Bernstein $\sqrt{\log(\mathcal{N}/\delta)/N_{\min,t}(s)}$ and $\log(\mathcal{N}/\delta)/N_{\min,t}(s)$ rates; all abstraction, warmup, and drift effects are bundled into $\mathrm{Bias}_t(s;\delta)$. A complete statement and proof are deferred to Appendix C.2.

Soft-BoN approximation to the tilted distribution. Recall that the exact DPRM reveal law is the Gibbs rule in (3). In practice, we approximate this law stagewise: at each visited state, we draw a shortlist of candidate actions from the base proposal and apply Soft-BoN reweighting with the exact process reward $R_t^\star(\cdot;s)$.

Let ν_β denote the terminal distribution induced by the exact DPRM Gibbs reveal law, and let $\widehat{\nu}_N$ denote the terminal distribution induced by the stagewise Soft-BoN approximation with uniform shortlist size N.

Theorem 3.3 (Soft-BoN terminal approximation)**.** *Assume that the reveal horizon satisfies $T<\infty$, each candidate set $\mathcal{A}(s)$ is finite, and the terminal reward satisfies $R(X_T)\in[0,1]$. Then*

$$\mathbb{E}[\mathrm{KL}(\nu_\beta \,\|\, \widehat{\nu}_N)] \le \frac{T\,\sinh(\beta/2)^2}{N}. \tag{12}$$

Hence the terminal approximation error decays at rate $O(1/N)$.

This gives the first approximation layer in the theory: even if we cannot realize the exact Gibbs reveal law, a stagewise Soft-BoN approximation converges to it at the natural $1/N$ rate. The appendix proves a stronger pathwise KL statement and then derives (12) by marginalization; see Appendix C.3.

Finite-sample optimization advantage. Proposition 3.1 shows that admissible token orders do not change the population optimum, so the relevant issue is finite-sample optimization rather than asymptotic consistency. The assumption below packages three regularities that are both theoretically-friendly and empirically plausible. First, importance-sampling SGD theory shows that variance-optimal sampling is proportional to local gradient scale, so it is natural to model confidence as a proxy for a local importance score (Zhao & Zhang, 2015; El Hanchi et al., 2022; Chen et al., 2023). Second, PUMA argues that train–test aligned masking concentrates optimization on a much smaller structured family than random masking (Kim et al., 2026). Third, Fang et al. (2026) show that confidence-based decoding can improve myopic quality while imposing a formal entropy cap, which makes late-stage exploration bottlenecks a natural concern.

Assumption 3.4 (Informal)**.** At each stage t, training operates on an aligned local order family. Early in training, confidence is a κ_t-accurate proxy for the local importance score, and most of the gradient mass lies on an exponentially smaller subfamily, quantified by a difficulty parameter d_t. Later, there remains a residual order family that is still necessary to drive the forward KL to zero, but confidence-only training visits it with exponentially small probability, quantified by a bottleneck parameter h_t; moreover, this residual family has a positive exact-DPRM score gap over its complement.

Appendix B.10 reports instrumented DMPO-Countdown reruns designed to probe observable consequences of these assumptions. These reruns log confidence bins, selected tokens, CE loss, true-token log-probability, CE-gradient proxies, reward, and DPRM scores during training. For any $\varepsilon>0$, let $T_t^{\mathrm{rand}}(\varepsilon)$ and $T_t^{\mathrm{conf}}(\varepsilon)$ denote the numbers of early-stage updates needed to reduce the stagewise forward KL below ε under random aligned-order training and confidence-driven progressive training, respectively. Likewise, let $T_{t,\mathrm{late}}^{\mathrm{conf}}(\varepsilon)$ and $T_{t,\mathrm{late}}^{\mathrm{DPRM}}(\varepsilon)$ denote the numbers of late-stage updates needed to reduce the remaining forward KL below ε under confidence-only and DPRM-corrected training. All algorithms considered here preserve the train–test alignment condition in Proposition 3.1. Full assumptions, theoretical statements and proofs are delayed to Appendix C.4.

Theorem 3.5 (Informal)**.** *Under Assumption 3.4 and standard stagewise smoothness and PL conditions, Progressive Online DPRM enjoys two sample-complexity advantages.*

- Early stage: $T_t^{\mathrm{conf}}(\varepsilon) = \widetilde{O}\Big(\frac{\kappa_t^2}{\varepsilon}\Big)$, $T_t^{\mathrm{rand}}(\varepsilon) = \widetilde{\Omega}\Big(\frac{e^{a_t d_t}}{\varepsilon}\Big)$ *for some stage-difficulty constant $a_t>0$. Thus confidence-driven progressive training is exponentially faster than random-order training.*
- Late stage: *if the warmup time T_{warm} is chosen so that the stage-2 score error is below one half of the exact DPRM gap on the residual family, then $T_{t,\mathrm{late}}^{\mathrm{DPRM}}(\varepsilon) = \widetilde{O}\big(\log\frac{1}{\varepsilon}\big)$, $T_{t,\mathrm{late}}^{\mathrm{conf}}(\varepsilon) = \widetilde{\Omega}\big(e^{b_t h_t}\log\frac{1}{\varepsilon}\big)$ for some bottleneck constant $b_t>0$. Thus DPRM is exponentially faster than confidence-only training in the late stage.*

Overall, Progressive Online DPRM can achieve an exponential speedup twice: first over random aligned-order training,

Table 2. Natural-language ordering results across pretraining, reasoning post-training, and test-time scaling. Each comparison changes only the token-ordering controller whenever possible.

(a) PUMA versus DPRM-PUMA on GSM8K at EMA checkpoint step 1.53M.

Method	Unmask-2	Unmask-3	Mean
PUMA (confidence)	29.95	28.73	29.34
DPRM-PUMA	**34.12**	**34.42**	**34.27**

(b) DPRM-Prism on GSM8K with LLaDA-2.0-mini.

Method	Voted	Rank-1	Any-4
Prism	82.41	82.11	84.61
DPRM-Prism	**83.85**	**83.70**	**86.58**

(c) Arithmetic mean of pass@K over $K \in \{1, 2, 4, 8, 16, 32\}$ on LLaDA-8B-Instruct post-training runs.

Model	GSM8K	GSM8K Hard	MATH	MATH Hard	Countdown	Countdown Hard
Base Model	75.8	60.9	49.2	44.0	31.9	8.4
DMPO	88.1	73.6	47.1	43.0	7.8	0.0
Progressive DMPO	**88.7**	74.4	50.4	44.3	53.4	29.6
DMPO-DPRM	88.6	**75.3**	**52.5**	**47.9**	**55.0**	**33.4**

For Prism, mean NFE increases from 609 to 1,071 while SVF calls remain 29. The full Prism table is reported in Appendix B.4.

and then over confidence-only progressive training.

4. Experiments

We evaluate DPRM as a *plug-in token-ordering module* in seven host systems spanning language pretraining, reasoning post-training, test-time scaling, and scientific generation. The hosts are PUMA (Kim et al., 2026) for masked-diffusion pretraining, DMPO (Zhu et al., 2025c) for reasoning post-training, Prism (Bai et al., 2026) for test-time scaling, DPLM-2 Bit (Bhattacharya et al., 2026) for multimodal protein inverse folding, DCM (Lee et al., 2025) for single-cell gene-expression diffusion, GenMol V2 for molecular SAFE diffusion, and SDPO (Wang et al., 2025a) for DNA diffusion. In every case, we keep the host model, objective, reward oracle, and data pipeline fixed as much as possible, and modify only the token-ordering controller. The main text focuses on the empirical effect of this intervention; task utilities, controller schedules, preprocessing details, and implementation choices are deferred to Tabs. 4 and 5 and Secs. B.2 to B.8. For non-natural-language domains, we additionally test a random-to-DPRM variant. The motivation is that early random masking may encourage broader exploration before reward-guided ordering becomes reliable, which may be useful when the data do not have a natural linguistic order.

4.1. DPRM on Natural Language

We first evaluate DPRM across the natural-language pipeline: masked-diffusion pretraining, reward-aware post-training, and test-time scaling. These settings use different hosts and evaluation protocols, but share the same intervention principle: keep the host model, data pipeline, objective or search scaffold fixed, and replace only the token-ordering controller. Implementation details and controller schedules are deferred to Secs. B.2 to B.4.

Tab. 2 shows that DPRM improves ordering across all three stages. In pretraining, DPRM-PUMA raises the GSM8K validation mean from 29.34 to 34.27, showing that confidence ordering remains improvable even after PUMA has already aligned train-time and test-time masks. In post-training, DMPO-DPRM gives the clearest gains on harder and more exploratory reasoning settings, improving MATH Hard from 44.3 to 47.9 and Countdown Hard from 29.6 to 33.4, while remaining essentially tied on the easier GSM8K average. At inference time, DPRM-Prism improves voted, rank-1, and any-of-4 GSM8K accuracy under the same Prism search scaffold, at the cost of additional ordering computation. Together, these results support DPRM as a reusable ordering intervention for natural-language diffusion models rather than a host-specific trick.

4.2. DPRM on Scientific Data

We next test whether token ordering remains consequential beyond natural language, using protein inverse folding, single-cell gene-expression diffusion, molecular SAFE diffusion, and DNA reward optimization. These domains often lack a canonical left-to-right semantic order, so we additionally include random-to-DPRM variants for the scientific hosts. The motivation is exploratory: early random masking may expose a broader set of partial states before reward-guided ordering becomes reliable. Apart from the ordering controller, the host architecture, representation, objective, data pipeline, and evaluation protocol are kept fixed whenever possible; detailed settings are deferred to Secs. B.5 to B.8.

Tab. 3 shows that scientific-domain ordering is high-leverage but more multi-objective than the natural-language cases. DPLM ordering variants improve forward folding over the DPLM-2 Bit baseline, while co-generation metrics reveal trade-offs among RMSD, TM-score, pLDDT, and designability. DCM is the cleanest scientific-domain gain:

Table 3. Scientific-domain ordering results. Each subtable reports a matched plug-in intervention in which the host architecture, objective, and data pipeline are kept fixed as much as possible, and only the token-ordering controller is changed.

(a) DPLM-2 Bit under the evaluation protocol of Hsieh et al. (2025). Progressive DPLM-2 Bit uses confidence-aligned progressive ordering; DPRM variants replace only the token-ordering controller.

Method	FF RMSD ↓	FF TM ↑	CoGen RMSD ↓	CoGen TM ↑	CoGen pLDDT ↑	Designable (%) ↑
DPLM-2 Bit	35.47	0.3071	**66.73**	0.4063	67.49	23.6
Progressive DPLM-2 Bit	**29.43**	**0.3321**	91.64	**0.4621**	**76.51**	**40.4**
DPRM-DPLM-2 Bit	**29.43**	**0.3321**	75.70	0.4605	68.77	40.0
DPRM(random)-DPLM-2 Bit	**29.43**	**0.3321**	80.23	0.4527	74.09	39.6

(b) DPRM-DCM on Dentate Gyrus validation cells. Means use 95% bootstrap intervals over 293 held-out cells. Token recovery and zero accuracy are percentages; MAE is over discretized expression bins.

Method	Token recovery (%) ↑	MAE ↓	Zero accuracy (%) ↑
DCM-random	63.97 [63.36, 64.54]	0.821 [0.805, 0.838]	78.39 [78.09, 78.70]
Progressive-DCM	75.83 [74.83, 76.79]	0.655 [0.622, 0.689]	99.77 [99.76, 99.78]
DPRM(random)-DCM	**75.92** [74.87, 76.87]	**0.654** [0.621, 0.688]	**99.90** [99.90, 99.91]
DPRM(conf.)-DCM	75.85 [74.83, 76.80]	0.663 [0.630, 0.697]	99.88 [99.88, 99.88]

(c) GenMol V2 ordering pilot. De novo metrics use 1,000 generated molecules per method with 95% bootstrap intervals; fragment metrics are mean values over the stable fragment subset. Higher is better.

Method	De novo validity	De novo quality	De novo uniqueness	De novo diversity	Linker validity	Scaffold quality
GenMol V2	0.984 [0.976, 0.991]	**0.854** [0.833, 0.876]	**0.582** [0.562, 0.601]	0.828 [0.825, 0.832]	0.142	0.429
Progressive-GenMol V2	0.990 [0.983, 0.996]	0.576 [0.546, 0.607]	0.480 [0.460, 0.499]	**0.853** [0.849, 0.857]	0.142	**0.712**
DPRM(random)-GenMol V2	**0.997** [0.993, 1.000]	0.829 [0.806, 0.852]	0.364 [0.347, 0.381]	0.778 [0.772, 0.784]	**0.429**	**0.712**
DPRM(conf.)-GenMol V2	0.968 [0.957, 0.978]	0.603 [0.573, 0.633]	0.313 [0.296, 0.330]	0.810 [0.804, 0.817]	0.142	**0.712**

(d) SDPO ordering comparison on the GOSAI DNA setting of Wang et al. (2025a). Metrics use 640 generated DNA samples per method with 95% bootstrap intervals. Higher is better; log-likelihood is better when less negative.

Method	Total	HepG2	ATAC	K-mer Pearson	Log lik.
SDPO	**1.1903** [1.0554, 1.3305]	3.9482 [3.8420, 4.0602]	**0.3590** [0.3219, 0.3954]	**0.8397** [0.8332, 0.8469]	-238.4408 [-239.0338, -237.8899]
Progressive-SDPO	0.1857 [0.1325, 0.2428]	**4.6042** [4.5467, 4.6620]	0.0746 [0.0531, 0.0969]	0.5405 [0.5333, 0.5477]	**-212.8601** [-213.7717, -211.9579]
DPRM-SDPO	0.8925 [0.7733, 1.0070]	4.0644 [3.9713, 4.1628]	0.3093 [0.2719, 0.3453]	0.7096 [0.7023, 0.7168]	-232.5601 [-233.1347, -231.9712]
DPRM(random)-SDPO	0.9123 [0.7906, 1.0281]	4.1156 [4.0196, 4.2133]	0.3108 [0.2734, 0.3453]	0.7131 [0.7059, 0.7201]	-232.5827 [-233.2017, -231.9412]

DPRM(random)-DCM improves token recovery, MAE, and zero-expression accuracy over random ordered masking. GenMol and SDPO further show that changing token order can improve selected fragment-constrained, structural, accessibility, or reward-specific metrics, but no single generic ordering rule dominates every quality measure. These results suggest that DPRM transfers beyond language as a plug-in control mechanism, while scientific applications may require domain-specific reward design and ordering schedules to choose the desired point on the trade-off surface.

5. Conclusion, Limitation and Future Work

We introduced DPRM, a plug-in token-ordering module for diffusion language models and adjacent masked discrete diffusion systems. DPRM treats token order as a reusable control variable: it starts from confidence-guided ordering and gradually incorporates online process-reward guidance to exact DPRM. Theoretically, we characterize the exact DPRM policy as a reward-tilted Gibbs reveal law, prove $O(1/N)$ convergence of its stagewise Soft-BoN approximation, show that the online bucketized controller tracks the exact DPRM score at standard concentration rates, and establish a sample-complexity advantage under tractable assumptions. Empirically, DPRM improves or reshapes the performance frontier across natural-language pretraining, post-training and test-time scaling, as well as protein, single-cell, molecular and DNA discrete diffusion tasks, while changing only the ordering controller whenever possible. The main limitations are that our online controller uses coarse bucketized estimates rather than exact Doob guidance, and our optimization theory relies on assumptions that are only indirectly supported by diagnostics such as CE loss, true-token log-probability and CE-gradient proxies (see Appendix B.10). Future work should develop tighter theoretical guarantees and richer domain-specific process rewards. It should also further validate DPRM beyond natural and scientific-language domains, including visual discrete diffusion and masked generation for images (Austin et al., 2021; Gu et al., 2022; Chang et al., 2022), discrete or masked diffusion models for video generation and control (Deng et al., 2026; Jain et al., 2024), and predictive representation-learning frameworks such as I-JEPA and V-JEPA, where the choice of context and target regions also plays a central role (Assran et al., 2023; Bardes et al., 2024).

References

Assran, M., Duval, Q., Misra, I., Bojanowski, P., Vincent, P., Rabbat, M., LeCun, Y., and Ballas, N. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15619–15629, 2023.

Austin, J., Johnson, D. D., Ho, J., Tarlow, D., and van den Berg, R. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, pp. 17981–17993, 2021.

Avdeyev, P., Shi, C., Tan, Y., Dudnyk, K., and Zhou, J. Dirichlet diffusion score model for biological sequence generation. In *International Conference on Machine Learning*, pp. 1276–1301. PMLR, 2023.

Avsec, Ž., Agarwal, V., Visentin, D., Ledsam, J. R., Grabska-Barwinska, A., Taylor, K. R., Assael, Y., Jumper, J., Kohli, P., and Kelley, D. R. Effective gene expression prediction from sequence by integrating long-range interactions. *Nature Methods*, 18(10):1196–1203, 2021. doi: 10.1038/s41592-021-01252-x.

Bai, J., Li, Y., Zhu, Y., Xin, Y., Shi, Q., Feng, A., Liu, X., Tao, M., Xue, J., Li, X., and Yang, M.-H. PRISM: Efficient test-time scaling via hierarchical search and self-verification for discrete diffusion language models. *arXiv preprint arXiv:2602.01842*, 2026.

Bardes, A., Garrido, Q., Ponce, J., Chen, X., Rabbat, M., LeCun, Y., Assran, M., and Ballas, N. Revisiting feature prediction for learning visual representations from video. *Transactions on Machine Learning Research*, 2024.

Besbes, O., Gur, Y., and Zeevi, A. Non-stationary stochastic optimization. *Operations research*, 63(5):1227–1244, 2015.

Bhattacharya, S., Gensbigler, C., Karim, S., and Lees, J. Discrete diffusion for single-cell gene expression modeling. MLGenX 2026 poster / bioRxiv preprint, 2026. URL https://openreview.net/forum?id=GPR1YXdE4U.

Bie, T., Cao, M., Chen, K., Du, L., Gong, M., Gong, Z., Gu, Y., Hu, J., Huang, Z., Lan, Z., et al. Llada2. 0: Scaling up diffusion language models to 100b. *arXiv preprint arXiv:2512.15745*, 2025.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Bu, D., Huang, W., Suzuki, T., Cheng, J., Zhang, Q., Xu, Z., and Wong, H.-S. Provably neural active learning succeeds via prioritizing perplexing samples. *arXiv preprint arXiv:2406.03944*, 2024.

Bu, D., Huang, W., Han, A., Nitanda, A., Wong, H.-S., Zhang, Q., and Suzuki, T. Provable benefit of curriculum in transformer tree-reasoning post-training. *arXiv preprint arXiv:2511.07372*, 2025a.

Bu, D., Huang, W., Han, A., Nitanda, A., Xue, B., Zhang, Q., Wong, H.-S., and Suzuki, T. Post-training as reweighting: A stochastic view of reasoning trajectories in language models. *arXiv preprint arXiv:2511.07368*, 2025b.

Bunne, C., Stark, S. G., Gut, G., del Castillo, J. S., Levesque, M., Lehmann, K.-V., Pelkmans, L., Krause, A., and Rätsch, G. Learning single-cell perturbation responses using neural optimal transport. *Nature Methods*, 20(11):1759–1768, 2023. doi: 10.1038/s41592-023-01969-x.

Cai, C. and Li, G. Confidence-based decoding is provably efficient for diffusion language models. *arXiv preprint arXiv:2603.22248*, 2026.

Chang, H., Zhang, H., Jiang, L., Liu, C., and Freeman, W. T. MaskGIT: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11315–11325, 2022.

Chen, Y., Georgiou, T. T., and Pavon, M. Stochastic control liaisons: Richard Sinkhorn meets Gaspard Monge on a Schrödinger bridge. *SIAM Review*, 63(2):249–313, 2021. doi: 10.1137/20M1339982. URL https://doi.org/10.1137/20M1339982.

Chen, Z., Lu, J., Qian, H., Wang, X., and Yin, W. Hetersgd: Tackling heterogeneous sampling costs via optimal reweighted stochastic gradient descent. In Ruiz, F., Dy, J., and van de Meent, J.-W. (eds.), *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pp. 10732–10781. PMLR, 2023. URL https://proceedings.mlr.press/v206/chen23i.html.

Cui, H., Wang, C., Maan, H., Pang, K., Luo, F., Duan, N., and Wang, B. scgpt: toward building a foundation model

for single-cell multi-omics using generative ai. *Nature Methods*, 21:1470–1480, 2024. doi: 10.1038/s41592-024-02201-0.

DaSilva, L. F., Senan, S., Kribelbauer-Swietek, J. F., Patel, Z. M., Louis, L. K., Reddy, A. J., Gabbita, S., Rosen, J. D., Nussbaum, Z., Córdova, C. M. V., et al. Designing synthetic regulatory elements using the generative AI framework DNA-Diffusion. *Nature Genetics*, 58:180–194, 2026. doi: 10.1038/s41588-025-02441-6.

de Almeida, B. P., Reiter, F., Pagani, M., and Stark, A. DeepSTARR predicts enhancer activity from DNA sequence and enables the de novo design of synthetic enhancers. *Nature Genetics*, 54(5):613–624, 2022. doi: 10.1038/s41588-022-01048-5.

Deng, H., Pan, T., Zhang, F., Liu, Y., Luo, Z., Cui, Y., Wang, W., Shen, C., Shan, S., Zhang, Z., and Wang, X. Uniform discrete diffusion with metric path for video generation. In *International Conference on Learning Representations*, 2026.

Domingo-Enrich, C., Han, J., Amos, B., Bruna, J., and Chen, R. T. Q. Stochastic optimal control matching. In Globerson, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J., and Zhang, C. (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 112459–112504. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/cc32ec39a5073f61d38c338d963df30d-Paper-Conference.pdf.

Doob, J. L. Conditional brownian motion and the boundary limits of harmonic functions. *Bulletin de la Société Mathématique de France*, 85:431–458, 1957.

El Hanchi, A., Stephens, D. A., and Maddison, C. J. Stochastic reweighted gradient descent. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 8359–8374. PMLR, 2022. URL https://proceedings.mlr.press/v162/hanchi22a.html.

Fang, L., Liu, A., Zou, H. P., Chen, Y., Ma, E., Pan, L., Miao, C., Huang, W.-C., Liu, X., and Yu, P. S. Locally confident, globally stuck: The quality-exploration dilemma in diffusion language models. *arXiv preprint arXiv:2604.00375*, 2026.

Gayoso, A., Steier, Z., Lopez, R., Regier, J., Nazor, K. L., Streets, A., and Yosef, N. Joint probabilistic modeling of single-cell multi-omic data with totalvi. *Nature Methods*, 18(3):272–282, 2021. doi: 10.1038/s41592-020-01050-x.

Gong, S., Zhang, R., Zheng, H., Gu, J., Jaitly, N., Kong, L., and Zhang, Y. DiffuCoder: Understanding and improving masked diffusion models for code generation. *arXiv preprint arXiv:2506.20639*, 2025.

Gower, R. M., Loizou, N., Qian, X., Sailanbayev, A., Shulgin, E., and Richtárik, P. SGD: General analysis and improved rates. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5200–5209. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/qian19b.html.

Gruver, N., Stanton, S. D., Frey, N. C., Rudner, T. G. J., Hotzel, I., Lafrance-Vanasse, J., Rajpal, A., Cho, K., and Wilson, A. G. Protein design with guided discrete diffusion. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=MfiK69Ga6p.

Gu, S., Chen, D., Bao, J., Wen, F., Zhang, B., Chen, D., Yuan, L., and Guo, B. Vector quantized diffusion model for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10696–10706, 2022.

Guo, W., Zhu, Y., Tao, M., and Chen, Y. Plug-and-play controllable generation for discrete masked models. *arXiv preprint arXiv:2410.02143*, 2024.

Hao, M., Gong, J., Zeng, X., Liu, C., Guo, Y., Cheng, X., Wang, T., Ma, J., Zhang, X., and Song, L. Large-scale foundation model on single-cell transcriptomics. *Nature Methods*, 21:1481–1491, 2024. doi: 10.1038/s41592-024-02305-7.

Hong, F., Yu, G., Ye, Y., Huang, H., Zheng, H., Zhang, Y., Wang, Y., and Yao, J. Wide-in, narrow-out: Revokable decoding for efficient and effective DLLMs. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=XtLQHlNLxy.

Hsieh, C.-Y., Wang, X., Zhang, D., Xue, D., Ye, F., Huang, S., Zheng, Z., and Gu, Q. Elucidating the design space of multimodal protein language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, 2025.

Hu, Z., Meng, J., Akhauri, Y., Abdelfattah, M. S., Seo, J.-s., Zhang, Z., and Gupta, U. Accelerating diffusion language model inference via efficient KV caching and guided diffusion. *arXiv preprint arXiv:2505.21467*, 2025.

Israel, D. M., Jin, T., Cheng, E. Y., Van den Broeck, G., Grover, A., Subramanian, S., and Carbin, M. Planned diffusion. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=wZN8debH4W.

Jain, Y., Nasery, A., Vineet, V., and Behl, H. Peekaboo: Interactive video generation via masked-diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8079–8088, 2024.

Kawata, R., Oko, K., Nitanda, A., and Suzuki, T. Direct distributional optimization for provable alignment of diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Kim, J., Shah, K., Kontonis, V., Kakade, S. M., and Chen, S. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=DjJmre5IkP.

Kim, J., Geuter, J., Alvarez-Melis, D., Kakade, S., and Chen, S. Stop training for the worst: Progressive unmasking accelerates masked diffusion training. *arXiv preprint arXiv:2602.10314*, 2026.

Lee, S., Kreis, K., Veccham, S. P., Liu, M., Reidenbach, D., Peng, Y., Paliwal, S., Nie, W., and Vahdat, A. GenMol: A drug discovery generalist with discrete diffusion. *arXiv preprint arXiv:2501.06158*, 2025.

Lee, S., Kim, S., Park, J., and Park, D. Lookahead unmasking elicits reliable decoding in diffusion language models. *OpenReview*, 2026. URL https://openreview.net/forum?id=SVI1ZnmFmx. Submitted to ICLR 2026.

Léonard, C. A survey of the Schrödinger problem and some of its connections with optimal transport. *Discrete and Continuous Dynamical Systems - Series A*, 34(4):1533–1574, 2014. URL https://hal.science/hal-00849930.

Liu, S., Nam, J., Campbell, A., Stark, H., Xu, Y., Jaakkola, T., and Gomez-Bombarelli, R. Think while you generate: Discrete diffusion with planned denoising. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=MJNywBdSDy.

Liu, Z., Yang, Y., Zhang, Y., Chen, J., Zou, C., Wei, Q., Wang, S., and Zhang, L. dLLM-Cache: Accelerating diffusion large language models with adaptive caching. *arXiv preprint arXiv:2506.06295*, 2025b.

Lopez, R., Regier, J., Cole, M. B., Jordan, M. I., and Yosef, N. Deep generative modeling for single-cell transcriptomics. *Nature Methods*, 15(12):1053–1058, 2018. doi: 10.1038/s41592-018-0229-2.

Lotfollahi, M., Wolf, F. A., and Theis, F. J. scgen predicts single-cell perturbation responses. *Nature Methods*, 16 (8):715–721, 2019. doi: 10.1038/s41592-019-0494-8.

Lotfollahi, M., Susmelj, A. K., De Donno, C., Ji, Y., Ibarra, I. L., Wolf, F. A., Yakubova, N., Theis, F. J., and Lopez-Paz, D. Compositional perturbation autoencoder for single-cell response modeling, 2021.

Luo, E., Hao, M., Wei, L., and Zhang, X. scdiffusion: conditional generation of high-quality single-cell data using diffusion model. *Bioinformatics*, 40(9):btae518, 2024. doi: 10.1093/bioinformatics/btae518.

Ma, X., Yu, R., Fang, G., and Wang, X. dKV-Cache: The cache for diffusion language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=Gppo2JImHs.

Maurer, A. and Pontil, M. Empirical bernstein bounds and sample variance penalization. *arXiv preprint arXiv:0907.3740*, 2009.

Nie, S., Wang, W., Ren, Z., Mao, B., Zhang, B., Liu, Z., Yin, S., Feng, C., Yin, M., Wang, Z., et al. Large language diffusion models. *arXiv preprint arXiv:2502.09992*, 2025.

Roohani, Y., Huang, K., and Leskovec, J. Predicting transcriptional outcomes of novel multigene perturbations with gears. *Nature Biotechnology*, 42:927–935, 2024. doi: 10.1038/s41587-023-01905-6.

Sahoo, S. S., Arriola, M., Gokaslan, A., Marroquin, E. M., Rush, A. M., Schiff, Y., Chiu, J. T., and Kuleshov, V. Simple and effective masked diffusion language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=L4uaAR4ArM.

Shi, C., Xu, M., Zhu, Z., Zhang, W., Zhang, M., and Tang, J. Graphaf: a flow-based autoregressive model for molecular graph generation. *arXiv preprint arXiv:2001.09382*, 2020.

Verdun, C. M., Oesterling, A., Lakkaraju, H., and Calmon, F. P. Soft best-of-n sampling for model alignment. In *2025 IEEE International Symposium on Information Theory (ISIT)*, pp. 1–6. IEEE, 2025.

Vignac, C., Krawczuk, I., Siraudin, A., Wang, B., Cevher, V., and Frossard, P. Digress: Discrete denoising diffusion for graph generation. *arXiv preprint arXiv:2209.14734*, 2022.

Wang, C., Uehara, M., He, Y., Wang, A., Lal, A., Jaakkola, T., Levine, S., Regev, A., Hanchen, and Biancalani, T. Fine-tuning discrete diffusion models via reward optimization with applications to DNA and protein design. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=G328D1xt4W.

Wang, X., Zheng, Z., Ye, F., Xue, D., Huang, S., and Gu, Q. Diffusion language models are versatile protein learners. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 52309–52333. PMLR, 21–27 Jul 2024a. URL https://proceedings.mlr.press/v235/wang24ct.html.

Wang, X., Zheng, Z., Ye, F., Xue, D., Huang, S., and Gu, Q. DPLM-2: A multimodal diffusion protein language model. *arXiv preprint arXiv:2410.13782*, 2024b.

Wang, X., Zheng, Z., Ye, F., Xue, D., Huang, S., and Gu, Q. Dplm-2: Multimodal diffusion protein language models. *arXiv preprint arXiv:2502.10634*, 2025b.

Xu, M., Yu, L., Song, Y., Shi, C., Ermon, S., and Tang, J. Geodiff: A geometric diffusion model for molecular conformation generation. *arXiv preprint arXiv:2203.02923*, 2022.

Yang, Z., Liu, H., Cao, C., and Su, B. D3LM: A discrete DNA diffusion language model for bidirectional DNA understanding and generation, 2026. URL https://arxiv.org/abs/2603.01780.

Ye, J., Gong, S., Chen, L., Zheng, L., Gao, J., Shi, H., Wu, C., Jiang, X., Li, Z., Bi, W., and Kong, L. Diffusion of thought: Chain-of-thought reasoning in diffusion language models. In *Advances in Neural Information Processing Systems*, 2025.

Zhao, P. and Zhang, T. Stochastic optimization with importance sampling for regularized loss minimization. In Bach, F. and Blei, D. (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1–9. PMLR, 2015. URL https://proceedings.mlr.press/v37/zhaoa15.html.

Zhao, S., Gupta, D., Zheng, Q., and Grover, A. d1: Scaling reasoning in diffusion large language models via reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025a. URL https://openreview.net/forum?id=7ZVRlBFuEv.

Zhao, S., Liu, M., Huang, J., Liu, M., Wang, C., Liu, B., Tian, Y., Pang, G., Bell, S., Grover, A., and Chen, F. Inpainting-guided policy optimization for diffusion large language models. *arXiv preprint arXiv:2509.10396*, 2025b.

Zhu, F., Wang, R., Nie, S., Zhang, X., Wu, C., Hu, J., Zhou, J., Chen, J., Lin, Y., Wen, J.-R., et al. LLaDA 1.5: Variance-reduced preference optimization for large language diffusion models. *arXiv preprint arXiv:2505.19223*, 2025a.

Zhu, Q., Ye, Z., Liu, H., Wang, Z., and Chen, M. Training-free adaptation of diffusion models via doob's h-transform. *arXiv preprint arXiv:2602.16198*, 2026.

Zhu, Y., Guo, W., Choi, J., Liu, G.-H., Chen, Y., and Tao, M. MDNS: Masked diffusion neural sampler via stochastic optimal control. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=xIH95kXNR2.

Zhu, Y., Guo, W., Choi, J., Molodyk, P., Yuan, B., Tao, M., and Chen, Y. Enhancing reasoning for diffusion LLMs via distribution matching policy optimization. *arXiv preprint arXiv:2510.08233*, 2025c.

Zrimec, J., Fu, X., Muhammad, A. S., Skrekas, C., Jauniskis, V., Speicher, N. K., Börlin, C. S., Verendel, V., Chehreghani, M. H., Dubhashi, D., et al. Controlling gene expression with deep generative design of regulatory DNA. *Nature Communications*, 13:5099, 2022. doi: 10.1038/s41467-022-32818-8.

A. Additional Related Work

Train–test alignment and token ordering in masked diffusion. Recent work has made clear that token ordering is not a minor implementation detail in masked diffusion models. Kim et al. (2025) show that inference-time adaptive ordering can substantially change performance even when training follows the standard masking setup. Kim et al. (2026) sharpen this point by identifying the mismatch between random training masks and the structured masks induced by progressive inference, and propose teacher-forced progressive unmasking to align the two. We build directly on this viewpoint, but argue that train–test alignment alone does not solve the reveal-order problem. Once random masking is abandoned, confidence top-k remains a computationally convenient heuristic rather than the true future-reward-aware order.

Confidence-based decoding, planning, and exploration in dLLMs. A growing line of work studies denoising order and remasking policy as central ingredients of dLLM decoding. On the positive side, Cai & Li (2026) provide the first formal efficiency guarantees for confidence-based decoding, showing that confidence can be provably effective in low-entropy regimes. On the negative side, Fang et al. (2026) show that low-confidence remasking improves single-sample quality while suppressing exploration and limiting pass@K gains, creating a quality–exploration dilemma. Related work has therefore introduced additional planning or verification over unmasking paths, including planned denoising (Liu et al., 2025a), lookahead path selection (Lee et al., 2026), revokable draft-and-verify decoding (Hong et al., 2026), and learned denoising-order planning (Israel et al., 2026). We share the diagnosis that local confidence is not the whole story, but take a different route: DPRM is designed as a generic reward-aware ordering module that can intervene during training as well as at inference, rather than as a purely decoding-time planner or verifier.

Trajectory reweighting, guided sampling, and Doob-style control. Our perspective is also related to work that interprets alignment or guidance as trajectory reweighting. Bu et al. (2025b) view post-training as reweighting over stochastic reasoning trajectories, which motivates using future trajectory value rather than local likelihood alone. At a more classical level, Doob transforms and related control formulations provide a principled way to tilt Markov processes toward terminal events or rewards (Doob, 1957; Léonard, 2014; Chen et al., 2021). Similar ideas have recently reappeared in diffusion and neural-sampling settings, including direct distributional optimization, training-free Doob-style adaptation, and stochastic-optimal-control formulations of guided generation (Kawata et al., 2025; Zhu et al., 2026; Domingo-Enrich et al., 2024; Zhu et al., 2025b). DPRM specializes this line of thought to the token-ordering problem in diffusion language models, and develops an online bucketized approximation that is lightweight enough to act as a practical plug-in controller.

RL and post-training for diffusion LLMs. A recent line of work adapts post-training and reinforcement learning from autoregressive LLMs to diffusion LLMs. Zhao et al. (2025a) introduce diffu-GRPO for masked dLLM reasoning, while Gong et al. (2025) study RL-style improvement in masked diffusion models for code generation. Zhao et al. (2025b) exploit inpainting as a diffusion-specific exploration mechanism during RL, and Zhu et al. (2025a) study preference-style post-training for LLaDA. DMPO takes a different route by formulating post-training as matching a reward-tilted target distribution through a forward denoising objective (Zhu et al., 2025c). Our work is orthogonal to this target-design question: in DPRM-DMPO, we keep the DMPO target fixed and intervene only on the masked-state ordering policy used to realize it.

Planning, scheduling, and revision in diffusion-language-model decoding. Beyond confidence heuristics, several recent works explicitly redesign the denoising schedule. Liu et al. (2025a) introduce a planner–denoiser decomposition in which a planner selects which positions should be denoised next. Israel et al. (2026) similarly train a model to determine its own denoising order through an explicit planning stage. Lee et al. (2026) reformulate decoding as path selection over candidate unmasking orders, and Hong et al. (2026) introduce revokable draft-and-verify decoding that re-masks suspicious tokens during generation. These methods reinforce the same broader point as our paper: denoising order is a real algorithmic design dimension in dLLMs. Our focus is to provide a generic reward-aware ordering module with both theoretical guarantees and host-level transfer, rather than a single decoding procedure specialized to one inference setting.

Test-time scaling and inference-time control in diffusion LMs. Prism shows that hierarchical search, local branching, and self-verification can substantially improve test-time scaling for discrete diffusion LMs (Bai et al., 2026). More broadly, recent dLLM work has begun to address inference-time efficiency through adaptive caching, KV reuse, and decoding-time control (Liu et al., 2025b; Ma et al., 2025; Hu et al., 2025). Our view is complementary: once a search or inference scaffold is fixed, there remains a distinct ordering question about which tokens or trajectories should be committed, revised, or expanded under a finite budget. DPRM-Prism targets precisely this layer, while leaving the surrounding search framework

intact.

Protein diffusion and multimodal protein language models. Diffusion models have become increasingly competitive in biomolecular sequence generation and protein design (Avdeyev et al., 2023; Gruver et al., 2023; Wang et al., 2024a). DPLM established diffusion language models as strong protein learners, and DPLM-2 extended this direction to multimodal sequence–structure modeling (Wang et al., 2025b). Hsieh et al. (2025) further show that performance in multimodal protein language models depends strongly on tokenization and structural prediction design choices. Our contribution in this domain is intentionally conservative: we do not redesign the multimodal architecture, tokenizer, or denoising loss, and instead isolate the effect of changing only the sequence ordering controller.

Single-cell gene-expression modeling. Single-cell generative modeling has developed along several complementary lines. Probabilistic latent-variable models such as scVI and totalVI model expression counts through continuous latent representations and have become standard tools for uncertainty-aware representation learning, batch correction, and multi-omic integration (Lopez et al., 2018; Gayoso et al., 2021). A second line focuses on perturbation-response prediction, including latent-space perturbation models such as scGen and CPA, neural optimal transport methods such as CellOT, and knowledge-informed models such as GEARS for genetic perturbations (Lotfollahi et al., 2019; 2021; Bunne et al., 2023; Roohani et al., 2024). More recently, large single-cell foundation models such as scGPT and scFoundation have treated gene-expression profiles as structured high-dimensional inputs for transferable representation learning and downstream perturbation prediction (Cui et al., 2024; Hao et al., 2024). Diffusion-based approaches have also entered this space, but most existing methods, such as scDiffusion, operate through continuous latent embeddings rather than directly over discrete count tokens (Luo et al., 2024).

DCM (Bhattacharya et al., 2026) takes a different route by treating sparse expression profiles as discrete count-bin sequences and training a discrete diffusion model directly in token space. This makes DCM a natural stress test for DPRM beyond ordinary language: genes are not ordered left-to-right, expression profiles are sparse and high-dimensional, and the reveal order determines which expression bins are reconstructed first. Our DPRM-DCM experiment is deliberately narrow and ordering-focused. We keep the DCM architecture, count-bin preprocessing, data split, optimizer, and denoising loss fixed, and change only the masked gene-position reveal policy.

Molecular discrete diffusion for drug design. Molecular generation has also begun to use masked or discrete diffusion over sequence-like chemical representations. GenMol (Lee et al., 2025) applies discrete diffusion to SAFE molecular strings and supports de novo generation, fragment-constrained generation, hit generation, and lead optimization. This setting is useful for DPRM because molecule generation exposes many ordering-dependent decisions over masked SAFE tokens while already providing cheap validity, quality, and fragment-retention metrics. Our DPRM-GenMol pilot therefore isolates the ordering layer inside an otherwise fixed molecular generator, complementing broader work on graph and molecular diffusion models (Shi et al., 2020; Vignac et al., 2022; Xu et al., 2022).

DNA regulatory-sequence modelling. Generative modelling of regulatory DNA has increasingly moved from unconditional sequence generation toward function-aware design. Sequence-to-function predictors such as DeepSTARR and Enformer provide strong oracles for enhancer activity, gene expression, chromatin state and variant effects, making it possible to score candidate regulatory sequences in silico (de Almeida et al., 2022; Avsec et al., 2021). Generative approaches have also begun to directly design regulatory elements, ranging from deep generative models for promoter and enhancer design to diffusion-based frameworks such as DNA-Diffusion, which generates cell-type-specific regulatory elements and validates them with STARR-seq and endogenous perturbation assays (Zrimec et al., 2022; DaSilva et al., 2026). More recently, masked discrete diffusion has been explored as a DNA foundation-model paradigm: D3LM reformulates DNA generation as discrete masked diffusion and reports strong regulatory-element generation performance, further suggesting that non-left-to-right generation is well matched to regulatory sequence design (Yang et al., 2026).

SDPO (Wang et al., 2025a) is especially relevant to our setting because it fine-tunes discrete diffusion models with reward optimization and studies DNA regulatory-sequence generation using oracle scores for expression, chromatin accessibility and sequence-distribution similarity. This setting is a useful additional test for DPRM because the host already exposes reward-oriented DNA metrics while the sequence generator remains a masked discrete diffusion model. Our SDPO pilot is deliberately narrow and ordering-focused: we keep the reward-optimization objective, oracle suite, data preprocessing, optimizer and denoising loss fixed, and test only whether alternative reveal-order controllers change the resulting reward trade-off.

Guidance, controllability, and stochastic-control views of diffusion. Outside language, a broad literature studies guided or controlled diffusion through reward shaping, guidance, or optimal-control formulations. Examples include plug-and-play controllable generation for masked discrete models (Guo et al., 2024), direct distributional optimization for diffusion alignment (Kawata et al., 2025), and training-free Doob-style adaptation (Zhu et al., 2026). Related stochastic-control perspectives also appear in optimal-control matching and diffusion neural samplers (Domingo-Enrich et al., 2024; Zhu et al., 2025b). DPRM differs from these approaches in both granularity and objective: we do not guide the entire generative process by altering the model or sampler globally, but instead learn a lightweight guidance to the local token-ordering rule inside an otherwise fixed diffusion-language-model host.

B. Additional Experiments and Details

B.1. DPRM Utility Oracles and Temperature Settings

DPRM requires a scalar utility for completed or partially completed trajectories. The host algorithm supplies this utility; DPRM only reuses it to update bucketized estimates for ordering. In all experiments the local score has the form

$$g_i(s) = \log \psi_i(s) + \eta_i(s)\, \beta \widehat{R}_{\phi,b_i}, \qquad \widehat{R}_{\phi,b} = \frac{1}{\beta} \log \frac{\sum_{j \in \mathcal{H}_{\phi,b}} \exp(\beta R_j)}{|\mathcal{H}_{\phi,b}|},$$

where ψ_i is the host proposal, η_i is the warmup/readiness gate, and β is the log-moment reward temperature. For DMPO, this DPRM temperature β is distinct from the DMPO target temperature α in $p^\star(x) \propto p_{\mathrm{ref}}(x) \exp(r(x)/\alpha)$.

Table 4. Task-dependent utility oracles used to update DPRM statistics. The same host objective, model, and data pipeline are kept fixed within each comparison; only the ordering controller receives the listed utility.

Experiment	DPRM utility R	β
DPRM-PUMA	Self-supervised teacher-forced utility: average log-probability assigned to the ground-truth tokens newly revealed by the selected action. At validation, the decoder uses the checkpoint-local DPRM state.	1.0
DPRM-DMPO	The task reward already computed by the DMPO rollout pipeline: GSM8K uses format, integer-answer, and exact-answer rewards; MATH uses boxed-answer correctness and answer-tag format rewards; Countdown uses the arithmetic-expression verifier.	1.0
DPRM-Prism	Test-time self-verification utility (SVF): the Prism verifier's probability that a candidate reasoning path is sound, implemented as the normalized yes/no probability of the verifier prompt.	1.0
DPRM-DPLM	Protein inverse-folding utility: provisional amino-acid recovery (AAR), i.e., the fraction of design positions whose current/provisional sequence matches the ground-truth sequence under the CATH 4.3 training sample.	8.0
DPRM-DCM	Self-supervised single-cell reconstruction utility: token recovery on the selected discretized gene-expression bins, averaged over selected positions in the current cell.	1.0
DPRM-GenMol	Molecular generation utility: valid molecule indicator combined with QED/SA quality when RDKit scoring is available; fragment-constrained evaluation additionally records fragment validity and task retention metrics.	1.0
DPRM-SDPO	DNA regulatory-sequence utility: the GOSAI reward product used by the SDPO evaluation, combining HepG2 expression score, ATAC classifier success, and high-expression k-mer Pearson correlation. Reference log-likelihood is tracked as a separate distributional-quality metric.	1.0

The utility in Tab. 4 determines $\widehat{R}_{\phi,b}$, but the realized ordering behavior also depends on the controller schedule. In particular, the experiments differ in their phase discretization, warmup/switch thresholds, readiness count, host-specific reveal budget m_t, and the Soft-BoN shortlist size N_t. We summarize the actual settings used in the reported runs in Tab. 5.

B.2. Additional Details over DPRM-PUMA

This subsection provides the full checkpoint-evaluation details for the DPRM-PUMA results reported in Tab. 2a.

Training and checkpoint selection. Both PUMA runs use the same TinyGSM masked-diffusion pretraining configuration: hidden size 512, 14 layers, 8 attention heads, progressive strategy, confidence-collapse mode with threshold 0.9, block size

256, batch size 32, learning rate 3×10^{-4}, weight decay 0.01, 20 epochs, and EMA decay 0.9999. The progressive horizon is scheduled from $K = 12$ to $K = 42$ by step 330k. In training, this means that the reveal budget is not a fixed constant but the phase-induced count $m_t = \max\{\mathrm{round}(r_{t+1}L_{\mathrm{eff}}) - u_t, 0\}$, where r_{t+1} is sampled from the next interval, L_{eff} is the effective non-prompt length, and u_t is the current number of revealed tokens. The only difference is the token-ordering controller: vanilla PUMA uses confidence top-k, while DPRM-PUMA uses our DPRM Soft-BoN controller with 16 bins, reward temperature β=1.0, $(T_{\mathrm{warm}}, T_{\mathrm{switch}}, N_{\mathrm{ready}}) = (2\mathrm{k}, 60\mathrm{k}, 128)$, and sampled shortlist $N_t = \min\{64, \max(8, 4m_t)\}$. For a fair comparison we report the latest retained EMA checkpoint common to both long runs, namely step 1.53M. The confidence baseline continues beyond this point, but the DPRM-PUMA run's latest shared persisted checkpoint is 1.53M, so this is the maximal apples-to-apples comparison available.

Evaluation protocol. We follow the official PUMA validation configuration on GSM8K. The model is evaluated zero-shot with temperature 0.0 under the two validation settings `unmasking_num`$\in \{2, 3\}$. In the confidence baseline these correspond to `top_k_unmasking_2` and `top_k_unmasking_3`; in DPRM-PUMA they become `dprm_soft_bon_unmasking_2` and `dprm_soft_bon_unmasking_3`. Thus the decode budget is fixed to $m_t \in \{2, 3\}$, while the DPRM controller uses 16 phase buckets, $(T_{\mathrm{warm}}, T_{\mathrm{switch}}, N_{\mathrm{ready}}) = (0, 16, 16)$, and the checkpoint-local DPRM state so that the test-time ordering rule is aligned with the pretraining controller.

Observed gains. The gains are now materially larger than in our earlier intermediate-checkpoint comparison and remain consistent across both official PUMA settings. On unmasking-2, accuracy improves from 29.95 to 34.12; on unmasking-3, it improves from 28.73 to 34.42. The larger improvement under unmasking-3 is notable because it corresponds to a less conservative reveal schedule, where the myopia of confidence ranking should matter more.

Per-question logging and paired-bootstrap robustness. To move beyond Wilson-style aggregate intervals, we evaluate the shared 1.53M checkpoint pair with per-question JSONL logging enabled for every GSM8K example. This produces a paired Bernoulli outcome for each question under both top-k PUMA and DPRM-PUMA, which allows a direct paired bootstrap over the shared evaluation set. Tab. 6 summarizes the resulting 95% percentile intervals from 5,000 paired bootstrap resamples. The point estimates remain favorable to DPRM-PUMA under both official PUMA validation settings: $29.95\% \rightarrow 34.12\%$ for unmasking-2 and $28.73\% \rightarrow 34.42\%$ for unmasking-3. Unlike our earlier lower-step comparison, the paired-bootstrap deltas now exclude zero in both settings. We therefore view the pretraining-side gain as statistically supported at this shared late checkpoint, while still emphasizing that the effect remains smaller than the post-training improvements on Countdown and MATH.

Interpretation. PUMA already addresses the dominant train–test mismatch by teacher-forcing the same progressive order used at decoding time. The checkpoint comparison in Fig. 2 therefore isolates a narrower question: once train–test alignment is already enforced, is confidence itself the right ordering signal? The answer from this experiment remains no. DPRM-PUMA yields a consistent improvement without touching the denoising loss, optimizer, or progressive masking scaffold, and the per-question bootstrap analysis in Tab. 6 and Fig. 3 now shows that this direction is statistically supported across both official PUMA validation settings at the shared late checkpoint. At the same time, the absolute gain is still smaller than the post-training improvements on Countdown and MATH. This is exactly the plug-in behavior we want from DPRM in a pretraining regime: a clean, localized intervention whose effect survives stronger checkpoint matching and stricter uncertainty accounting.

B.3. Additional Details over DPRM-DMPO

Details of DPRM-DMPO The practical algorithms for DMPO (Zhu et al., 2025c) used in our experiments are below. Training remains teacher-forced: once a reveal set is chosen, newly revealed positions are filled with the ground-truth token. Thus DPRM-DMPO changes only the order in which the denoiser sees partial responses; it does not change the labels used by WDCE.

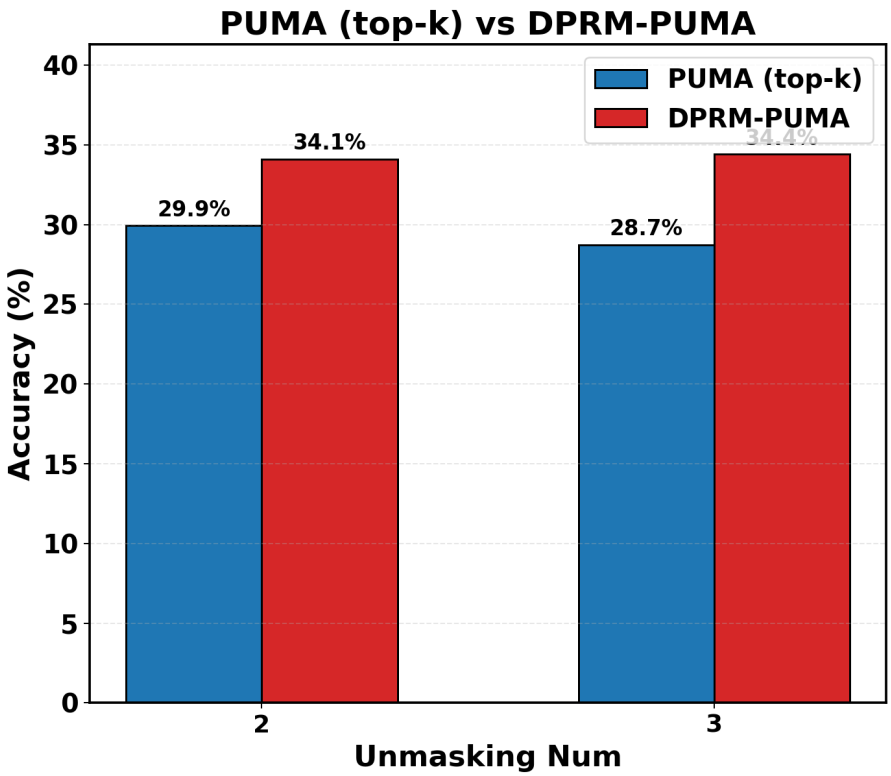

Figure 2. PUMA vs. DPRM-PUMA on GSM8K at the shared 1.53M EMA checkpoint. We use the two official PUMA validation settings, `unmasking_num`$\in \{2, 3\}$. DPRM-PUMA improves both.

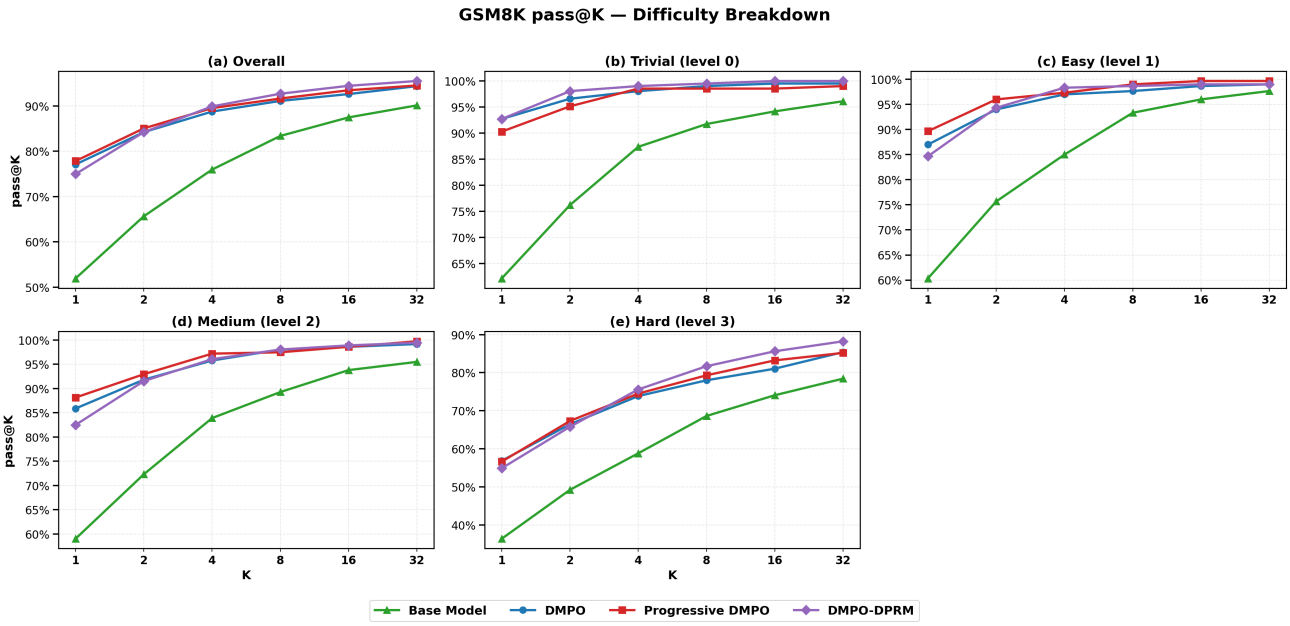

Figure 4. GSM8K pass@K curves by difficulty level (0: trivial, 1: easy, 2: medium, 3: hard). DMPO-DPRM's advantage over Progressive DMPO is most visible on harder levels and at larger K.

This subsection provides the full experimental details for the DPRM-DMPO results reported in Sec. 4.1.

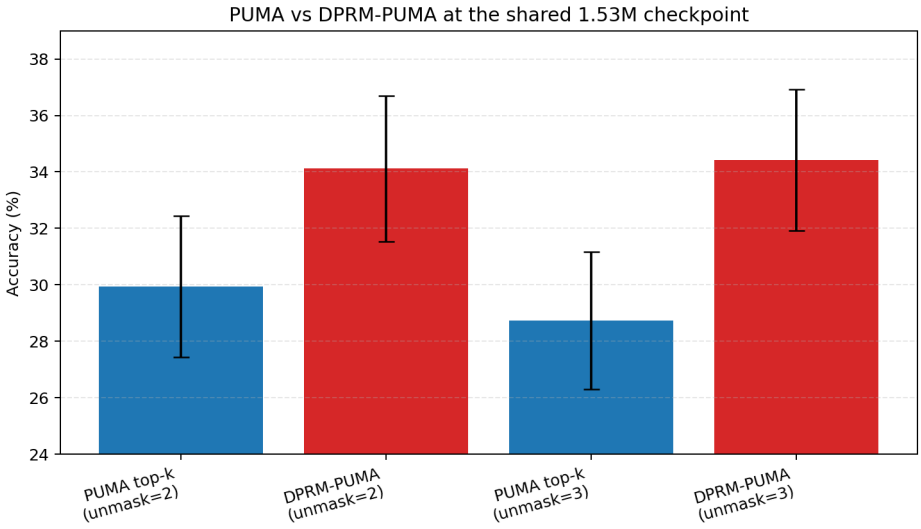

Figure 3. Bootstrap confidence intervals for PUMA and DPRM-PUMA at the shared 1.53M checkpoint. The two official unmasking settings both favor DPRM-PUMA, and the paired-bootstrap deltas in Tab. 6 exclude zero at the 95% level.

Benchmarks and metrics. We evaluate completed post-training runs on GSM8K, MATH, and Countdown using pass@K curves for the six tested values $K \in \{1, 2, 4, 8, 16, 32\}$. For compact comparison in the main paper, we report the arithmetic mean of pass@K over exactly these six values.

Difficulty stratification. Each benchmark provides per-example difficulty labels that allow us to disaggregate performance by problem complexity:

- **GSM8K**: The full 1,319-example test set from `divelab/gsm8k` with four difficulty levels (0–3) based on the base model's per-example pass@1 probability. Level 0 (trivial) corresponds to pass@1 $\in [0.0, 0.3)$; level 1 (easy) to $[0.3, 0.5)$; level 2 (medium) to $[0.5, 0.7)$; level 3 (hard) to $[0.7, 1.0)$. The overall results aggregate all levels.

- **MATH**: A fixed 500-example subset from the local DMPO evaluation pipeline with five difficulty levels (1–5). Levels 1–4 correspond to the standard MATH difficulty tiers (trivial, easy, medium, hard); level 5 is an out-of-distribution (OOD) subset containing problems whose topics are underrepresented in the training split.

- **Countdown**: The full 5,120-example test split from `divelab/countdown`, where difficulty is defined by the number of target operands in the countdown equation: level 2 (trivial, two operands), level 3 (easy, three operands), level 4 (medium, four operands), level 5 (hard, five operands), and level 6 (OOD, six operands, which rarely appear in standard arithmetic curricula).

Compared methods. We compare four systems built from the same LLaDA-8B-Instruct base checkpoint:

1. **Base Model**: the pre-finetuning LLaDA-8B-Instruct checkpoint;

2. **DMPO**: the original DMPO fine-tuning recipe with random masking (Zhu et al., 2025c);

3. **Progressive DMPO**: our PUMA-inspired finetuning baseline, which uses teacher-forced progressive masking together with the decoder's confidence top-k reveal order;

4. **DMPO-DPRM**: our method, which keeps the Progressive DMPO hyperparameters fixed and changes only the reveal policy to DPRM Soft-BoN.

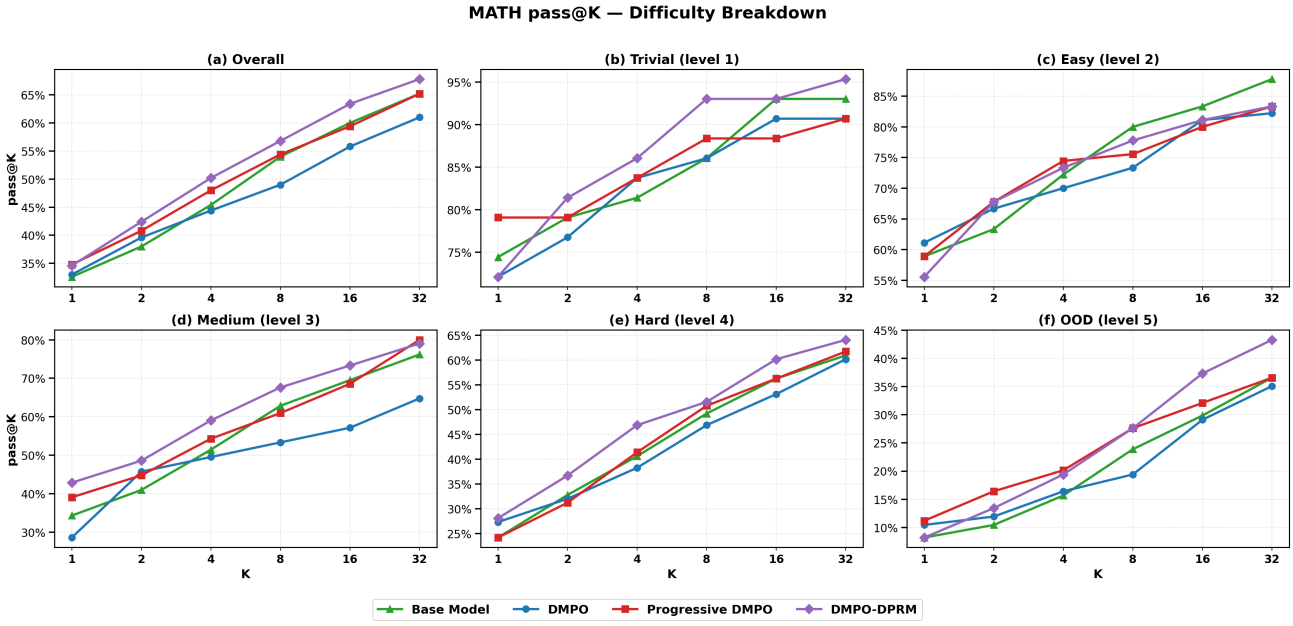

Figure 5. MATH pass@K curves by difficulty level (1: trivial, 2: easy, 3: medium, 4: hard, 5: OOD). DPRM-DMPO provides consistent gains on hard and OOD subsets.

Inference configuration. For **DMPO-DPRM** decoding we use its aligned inference rule: `fast_dllm` with `pd_cache_prefix`, `remasking=dprm_soft_bon`, block length 32, temperature 0.2, and the checkpoint-local `dprm_estimator.json`. In the reported evaluation scripts, the host decoder keeps the native Fast-dLLM transfer schedule inside each block, so m_t is produced by uniform token transfer over the local denoising steps; with generation length 256, block length 32, and 128 denoising steps, this becomes 16 local steps per block and an initial budget of 2 tokens per step. The loaded estimator is used with the force-full gate so that the decode-side DPRM guidance is fully active. The other three models are decoded with the standard low-confidence remasking rule. We intentionally keep this asymmetry because our claim is precisely that reveal-order alignment should hold across training and inference.

Fairness of the comparison. For each task, DPRM-DMPO is matched against the corresponding Progressive DMPO baseline rather than against DMPO-random. Thus, the comparison isolates the effect of the reveal-order policy under an otherwise fixed progressive post-training setup.

On GSM8K, we keep the same 5,000 training steps, learning rate 1×10^{-6}, number of generations 8, replay reuse length 8, per-device batch size 4, generation batch size 4, phase count $K = 8$, threshold $\tau = 0.9$, diffusion steps 128, and temperature 0.2. DMPO-DPRM uses 16 confidence bins, $(T_{\text{warm}}, T_{\text{switch}}, N_{\text{ready}}) = (500, 2000, 128)$, and sampled shortlist $N_t = \min\{32, \max(8, 4m_t)\}$, while the reveal budget is always the host progressive budget $m_t = \lceil M_t/P_t \rceil$.

On MATH, we likewise match Progressive DMPO with learning rate 3×10^{-6}, per-device batch size 4, gradient accumulation 4, generations 8, and the same decoding budget; the DPRM controller settings remain the same.

On Countdown, we use 5,000 training steps, learning rate 1×10^{-6}, per-device batch size 4, generations 8, replay reuse length 8, diffusion steps 128, and temperature 0.2; again, the DPRM controller settings remain the same.

Therefore, relative to Progressive DMPO, DPRM-DMPO changes only the reveal-order policy during training and the aligned decoder during testing.

Paired-bootstrap uncertainty. Using the protocol of Sec. B.9, we recompute uncertainty directly from the saved raw success matrices rather than from aggregate pass@K curves. Tab. 7 reports 95% paired-bootstrap intervals for the two central deltas in this section: the gain from switching from random masking to confidence-aligned progressive masking, and the further gain from switching from confidence to DPRM within the progressive scaffold. The bootstrap view sharpens the empirical picture. The curriculum effect of Progressive DMPO over random masking is statistically clear on MATH and

Countdown, and the DPRM guidance is clearly supported on Countdown. On GSM8K and MATH, the DPRM deltas remain positive or near-zero but are not separated from zero at the 95% level in a single-seed comparison, so we describe those gains as suggestive rather than definitive.

Additional observations on difficulty breakdowns. The main gains of DPRM-DMPO are concentrated on harder evaluation regimes. On GSM8K hard questions (level 3), the mean pass@K increases from 74.4 for Progressive DMPO to 75.3 for DPRM-DMPO, and the pass@32 endpoint increases from 85.2 to 88.2. On MATH hard, the mean pass@K increases from 44.3 to 47.9; on the OOD subset it increases from 24.0 to 24.9, while the pass@32 endpoint rises from 36.6 to 43.3. On Countdown hard (level 3, five operands), the mean pass@K increases from 29.6 to 33.4, and the pass@32 endpoint increases from 47.9 to 60.0.

These patterns suggest that DPRM does not uniformly dominate confidence ranking at every operating point. Instead, it changes the exploration profile, yielding better reward-aware coverage on harder, more combinatorial, or more out-of-distribution parts of the state space.

Pass@K curves by difficulty level. Figs. 4 to 6 show the full pass@K curves disaggregated by difficulty level for each benchmark. Each panel plots pass@K against $K \in \{1, 2, 4, 8, 16, 32\}$ for all four methods.

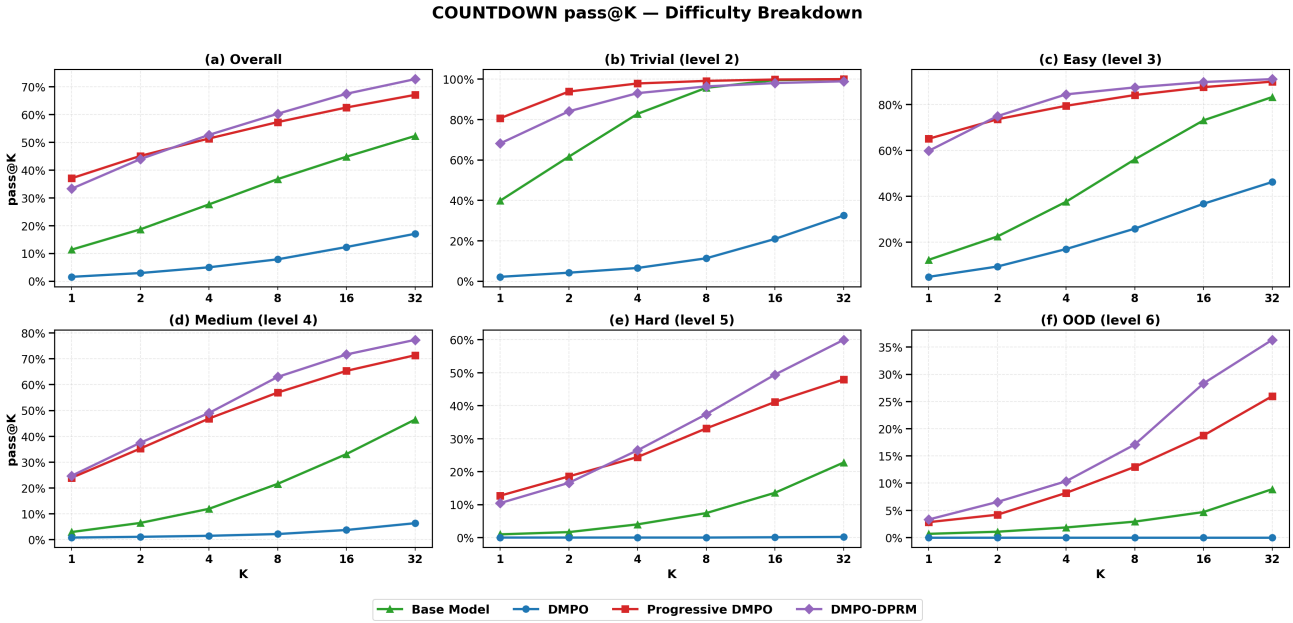

Figure 6. Countdown pass@K curves by number of target operands (2–6). Vanilla DMPO collapses on this task, falling below the base model at every difficulty level. DPRM-DMPO achieves the strongest performance across all levels.

Interpretation. Taken together, the experiments support a two-step interpretation. Random masking is a poor state sampler for post-training because it allocates denoising capacity to states that decoding rarely visits; on Countdown, it is even worse than no fine-tuning at all. Confidence-based progressive masking, as instantiated by Progressive DMPO, fixes much of this train–test mismatch but remains locally greedy. DPRM-DMPO preserves the same confidence-aligned progressive training scaffold while replacing the reveal-order heuristic by an online process-reward guidance. Empirically, this matters most when the task is hard, out-of-distribution, or evaluated under larger sampling budgets.

B.4. Additional Details over DPRM-Prism

This subsection provides the full experimental details for the DPRM-Prism results reported in Tab. 2b.

Setup. We evaluate on GSM8K using the LLaDA-2.0-mini model (Bie et al., 2025). The Prism framework (Bai et al., 2026) performs hierarchical trajectory search (HTS) with self-verification as reward (SVF). We run two configurations that

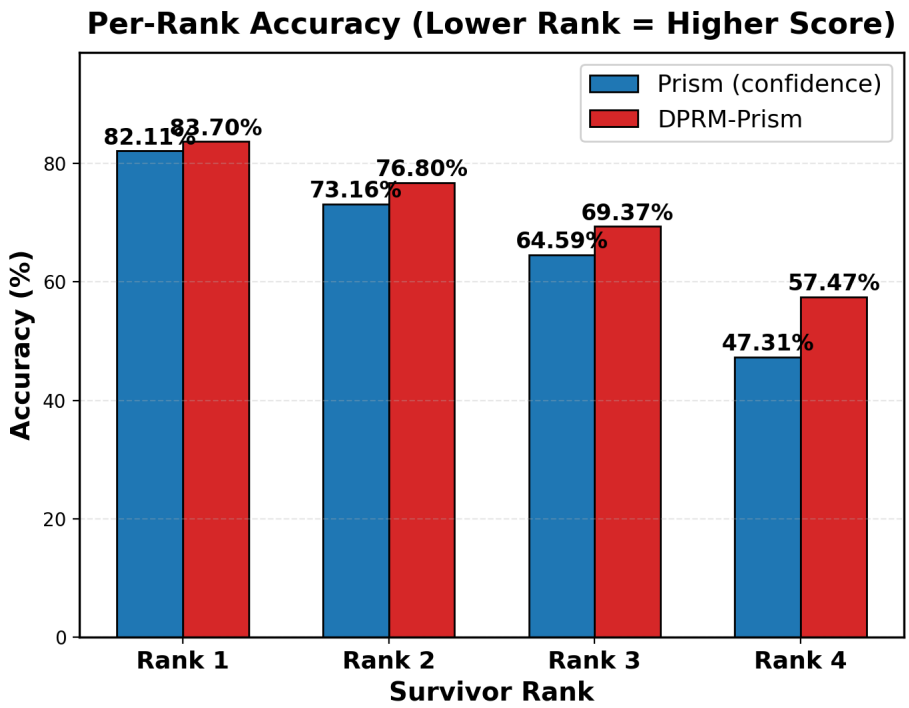

Figure 7. Per-rank accuracy comparison on GSM8K. Rank 1 is the highest-scored survivor after pruning. DPRM-Prism improves at every rank position.

differ only in the token-ordering policy used during trajectory pruning and remasking:

1. **Prism (confidence)**: the original Prism baseline, which uses confidence top-k to rank and select tokens at each unmasking step;
2. **DPRM-Prism**: our method, which replaces confidence top-k by DPRM Soft-BoN. The DPRM controller uses 8 phase buckets, 16 confidence bins, reward temperature $\beta = 1.0$, $(T_{\text{warm}}, T_{\text{switch}}, N_{\text{ready}}) = (6, 22, 64)$ over the 32 decode steps, and sampled shortlist $N_t = \min\{64, \max(8, 4m_t)\}$.

Shared hyperparameters. Both configurations use identical Prism search scaffolds: initial branching width N=16, final survivors K=4, survivor count per pruning event S=2, decay factor d=1.8, pruning window $[0.1, 0.6]$ of the generation length, pruning interval 3 steps, block length 32, generation steps 32, generation length 256, temperature 0.7, and SVF reward mode with task type `math`. Under this host schedule, the token commit budget is $m_t = 1$ per step inside each block, and the early low-gate regime retains Prism's threshold fallback before fully switching to DPRM selection. We evaluate on the full GSM8K test set (1,319 examples) with zero-shot prompting. Each configuration is run once (no repeated seeds), and accuracy is computed by weighted majority vote among the top 60% of scored survivors, following the original Prism evaluation protocol.

Per-rank accuracy. Fig. 7 breaks down accuracy by survivor rank. Rank 1 corresponds to the highest-scored trajectory after the final pruning stage. DPRM-Prism outperforms the baseline at every rank, with the largest absolute gap at Rank 4 (+10.2pp), indicating that DPRM produces better-quality lower-ranked survivors as well. This is consistent with DPRM improving the overall exploration profile rather than merely sharpening the top candidate.

NFE analysis. Fig. 8 provides two complementary views of inference cost. The left panel plots voted accuracy against mean NFE on a log scale, with reference points from the original Prism paper (LLaDA-2.0-mini, N=1 and Best-of-16). Both Prism and DPRM-Prism substantially outperform these baselines, and DPRM-Prism achieves higher accuracy at moderate additional NFE cost. The right panel shows the per-sample NFE distributions: DPRM-Prism's distribution is shifted rightward by the ordering overhead, but the two distributions have comparable spread.

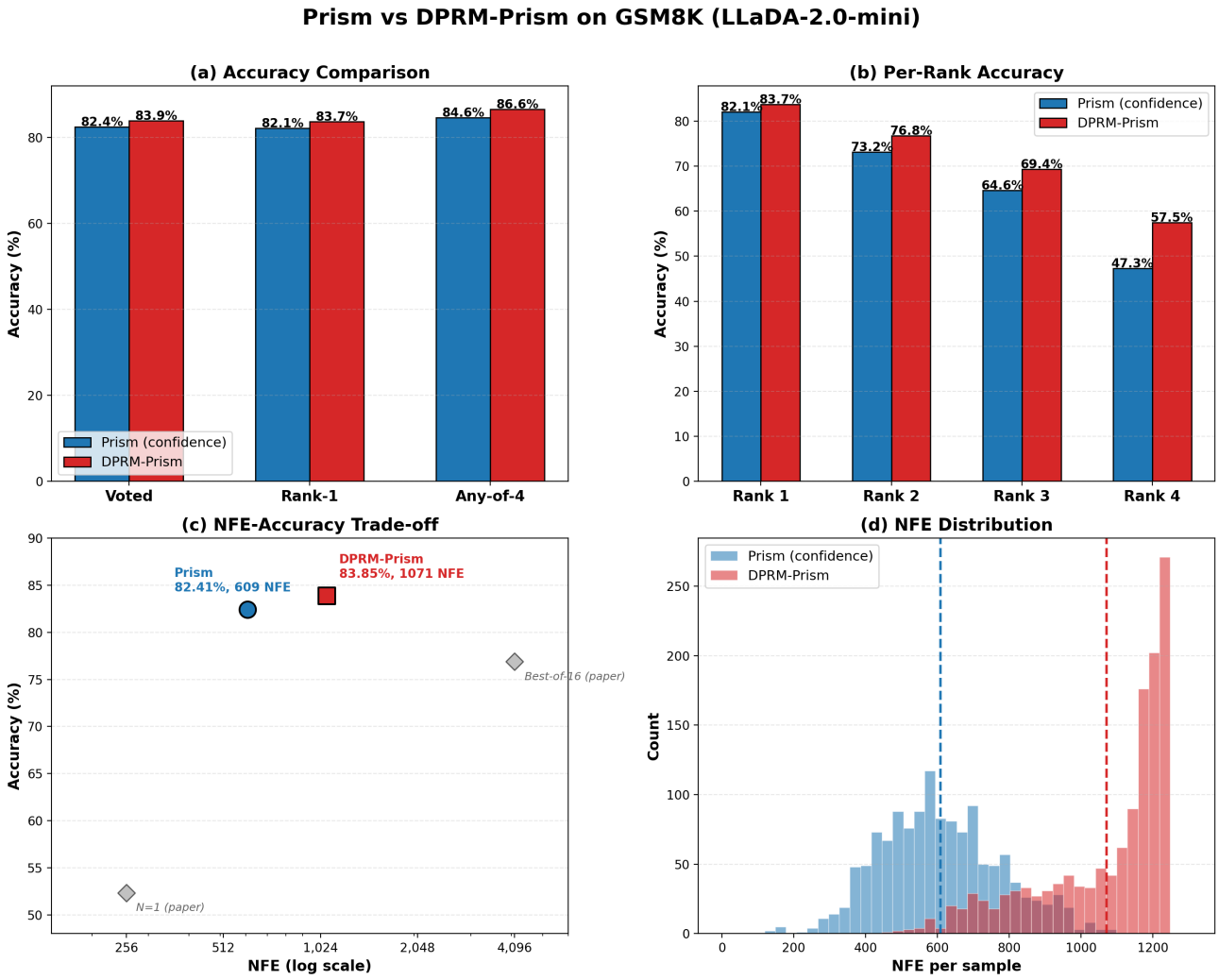

Figure 8. Left: NFE–accuracy trade-off. The diamond markers show reference baselines from the Prism paper. Right: per-sample NFE distributions.

NFE overhead. The $\times 1.76$ NFE increase of DPRM-Prism over the baseline originates entirely from the DPRM ordering layer: at each unmasking step, Soft-BoN evaluates multiple candidate token orderings before selecting one. The Prism search scaffold (HTS branching, SVF calls, pruning events) is completely unchanged. In exchange for this overhead, DPRM-Prism improves voted accuracy by $+1.44$pp and any-of-4 accuracy by $+1.97$pp. The NFE per SVF call remains identical (29), confirming that the verification budget is unaffected.

Interpretation. The DPRM-Prism results confirm that the plug-in ordering principle transfers cleanly from training-time interventions (DPRM-DMPO) to test-time scaling. Because the Prism scaffold handles trajectory-level search while DPRM handles token-level ordering, the two modules compose naturally: DPRM improves the quality of each unmasking decision, and the hierarchical search propagates these gains through the pruning tree. The consistent improvement across all rank positions suggests that DPRM's benefit is not limited to the final selection step but improves the entire generation trajectory.

B.5. Additional Details over DPRM-DPLM

This subsection provides the full experimental details for the DPLM-2 Bit ordering ablation reported in Tab. 3a.

We compare validation-best checkpoints produced by four matched 650M training runs: **DPLM-2 Bit**, **Progressive DPLM-2 Bit**, **DPRM-DPLM-2 Bit**, and **DPRM(random)-DPLM-2 Bit**. The protein-specific model stack is otherwise unchanged:

same DPLM-2 Bit architecture, same multimodal conditioning, same tokenizer and structure abstraction, and the same denoising objective. The only controlled difference is the sequence token-ordering controller. All ordering-aware variants use 8 phases, 16 confidence bins, one active structural bucket, and sampled shortlist $N_t = \min\{32, \max(8, 4m_t)\}$. Progressive DPLM uses confidence-aligned progressive ordering with $(T_{\text{warm}}, T_{\text{switch}}, N_{\text{ready}}) = (0, 0, 256)$. DPRM-DPLM warms up with confidence and then shifts to the online DPRM Soft-BoN controller with $(2000, 20000, 256)$, and DPRM(random)-DPLM replaces the confidence warmup by random ordering before the same DPRM phase. In all three variants the phase-local update budget is $m_t = \lceil M_t/P_t \rceil$, where M_t is the remaining masked design budget and P_t the remaining phases.

Evaluation protocol. We follow the evaluation routines exposed by the original DPLM repository and paper (Hsieh et al., 2025). First, we run **forward folding** on the CAMEO2022 benchmark and report average backbone RMSD and TM-score to the ground-truth structure over 163 targets. Second, we run **unconditional co-generation** at lengths $100, 200, 300, 400, 500$, generating 50 samples per length, and evaluate them with the repository's self-consistency pipeline based on ESMFold. The reported metrics are mean ca-RMSD, mean backbone TM-score, mean pLDDT, and the designable rate, where a sample is counted as designable when its self-consistency TM-score is at least 0.5.

Statistical uncertainty. For forward folding, all models are evaluated on the same CAMEO2022 target set, so we report paired bootstrap intervals over the 163 targets when comparing each ordering variant to DPLM-2 Bit. For co-generation, the generated structures are not paired sample-by-sample across methods, so we report ordinary bootstrap intervals over the 250 generated samples pooled across the five lengths, and independent bootstrap intervals for deltas against DPLM-2 Bit. All intervals use 5,000 bootstrap resamples. The raw summaries, deltas, and plotting artifacts are stored under `eval_outputs/comparisons/dplm2bit_order_ablation`.

Results under the original DPLM metrics. In forward folding, all three ordering-aware variants are statistically indistinguishable from one another and uniformly better than DPLM-2 Bit: each reduces backbone RMSD by 6.03 with paired-bootstrap 95% interval $[-6.53, -5.54]$ and improves TM-score by 0.025 with interval $[0.0186, 0.0317]$. In unconditional co-generation, all three ordering-aware variants again improve designable rate over DPLM-2 Bit, by $+16.8$pp for Progressive DPLM, $+16.4$pp for DPRM-DPLM, and $+16.0$pp for DPRM(random)-DPLM, with intervals $[+8.8, +24.8]$pp, $[+8.4, +24.8]$pp, and $[+8.0, +24.0]$pp. The confidence-progressive controller is strongest on the repository's global foldability metrics, with bb-TM gain $+0.0558$ $[0.0348, 0.0766]$ and pLDDT gain $+9.01$ $[6.71, 11.25]$. DPRM-DPLM remains competitive on bb-TM, with gain $+0.0542$ $[0.0343, 0.0738]$, and has the smallest ca-RMSD penalty among the ordering-aware variants, $+8.97$ $[3.42, 14.36]$, compared with $+24.91$ $[18.79, 31.08]$ for Progressive DPLM and $+13.50$ $[7.78, 18.94]$ for DPRM(random)-DPLM. We therefore interpret the protein experiment as genuinely multi-objective: ordering matters a great deal, but the best controller depends on which structural metric one prioritizes.

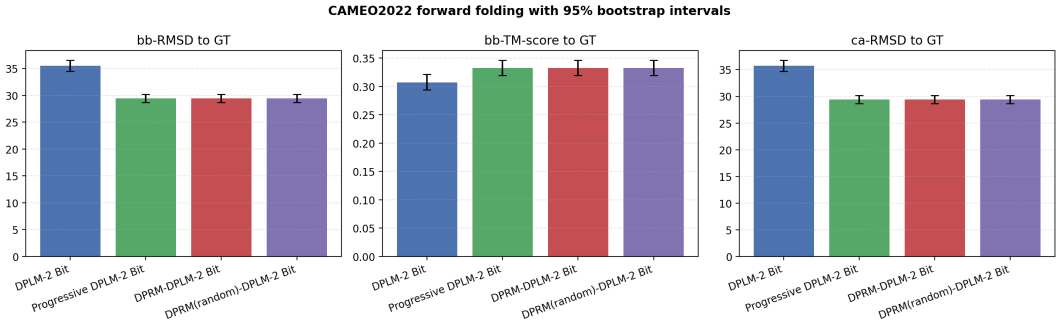

Figure 9. Forward-folding comparison on CAMEO2022 with 95% bootstrap intervals over targets. In the experiments, all three ordering-aware variants are statistically indistinguishable on the forward-folding metrics and all improve over DPLM-2 Bit.

Interpretation. The DPLM results are useful because they separate several notions of protein quality: local structural agreement, global fold similarity, model confidence, and designability. Unlike GSM8K-style reasoning, these metrics need not move together. The clean four-way ablation still shows that ordering alone materially changes the protein diffusion trajectory under a fixed architecture, but it changes the earlier conclusion about *which* ordering rule looks best. Under the repository's current metrics, the confidence-progressive controller is strongest overall in co-generation, whereas

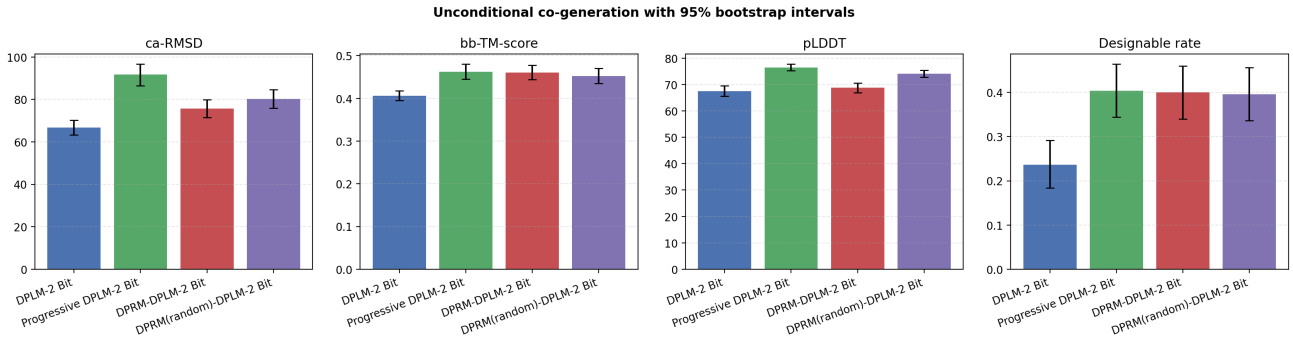

Figure 10. Unconditional co-generation self-consistency over lengths 100–500 with 95% bootstrap intervals over generated samples. Progressive DPLM-2 Bit is strongest on bb-TM, pLDDT, and designable rate, while DPRM-DPLM-2 Bit incurs a smaller ca-RMSD penalty than the other ordering-aware variants.

DPRM-DPLM offers a milder trade-off with comparable bb-TM and designable rate but a substantially smaller ca-RMSD penalty. This still supports the plug-in claim, but it also suggests that future protein-specific DPRM controllers should use richer utilities than the current generic AAR-based estimator if the goal is to dominate confidence on all structural metrics simultaneously.

B.6. Additional Details over DPRM-DCM

This subsection documents the single-cell discrete diffusion experiment reported in Tab. 3b. The upstream DCM paper (Bhattacharya et al., 2026) evaluates a broader set of distributional single-cell benchmarks, including Dentate Gyrus, Replogle Perturb-seq, and PBMC cytokine perturbation settings. Our goal here is narrower: to test whether the DPRM ordering module can be inserted into an existing masked discrete diffusion codebase outside language without changing its model or denoising objective.

Dataset and preprocessing. We use the public Dentate Gyrus dataset from the DCM repository's documented Figshare source. The raw matrix has 2,930 cells and 13,913 genes. For a compact ordering diagnostic, we select the top 5,000 high-variance genes, apply $\log(1+x)$, and discretize nonzero expression values by quantile bins. The resulting validation protocol uses the same fixed $90/10$ split for every method, giving 293 held-out cells. This small dataset and small DCM configuration explain the short training runtime: the run is a fast Dentate-ordering reproduction, not the full benchmark scale of Bhattacharya et al. (2026).

Model and training. All four methods use the same SEDD-style DCM transformer: hidden size 128, 4 layers, 4 attention heads, dropout 0.1, batch size 8, learning rate 10^{-4}, weight decay 0.01, mask ratio 0.15, bfloat16 mixed precision, and 50 epochs. The baseline uses random ordered masking. Progressive-DCM uses current-model confidence ordering throughout. DPRM(random)-DCM starts from random ordering and shifts to DPRM; DPRM(conf.)-DCM starts from confidence ordering and shifts to DPRM. Both DPRM variants use 8 phases, 16 confidence bins, $\beta = 1.0$, $(T_{\text{warm}}, T_{\text{switch}}, N_{\text{ready}}) = (500, 2000, 128)$, and sampled shortlist $N_t = \min\{64, \max(8, 4m_t)\}$.

Training sanity check. The runs finish quickly because the host is small and the processed Dentate subset contains fewer than three thousand cells. The final and best validation losses follow the same qualitative pattern as the decoding evaluation: random ordered masking is much worse than ordering-aware variants. The best validation losses are 1418.94 for DCM-random, 797.39 for Progressive-DCM, 796.22 for DPRM(random)-DCM, and 1228.30 for DPRM(conf.)-DCM. We therefore treat the short runtime as normal for this compact experiment rather than as a failed or under-trained run.

Aligned evaluation and uncertainty. Evaluation decodes each held-out cell from the all-mask state for 32 reveal steps. The reveal budget is $m_t = \lceil M_t/P_t \rceil$, where M_t is the number of still-masked genes and P_t is the number of remaining phases. Random, confidence, and DPRM checkpoints are evaluated with their corresponding train-time ordering families; DPRM checkpoints load their saved online estimator and use the full DPRM gate at test time. Each method is sampled four times per cell, and metrics are averaged per cell before bootstrap. Fig. 11 visualizes the resulting 95% bootstrap intervals.

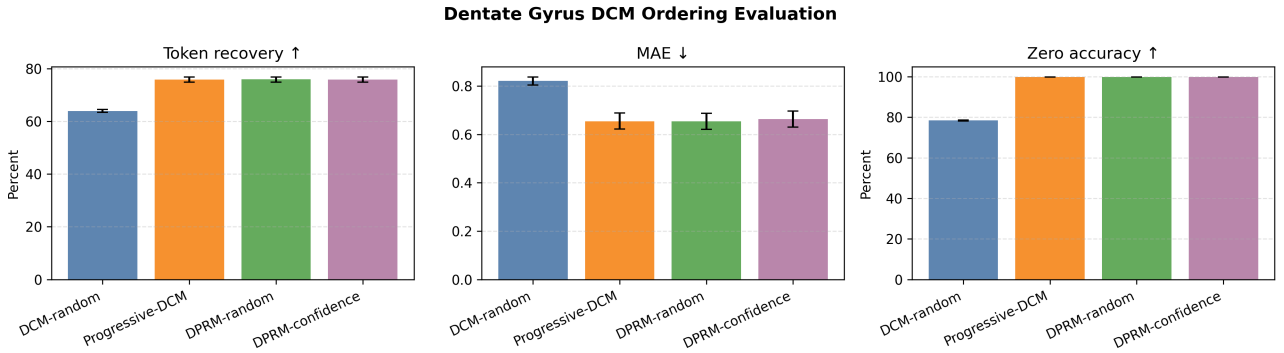

Figure 11. Dentate Gyrus DCM ordering evaluation with 95% bootstrap intervals over 293 validation cells. All ordering-aware variants improve token recovery, MAE, and zero-expression accuracy over random ordered masking. DPRM(random)-DCM is strongest on MAE and zero-expression accuracy in this compact setting.

Interpretation. The DCM result should be read as evidence for the plug-in nature of DPRM rather than as a claim of state-of-the-art single-cell generation. We do not modify DCM's model family, count-bin representation, loss, or optimizer, and we do not evaluate the full MMD/Wasserstein benchmark suite from the original paper. Nevertheless, the large paired-bootstrap gains show that the same ordering controller used in language and protein settings can transfer to sparse scientific count data. The close performance of Progressive-DCM and the two DPRM variants also indicates that, in this small Dentate setting, the main improvement comes from moving away from random ordered masking; richer task utilities may be needed for DPRM to separate more strongly from confidence on downstream biological distributional metrics.

B.7. Additional Details over DPRM-GenMol

This subsection documents the molecular drug-design ordering pilot reported in Tab. 3c. GenMol V2 (Lee et al., 2025) is a generalist molecular generator that applies masked discrete diffusion to SAFE molecular strings and supports de novo generation, linker design, motif extension, scaffold decoration, and superstructure generation. Our goal is narrower than the full GenMol benchmark: we test whether the same DPRM ordering module can be inserted into a molecular discrete diffusion sampler while keeping the GenMol V2 checkpoint, SAFE representation, and molecular metrics fixed.

Model and variants. All four runs use the public GenMol V2 path with bracket-SAFE enabled. We compare the original GenMol V2 ordering, confidence-progressive ordering, DPRM(random)-GenMol, and DPRM(conf.)-GenMol. The DPRM variants use 8 reveal phases, 16 confidence bins, $\beta = 1.0$, $(T_{\text{warm}}, T_{\text{switch}}, N_{\text{ready}}) = (500, 2000, 128)$, and sampled shortlist $N_t = \min\{64, \max(8, 4m_t)\}$. DPRM(random)-GenMol starts from random reveal proposals before switching to DPRM; DPRM(conf.)-GenMol starts from confidence reveal proposals before switching to DPRM. No molecular tokenizer, denoising objective, checkpoint format, sampler temperature, or RDKit metric is changed.

Evaluation protocol. For de novo generation, each method generates 1,000 molecules using the GenMol V2 de novo configuration: softmax temperature 1.0, randomness 0.3, and minimum added length 60. We report validity, uniqueness, QED/SA quality, and Morgan-fingerprint diversity with 5,000 ordinary bootstrap resamples over generated molecules. For fragment-constrained generation, we use the repository's fragment demo CSV and the five supported tasks. GenMol's native `fragment_linking` sampler exits at the native-library level for several fragment rows under at least one ordering checkpoint. To avoid method-dependent crashes, the reported pilot uses the common stable subset that excludes ELIGLUSTAT, ERLOTINIB, and FUTIBATINIB for every method. We generate one molecule per fragment example and task, giving $7 \times 5 = 35$ fragment-task units per method, and bootstrap over these units. This makes the comparison stable and method-matched, but it should not be read as a full GenMol V2 benchmark reproduction.

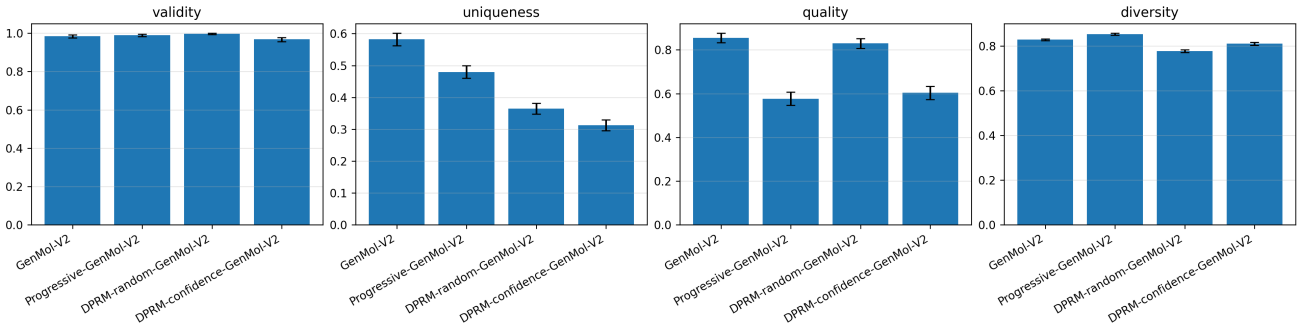

Figure 12. GenMol V2 de novo molecular generation with 95% bootstrap intervals over 1,000 generated molecules per method. GenMol V2 remains strongest on quality and uniqueness; DPRM(random)-GenMol has the highest validity; Progressive-GenMol has the highest diversity.

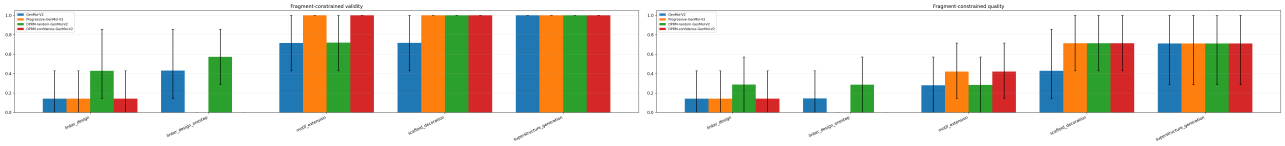

Figure 13. GenMol V2 fragment-constrained generation on the common stable seven-fragment subset. Error bars show 95% bootstrap intervals over fragment-task units. DPRM(random)-GenMol improves linker and linker-onestep validity, while Progressive and DPRM-confidence improve motif-extension and scaffold-decoration quality.

Interpretation. The GenMol pilot is intentionally conservative. It supports the claim that token ordering is a portable control surface even in molecular SAFE diffusion, but it does not show uniform dominance over GenMol V2. In de novo generation, the original checkpoint remains best on quality and uniqueness, suggesting that the generic DPRM utility used here is not yet aligned with every molecular-design objective. In fragment-constrained generation, however, ordering-aware variants improve several task-specific validity and quality metrics under the same sampler. Future molecular-specific DPRM variants should use richer property or docking oracles if the goal is to optimize drug-design utility rather than demonstrate plug-in feasibility.

B.8. Additional Details over DPRM-SDPO

This subsection documents the DNA reward-optimization pilot reported in Tab. 3d. SDPO (Wang et al., 2025a) fine-tunes discrete diffusion models with reward optimization and includes DNA regulatory-sequence design experiments. Our goal here is narrower than a full SDPO reproduction: we test whether the DPRM ordering module can be inserted into the SDPO code path while keeping the pretrained diffusion checkpoint, reward optimizer, oracle suite, and sequence representation fixed.

Model and variants. All four runs start from the same public pretrained DNA diffusion checkpoint and use the same SDPO fine-tuning configuration: $K = 2000$, two epochs, learning rate 10^{-5}, and SDPO temperature 0.5. We compare the original SDPO ordering, confidence-progressive ordering, DPRM-SDPO, and DPRM(random)-SDPO. The DPRM variants use 8 reveal phases, 10 confidence bins, $\beta = 1.0$, $(T_{\text{warm}}, T_{\text{switch}}, N_{\text{ready}}) = (100, 400, 64)$, and Soft-BoN shortlist size $N_t = 64$. DPRM-SDPO starts from confidence ordering before switching to DPRM, whereas DPRM(random)-SDPO starts from random ordering before the same DPRM phase.

Evaluation protocol. Each method generates 10 batches of 64 DNA sequences, for 640 samples per method. We report the GOSAI HepG2 expression score, ATAC success rate, high-expression k-mer Pearson correlation, reference-model log-likelihood, and the product-style total metric used by the SDPO evaluation. Intervals are 95% ordinary bootstrap intervals over generated samples with 1,000 resamples. This uncertainty measures generated-sample variability under a fixed checkpoint; it does not average over independent SDPO fine-tuning seeds.

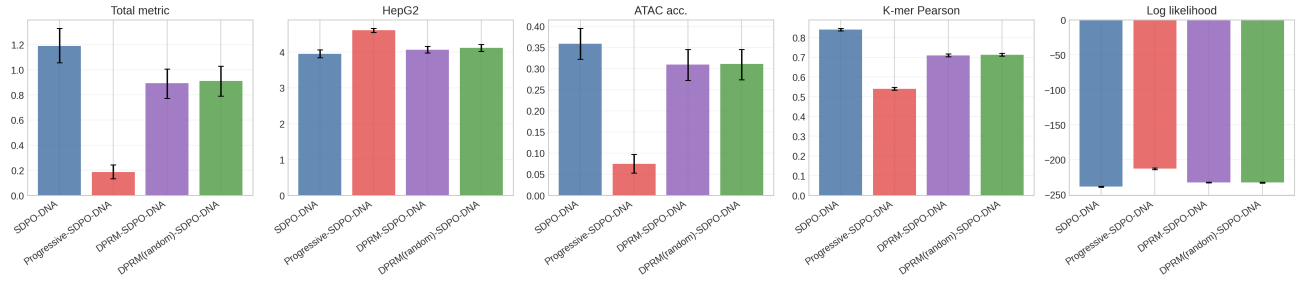

Figure 14. SDPO ordering comparison with 95% bootstrap intervals over 640 generated DNA samples per method. Confidence-progressive ordering improves HepG2 and log-likelihood but collapses ATAC and k-mer quality. DPRM variants preserve substantially more ATAC and k-mer quality while still improving HepG2 over the SDPO baseline.

Interpretation. The SDPO experiment reinforces the multi-objective nature of scientific diffusion design. Confidence-only progressive ordering makes the generated sequences look more likely under the reference model and improves the expression oracle, but it severely reduces accessibility and k-mer-distribution quality. DPRM partially counteracts that collapse: both DPRM variants recover most of the baseline ATAC success and a large part of the k-mer correlation, while improving HepG2 over baseline. The product-style total metric still favors the original SDPO baseline, so we do not claim that the current generic DPRM utility is optimal for DNA regulatory design. The useful conclusion is more precise: changing only token ordering substantially moves the reward-quality trade-off surface, and DPRM is a safer reward-aware controller than confidence-only progressive ordering in this pilot.

B.9. Statistical Uncertainty Protocol

All uncertainty summaries reported below use $5{,}000$ nonparametric bootstrap resamples and 95% percentile confidence intervals whenever raw per-example outputs are available. When two methods are evaluated on the same set of examples, we use a *paired* bootstrap over example indices; when only a single method is summarized, we bootstrap the method-specific metric over the underlying evaluation units. Wilson score intervals are used only as a fallback when the original experiment saved aggregate accuracies but not raw per-example decisions.

DMPO family. For DMPO-style pass@K evaluation, the canonical raw artifact is a Boolean matrix

$$S \in \{0,1\}^{n \times K_{\max}},$$

where $S_{e,j} = 1$ iff the j-th decoded sample for example e is correct. For the six reported values $K \in \{1,2,4,8,16,32\}$, we define the per-example average pass@K statistic

$$\bar{p}_e = \frac{1}{6} \sum_{K \in \{1,2,4,8,16,32\}} \mathbf{1}\left\{ \max_{j \le K} S_{e,j} = 1 \right\}.$$

Method deltas are then computed by paired bootstrap over the shared examples using $\bar{p}_e^{(B)} - \bar{p}_e^{(A)}$. In addition to the raw success matrix, the updated evaluators now emit a human-readable `per_example_summary.jsonl` file that records each example's metadata, the sample-wise correctness vector, and the induced pass@K curve.

PUMA family. For PUMA, each checkpoint evaluation now saves one JSONL record per GSM8K question, including the example index, the decoded sample, the predicted answer, and the final correctness bit. At a fixed unmasking setting, this gives a Bernoulli outcome $Y_e \in \{0,1\}$ for every evaluation question. The reported method delta is the paired-bootstrap estimate of $\mathbb{E}[Y_e^{\text{DPRM}} - Y_e^{\text{top-}k}]$ over the shared GSM8K questions. In the revised study, both the main-text checkpoint comparison and the paired-bootstrap uncertainty analysis use the latest shared persisted checkpoint at step 1.53M, so the point estimates and the uncertainty table are fully aligned.

Prism family. For DPRM-Prism, the evaluation unit is a GSM8K question under a fixed hierarchical trajectory-search scaffold. For each question and method, the evaluator stores the final weighted-vote correctness bit, the correctness of each retained survivor rank, the any-of-4 correctness bit, and the number of function evaluations. Accuracy intervals in Tab. 2b

are computed by bootstrapping questions. Deltas between Prism and DPRM-Prism use a paired bootstrap over the same 1,319 GSM8K questions, because both methods are evaluated on the identical prompt set. NFE summaries are bootstrapped over questions as method-specific cost statistics. We do not bootstrap over repeated decoding seeds because the reported Prism comparison uses one fixed run per method, matching the original test-time-scaling protocol; the uncertainty therefore quantifies dataset-level variability rather than stochastic-run variability.

DPLM family. For DPLM-2 Bit, uncertainty is computed at the natural unit of each evaluation benchmark. In forward folding, all methods are evaluated on the same CAMEO2022 target set, so RMSD and TM-score deltas are estimated with a paired bootstrap over the 163 shared targets. In unconditional co-generation, generated samples are not paired across methods: each method produces samples over the five length buckets $100, 200, 300, 400, 500$, and the repository metrics are computed from the resulting ESMFold self-consistency outputs. We therefore report ordinary bootstrap intervals over generated samples for method-level co-generation metrics, and independent-bootstrap intervals for deltas against DPLM-2 Bit. This distinction is important: forward-folding intervals control for the same target structures, whereas co-generation intervals measure variability of the generated sample population under each ordering controller.

DCM family. For the single-cell experiment, the evaluator saves one CSV row per held-out cell and method. Each row contains token recovery, mean absolute error over discretized expression bins, and zero-expression accuracy, averaged over four stochastic decodes from the all-mask state. Since all methods use the same 293 validation cells, deltas in Tab. 3b are computed by paired bootstrap over cell indices. This protocol measures reconstruction under train–test aligned masked decoding; it is not intended to reproduce the full distributional MMD or Wasserstein benchmark suite of Bhattacharya et al. (2026).

GenMol family. For the GenMol V2 pilot, de novo metrics are bootstrapped over generated molecules, because samples are not paired across ordering methods. Fragment-constrained metrics are bootstrapped over the common stable fragment-task units after applying the same method-independent row filter to every checkpoint. The fragment analysis is therefore an ordinary bootstrap over shared task units rather than a paired bootstrap over identical generated molecules. This is appropriate for the pilot because the generated molecules themselves are stochastic and method-specific, while the conditioning fragments and task definitions are shared.

SDPO family. For SDPO, the evaluation unit is a generated DNA sequence. Each method generates 640 sequences from the same sampling budget, and the evaluator records HepG2 oracle score, ATAC success, high-expression k-mer Pearson correlation, reference log-likelihood, and their product-style total metric. Because generated sequences are stochastic and method-specific rather than paired, Tab. 3d uses ordinary bootstrap intervals over generated samples within each method. As with GenMol, these intervals quantify sample-level variability for a fixed checkpoint and seed; they do not substitute for a multi-seed SDPO fine-tuning study.

B.10. Additional Experimental Observations Supporting Theory

Sec. C.4 formalizes a finite-sample optimization story with two parts. The early-stage assumption says that confidence is a useful proxy for locally stable optimization: orders that the current model can reveal confidently should tend to have smaller CE loss and more predictable CE-gradient proxies, so confidence-aligned progressive training can improve faster than random masking under a fixed SGD budget. The late-stage assumption says that confidence can eventually become too narrow: some low-confidence order families may still be useful for improving terminal reward or reducing the remaining forward KL, but confidence-only training visits them too rarely. In the formal theorem these are called *residual families*; operationally, they simply mean under-covered but useful masked-state regions, not a new model component.

We probe these assumptions on Countdown, where difficulty labels are especially interpretable because they correspond to the number of operands in the target arithmetic expression. In addition to the ordinary pass@K evaluations, we ran five instrumented reruns: random aligned-order DMPO, confidence-only Progressive DMPO ($\beta = 0$), and DPRM-DMPO with $\beta \in \{0.5, 1.0, 2.0\}$. Every diagnostic run logs the progressive phase, confidence bin, selected-token indicator, CE loss, true-token log-probability, terminal reward, DPRM score, DPRM value, schedule mix, and CE true-logit gradient proxy. The diagnostics below are still indirect: they use CE and gradient proxies rather than exact forward-KL decrements, and the online DPRM score is bucketized rather than an exact Doob process reward. Their purpose is to check observable implications of the theorem's assumptions.

Optimization trace. We first inspect the training-reward trace in Fig. 15. Random DMPO was resumed from its early checkpoint, so the plotted curve stitches the initial and resumed runs; Progressive DMPO and DMPO-DPRM are plotted from their completed matched runs. This plot mainly targets the early-stage part of the theorem. The separation between train–test aligned progressive ordering and random masking is consistent with the assumption that confidence helps concentrate finite optimization budget on easier and more stable masked states. DPRM remains competitive with the confidence-only progressive scaffold under the same budget, which is expected because the DPRM controller is scheduled to inherit confidence behavior before its process-reward estimates become reliable.

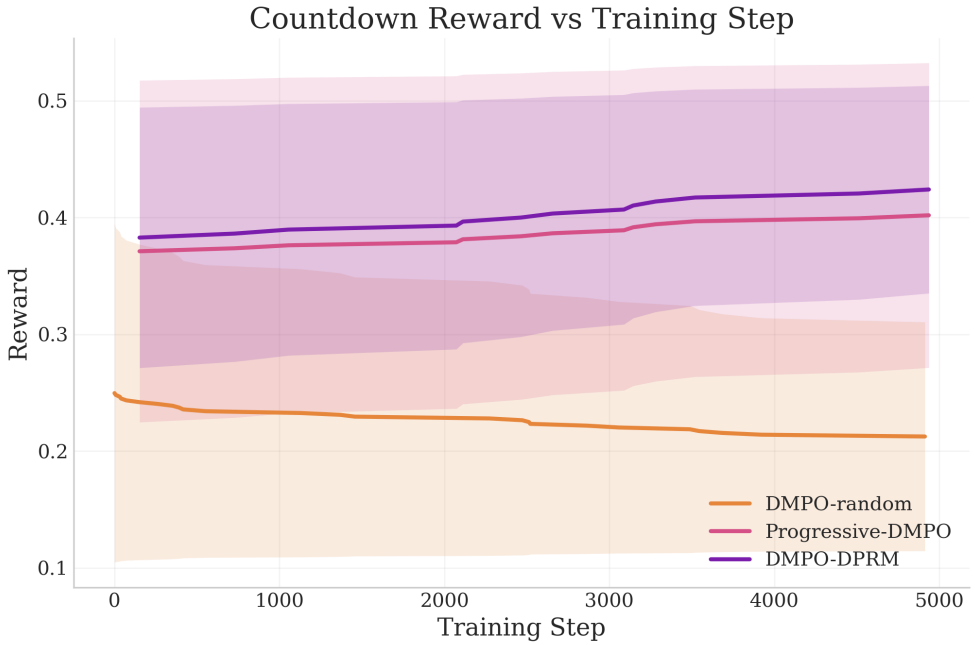

Figure 15. Countdown training reward versus global step from W&B logs. Bands use the logged reward standard deviation. The random DMPO curve stitches the initial run and its resume run; Progressive DMPO and DMPO-DPRM use their completed matched runs. This plot is an optimization diagnostic rather than an evaluation metric.

Token-level evidence for the early proxy assumption. Fig. 16 uses the instrumented reruns rather than final pass@K curves. Panel (a) shows that confidence bins strongly track local denoising difficulty: in the early stage, the weighted correlation between confidence bin and candidate CE loss is negative for all methods, ranging from -0.94 to -0.97 in Tab. 8. Panel (b) compares the selected tokens. Random ordering selects tokens with early CE 0.740, whereas Progressive DMPO and DPRM-DMPO reduce this to 0.477 and 0.442 for the $\beta = 1$ DPRM run. This supports the first theorem assumption in the precise sense used by the proof: confidence does not need to equal the true gradient norm; it only needs to select a locally easier and more stable subfamily often enough to improve finite-sample optimization.

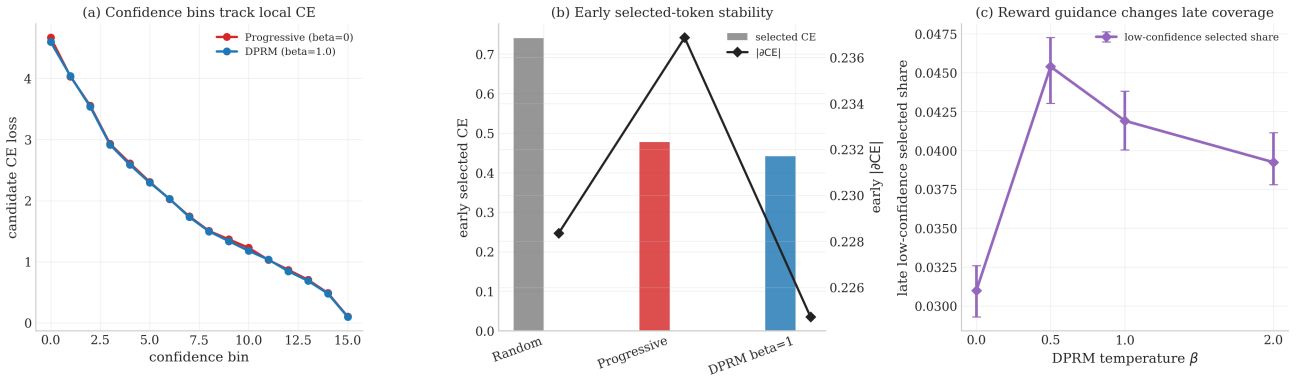

Figure 16. Instrumented Countdown diagnostics for the finite-sample ordering theory. Panel (a) shows that confidence bins are a strong proxy for local CE loss in early training. Panel (b) shows that confidence-aligned controllers select lower-CE tokens than random ordering early. Panel (c) shows the beta sensitivity of late low-confidence selected-token mass. Intervals in Panel (c) are bootstrap intervals over logged confidence-bin snapshots.

Table 8. Countdown instrumentation for the finite-sample ordering theory. Early metrics are computed over steps ≤ 1000; late metrics over steps ≥ 3000. The low-confidence selected share uses bins 0–5 out of 16. Intervals are bootstrap intervals over logged confidence-bin snapshots.

Method	β	**Early CE**	**Early $\lvert\partial\mathrm{CE}\rvert$**	**Corr(conf., CE)**	**Late low-conf. selected %**	**Late DPRM score**
Random	–	0.740	0.228	-0.975	11.4 [10.7, 12.0]	-0.286
Progressive ($\beta = 0$)	0.0	0.477	0.237	-0.942	3.1 [2.9, 3.3]	-0.132
DPRM ($\beta = 0.5$)	0.5	0.438	0.147	-0.948	4.5 [4.3, 4.7]	0.005
DPRM ($\beta = 1.0$)	1.0	0.442	0.225	-0.945	4.2 [4.0, 4.4]	0.206
DPRM ($\beta = 2.0$)	2.0	0.467	0.155	-0.946	3.9 [3.8, 4.1]	0.777

Coverage evidence for the late under-coverage assumption. Fig. 17 probes the second theorem assumption at the confidence-bin level. Confidence-only Progressive DMPO concentrates late reveal decisions almost entirely in high-confidence bins: only 3.1% of selected tokens fall in bins 0–5. DPRM increases this low-confidence selected mass to 4.5%, 4.2%, and 3.9% for $\beta = 0.5, 1.0, 2.0$, respectively, while its selected DPRM score becomes positive and increases with β. Random ordering has broader low-confidence coverage, but it is not reward-targeted and has much worse early selected CE; this is exactly the trade-off addressed by Progressive Online DPRM, which starts from the confidence-aligned scaffold and then adds reward-guided coverage once the process-reward estimates become usable.

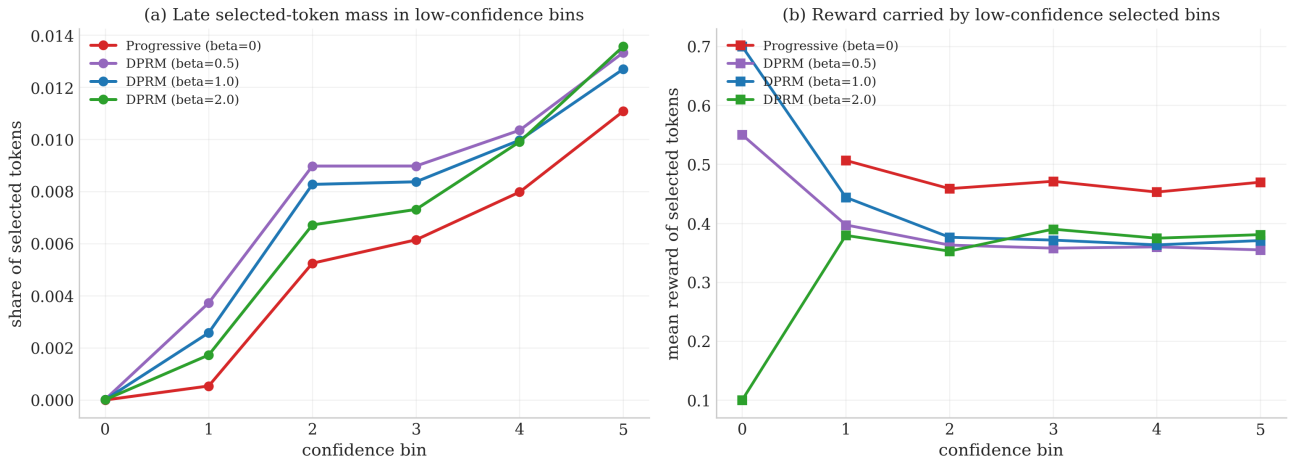

Figure 17. Late-stage coverage inside the low-confidence region on instrumented Countdown reruns. DPRM increases selected-token mass in bins 0–5 relative to confidence-only Progressive DMPO while assigning positive DPRM scores to selected tokens as β grows. This is the bin-level diagnostic corresponding to the theorem's late under-coverage assumption.

Outcome-level complement. Fig. 18 uses the completed pass@K curves to connect the instrumentation to task-level outcomes. Panel (a) again checks the early-stage proxy assumption: Progressive DMPO improves pass@1 over random DMPO most strongly on easier difficulty levels, where a confidence-driven order is most likely to match stable local progress. Panels (b)–(c) compare DMPO-DPRM with Progressive DMPO and target the second assumption. If confidence-only ordering were sufficient, the DPRM gain would not need to grow with sampling budget or difficulty. Instead, DPRM is not uniformly better at low K on the easiest subset, but its advantage grows with both K and difficulty: at pass@32, DPRM improves over Progressive DMPO by 12.0 points on hard Countdown and 10.4 points on OOD Countdown. This is consistent with the bin-level finding that DPRM restores some probability mass to low-confidence regions that confidence-only ordering under-covers.

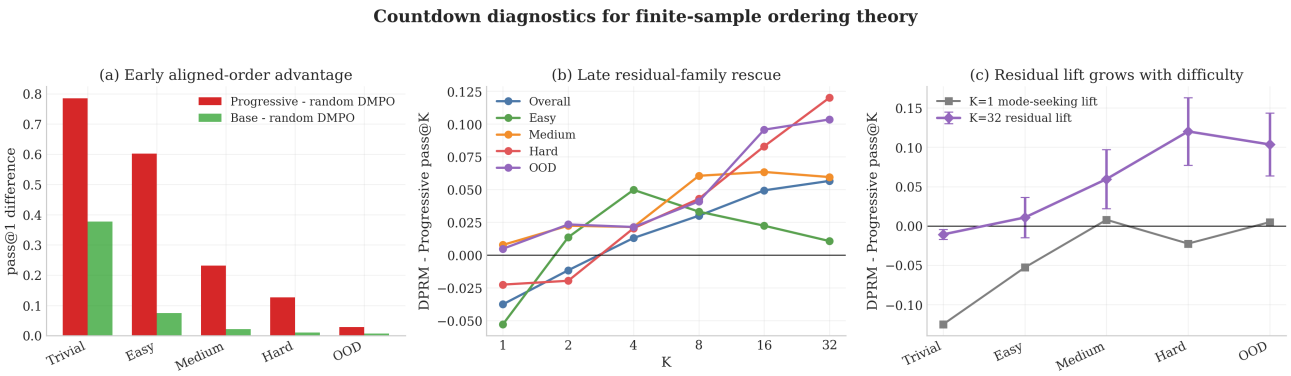

Figure 18. Outcome-level diagnostics for the finite-sample ordering theory on Countdown. Panel (a) supports the early-stage assumption that confidence is a useful local optimization proxy: confidence-aligned progressive training dominates random DMPO at pass@1, especially on easier operand-count subsets. Panels (b)–(c) support the late-stage confidence-undercoverage assumption: DPRM's gain over confidence-only Progressive DMPO grows with sampling budget K and is concentrated on hard and OOD subsets. Error bars are conservative independent-binomial intervals computed from aggregate pass@K curves; the token-level logging evidence is shown separately in Figs. 16 and 17.

C. Formal Statements and Proofs

C.1. Teacher-forced alignment, minimizer preservation, and statistical separation

For completeness and the ease of readers, we restate and prove in our notation the analogues of Proposition 1 and Appendix A.1–A.3 of Kim et al. (2026). We first prove teacher-forced marginal agreement. We then prove the forward-process invariance of the Bayes predictor, which yields minimizer preservation. Finally, we restate the latent-variable sample-complexity separation.

C.1.1. Teacher-forced alignment

We first formalize Proposition 3.1(1) in the single-index reveal setting used in Kim et al. (2026, Appendix A.1).

Proposition C.1 (Teacher-forced marginal agreement)**.** *Fix an integer $K \geq 1$ and a time grid $t_j := 1 - j/K$ for $j = 0, 1, \ldots, K$. Fix a prompt q and let $O \sim p_*(\cdot \mid q)$. Consider the following two Markov chains on partially masked response states, both initialized at the fully masked response state.*

1. ***Idealized posterior-based inference.*** *Given $Z_{t_j} = z$, sample an index*

$$I_j \sim g_\phi(\cdot \mid z, q, t_j),$$

then sample a token

$$U_j \sim p_*(O_{I_j} = \cdot \mid Z_{t_j} = z, Q = q),$$

and set

$$Z_{t_{j+1}} := z^{(I_j \leftarrow U_j)}.$$

Let $q_{t_j}(\cdot \mid q)$ denote the law of Z_{t_j}.

2. ***Teacher-forced chain.*** *First sample* $O \sim p_*(\cdot \mid q)$. *Given* $\widetilde{Z}_{t_j} = z$, *sample*

$$I_j \sim g_\phi(\cdot \mid z, q, t_j),$$

and set

$$\widetilde{Z}_{t_{j+1}} := z^{(I_j \leftarrow O_{I_j})}.$$

Let $\widetilde{q}_{t_j}(\cdot \mid q)$ *denote the marginal law of* $\widetilde{Z}_{t_j}$ *after marginalizing over* O.

Then

$$q_{t_j}(\cdot \mid q) = \widetilde{q}_{t_j}(\cdot \mid q) \qquad \textit{for every } j = 0, 1, \ldots, K.$$

Proof. We show that the two chains have the same one-step transition kernel.

Fix a step j and a masked state z. Under the teacher-forced chain, the event $\{\widetilde{Z}_{t_j} = z\}$ is possible only if the sampled response O agrees with z on the revealed coordinates. Thus, for some scalar $\alpha_j(q, z) \geq 0$ and every response o,

$$\mathbb{P}(\widetilde{Z}_{t_j} = z \mid Q = q, O = o) = \alpha_j(q, z)\, \mathbf{1}\{o_{\mathrm{um}(z)} = z_{\mathrm{um}(z)}\}.$$

By Bayes' rule,

$$\mathbb{P}(O = o \mid Q = q, \widetilde{Z}_{t_j} = z) \propto p_*(o \mid q)\, \mathbf{1}\{o_{\mathrm{um}(z)} = z_{\mathrm{um}(z)}\}.$$

Hence, for every masked coordinate $i \in M(z)$ and every token u,

$$\mathbb{P}(O_i = u \mid Q = q, \widetilde{Z}_{t_j} = z) = p_*(O_i = u \mid Q = q, Z_{t_j} = z).$$

Now fix $i \in M(z)$ and a token value u. Using the conditional identity above and the fact that the reveal policy depends only on the current visible state and time,

$$\begin{aligned}\mathbb{P}\Big(\widetilde{Z}_{t_{j+1}} = z^{(i \leftarrow u)} \mid Q = q, \widetilde{Z}_{t_j} = z\Big) &= g_\phi(i \mid z, q, t_j)\, \mathbb{P}(O_i = u \mid Q = q, \widetilde{Z}_{t_j} = z) \\ &= g_\phi(i \mid z, q, t_j)\, p_*(O_i = u \mid Q = q, Z_{t_j} = z).\end{aligned}$$

On the other hand, the idealized inference chain satisfies

$$\mathbb{P}\Big(Z_{t_{j+1}} = z^{(i \leftarrow u)} \mid Q = q, Z_{t_j} = z\Big) = g_\phi(i \mid z, q, t_j)\, p_*(O_i = u \mid Q = q, Z_{t_j} = z).$$

Thus the two chains have the same one-step transition kernel and the same initial state. An induction on j gives

$$q_{t_j}(\cdot \mid q) = \widetilde{q}_{t_j}(\cdot \mid q) \qquad \text{for all } j.$$

□

Remark C.2. Proposition C.1 is the formal version of Proposition 3.1(1). It is the direct analogue of the discrete-time single-index result in Kim et al. (2026, Appendix A.1).

C.1.2. Minimizer preservation

We now formalize Proposition 3.1(2). This is the Appendix A.2 argument of Kim et al. (2026): the forward-process weight cancels in Bayes' rule, so the Bayes predictor does not depend on the particular masking weights.

For a partially masked response state z, let

$$\mathrm{um}(z) := \{i : z_i \neq \mathsf{M}\}, \qquad M(z) := \{i : z_i = \mathsf{M}\}.$$

Lemma C.3 (Weighted cross-entropy is minimized by the true conditional)**.** *Let* (X, Y) *be jointly distributed random variables, where* Y *takes values in a finite set* $\mathcal{V}$. *Let* $w(X) \geq 0$ *be measurable. Consider*

$$\mathcal{J}(r) := \mathbb{E}\big[w(X)\,(-\log r(Y \mid X))\big],$$

where $r(\cdot \mid x)$ *ranges over all conditional distributions on* $\mathcal{V}$. *Then every minimizer* $r^\star$ *satisfies*

$$r^\star(\cdot \mid x) = \mathbb{P}(Y = \cdot \mid X = x)$$

for almost every x *with* $w(x) > 0$.

Proof. Conditioning on $X = x$ gives

$$\mathbb{E}[-\log r(Y \mid X) \mid X = x] = H(\mathbb{P}(Y = \cdot \mid X = x)) + \mathrm{KL}(\mathbb{P}(Y = \cdot \mid X = x) \,\|\, r(\cdot \mid x)).$$

Multiply by $w(x) \geq 0$ and integrate over X. The entropy term is independent of r, whereas the KL term is nonnegative and vanishes exactly at the true conditional. □

Proposition C.4 (Forward-process invariance of the Bayes predictor)**.** *Let the forward process take the form*

$$\mathbb{P}(Z_t = z \mid Q = q, O = o, t) \propto \alpha_t(q, z)\, \mathbf{1}\{o_{\mathrm{um}(z)} = z_{\mathrm{um}(z)}\}, \tag{13}$$

where $\alpha_t(q, z) \geq 0$ does not depend on o. Let

$$p_f(q, o, t, z) := \nu_*(q, o)\, \mathbf{1}_{[0,1]}(t)\, \mathbb{P}(Z_t = z \mid Q = q, O = o, t).$$

Consider the population loss

$$\mathcal{L}_{\mathrm{fwd}}(\theta) := \mathbb{E}_{(Q,O,t,Z)\sim p_f}\left[\frac{1}{t} \sum_{i \in M(Z)} -\log p_\theta(O_i \mid Z, Q)\right]. \tag{14}$$

Then:

1. *the posterior satisfies*
$$\mathbb{P}(O = o \mid Q = q, Z = z, t) \propto p_*(o \mid q)\, \mathbf{1}\{o_{\mathrm{um}(z)} = z_{\mathrm{um}(z)}\}, \tag{15}$$
so it is independent of $\alpha_t(q, z)$;
2. *every minimizer $\theta^\star$ of (14) satisfies*
$$p_{\theta^\star}(u \mid z, q) = \mathbb{P}(O_i = u \mid Q = q, Z = z) \qquad \text{for almost every } (q, z, i) \text{ with } i \in M(z).$$

Proof. Fix q, t, and z. By Bayes' rule and (13),

$$\mathbb{P}(O = o \mid Q = q, Z = z, t) \propto \mathbb{P}(Z_t = z \mid Q = q, O = o, t)\, p_*(o \mid q) \propto \alpha_t(q, z)\, \mathbf{1}\{o_{\mathrm{um}(z)} = z_{\mathrm{um}(z)}\}\, p_*(o \mid q).$$

Since $\alpha_t(q, z)$ does not depend on o, it cancels under normalization. This proves (15).

Now fix a coordinate i and define

$$X := (Q, Z), \qquad Y := O_i.$$

The contribution of coordinate i to (14) is

$$\mathcal{L}_i(\theta) = \mathbb{E}\left[\frac{1}{t}\, \mathbf{1}\{i \in M(Z)\}\, \big(-\log p_\theta(Y \mid X)\big)\right].$$

By the first part, the conditional law of Y given $X = (q, z)$ is the ground-truth posterior and does not depend on α_t. Applying Lemma C.3 with weight

$$w_i(Q, Z, t) := \frac{1}{t}\, \mathbf{1}\{i \in M(Z)\}$$

shows that every minimizer matches that conditional posterior on every active masked coordinate. Summing over i proves the claim. □

Corollary C.5 (Minimizer preservation for admissible teacher-forced masking)**.** *Suppose that an admissible teacher-forced progressive masking law q_π is induced by a forward process of the form (13). Then every minimizer of (10) satisfies*

$$p_{\theta^\star}(u \mid z, q) = \nu_*(O_i = u \mid Q = q, Z = z)$$

for $\bar{\rho}_\pi$-almost every (q, z, i) with $i \in M(z)$. Hence admissible progressive masking changes only the occupancy measure over masked states, not the Bayes-optimal denoiser.

Proof. This is exactly the second part of Proposition C.4, rewritten under the notation of (10). □

C.1.3. STATISTICAL SEPARATION

We now formalize Proposition 3.1(3). This is the latent-variable separation result corresponding to Kim et al. (2026, Proposition 3).

Lemma C.6 (Chernoff bound)**.** *Let P and Q be distributions on a finite space, and let $Z_1, \dots, Z_T \sim P$ be i.i.d. Then*

$$\mathbb{P}\left(\prod_{t=1}^{T} \frac{Q(Z_t)}{P(Z_t)} \geq 1\right) \leq e^{-TC(P,Q)},$$

where $C(P, Q)$ is the Chernoff information between P and Q.

Proof. For any $s \in (0, 1)$, Markov's inequality gives

$$\mathbb{P}\left(\prod_{t=1}^{T} \frac{Q(Z_t)}{P(Z_t)} \geq 1\right) = \mathbb{P}\left(\Big(\prod_{t=1}^{T} \frac{Q(Z_t)}{P(Z_t)}\Big)^{s} \geq 1\right) \leq \mathbb{E}\left[\prod_{t=1}^{T} \Big(\frac{Q(Z_t)}{P(Z_t)}\Big)^{s}\right].$$

By independence, the right-hand side equals

$$\Big(\sum_{z} P(z)^{1-s} Q(z)^{s}\Big)^{T}.$$

Minimizing over $s \in (0, 1)$ yields the bound. □

For the separation result, we follow the latent-variable setup of Kim et al. (2026, Proposition 3). Let the response be

$$O = \pi(U_{1:d}, Y),$$

where $U_1, \dots, U_d$ are latent coordinates and Y is the distinguished observation coordinate.

Assumption C.7 (Posterior collapse)**.** There exists a fixed distribution π_0 on $\mathcal{V}$ such that for every parameter θ and every strict subset $S \subsetneq [d]$, the posterior law of Y given U_S is π_0. Equivalently, all latents must be known before the observation can be inferred.

Assumption C.8 (Identifiability)**.** Let P_θ denote the joint law of $\pi(U_{1:d}, Y)$ under parameter θ. Assume that there exists $\kappa > 0$ such that

$$C(P_\theta, P_{\theta'}) \geq \kappa \qquad \text{for all } \theta \neq \theta'.$$

Proposition C.9 (Exponential-vs-linear sample-complexity separation)**.** *Let $\delta \in (0, 1)$, and let q be the masking probability under random masking. Under Assumptions C.7 and C.8, random masking requires at least*

$$n = \Omega\big(q^{-1}(1-q)^{-d} \log(1/\delta)\big)$$

samples to drive the error below δ. By contrast, oracle teacher-forced trajectories require only

$$n_{\text{traj}} = O\left((d+1)\frac{\log(|\Theta|/\delta)}{\kappa}\right)$$

samples for MAP recovery. For fixed q, the random-masking dependence is exponential in d, whereas the oracle-trajectory dependence is linear in d.

Proof. First consider random masking. Let F be the event that all latents $U_{1:d}$ are unmasked and Y is masked. Then

$$\mathbb{P}(F) = q(1-q)^d.$$

Hence

$$\mathbb{P}(F^c) = 1 - q(1-q)^d.$$

If all n samples fall in F^c, then by Assumption C.7 they contain no information that distinguishes the parameter values. Therefore no estimator can recover θ better than random guessing over Θ, so

$$\mathbb{P}(\widehat{\theta} \neq \theta) \geq \Big(1 - \frac{1}{|\Theta|}\Big)\big(1 - q(1-q)^d\big)^n.$$

Using $(1-a)^n \geq e^{-an}$ gives

$$\mathbb{P}(\widehat{\theta} \neq \theta) \geq \Big(1 - \frac{1}{|\Theta|}\Big)e^{-nq(1-q)^d}.$$

Thus achieving error at most δ requires

$$n \geq \frac{1}{q(1-q)^d}\log\Big(\frac{1 - 1/|\Theta|}{\delta}\Big).$$

Now consider oracle teacher-forced trajectories. Each trajectory contains one informative state in which all latents are revealed and Y is masked. Let these informative states be $Z_1, \ldots, Z_T \sim P_\theta$. Fix a wrong hypothesis $\theta' \neq \theta$. If MAP selects θ' over θ, then

$$\prod_{t=1}^{T} \frac{P_{\theta'}(Z_t)}{P_\theta(Z_t)} \geq 1.$$

By Lemma C.6 and Assumption C.8,

$$\mathbb{P}(\text{MAP selects } \theta' \text{ instead of } \theta) \leq e^{-TC(P_\theta, P_{\theta'})} \leq e^{-T\kappa}.$$

A union bound over the $|\Theta| - 1$ wrong hypotheses yields

$$\mathbb{P}(\widehat{\theta} \neq \theta) \leq (|\Theta| - 1)e^{-T\kappa}.$$

Hence it suffices to take

$$T \geq \frac{1}{\kappa}\big(\log(|\Theta| - 1) + \log(1/\delta)\big).$$

Since each trajectory has length $d+1$, the total number of states is

$$n_{\text{traj}} = (d+1)T = O\left((d+1)\frac{\log(|\Theta|/\delta)}{\kappa}\right).$$

□

Remark C.10. Proposition C.9 is the formal version of Proposition 3.1(3). It is a finite-sample statement. It does not contradict Corollary C.5, because it concerns the number of informative masked states needed for recovery, not a change in the Bayes-optimal predictor.

C.2. Formal statement and proof of the online tracking theorem

We now state the full version of the online tracking theorem and prove it. The proof has four steps. We first apply an empirical-Bernstein bound to the bucket mean estimator. We then pass to the log-moment reward estimator. Next we add a drift term, following the variation-budget viewpoint standard in non-stationary online learning. Finally we convert uniform score error into reveal regret.

For a reachable state $s = (q, z, t)$, define

$$\eta_t(s) := \max_{i \in M(z)} \eta_t(\phi, b_i(s)), \qquad B_t(s) := \sup_{i \in M(z)} |R_t^\star(i; s)|,$$

$$D_t(s) := \max_{i \in M(z)} D_{\phi, b_i(s)}(t), \qquad D_{\phi,b}(t) := \sum_{u=2}^{t} |\mu_{\phi,b,u} - \mu_{\phi,b,u-1}|.$$

Also define

$$\mathrm{rad}_t(s; \delta) := \max_{i \in M(z)} \frac{1}{\beta \underline{\mu}} \left[\sqrt{\frac{2\widehat{v}_{\phi,b_i(s),t}\log(3KBT^2/\delta)}{N_{\phi,b_i(s),t}}} + \frac{3(e^\beta - 1)\log(3KBT^2/\delta)}{N_{\phi,b_i(s),t}}\right] \tag{16}$$

and

$$\varepsilon_t(s;\delta) := \beta\varepsilon_{\mathrm{abs},t}(s) + \beta\eta_t(s)\,\mathrm{rad}_t(s;\delta) + \beta(1-\eta_t(s))\,B_t(s) + \frac{2}{\underline{\mu}}D_t(s). \tag{17}$$

The bias term in Theorem 3.2 is the explicit quantity

$$\mathrm{Bias}_t(s;\delta) := \beta\varepsilon_{\mathrm{abs},t}(s) + \beta(1-\eta_t(s))\,B_t(s) + \frac{2}{\underline{\mu}}D_t(s),$$

while the empirical-Bernstein part is captured by $\beta\eta_t(s)\,\mathrm{rad}_t(s;\delta)$. We now state the full theorem. For the union bound, let

$$\mathcal{N} := KBT^2,$$

the total number of bucket-time events controlled simultaneously. The quantity $\log(\mathcal{N}/\delta)$ is the term that appears in the informal theorem in the main text.

Theorem C.11 (Online score tracking)**.** *Fix a horizon T and a confidence level $\delta \in (0,1)$. Assume that:*

1. *$R(X_T) \in [0,1]$ almost surely;*
2. *for every reachable state $s = (q,z,t)$ and every $i \in M(z)$,*
$$|R_t^\star(i;s) - \bar{R}_{\phi,b_i(s),t}| \le \varepsilon_{\mathrm{abs},t}(s);$$
3. *$\mu_{\phi,b,t} \ge \underline{\mu} > 0$ for every bucket and time.*

Then, with probability at least $1-\delta$, for all reachable states $s = (q,z,t)$ and all $t \le T$,

$$\sup_{i\in M(z)} |\widehat{g}_t(i;s) - g_t^\star(i;s)| \le \varepsilon_t(s;\delta). \tag{18}$$

Moreover, if $\widehat{A}_t(s)$ is chosen by the practical controller and

$$A_t^\star(s) \in \arg\max_{|A|=m(s)} \sum_{i\in A} g_t^\star(i;s),$$

then

$$0 \le \sum_{i\in A_t^\star(s)} g_t^\star(i;s) - \sum_{i\in \widehat{A}_t(s)} g_t^\star(i;s) \le 2\,m(s)\,\varepsilon_t(s;\delta). \tag{19}$$

Notation. For a bucket (ϕ,b) at time t, define

$$Y_{\phi,b}^{(j)} := \exp(\beta R_j) \in [1,e^\beta], \qquad \mu_{\phi,b,t} := \mathbb{E}[Y_{\phi,b}^{(j)}], \qquad \widehat{\mu}_{\phi,b,t} := \frac{1}{N_{\phi,b,t}} \sum_{j=1}^{N_{\phi,b,t}} Y_{\phi,b}^{(j)},$$

and

$$\bar{R}_{\phi,b,t} := \frac{1}{\beta}\log\mu_{\phi,b,t}, \qquad \widehat{R}_{\phi,b,t} := \frac{1}{\beta}\log\widehat{\mu}_{\phi,b,t}.$$

C.2.1. Step 1: concentration for the bucket mean

Lemma C.12 (Bucket mean concentration)**.** *Fix a bucket (ϕ,b) and time t. Assume that $Y_{\phi,b}^{(1)},\dots,Y_{\phi,b}^{(N_{\phi,b,t})} \in [1,e^\beta]$ are conditionally independent with common mean $\mu_{\phi,b,t}$ and empirical variance $\widehat{v}_{\phi,b,t}$. Then, for every $\delta \in (0,1)$, with probability at least $1-\delta$,*

$$|\widehat{\mu}_{\phi,b,t} - \mu_{\phi,b,t}| \le \sqrt{\frac{2\widehat{v}_{\phi,b,t}\log(3/\delta)}{N_{\phi,b,t}}} + \frac{3(e^\beta-1)\log(3/\delta)}{N_{\phi,b,t}}. \tag{20}$$

Proof. This is the bounded empirical-Bernstein inequality; see, for example, the main empirical-Bernstein theorem of Maurer & Pontil (2009). We apply it to the bounded sample

$$Y_{\phi,b}^{(1)},\dots,Y_{\phi,b}^{(N_{\phi,b,t})} \in [1,e^\beta].$$

□

C.2.2. STEP 2: CONCENTRATION FOR THE LOG-MOMENT REWARD ESTIMATOR

Lemma C.13 (Bucket reward concentration)**.** *Assume also that $\mu_{\phi,b,t} \geq \underline{\mu} > 0$. Then, on the event in (20),*

$$|\widehat{R}_{\phi,b,t} - \bar{R}_{\phi,b,t}| \leq \frac{1}{\beta\underline{\mu}} \left[\sqrt{\frac{2\widehat{v}_{\phi,b,t} \log(3/\delta)}{N_{\phi,b,t}}} + \frac{3(e^{\beta}-1)\log(3/\delta)}{N_{\phi,b,t}} \right]. \tag{21}$$

Proof. Since

$$\widehat{R}_{\phi,b,t} - \bar{R}_{\phi,b,t} = \frac{1}{\beta}\big(\log \widehat{\mu}_{\phi,b,t} - \log \mu_{\phi,b,t}\big),$$

it suffices to control the log map. The derivative of $(1/\beta)\log x$ is $(\beta x)^{-1}$. Because $\mu_{\phi,b,t} \geq \underline{\mu}$, the mean-value theorem gives

$$\left| \frac{1}{\beta} \log \widehat{\mu}_{\phi,b,t} - \frac{1}{\beta} \log \mu_{\phi,b,t} \right| \leq \frac{1}{\beta\underline{\mu}} |\widehat{\mu}_{\phi,b,t} - \mu_{\phi,b,t}|.$$

Substituting (20) proves (21). □

C.2.3. STEP 3: DRIFT-AWARE UNIFORM SCORE ERROR

For each bucket (ϕ, b), define

$$D_{\phi,b}(t) := \sum_{u=2}^{t} |\mu_{\phi,b,u} - \mu_{\phi,b,u-1}|.$$

Lemma C.14 (Drift-aware bucket tracking)**.** *With probability at least $1-\delta$, uniformly over all buckets and all $t \leq T$,*

$$|\widehat{R}_{\phi,b,t} - \bar{R}_{\phi,b,t}| \leq \mathrm{rad}_{\phi,b,t}(\delta) + \frac{2}{\underline{\mu}} D_{\phi,b}(t), \tag{22}$$

where

$$\mathrm{rad}_{\phi,b,t}(\delta) := \frac{1}{\beta\underline{\mu}} \left[\sqrt{\frac{2\widehat{v}_{\phi,b,t} \log(3KBT^2/\delta)}{N_{\phi,b,t}}} + \frac{3(e^{\beta}-1)\log(3KBT^2/\delta)}{N_{\phi,b,t}} \right]. \tag{23}$$

Proof. Apply Thm. C.13 to every bucket and every time $t \leq T$ with failure probability

$$\delta' = \frac{\delta}{KBT^2}.$$

A union bound gives the concentration term uniformly.

It remains to control drift. The cumulative quantity $D_{\phi,b}(t)$ is a variation-budget term of the kind standard in non-stationary online learning; see Besbes et al. (2015). For each u,

$$\bar{R}_{\phi,b,u} - \bar{R}_{\phi,b,u-1} = \frac{1}{\beta}\big(\log \mu_{\phi,b,u} - \log \mu_{\phi,b,u-1}\big).$$

Since $\mu_{\phi,b,u} \geq \underline{\mu}$, the mean-value theorem yields

$$|\bar{R}_{\phi,b,u} - \bar{R}_{\phi,b,u-1}| \leq \frac{1}{\beta\underline{\mu}} |\mu_{\phi,b,u} - \mu_{\phi,b,u-1}|.$$

Summing over $u \leq t$ gives a cumulative drift guidance proportional to $D_{\phi,b}(t)$. Using the slightly looser coefficient $2/\underline{\mu}$ yields (22). □

Lemma C.15 (Uniform score approximation)**.** *Assume that for every reachable state $s = (q, z, t)$ and every $i \in M(z)$,*

$$|R_t^{\star}(i; s) - \bar{R}_{\phi,b_i(s),t}| \leq \varepsilon_{\mathrm{abs},t}(s).$$

Then, on the event of Thm. C.14, for every reachable state $s = (q, z, t)$,

$$\sup_{i \in M(z)} |\widehat{g}_t(i; s) - g_t^{\star}(i; s)| \leq \varepsilon_t(s; \delta),$$

where $\varepsilon_t(s; \delta)$ is the quantity in (17).

Proof. Fix $i \in M(z)$. Then

$$\begin{aligned}|\widehat{g}_t(i;s) - g_t^\star(i;s)| &= \beta \left|\eta_t(\phi, b_i(s))\widehat{R}_{\phi,b_i(s),t} - R_t^\star(i;s)\right| \\ &\le \beta|R_t^\star(i;s) - \bar{R}_{\phi,b_i(s),t}| + \beta\eta_t(\phi,b_i(s))|\widehat{R}_{\phi,b_i(s),t} - \bar{R}_{\phi,b_i(s),t}| \\ &\quad + \beta(1-\eta_t(\phi,b_i(s)))|R_t^\star(i;s)|. \end{aligned} \tag{24}$$

The first term is bounded by the abstraction error assumption. The second term is bounded by Thm. C.14. The third term is bounded by $B_t(s)$. Taking the supremum over $i \in M(z)$ yields exactly (17). □

C.2.4. Step 4: score error implies reveal regret

Lemma C.16 (Top-m regret from uniform score error)**.** *Let*

$$J_t^\star(A;s) := \sum_{i\in A} g_t^\star(i;s), \qquad \widehat{J}_t(A;s) := \sum_{i\in A}\widehat{g}_t(i;s),$$

and let

$$A_t^\star(s) \in \arg\max_{|A|=m(s)} J_t^\star(A;s), \qquad \widehat{A}_t(s) \in \arg\max_{|A|=m(s)} \widehat{J}_t(A;s).$$

If

$$\sup_{i\in M(z)} |\widehat{g}_t(i;s) - g_t^\star(i;s)| \le \varepsilon,$$

then

$$0 \le J_t^\star(A_t^\star(s);s) - J_t^\star(\widehat{A}_t(s);s) \le 2m(s)\varepsilon. \tag{25}$$

Proof. For any feasible set A,

$$|\widehat{J}_t(A;s) - J_t^\star(A;s)| \le \sum_{i\in A}|\widehat{g}_t(i;s) - g_t^\star(i;s)| \le m(s)\varepsilon.$$

Since $\widehat{A}_t(s)$ maximizes $\widehat{J}_t(\cdot;s)$,

$$\widehat{J}_t(\widehat{A}_t(s);s) \ge \widehat{J}_t(A_t^\star(s);s).$$

Therefore

$$\begin{aligned} J_t^\star(A_t^\star(s);s) &\le \widehat{J}_t(A_t^\star(s);s) + m(s)\varepsilon \\ &\le \widehat{J}_t(\widehat{A}_t(s);s) + m(s)\varepsilon \\ &\le J_t^\star(\widehat{A}_t(s);s) + 2m(s)\varepsilon. \end{aligned} \tag{26}$$

This proves (25). □

Proof of Theorem C.11. The score bound (18) is exactly Thm. C.15. Applying Thm. C.16 with $\varepsilon = \varepsilon_t(s;\delta)$ gives (19). □

Corollary C.17 (Exact recovery under a margin)**.** *Fix a reachable state $s = (q,z,t)$, and define*

$$\Delta_t(s) := \min_{i\in A_t^\star(s),\, j\notin A_t^\star(s)} \big(g_t^\star(i;s) - g_t^\star(j;s)\big).$$

If $\Delta_t(s) > 0$ and

$$\varepsilon_t(s;\delta) < \frac{\Delta_t(s)}{2},$$

then

$$\widehat{A}_t(s) = A_t^\star(s).$$

Proof. Fix $i \in A_t^\star(s)$ and $j \notin A_t^\star(s)$. Then

$$\widehat{g}_t(i;s) - \widehat{g}_t(j;s) \ge g_t^\star(i;s) - g_t^\star(j;s) - 2\varepsilon_t(s;\delta) > 0.$$

Thus every element of $A_t^\star(s)$ still ranks above every element outside it under $\widehat{g}_t$. Hence the top-$m(s)$ set is unchanged. □

C.3. Proof of Theorem 3.3

We prove a stronger pathwise statement and then pass to the terminal marginal. The derivation has two ingredients. First, the exact DPRM reveal law is a Doob h-transform of the base reveal chain; see the classical Doob h-transform construction and the DPRM derivation in Bu et al. (2025b). Second, the stagewise Soft-BoN approximation follows from the main KL approximation theorem of Verdun et al. (2025).

Setup. Let

$$\tau = (S_0, A_0, S_1, A_1, \dots, S_{T-1}, A_{T-1}, S_T)$$

be a reveal trajectory. At state $s \in \mathcal{S}_t$, the action $a \in \mathcal{A}(s)$ determines the next state $s^a \in \mathcal{S}_{t+1}$ under teacher forcing. Let $R(X_T) \in [0,1]$ be the terminal reward. Using the base proposal $q_0(\cdot \mid s)$ from (1), define

$$\mathbb{P}_0(\tau) = \rho_0(S_0) \prod_{t=0}^{T-1} q_0(A_t \mid S_t), \qquad \mathbb{P}_\beta(\tau) = \frac{\exp(\beta R(X_T(\tau)))}{Z_\beta}\, \mathbb{P}_0(\tau),$$

where

$$Z_\beta := \mathbb{E}_{\mathbb{P}_0}[\exp(\beta R(X_T))]\,.$$

Also define

$$h_t(s) := \mathbb{E}_{\mathbb{P}_0}[\exp(\beta R(X_T)) \mid S_t = s]\,.$$

Lemma C.18 (Backward recursion for h_t)**.** *For every non-terminal state $s \in \mathcal{S}_t$,*

$$h_t(s) = \sum_{a \in \mathcal{A}(s)} q_0(a \mid s)\, h_{t+1}(s^a),$$

and

$$h_T(s) = \exp(\beta R(x(s))),$$

where $x(s)$ is the fully revealed response represented by s.

Proof. By the tower property,

$$h_t(s) = \mathbb{E}_{\mathbb{P}_0}[\mathbb{E}_{\mathbb{P}_0}[\exp(\beta R(X_T)) \mid S_{t+1}] \mid S_t = s] = \mathbb{E}_{\mathbb{P}_0}[h_{t+1}(S_{t+1}) \mid S_t = s]\,.$$

Since $S_{t+1} = s^a$ once $A_t = a$ is chosen,

$$\mathbb{E}_{\mathbb{P}_0}[h_{t+1}(S_{t+1}) \mid S_t = s] = \sum_{a \in \mathcal{A}(s)} q_0(a \mid s)\, h_{t+1}(s^a).$$

At time T, there is no future randomness, so

$$h_T(s) = \mathbb{E}_{\mathbb{P}_0}[\exp(\beta R(X_T)) \mid S_T = s] = \exp(\beta R(x(s))).$$

□

Proposition C.19 (Exact reveal rule under the tilted path law)**.** *For every non-terminal state $s = (q, z, t)$, the one-step conditional of $\mathbb{P}_\beta$ is exactly the Gibbs reveal law in (3).*

Proof. This is the standard Doob h-transform form of the tilted path law; see the classical construction and the DPRM specialization in Bu et al. (2025b). For completeness, we verify the identity in our notation.

Fix s and a. By the definition of $\mathbb{P}_\beta$,

$$\mathbb{P}_\beta(A_t = a \mid S_t = s) \propto q_0(a \mid s)\, \mathbb{E}_{\mathbb{P}_0}[\exp(\beta R(X_T)) \mid S_t = s, A_t = a]\,.$$

Under teacher forcing, the event $A_t = a$ implies $S_{t+1} = s^a$. Therefore

$$\mathbb{E}_{\mathbb{P}_0}[\exp(\beta R(X_T)) \mid S_t = s, A_t = a] = \mathbb{E}_{\mathbb{P}_0}[\exp(\beta R(X_T)) \mid S_{t+1} = s^a] = h_{t+1}(s^a).$$

Hence

$$\mathbb{P}_\beta(A_t = a \mid S_t = s) \propto q_0(a \mid s)\, h_{t+1}(s^a).$$

Using $R_t^\star(a; s) = \beta^{-1} \log h_{t+1}(s^a)$, this is exactly (3). □

Lemma C.20 (Bounded process reward). *If $R(X_T) \in [0,1]$, then $R_t^\star(a;s) \in [0,1]$ for every non-terminal state s and action $a \in \mathcal{A}(s)$.*

Proof. Since $R(X_T) \in [0,1]$,

$$1 \le \exp(\beta R(X_T)) \le e^\beta.$$

Conditioning on $S_{t+1} = s^a$ preserves these bounds:

$$1 \le \mathbb{E}[\exp(\beta R(X_T)) \mid S_{t+1} = s^a] \le e^\beta.$$

Applying $(1/\beta)\log$ gives

$$0 \le R_t^\star(a;s) \le 1.$$

□

Statewise Soft-BoN approximation. Fix a non-terminal state s, and let

$$A_{t,1}, \dots, A_{t,N} \overset{\text{i.i.d.}}{\sim} q_0(\cdot \mid s).$$

Given this candidate multiset, define

$$\widehat{\pi}_{t,N}(a \mid s; A_{t,1:N}) := \frac{\sum_{j=1}^N \mathbf{1}\{A_{t,j} = a\} \exp(\beta R_t^\star(a;s))}{\sum_{j=1}^N \exp(\beta R_t^\star(A_{t,j};s))}.$$

Proposition C.21 (Local Soft-BoN KL bound). *Fix a non-terminal state s. Assume that $\mathcal{A}(s)$ is finite and $R_t^\star(a;s) \in [0,1]$ for all $a \in \mathcal{A}(s)$. Then*

$$\mathbb{E}[\mathrm{KL}(\pi_t^\star(\cdot \mid s) \,\|\, \widehat{\pi}_{t,N}(\cdot \mid s; A_{t,1:N}))] \le \frac{\sinh(\beta/2)^2}{N}. \tag{27}$$

Proof. Apply the main KL approximation theorem of Verdun et al. (2025) to the finite proposal distribution

$$P(\cdot) = q_0(\cdot \mid s)$$

and reward function

$$r(a) = R_t^\star(a;s) \in [0,1].$$

Choose the theorem's temperature parameter so that the resulting exponential tilt matches β. Then the tilted target becomes

$$P^\star(a) \propto P(a)\exp(\beta r(a)) = q_0(a \mid s)\exp(\beta R_t^\star(a;s)) = \pi_t^\star(a \mid s).$$

Their empirical Soft-BoN law is exactly $\widehat{\pi}_{t,N}(\cdot \mid s; A_{t,1:N})$. Therefore their KL bound yields (27). □

Pathwise lifting. Let $N_t(S_t)$ be the shortlist size used at step t. Conditioned on all shortlist randomness, let $\widehat{\mathbb{P}}_{\{N_t\}}$ be the resulting approximate reveal trajectory law.

Lemma C.22 (KL chain rule for reveal trajectories). *Assume that $\mathbb{P}_\beta$ and $\widehat{\mathbb{P}}_{\{N_t\}}$ have the same initial distribution and the same deterministic state-update map $s \mapsto s^a$, and differ only in their one-step action conditionals $\pi_t^\star(\cdot \mid s)$ and $\widehat{\pi}_{t,N_t(s)}(\cdot \mid s)$. Then, conditioned on the shortlist randomness,*

$$\mathrm{KL}\Big(\mathbb{P}_\beta \,\Big\|\, \widehat{\mathbb{P}}_{\{N_t\}}\Big) = \mathbb{E}_{\mathbb{P}_\beta}\left[\sum_{t=0}^{T-1} \mathrm{KL}\big(\pi_t^\star(\cdot \mid S_t) \,\big\|\, \widehat{\pi}_{t,N_t(S_t)}(\cdot \mid S_t)\big)\right]. \tag{28}$$

Proof. Under the assumptions, both path laws factorize as

$$\mathbb{P}_\beta(\tau) = \rho_0(S_0) \prod_{t=0}^{T-1} \pi_t^\star(A_t \mid S_t)$$

and

$$\widehat{\mathbb{P}}_{\{N_t\}}(\tau) = \rho_0(S_0) \prod_{t=0}^{T-1} \widehat{\pi}_{t,N_t(S_t)}(A_t \mid S_t).$$

Hence

$$\log \frac{d\mathbb{P}_\beta}{d\widehat{\mathbb{P}}_{\{N_t\}}}(\tau) = \sum_{t=0}^{T-1} \log \frac{\pi_t^\star(A_t \mid S_t)}{\widehat{\pi}_{t,N_t(S_t)}(A_t \mid S_t)}.$$

Taking expectation under $\mathbb{P}_\beta$ and conditioning on S_t gives (28). □

Proposition C.23 (Pathwise lifting of the local Soft-BoN bound)**.** *Under the assumptions of Theorem 3.3,*

$$\mathbb{E}\Big[\mathrm{KL}\Big(\mathbb{P}_\beta \,\Big\|\, \widehat{\mathbb{P}}_{\{N_t\}}\Big)\Big] \le \sinh(\beta/2)^2 \, \mathbb{E}_{\mathbb{P}_\beta}\left[\sum_{t=0}^{T-1} \frac{1}{N_t(S_t)}\right]. \tag{29}$$

In particular, if $N_t(s) \equiv N$, then

$$\mathbb{E}\Big[\mathrm{KL}\Big(\mathbb{P}_\beta \,\Big\|\, \widehat{\mathbb{P}}_N\Big)\Big] \le \frac{T \, \sinh(\beta/2)^2}{N}. \tag{30}$$

Proof. Condition on the shortlist randomness. By Thm. C.22,

$$\mathrm{KL}\Big(\mathbb{P}_\beta \,\Big\|\, \widehat{\mathbb{P}}_{\{N_t\}}\Big) = \mathbb{E}_{\mathbb{P}_\beta}\left[\sum_{t=0}^{T-1} \mathrm{KL}\big(\pi_t^\star(\cdot \mid S_t) \,\big\|\, \widehat{\pi}_{t,N_t(S_t)}(\cdot \mid S_t)\big)\right].$$

Now take expectation over the shortlist randomness. Since the sum is finite, Tonelli's theorem gives

$$\mathbb{E}\Big[\mathrm{KL}\Big(\mathbb{P}_\beta \,\Big\|\, \widehat{\mathbb{P}}_{\{N_t\}}\Big)\Big] = \mathbb{E}_{\mathbb{P}_\beta}\left[\sum_{t=0}^{T-1} \mathbb{E}\big[\mathrm{KL}\big(\pi_t^\star(\cdot \mid S_t) \,\big\|\, \widehat{\pi}_{t,N_t(S_t)}(\cdot \mid S_t)\big) \,\big|\, S_t\big]\right].$$

For each realized state $S_t = s$, Thm. C.21 yields

$$\mathbb{E}\big[\mathrm{KL}\big(\pi_t^\star(\cdot \mid s) \,\big\|\, \widehat{\pi}_{t,N_t(s)}(\cdot \mid s)\big)\big] \le \frac{\sinh(\beta/2)^2}{N_t(s)}.$$

Substituting this bound gives (29). If $N_t(s) \equiv N$, then

$$\sum_{t=0}^{T-1} \frac{1}{N_t(S_t)} = \frac{T}{N},$$

so (30) follows. □

Proof of Theorem 3.3. By Thm. C.23,

$$\mathbb{E}\Big[\mathrm{KL}\Big(\mathbb{P}_\beta \,\Big\|\, \widehat{\mathbb{P}}_N\Big)\Big] \le \frac{T \, \sinh(\beta/2)^2}{N}.$$

The terminal output is a measurable function of the full reveal trajectory. Hence the data-processing inequality for KL divergence gives

$$\mathbb{E}[\mathrm{KL}(\nu_\beta \,\|\, \widehat{\nu}_N)] \le \mathbb{E}\Big[\mathrm{KL}\Big(\mathbb{P}_\beta \,\Big\|\, \widehat{\mathbb{P}}_N\Big)\Big] \le \frac{T \, \sinh(\beta/2)^2}{N},$$

which is exactly (12). □

C.4. Finite-sample forward-KL separations for Progressive Online DPRM

This appendix studies a finite-sample question left open by Proposition 3.1. Admissible token orders do not change the population minimizer, so the relevant issue is not asymptotic consistency but optimization complexity: which aligned order family drives the training KL below a target level ε fastest?

We analyze this question in two stages. In the early stage, we show that if confidence is a good proxy for local gradient scale, then confidence-driven progressive training is exponentially faster than random aligned-order sampling. This view mirrors curriculum learning (Bu et al., 2025a) and active learning (Bu et al., 2024). In the late stage, we show that if confidence-only training undersamples a residual order family that is still necessary for final KL convergence, then a sufficiently accurate DPRM guidance yields an exponential speedup over confidence-only training.

The analysis uses three standard ingredients. First, in unbiased importance-weighted SGD, the conditional second moment is minimized by sampling in proportion to gradient norm (Zhao & Zhang, 2015; El Hanchi et al., 2022). Second, arbitrary-sampling SGD admits convergence recursions controlled by that second moment (Gower et al., 2019; Chen et al., 2023). Third, confidence gating imposes a hard entropy cap and thus a support-narrowing effect (Fang et al., 2026, Proposition 2).

Stagewise forward-KL objective. Fix a training stage t. For each sample x, let $\mathcal{O}_t(x)$ be an aligned local order family. This may consist of reveal actions, reveal prefixes, or any other local order objects that can be visited both by teacher-forced training and by the corresponding inference-time controller at stage t. Let $u_t(\cdot \mid x)$ be a reference aligned sampling law on $\mathcal{O}_t(x)$.

Let $p_t^\star(\cdot \mid x, o)$ be the target conditional at stage t and let $p_\theta(\cdot \mid x, o)$ be the model conditional. Define the stagewise forward-KL objective

$$\mathcal{K}_t(\theta) := \mathbb{E}_{x\sim\widehat{P}_n}\mathbb{E}_{o\sim u_t(\cdot|x)}\Big[\mathrm{KL}\big(p_t^\star(\cdot \mid x, o) \,\|\, p_\theta(\cdot \mid x, o)\big)\Big]. \tag{31}$$

Equivalently, $\mathcal{K}_t(\theta)$ is the excess stagewise cross-entropy up to an additive constant independent of θ.

Let $\ell_t(\theta; x, o, \xi)$ be a stochastic per-order loss whose conditional expectation equals the integrand of (31), and define

$$g_t(\theta; x, o, \xi) := \nabla_\theta \ell_t(\theta; x, o, \xi), \qquad \nu_t(\theta; x, o) := \Big(\mathbb{E}_\xi \|g_t(\theta; x, o, \xi)\|_2^2\Big)^{1/2}.$$

If $O_t \sim p_t(\cdot \mid X_t)$, the unbiased importance-weighted gradient estimator is

$$\widehat{G}_t^{(p)} := \frac{u_t(O_t \mid X_t)}{p_t(O_t \mid X_t)}\, g_t(\theta_t; X_t, O_t, \xi_t), \tag{32}$$

with conditional second moment

$$\mathcal{M}_t(p) := \mathbb{E}\Big[\|\widehat{G}_t^{(p)}\|_2^2 \mid \theta_t\Big]. \tag{33}$$

Assumption C.24 (Stagewise smoothness and PL geometry)**.** At stage t, the forward-KL objective $\mathcal{K}_t$ is L_t-smooth and satisfies the Polyak–Łojasiewicz inequality with constant $\mu_t > 0$:

$$\frac{1}{2}\|\nabla\mathcal{K}_t(\theta)\|_2^2 \geq \mu_t\big(\mathcal{K}_t(\theta) - \mathcal{K}_t^\star\big) \qquad \text{for all relevant } \theta.$$

Assumption C.25 (Confidence as a proxy for local gradient scale)**.** There exists a confidence-induced order score $c_t(x, o) \geq 0$ and a constant $\kappa_t \geq 1$ such that

$$\kappa_t^{-1}\nu_t(\theta_t; x, o) \leq c_t(x, o) \leq \kappa_t \nu_t(\theta_t; x, o) \qquad \text{for all } x, o.$$

Early stage: confidence beats random.

Assumption C.26 (Early importance concentration)**.** For each sample x, there exists a subset $\mathcal{E}_t(x) \subseteq \mathcal{O}_t(x)$ such that:

1. **Most gradient mass lies on $\mathcal{E}_t(x)$:**

$$\sum_{o\in\mathcal{E}_t(x)} u_t(o \mid x)\, \nu_t(\theta_t; x, o) \geq (1 - \rho_t) \sum_{o\in\mathcal{O}_t(x)} u_t(o \mid x)\, \nu_t(\theta_t; x, o)$$

   for some $\rho_t \in [0, 1)$;

2. **The easy family is exponentially smaller than the full aligned family:**

$$\frac{|\mathcal{O}_t(x)|}{|\mathcal{E}_t(x)|} \geq e^{a_t d_t}$$

for some stage-difficulty parameter $d_t \geq 1$ and constant $a_t > 0$.

Lemma C.27 (Variance-optimal proposal over aligned order families)**.** *For fixed θ_t, the proposal minimizing $\mathcal{M}_t(p)$ satisfies*

$$p_t^{\mathrm{opt}}(o \mid x) \propto u_t(o \mid x)\, \nu_t(\theta_t; x, o). \tag{34}$$

Proof. Condition on x. By (32) and (33),

$$\mathcal{M}_t(p \mid x) = \sum_{o \in \mathcal{O}_t(x)} \frac{u_t(o \mid x)^2}{p_t(o \mid x)}\, \nu_t(\theta_t; x, o)^2.$$

Hence, for each fixed x, the problem is

$$\min_{p(\cdot|x) \in \Delta(\mathcal{O}_t(x))} \sum_{o \in \mathcal{O}_t(x)} \frac{u_t(o \mid x)^2 \nu_t(\theta_t; x, o)^2}{p(o \mid x)}.$$

This is exactly the conditional-variance minimization problem of importance-sampling SGD (Zhao & Zhang, 2015; El Hanchi et al., 2022). By Cauchy–Schwarz or a Lagrange multiplier, the minimizer is

$$p_t^{\mathrm{opt}}(o \mid x) = \frac{u_t(o \mid x)\nu_t(\theta_t; x, o)}{\sum_{o'} u_t(o' \mid x)\nu_t(\theta_t; x, o')}.$$

□

Define the confidence proposal

$$p_t^{\mathrm{conf}}(o \mid x) := \frac{u_t(o \mid x)c_t(x, o)}{\sum_{o'} u_t(o' \mid x)c_t(x, o')}, \tag{35}$$

and the random aligned proposal

$$p_t^{\mathrm{rand}}(o \mid x) := \frac{1}{|\mathcal{O}_t(x)|}. \tag{36}$$

Lemma C.28 (Second-moment comparison)**.** *Under Assumptions C.25 and C.26,*

$$\mathcal{M}_t\big(p_t^{\mathrm{conf}}\big) \leq \kappa_t^2\, \mathcal{M}_t\big(p_t^{\mathrm{opt}}\big), \tag{37}$$

and

$$\mathcal{M}_t\big(p_t^{\mathrm{rand}}\big) \geq (1 - \rho_t)^2 e^{a_t d_t}\, \mathcal{M}_t\big(p_t^{\mathrm{opt}}\big). \tag{38}$$

Proof. The first bound follows from Assumption C.25: the proposal p_t^{conf} is a κ_t^2-multiplicative approximation to p_t^{opt}, so substituting into (33) yields (37).

For the second bound, condition on x and write

$$a_o := u_t(o \mid x)\nu_t(\theta_t; x, o).$$

Then

$$\mathcal{M}_t(p_t^{\mathrm{rand}} \mid x) = |\mathcal{O}_t(x)| \sum_{o \in \mathcal{O}_t(x)} a_o^2 \geq |\mathcal{O}_t(x)| \sum_{o \in \mathcal{E}_t(x)} a_o^2.$$

By Cauchy–Schwarz,

$$\sum_{o \in \mathcal{E}_t(x)} a_o^2 \geq \frac{1}{|\mathcal{E}_t(x)|} \left(\sum_{o \in \mathcal{E}_t(x)} a_o \right)^2.$$

Using Assumption C.26,

$$\sum_{o\in\mathcal{E}_t(x)} a_o \ge (1-\rho_t)\sum_{o\in\mathcal{O}_t(x)} a_o.$$

Therefore

$$\mathcal{M}_t(p_t^{\mathrm{rand}} \mid x) \ge \frac{|\mathcal{O}_t(x)|}{|\mathcal{E}_t(x)|}(1-\rho_t)^2\left(\sum_{o\in\mathcal{O}_t(x)} a_o\right)^2.$$

But

$$\left(\sum_{o\in\mathcal{O}_t(x)} a_o\right)^2 = \mathcal{M}_t(p_t^{\mathrm{opt}} \mid x),$$

and $|\mathcal{O}_t(x)|/|\mathcal{E}_t(x)| \ge e^{a_t d_t}$. Averaging over x proves (38). □

Define the ε-sample complexity at stage t by

$$T_t^{(p)}(\varepsilon) := \min\Big\{N : \mathbb{E}[\mathcal{K}_t(\theta_N) - \mathcal{K}_t^\star] \le \varepsilon\Big\}.$$

Theorem C.29 (Early-stage exponential separation: confidence vs random)**.** *Suppose Assumptions C.24, C.25, and C.26 hold. Consider constant-step unbiased SGD using the estimator* (32)*.*

Then there exists a stepsize choice for confidence-driven aligned training such that

$$T_t^{(\mathrm{conf})}(\varepsilon) = O\left(\frac{L_t\kappa_t^2\,\mathcal{M}_t(p_t^{\mathrm{opt}})}{\mu_t^2\,\varepsilon}\log\frac{\mathcal{K}_t(\theta_0)-\mathcal{K}_t^\star}{\varepsilon}\right). \tag{39}$$

Moreover, any constant-stepsize random aligned-order SGD whose asymptotic error floor is at most $\varepsilon/2$ must satisfy

$$T_t^{(\mathrm{rand})}(\varepsilon) = \Omega\left(\frac{L_t(1-\rho_t)^2 e^{a_t d_t}\,\mathcal{M}_t(p_t^{\mathrm{opt}})}{\mu_t^2\,\varepsilon}\log\frac{\mathcal{K}_t(\theta_0)-\mathcal{K}_t^\star}{\varepsilon}\right). \tag{40}$$

Hence confidence-driven progressive training enjoys an exponential sample-complexity improvement over random aligned-order training, up to the calibration factor κ_t^2.

Proof. For any proposal p, smoothness and unbiasedness give the standard recursion

$$\mathbb{E}[\mathcal{K}_t(\theta_{k+1}) - \mathcal{K}_t^\star \mid \theta_k] \le (1-\mu_t\eta_t)\big(\mathcal{K}_t(\theta_k)-\mathcal{K}_t^\star\big) + \frac{L_t\eta_t^2}{2}\mathcal{M}_t(p),$$

as in the arbitrary-sampling SGD literature (Gower et al., 2019; Chen et al., 2023).

For the confidence proposal, choose

$$\eta_t = \Theta\left(\frac{\mu_t\varepsilon}{L_t\mathcal{M}_t(p_t^{\mathrm{conf}})}\right).$$

Then the additive noise floor is $O(\varepsilon)$, and unrolling the recursion yields

$$T_t^{(\mathrm{conf})}(\varepsilon) = O\left(\frac{L_t\mathcal{M}_t(p_t^{\mathrm{conf}})}{\mu_t^2\,\varepsilon}\log\frac{\mathcal{K}_t(\theta_0)-\mathcal{K}_t^\star}{\varepsilon}\right).$$

Using (37) gives (39).

For the random proposal, any constant stepsize with asymptotic floor at most $\varepsilon/2$ must satisfy

$$\frac{L_t\eta_t}{2\mu_t}\mathcal{M}_t(p_t^{\mathrm{rand}}) \le \frac{\varepsilon}{2},$$

hence

$$\eta_t \le \frac{\mu_t \varepsilon}{L_t \mathcal{M}_t(p_t^{\mathrm{rand}})}.$$

Therefore the contraction factor cannot exceed $1 - \mu_t \eta_t$, so to shrink the initial KL down to ε requires at least

$$\Omega\left(\frac{L_t \mathcal{M}_t(p_t^{\mathrm{rand}})}{\mu_t^2\, \varepsilon} \log \frac{\mathcal{K}_t(\theta_0) - \mathcal{K}_t^\star}{\varepsilon}\right)$$

iterations. Using (38) yields (40). □

Late stage: DPRM beats confidence-only. We now analyze the stage after warmup. Here the issue is no longer variance-optimality, but undercoverage. We assume that there remains a residual family of orders that is necessary for final KL convergence, but confidence-only training visits it too rarely.

We first restate the entropy-cap result of Fang et al. (2026, Proposition 2).

Proposition C.30 (Entropy cap under confidence gating (Fang et al., 2026, Prop. 2))**.** *Assume the decoder is $(1-\delta)$-gated. Then the induced sequence distribution X satisfies*

$$H(X) \le L h_V(\delta), \qquad B_{\mathrm{eff}} := \exp(H(X)/L) \le \exp(h_V(\delta)), \tag{41}$$

where $h_V(\delta) = h_b(\delta) + \delta \log(|V| - 1)$.

Assumption C.31 (Late-stage residual family and confidence undercoverage)**.** For each late-stage state s, let $\mathcal{O}_t(s)$ be the reachable local order family and let $\mathcal{R}_t(s) \subseteq \mathcal{O}_t(s)$ be a residual family with the following properties.

1. **Residual score gap.** The exact DPRM score gap

$$\Delta_t^{\mathrm{res}}(s) := \inf_{o \in \mathcal{R}_t(s)} g_t^\star(o; s) - \sup_{o \in \mathcal{O}_t(s) \setminus \mathcal{R}_t(s)} g_t^\star(o; s)$$

   is strictly positive.

2. **Confidence undercoverage.** Under confidence-only training,

$$\pi_t^{\mathrm{conf}}(\mathcal{R}_t(s) \mid s) \le C_t e^{-b_t h_t}$$

   for some structural difficulty parameter $h_t \ge 1$, constant $b_t > 0$, and $C_t \ge 1$.

3. **Residual KL contraction.** Let $\mathcal{K}_t^{\mathrm{res}}(\theta)$ denote the contribution of $\mathcal{R}_t$ to the stagewise forward KL. There exists $\gamma_t \in (0, 1]$ such that, conditional on sampling an order from $\mathcal{R}_t(s)$,

$$\mathbb{E}[\mathcal{K}_{t+1}^{\mathrm{res}}(\theta) \mid \theta_t,\, O_t \in \mathcal{R}_t(s)] \le (1 - \gamma_t) \mathcal{K}_t^{\mathrm{res}}(\theta_t),$$

   whereas conditional on sampling outside $\mathcal{R}_t(s)$,

$$\mathbb{E}[\mathcal{K}_{t+1}^{\mathrm{res}}(\theta) \mid \theta_t,\, O_t \notin \mathcal{R}_t(s)] \le \mathcal{K}_t^{\mathrm{res}}(\theta_t).$$

Theorem C.32 (Late-stage exponential separation: DPRM vs confidence-only)**.** *Assume Assumption C.31. Let*

$$\epsilon_t(s) := \sup_{o \in \mathcal{O}_t(s)} |\widehat{g}_t(o; s) - g_t^\star(o; s)|$$

be the total stage-2 score error, including online estimation and any shortlist approximation error. Suppose that after warmup,

$$\epsilon_t(s) < \frac{\Delta_t^{\mathrm{res}}(s)}{2} \qquad \textit{for all relevant } s.$$

Then:

1. the practical stage-2 proposal over $\mathcal{O}_t(s)$ assigns residual-family mass at least

$$\widehat{\pi}_t(\mathcal{R}_t(s) \mid s) \geq \frac{|\mathcal{R}_t(s)|\, e^{\Delta_t^{\mathrm{res}}(s)-2\epsilon_t(s)}}{|\mathcal{R}_t(s)|\, e^{\Delta_t^{\mathrm{res}}(s)-2\epsilon_t(s)} + |\mathcal{O}_t(s) \setminus \mathcal{R}_t(s)|}; \tag{42}$$

2. if, in addition,

$$\Delta_t^{\mathrm{res}}(s) - 2\epsilon_t(s) \geq b_t h_t + \log \frac{|\mathcal{O}_t(s) \setminus \mathcal{R}_t(s)|}{|\mathcal{R}_t(s)|} + c_t$$

for some $c_t > 0$, then

$$\widehat{\pi}_t(\mathcal{R}_t(s) \mid s) \geq \frac{1}{1+e^{-c_t}} =: p_t^{\mathrm{DPRM}}, \tag{43}$$

whereas

$$\pi_t^{\mathrm{conf}}(\mathcal{R}_t(s) \mid s) \leq C_t e^{-b_t h_t}.$$

Consequently, if

$$T_{t,\mathrm{late}}^{(\mathrm{conf})}(\varepsilon) \quad \textit{and} \quad T_{t,\mathrm{late}}^{(\mathrm{DPRM})}(\varepsilon)$$

denote the numbers of late-stage updates needed to drive the residual forward KL below ε, then

$$T_{t,\mathrm{late}}^{(\mathrm{conf})}(\varepsilon) = \Omega\left(\frac{e^{b_t h_t}}{\gamma_t C_t} \log \frac{\mathcal{K}_t^{\mathrm{res}}(\theta_{T_{\mathrm{warm}}})}{\varepsilon}\right), \tag{44}$$

whereas

$$T_{t,\mathrm{late}}^{(\mathrm{DPRM})}(\varepsilon) = O\left(\frac{1}{\gamma_t p_t^{\mathrm{DPRM}}} \log \frac{\mathcal{K}_t^{\mathrm{res}}(\theta_{T_{\mathrm{warm}}})}{\varepsilon}\right). \tag{45}$$

Thus, under the stated score-gap condition, stage-2 DPRM enjoys an exponential sample-complexity improvement over confidence-only training in the late stage.

Proof. For any $o \in \mathcal{R}_t(s)$ and $o' \notin \mathcal{R}_t(s)$,

$$\widehat{g}_t(o;s) - \widehat{g}_t(o';s) \geq g_t^\star(o;s) - g_t^\star(o';s) - 2\epsilon_t(s) \geq \Delta_t^{\mathrm{res}}(s) - 2\epsilon_t(s).$$

Exponentiating and summing over $\mathcal{R}_t(s)$ and its complement yields (42). If the stronger gap condition holds, the right-hand side is at least

$$\frac{1}{1+e^{-c_t}},$$

which proves (43).

For the residual forward KL, let p be the probability that the current proposal samples from $\mathcal{R}_t(s)$. By Assumption C.31(3),

$$\mathbb{E}[\mathcal{K}_{k+1}^{\mathrm{res}} \mid \theta_k] \leq (1-\gamma_t p)\, \mathcal{K}_k^{\mathrm{res}}.$$

Iterating gives

$$\mathbb{E}[\mathcal{K}_k^{\mathrm{res}}] \leq (1-\gamma_t p)^k\, \mathcal{K}_0^{\mathrm{res}} \leq e^{-\gamma_t p k}\, \mathcal{K}_0^{\mathrm{res}}.$$

Hence reaching ε requires

$$k \geq \frac{1}{\gamma_t p} \log \frac{\mathcal{K}_0^{\mathrm{res}}}{\varepsilon}.$$

Under confidence-only training, $p \leq C_t e^{-b_t h_t}$, which yields (44). Under DPRM, $p \geq p_t^{\mathrm{DPRM}}$, which yields (45). □

Corollary C.33 (Practical stage-2 separation under online tracking)**.** *Assume the conditions of Theorem C.32. Suppose further that the online score-tracking theorem gives*

$$\epsilon_t(s) = O\left(\beta \eta_t(s) \sqrt{\frac{\log(\mathcal{N}/\delta)}{N_{\min,t}(s)}} + \beta \eta_t(s) \frac{\log(\mathcal{N}/\delta)}{N_{\min,t}(s)} + \mathrm{Bias}_t(s;\delta)\right)$$

with probability at least $1-\delta$, as in Theorem 3.2. If T_{warm} is chosen so that this bound is smaller than $\Delta_t^{\mathrm{res}}(s)/2$ for all relevant late-stage states, then the exponential late-stage separation (44) and (45) holds for the practical Progressive Online DPRM controller.

Proof. Immediate from Theorem 3.2 and Theorem C.32. □

Remark C.34. The results above separate the roles of the two stages. The early confidence stage is justified by optimization-noise reduction and can be exponentially faster than random aligned-order training under the importance-concentration assumption. The late DPRM stage is justified by residual-family rescue and can be exponentially faster than confidence-only training once the online score is accurate enough and the residual family remains necessary for final KL convergence. Together, these results provide a finite-sample explanation for why Progressive Online DPRM can help even though admissible orders do not change the population minimizer.

Table 5. Actual DPRM controller settings used in the reported experiments. Here m_t denotes the host reveal/update budget at step t, and N_t denotes the sampled Soft-BoN shortlist size.

Experiment	Phase / bucketization	$T_{\text{warm}}, T_{\text{switch}}, N_{\text{ready}}$	Host update budget m_t	Shortlist N_t
DPRM-PUMA	Train: progressive horizon K scheduled from 12 to 42, with 16 confidence bins. Decode: 16 phase buckets.	Train: $(2\text{k}, 60\text{k}, 128)$. Decode: $(0, 16, 16)$.	Train: phase-induced reveal count $m_t = \max\{\text{round}(r_{t+1}L_{\text{eff}}) - u_t, 0\}$, where r_{t+1} is the next sampled interval ratio and u_t is the current number of revealed tokens; confidence-collapse extras use $\tau = 0.9$. Decode: fixed $m_t \in \{2, 3\}$.	Sampled Soft-BoN with $N_t = \min\{64, \max(8, 4m_t)\}$.
DPRM-DMPO	8 progressive phases and 16 confidence bins.	Train: $(500, 2000, 128)$. Decode: checkpoint-local estimator with force-full gate.	Train: $m_t = \lceil M_t/P_t \rceil$, where M_t is the remaining masked completion budget and P_t the remaining phases; optional confidence-collapse extras use $\tau = 0.9$. Decode: the native Fast-dLLM transfer schedule inside each 32-token block over the local denoising steps.	Sampled Soft-BoN with $N_t = \min\{32, \max(8, 4m_t)\}$.
DPRM-Prism	8 phase buckets and 16 confidence bins.	With $T = 32$ steps, $(T_{\text{warm}}, T_{\text{switch}}, N_{\text{ready}}) = (6, 22, 64)$, corresponding to $(0.2T, 0.7T, 64)$.	Prism's native HTS transfer schedule with block length 32 and 32 denoising steps, so $m_t = 1$ per step inside each block. During low-gate early steps, the threshold fallback remains active.	Sampled Soft-BoN with $N_t = \min\{64, \max(8, 4m_t)\}$.
DPRM-DPLM	8 phases, 16 confidence bins, and one active structural bucket.	Progressive baseline: $(0, 0, 256)$. DPRM and DPRM(random): $(2000, 20000, 256)$.	In both train and decode controllers, $m_t = \lceil M_t/P_t \rceil$ with M_t the remaining masked design positions and P_t the remaining phases. The reported runs set the confidence-collapse threshold to 0, so no extra threshold reveals occur.	Sampled Soft-BoN with $N_t = \min\{32, \max(8, 4m_t)\}$.
DPRM-DCM	8 reveal phases and 16 confidence bins over discretized gene-expression tokens.	Progressive baseline: $(0, 0, 128)$. DPRM variants: $(500, 2000, 128)$.	Train and decode use $m_t = \lceil M_t/P_t \rceil$, where M_t is the remaining masked gene-position budget and P_t is the number of remaining reveal phases.	Sampled Soft-BoN with $N_t = \min\{64, \max(8, 4m_t)\}$.
DPRM-GenMol	8 reveal phases and 16 confidence bins over SAFE molecular tokens.	Progressive baseline: $(0, 0, 128)$. DPRM variants: $(500, 2000, 128)$.	De novo and fragment-conditioned decoding use the host masked-token reveal budget induced by the remaining SAFE masks and reveal phases. The pilot keeps GenMol V2's sampler temperatures and randomness settings fixed.	Sampled Soft-BoN with $N_t = \min\{64, \max(8, 4m_t)\}$.
DPRM-SDPO	8 reveal phases and 10 confidence bins over DNA sequence tokens.	Progressive baseline: confidence-only schedule. DPRM variants: $(100, 400, 64)$.	Train and decode use $m_t = \lceil M_t/P_t \rceil$, where M_t is the remaining masked DNA-token budget and P_t is the remaining reveal phases. The SDPO objective uses $K = 2000$, 2 epochs, and the original reward-optimization hyperparameters.	Sampled Soft-BoN with $N_t = 64$.

Table 6. Per-question bootstrap and paired-bootstrap uncertainty for PUMA at the shared 1.53M checkpoint. Accuracies and deltas are reported in percentage points with 95% percentile intervals from 5,000 bootstrap resamples over the 1,319 GSM8K test questions.

Setting	PUMA top-k	DPRM-PUMA	Δ (DPRM $-$ top-k)
Unmasking-2	29.95 [27.45, 32.45]	34.12 [31.54, 36.69]	+4.17 [1.74, 6.60]
Unmasking-3	28.73 [26.31, 31.16]	34.42 [31.92, 36.92]	+5.69 [3.18, 8.26]

Table 7. Paired-bootstrap deltas in average pass@K over the six reported values $K \in \{1, 2, 4, 8, 16, 32\}$. All values are in percentage points, with 95% percentile intervals from 5,000 paired bootstrap resamples over shared evaluation examples.

Task	Progressive DMPO $-$ DMPO	DMPO-DPRM $-$ Progressive DMPO
GSM8K	+0.64 [−0.39, 1.71]	−0.05 [−1.09, 0.97]
MATH	+2.97 [0.20, 5.97]	+2.43 [−0.03, 4.90]
Countdown	+45.59 [44.44, 46.69]	+1.67 [0.84, 2.52]

Algorithm 2 Practical DPRM-DMPO Training with Progressive Teacher-Forced Unmasking

Require: Initial diffusion LM p_θ, reference model p_{ref}, reward $r(\cdot)$, progressive phases K, buffer reuse length U, confidence bins B, DPRM temperature β, warmup T_{warm}, switch step T_{switch}, readiness threshold N_{ready}

Initialize DPRM statistics $\{N_{\phi,b}, S_{\phi,b}\}_{\phi=0\ b=0}^{K-1\ B-1} \leftarrow 0$
for rollout refresh cycle $m = 1, 2, \ldots$ **do**
  Sample prompts $q \sim \mathcal{D}$ and generate clean completions $x \sim \pi_\theta(\cdot \mid q)$ using the standard DMPO rollout sampler
  Compute rewards $R \leftarrow r(q, x)$ and importance weights / advantages $w(x)$
  Replicate (q, x, R, w) according to the WDCE training scheme
  Initialize progressive teacher-forced states $z^{(0)} \leftarrow \textsc{InitProgressiveState}(x, K)$
  for $u = 0$ to $U - 1$ **do**
    $t \leftarrow$ current global optimization step; compute logits $\ell \leftarrow f_\theta(z^{(u)})$
    Compute weighted denoising loss $\mathcal{L}_{\mathrm{WDCE}}(\theta; x, z^{(u)}, w)$ on masked positions
    for each sample n in the mini-batch **do**
      Let $M_n = \{i : z_{n,i}^{(u)} = \texttt{[MASK]}\}$, $\phi_n = \textsc{Phase}(z_n^{(u)})$, and $m_n = \textsc{RevealBudget}(\phi_n)$
      for each $i \in M_n$ **do**
        $p_{n,i} \leftarrow \max_v \mathrm{softmax}(\ell_{n,i})_v, \quad b_{n,i} \leftarrow \textsc{Bin}(p_{n,i})$
        $\widehat{R}_{n,i} \leftarrow \frac{1}{\beta} \log\left(\frac{S_{\phi_n, b_{n,i}}}{\max(N_{\phi_n, b_{n,i}}, 1)}\right)$
        $\eta_{n,i} \leftarrow \mathrm{clip}\left(\frac{t - T_{\mathrm{warm}}}{T_{\mathrm{switch}} - T_{\mathrm{warm}}}, 0, 1\right) \cdot \min\left(\frac{N_{\phi_n, b_{n,i}}}{N_{\mathrm{ready}}}, 1\right)$
        $s_{n,i} \leftarrow \log p_{n,i} + \eta_{n,i} \widehat{R}_{n,i}$
      end for
      Optionally sample shortlist $C_n \subseteq M_n$ from proposal $q(i) \propto p_{n,i}$
      Reveal set $A_n \leftarrow \mathrm{TopK}_{i \in C_n \text{ or } M_n}(s_{n,i}, m_n)$
      Teacher-force reveal: set $z_{n,i}^{(u+1)} \leftarrow x_{n,i}^\star$ for all $i \in A_n$
      for each $i \in A_n$ **do**
        $N_{\phi_n, b_{n,i}} \leftarrow N_{\phi_n, b_{n,i}} + 1$
        $S_{\phi_n, b_{n,i}} \leftarrow S_{\phi_n, b_{n,i}} + \exp(\beta R_n)$
      end for
      Optionally confidence-collapse positions with $p_{n,i} > \tau$
      if $z_n^{(u+1)}$ reaches the terminal phase **then**
        Reinitialize sample n from phase 0
      end if
    end for
    Update θ using $\nabla_\theta \mathcal{L}_{\mathrm{WDCE}}$
  end for
end for
return trained model p_θ and estimator $\{N_{\phi,b}, S_{\phi,b}\}$

Algorithm 3 Practical Aligned DPRM Soft-BoN Decoding for DPRM-DMPO

Require: Trained model p_θ, learned estimator $\{N_{\phi,b}, S_{\phi,b}\}$, prompt q, decoding horizon T

Initialize all non-prompt positions as `[MASK]`
for $t = 0, 1, \ldots, T-1$ **do**
  Compute logits $\ell \leftarrow f_\theta(z^{(t)})$ and provisional token predictions $\hat{x}_i$ on masked positions
  Let $p_i = p_\theta(\hat{x}_i \mid z^{(t)})$ and $\phi \leftarrow \textsc{PhaseFromDecodeStep}(t)$
  for each masked position i **do**
    $b_i \leftarrow \textsc{Bin}(p_i)$
    $\widehat{R}_i \leftarrow \frac{1}{\beta} \log\left(\frac{S_{\phi,b_i}}{\max(N_{\phi,b_i},1)}\right)$
    $\eta_i \leftarrow \min\left(\frac{N_{\phi,b_i}}{N_{\text{ready}}}, 1\right)$
    $s_i \leftarrow \log p_i + \eta_i \widehat{R}_i$
  end for
  Optionally sample shortlist C from $q(i) \propto p_i$
  Reveal positions $A \leftarrow \mathrm{TopK}_{i \in C \text{ or all masked}}(s_i, m_t)$
  Set $z_i^{(t+1)} \leftarrow \hat{x}_i$ for all $i \in A$
end for
return decoded sample $\hat{x}$