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

Dake Bu [1 2] **Wei Huang** [3 4] **Andi Han** [5] **Hau-San Wong** [1] **Qingfu Zhang** [1] **Taiji Suzuki** [6 3] **Atsushi Nitanda** [2 7]

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 stagewise Soft-BoN approximation, show that the online bucketized controller tracks the exact DPRM score at empirical-Bernstein rates, and derive conditional sample-complexity advantages under explicit optimization assumptions. Across nine hosts covering language reasoning, test-time scaling, protein, single-cell, molecular, DNA, text-to-image generation, and VQA, DPRM order variants improve several language, DNA, and multimodal settings while also identifying boundary cases where confidence-only ordering or task-specific utilities are preferable. Code is available at: https://github.com/DakeBU/DPRM-DLLM.

1. Introduction

Autoregressive LLMs have achieved remarkable performance in mathematics, coding, and agentic problem solving (Jaech et al., 2024; Anthropic, 2025; Guo et al., 2025; Kimi Team et al., 2025), but left-to-right generation is a modeling choice rather than a universal property of the data. Many scientific objects are governed by global constraints and bidirectional dependencies: diffusion protein language models outperform same-scale autoregressive protein LMs on function, localization, interaction, annotation, and foldability tasks (Wang et al., 2024); masked diffusion improves regulatory DNA generation by modeling DNA bidirectionally (Yang et al., 2026); and masked discrete diffusion is a natural fit for sparse, non-sequential single-cell transcriptomes (Zhang et al., 2026). These results suggest that scientific generation may need global, flexible-order denoising rather than a fixed serial order. The same shift is also visible in natural language: large diffusion LMs such as LLaDA (Nie et al., 2025) and Dream (Ye et al., 2025b) show that autoregression is no longer the only viable default, diffusion-based reasoning methods continue to close the quality gap, and recent systems such as Mercury (Inception Labs et al., 2025) and Gemini Diffusion (DeepMind, 2025) suggest that flexible-order generation can also offer practical throughput advantages (Ye et al., 2025a; Zhu et al., 2025c).

This flexibility, however, exposes a new algorithmic question that autoregressive models largely hide:

how should a diffusion language model order its tokens?

Related non-diffusion work reaches a similar conclusion for graphs and molecules, where learned-order autoregressive models treat the generation order itself as a state-dependent policy (Wang et al., 2025c). Recent masked-diffusion work has likewise 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) and Park et al. (2026) also show its limits: confidence-

[1]City University of Hong Kong [2]CFAR and IHPC, A*STAR [3]RIKEN AIP [4]The Institute of Statistical Mathematics [5]University of Sydney [6]The University of Tokyo [7]Nanyang Technological University. Correspondence to: Atsushi Nitanda <atsushi_nitanda@a-star.edu.sg>, Hau-San Wong <cshswong@cityu.edu.hk>.

Published as a paper at the 1st FoGen workshop, ICML 2026, Seoul, South Korea, 2026. Copyright 2026 by the author(s).

Table 1. Nine host settings used to test DPRM as a plug-in token-ordering module. The first seven are the main language/scientific hosts; the last two are multimodal extensions for visual-token and image-conditioned text-token ordering.

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)
DPRM-Omni	Omni-Diffusion	Generative modeling	Text-to-image visual-token generation	Li et al. (2026)
DPRM-LLaDA-V	LLaDA-V	Vision-language decoding	Image-conditioned VQA	You et al. (2025)

based decoding can be myopic, suppress useful exploration, prematurely reveal end-of-text tokens, and create a quality–exploration trade-off when globally valuable trajectories are not locally the most confident ones.

This raises the central question of the paper: *can token ordering use task reward information beyond local confidence without changing the host diffusion model?* We answer yes by treating confidence-only progressive ordering and reward-tilted ordering as points in the same DPRM family. Confidence provides the efficient default and corresponds to the limit where the reward gate never contributes. Reward-tilted DPRM becomes useful when the controller has seen enough comparable phase/confidence buckets to identify order classes that are not locally most confident but historically lead to higher terminal utility.

The construction is inspired by Bu et al. (2026), which interprets reward learning as reweighting Markov trajectories. Token ordering in a DLLM is also a Markov process over partially revealed states. DPRM therefore changes only the reveal policy: the host model, loss, data, and token sampler remain fixed. The exact controller is a Doob-style conditional expectation over completed rewards; the implemented controller approximates it online with low-dimensional buckets and readiness gates.

We provide theoretical support at three levels. Stagewise Soft-BoN converges to the exact DPRM target at rate $O(1/N)$ in terminal KL. The bucketized estimator tracks the exact score at empirical-Bernstein rates, up to coarsening, warmup, and nonstationarity terms. Under explicit optimization assumptions, the theory predicts when confidence alignment beats random masking and when reward-tilted ordering can improve over the confidence-only limit. The experiments therefore report task metrics together with cost, bucket-coverage, and entropy-control diagnostics.

We evaluate nine host settings: natural-language pretraining, reasoning post-training, test-time scaling, protein diffusion, single-cell diffusion, molecular diffusion, DNA reward optimization, text-to-image visual-token generation, and image-conditioned vision-language decoding. DPRM-PUMA improves the shared-checkpoint GSM8K mean from 29.34 to 34.27; DMPO-DPRM improves MATH Hard from 44.3 to 47.9 and Countdown Hard from 29.6 to 33.4; DPRM-Prism improves GSM8K voted accuracy from 82.41 to 83.85 with higher NFE; DPRM(random)-SDPO improves the DNA total metric from 1.141 to 2.119; DPRM-confidence improves Omni-Diffusion CLIP-L/14 cosine from 0.24744 to 0.24915 and LLaDA-V AI2D accuracy from 0.658 to 0.692. Other rows are deliberately mixed: DPLM is near-tied or non-reward-tilt-favorable, GenMol is metric-specific, and LLaDA-V RealWorldQA favors confidence-progressive decoding. These outcomes support a precise claim: token order is a reusable control axis, and reward-tilted DPRM helps when the bucketized conditional utility is covered and aligned with the task. Extended related work is deferred 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 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

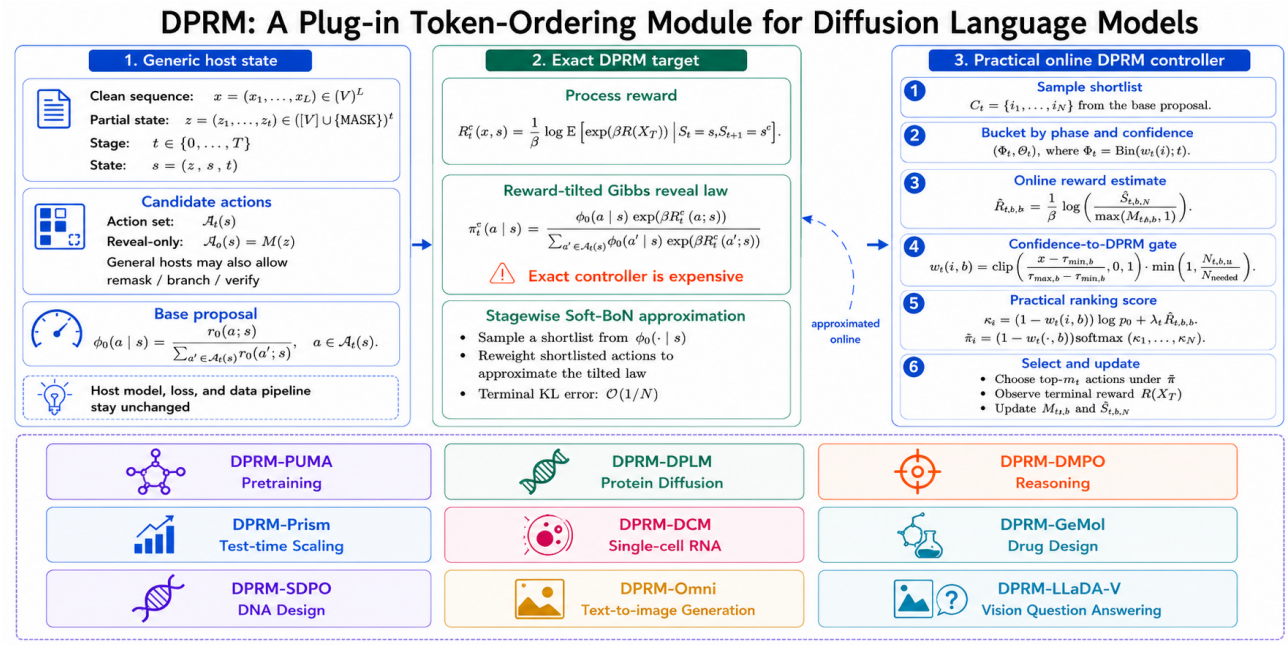

Figure 1. DPRM as a plug-in token-ordering module. The host provides a local proposal over candidate actions, and DPRM replaces only the ordering rule. In practice, DPRM starts from confidence-based warmup and then uses shortlist-based Soft-BoN reweighting together with an online reward estimate to approximate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

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 $(S_0, S_1, \dots, S_T)$ 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 $q_0(\cdot|\cdot)$

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

where $\psi(a;s)$ is the host's incumbent local score. The target distribution is $\pi_T^* \propto q_0 \exp(\beta R(X_T))$, with temperature β controls the trade-off between maintaining base model's coverage or obey the terminal reward.

The exact DPRM process reward is the log-moment future

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 $C_t = \{i_1, \dots, i_{N_t}\}$ from the base proposal over $\mathcal{I}(s_t)$
for each shortlisted candidate $i \in C_t$ **do**
  compute bucket index $b_i \leftarrow \mathrm{Bin}(\psi_i(s_t))$
  estimate process reward $\widehat{R}_{\phi_t,b_i} \leftarrow \frac{1}{\beta}\log\left(\frac{S_{\phi_t,b_i}}{\max(N_{\phi_t,b_i},1)}\right)$
  compute gate $\eta_{t,i} \leftarrow \eta_t(\phi_t, b_i)$ by (7)
  set confidence score $u_i \leftarrow \log \psi_i(s_t)$
  set DPRM score $g_i \leftarrow \log \psi_i(s_t) + \beta \widehat{R}_{\phi_t,b_i}$
  combine $s_i \leftarrow (1-\eta_{t,i})u_i + \eta_{t,i} g_i$
end for
rank the shortlist C_t by s_i
let $\widehat{A}_t(s_t)$ 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 $i \in \widehat{A}_t(s_t)$ **do**
  update $N_{\phi_t,b_i} \leftarrow N_{\phi_t,b_i} + 1$ and $S_{\phi_t,b_i} \leftarrow S_{\phi_t,b_i} + \exp(\beta R(X_T))$
end for

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. 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, which assumes that useful ordering behavior can often be organized by low-dimensional variables rather than by exact partial sequences. This assumption is also supported by recent confidence-decoding analyses: Park et al. (2026), for example, identify EOT status and suffix-anchor proximity as low-dimensional variables that strongly affect diffusion decoding behavior. A *phase* is a coarse label for where the host is in its reveal or denoising process, for example early, middle, or late. This is the variable called `phase_id` in the implementation. A *confidence bucket* is a coarse bin of the base score $\psi_i(s_t)$, so candidates with similar confidence share statistics. This is the variable called `confidence_bin`. The pair (ϕ, b) defines a small decision cell: decisions made at a similar stage and with similar confidence. The exact state s_t is still the full partial sequence state, including the current visible tokens, mask pattern, prompt or conditioning variables, and step index. The bucketized controller does not memorize this full state; it projects the candidate decision (s_t, i) to a cell

$$\zeta_t(s_t, i) = (\phi_t(s_t), b_t(s_t, i)), \qquad b_t(s_t, i) = \mathrm{Bin}(\psi_i(s_t)).$$

Thus an unseen partial sequence can still receive nonzero DPRM guidance if its current host confidence and phase place it in a bucket that was populated by earlier trajectories. 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 candidates.

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 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 addresses the statistical waste of random masking in regimes where confidence is an informative proxy, and it inherits the relevant low-entropy efficiency intuition 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

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

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 when the aligned trajectory family is much smaller than the random-mask family. Second, Theorem 3.2 shows that 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 gives a conditional sample-complexity separation between DPRM, confidence-driven order, and random order under explicit stagewise optimization 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

$$\nu_*(q, o) = \nu_Q(q)\, p_*(o \mid q) \tag{9}$$

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

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

Also define the weighted masked-state occupancy measure

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

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, and the empirical-Bernstein radius diagnostic is reported in Appendix B.15.

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 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.15 reports diagnostics designed to probe observable consequences of these assumptions rather than only final task scores. Countdown logs confidence bins, selected tokens, CE loss, true-token log-probability, CE-gradient proxies, reward, and DPRM scores during training. SDPO-DNA adds entropy-only, shuffled-bucket, and gate/count controls, while LLaDA-V adds entropy, EOT-suppression, and suffix-anchor controls inspired by Park et al. (2026). For the optimization-theory notation, 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.

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	Mean NFE
Prism	82.41	82.11	84.61	609
DPRM-Prism	**83.85**	**83.70**	**86.58**	1,071

(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, the accuracy gain is a quality–cost trade-off: mean NFE increases from 609 to 1,071 while SVF calls remain 29. Appendix B.5 reports the compact cost-quality sweep used to measure this overhead.

Table 3. Multimodal ordering results. Omni-Diffusion tests visual-codebook token ordering in text-to-image generation, while LLaDA-V tests generated text-token ordering under image conditioning. All metrics are computed on the same split within each row, and higher is better.

Task	Metric / ordered object	Random	Progressive	DPRM(conf.)	DPRM(random)	Interpretation
Omni-Diffusion T2I	CLIP-L/14 image–text cosine over 64 official-step prompts; visual codebook-token positions.	0.22184	0.24744	**0.24915**	0.21456	Small DPRM-confidence gain; DPRM-random is a negative warmup choice.
LLaDA-V AI2D	Target-normalized accuracy over 500 image-conditioned VQA examples; generated text-token positions.	0.628	0.658	**0.692**	0.654	DPRM-confidence improves over both random and progressive confidence.
LLaDA-V Real-WorldQA	Target-normalized accuracy over 765 image-conditioned VQA examples; generated text-token positions.	0.44575	**0.46013**	0.41699	0.43791	Negative transfer under the coarse text-token bucket table; progressive confidence remains best.

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

- Early stage: $T_t^{\mathrm{conf}}(\varepsilon) = \widetilde{O}\left(\frac{\kappa_t^2}{\varepsilon}\right)$, $T_t^{\mathrm{rand}}(\varepsilon) = \widetilde{\Omega}\left(\frac{e^{a_t d_t}}{\varepsilon}\right)$ *for some stage-difficulty constant* $a_t > 0$. *Thus confidence-driven progressive training can be exponentially faster than random-order training in the latent-family regime described by the assumption.*
- 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* $T_{t,\mathrm{late}}^{\mathrm{DPRM}}(\varepsilon) = \widetilde{O}\left(\log \frac{1}{\varepsilon}\right)$, $T_{t,\mathrm{late}}^{\mathrm{conf}}(\varepsilon) = \widetilde{\Omega}\left(e^{b_t h_t} \log \frac{1}{\varepsilon}\right)$ *for some bottleneck constant* $b_t > 0$. *Thus DPRM can have exponentially smaller late-stage sample complexity than confidence-only training when the residual family and score-gap conditions hold.*

Overall, the theorem predicts two possible finite-sample separations: first from confidence alignment over random masking, and then from reward-guided DPRM over confidence-only ordering. The constants and residual-family assumptions are not treated as directly measured empirical quantities; the experiments in Appendix B.15 report observable diagnostics that are consistent with, but do not fully identify, these assumptions.

4. Experiments

We evaluate DPRM as a *plug-in token-ordering module* in nine hosts: PUMA (Kim et al., 2026), DMPO (Zhu et al., 2025c), Prism (Bai et al., 2026), DPLM-2 Bit (Wang et al., 2025b; Hsieh et al., 2025), DCM (Bhattacharya et al., 2026), GenMol V2 (Lee et al., 2025), SDPO (Wang et al., 2025a), Omni-Diffusion (Li et al., 2026), and LLaDA-V (You et al., 2025). Within each host, the model, objective, reward oracle, data pipeline, and evaluation protocol are fixed as far as the host allows; only the ordering controller changes. We compare random/default ordering, confidence-only progressive DPRM, and reward-tilted DPRM variants. Task utilities and schedules are in Tabs. 5 and 6 and Secs. B.3 to B.5, B.6.1 and B.8 to B.12; Inference cost and time cost are deferred to Appendix B.2 and B.13.

4.1. DPRM on Natural Language

We first evaluate masked-diffusion pretraining, reward-aware post-training, and test-time scaling. The intervention is the same in all three: keep the host pipeline fixed and

Table 4. 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 ordering comparison. All rows use the same DPLM-2 Bit host and evaluation protocol; the method name specifies only the ordering policy. Default and confidence-progressive are non-reward-tilted reference orders, and the protein results are near-tied or non-reward-tilt-favorable rather than DPRM-dominant.

Method	CAMEO RMSD ↓	CAMEO TM ↑	CoGen TM ↑	CoGen pLDDT ↑
DPLM-2 Bit	35.47	**0.3071**	**0.4063**	67.49
Progressive-DPLM-2 Bit	35.47	0.3066	0.4043	**70.95**
DPRM-DPLM-2 Bit	**35.41**	0.3053	0.3565	59.35
DPRM(random)-DPLM-2 Bit	35.46	0.3055	0.3675	62.59

(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	66.76 [66.10, 67.41]	0.758 [0.739, 0.778]	82.83 [82.59, 83.07]
Progressive-DCM	**76.07** [75.06, 77.03]	**0.628** [0.596, 0.662]	99.75 [99.73, 99.77]
DPRM(random)-DCM	75.92 [74.87, 76.87]	0.650 [0.617, 0.683]	**99.86** [99.85, 99.87]
DPRM(conf.)-DCM	76.00 [74.98, 76.96]	0.642 [0.609, 0.675]	99.83 [99.81, 99.84]

(c) GenMol V2 ordering comparison. 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. The reported DPRM rows use the phase-independent bucket setting selected by the coverage/utility analysis. Higher is better; log-likelihood is better when less negative.

Method	Phases	Total	HepG2	ATAC	K-mer Pearson	Log lik.
SDPO	–	1.141 [1.003, 1.276]	3.918 [3.813, 4.028]	0.348 [0.311, 0.383]	0.836 [0.830, 0.844]	-238.39 [-239.00, -237.83]
Progressive-SDPO	–	0.384 [0.318, 0.454]	4.138 [4.058, 4.212]	0.168 [0.141, 0.195]	0.553 [0.544, 0.563]	**-224.31** [-225.41, -223.16]
DPRM-SDPO	1	1.423 [1.282, 1.571]	**4.304** [4.193, 4.412]	0.401 [0.363, 0.441]	0.825 [0.820, 0.831]	-233.61 [-234.95, -232.28]
DPRM(random)-SDPO	1	**2.119** [1.973, 2.262]	3.337 [3.233, 3.447]	**0.754** [0.720, 0.788]	**0.842** [0.837, 0.847]	-250.72 [-251.53, -249.88]

replace only the ordering controller. Tab. 2 shows when reward tilt adds to the confidence-only limit. DPRM-PUMA raises GSM8K validation mean from 29.34 to 34.27 at the shared 1.53M checkpoint. DMPO-DPRM is strongest on harder reasoning, improving MATH Hard from 44.3 to 47.9 and Countdown Hard from 29.6 to 33.4, while remaining near-tied on easier GSM8K averages. DPRM-Prism improves GSM8K voted, rank-1, and any-of-4 accuracy under the same search scaffold, but it also increases NFE. Thus the natural-language evidence is positive, with plausible cost per shown in Appendix B.2 and B.13.

4.2. DPRM on Multimodal Data

Tab. 3 probes two multimodal order types. Omni-Diffusion orders visual codebook tokens during text-to-image generation; DPRM-confidence gives a small CLIP-L/14 gain over the confidence-only limit, while DPRM-random is worse than random. LLaDA-V orders generated text positions under image conditioning; DPRM-confidence improves AI2D target-normalized accuracy, but RealWorldQA remains strongest under confidence-progressive decoding. These rows are useful because they show both transfer and boundary behavior: bucketized reward tilt can help structured diagram QA, but coarse text-token buckets are not sufficient for heterogeneous open-world VQA.

4.3. DPRM on Scientific Data

We next test protein inverse folding, single-cell gene-expression diffusion, molecular SAFE diffusion, and DNA reward optimization. These domains often lack a canonical left-to-right order, so we include both confidence-warmup and random-warmup DPRM variants. Random warmup is exploratory: it can expose broader partial-state classes before reward tilt becomes reliable, but it can also hurt when the host needs spatially or chemically coherent early commitments. Tab. 4 shows high leverage and clear trade-offs. DPLM is not a strong DPRM win: CAMEO differences are small, and co-generation favors the default or confidence-progressive rows. DCM gives a clean ordering effect, with all ordering-aware variants improving token recovery, MAE, and zero-expression accuracy over random ordered masking. GenMol is metric-specific because the bucket utility used for ordering is self-supervised molecular-token confidence, whereas validity, quality, uniqueness, di-

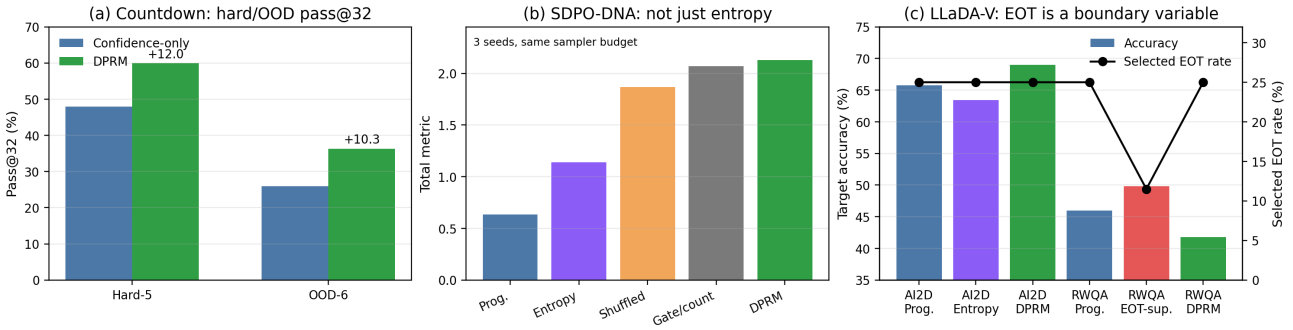

Figure 2. Theory-facing mechanism diagnostics. Panel (a) tests the finite-sample prediction that reward-tilted ordering should matter most on hard or OOD residual families; DPRM improves Countdown hard and OOD pass@32 over confidence-only Progressive DMPO. Panel (b) tests whether bucketized DPRM is more than entropy or over-confidence management on SDPO-DNA; DPRM remains strongest against reward-blind uncertainty, shuffled bucket values, and gate/count controls. Panel (c) tests low-dimensional order structure on LLaDA-V using the EOT/suffix-anchor phenomenon of Park et al. (2026); DPRM helps AI2D without an explicit EOT rule, while RealWorldQA exposes a boundary where direct EOT suppression is better.

versity, and fragment retention are downstream evaluation metrics. SDPO is the strongest scientific reward-tilt result: the phase-independent DPRM-random table reaches full selected-decision readiness and gives the best total metric, ATAC, and k-mer Pearson, while DPRM-confidence gives the best HepG2. These results reflect that the order in scientific modeling do matters and should be a non-negligible control factor.

4.4. Theory-Facing Mechanism Diagnostics

This section provide indirect mechanism diagnostics validating our theoretical guarantees. Extended details are delayed to Appendix B.15.

Sample-complexity separation. Thm. 3.5 predicts a two-stage effect. Confidence alignment can beat random masking early because it spends updates on locally easier denoising states. Reward-tilted DPRM should then help most on the remaining residual families: hard or out-of-distribution trajectories that are useful for terminal reward but are not selected often by confidence-only ordering. Fig. 2(a) is the main check of this prediction. On Countdown, DPRM improves hard and OOD pass@32 over confidence-only Progressive DMPO, matching the regime where the theorem expects residual-family coverage to matter.

Entropy-only and bucket ablations. To rule out the case where DPRM is merely increasing exploration by increasing uncertainty. We design the uncertainty control ranks positions by $u_i(s_t) = 1 - \max_v p_\theta(v \mid s_t)$, using the same reveal budget and no terminal-reward information. Shuffled-bucket, gate-only, and count-only rows keep the DPRM scaffold but remove or weaken learned terminal-utility values. On SDPO-DNA, entropy-only improves over confidence-progressive ordering but remains below DPRM, and shuffling bucket values lowers the total metric. This shows that the retained $P = 1$ bucket table contains useful reward-conditioned signal beyond a reward-blind uncertainty heuristic.

Low-dimensional order structure. Fig. 2(c) tests a concrete low-dimensional behavior in text decoding. EOT is the end-of-text token. Park et al. (2026) show that confidence decoding can reveal high-confidence EOT too early, and propose suffix-anchor confidence modulation (SACM) as a hand-designed correction. Interestingly, albeit DPRM has no EOT-specific rule since it only sees phase/confidence buckets and terminal utility, it could also have AI2D gain. This suggests that our bucketized DPRM can implicitly learn useful image-conditioned answer-order regularities.

5. Conclusion, Limitation and Future Work

We introduced DPRM, a plug-in controller that makes token order a reusable control variable for diffusion language models and adjacent masked discrete diffusion systems. DPRM contains confidence-progressive ordering as a confidence-only limit and adds reward tilt when online bucket statistics are reliable. The theory characterizes the exact Doob-guided policy, the Soft-BoN approximation, and the finite-sample error of the bucketized implementation. Empirically, ordering-aware DPRM variants are effective across several language, single-cell reconstruction, DNA reward-optimization, and multimodal settings, while protein co-generation, molecular design, DPRM-random image generation, and RealWorldQA expose multi-objective tradeoff or limits of coarse bucket transfer. The main limitations are our bucket abstraction bias, NFE overhead, and the theory assumptions that are supported by diagnostics rather than fully measured constants. Future work should develop tighter bucket abstractions, compute-matched decode controls, and richer domain-specific process rewards.

Acknowledgements

DB and HW are supported in part by the Research Grants Council of the Hong Kong Special Administrative Region (Project No. CityU 11206622). WH is supported by JSPS KAKENHI (24K20848) and JST BOOST (JPMJBY24G6). TS was partially supported by JSPS KAKENHI (24K02905) and JST CREST (PMJCR2015). This research is supported by the National Research Foundation, Singapore and the Ministry of Digital Development and Information under the AI Visiting Professorship Programme (award number AIVP-2024-004). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore and the Ministry of Digital Development and Information.

References

Anthropic. Introducing claude 4, May 2025. URL https://www.anthropic.com/news/claude-4. Accessed: 2025-09-01.

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.

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.

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*, 2025.

Bu, D., Huang, W., Han, A., Nitanda, A., Xue, B., Zhang, Q., Wong, H.-S., and Suzuki, T. Distributional biases in post-training: A markovian analysis of reasoning trajectories. *arXiv preprint arXiv:2511.07368*, 2026.

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.

DeepMind. Gemini diffusion. https://deepmind.google/models/gemini-diffusion/, 2025. Accessed: 2025-09-24.

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.

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

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, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. Deepseek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

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.

Inception Labs, Khanna, S., Kharbanda, S., Li, S., Varma, H., Wang, E., Birnbaum, S., Luo, Z., Miraoui, Y., Palrecha, A., et al. Mercury: Ultra-fast language models based on diffusion. *arXiv preprint arXiv:2506.17298*, 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.

Jaech, A., Kalai, A., Lerer, A., Richardson, A., El-Kishky, A., Low, A., Helyar, A., Madry, A., Beutel, A., Carney, A., et al. OpenAI o1 system card. *arXiv preprint arXiv:2412.16720*, 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.

Kimi Team, Du, A., Gao, B., Xing, B., Jiang, C., Chen, C., Li, C., Xiao, C., Du, C., Liao, C., et al. Kimi k1. 5: Scaling reinforcement learning with LLMs. *arXiv preprint arXiv:2501.12599*, 2025.

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.

Li, L., Long, Z., Shen, Y., Gao, H., Cao, H., Sun, X., Shan, C., He, R., and Fu, C. Omni-Diffusion: Unified multimodal understanding and generation with masked discrete diffusion. *arXiv preprint arXiv:2603.06577*, 2026. URL https://arxiv.org/abs/2603.06577.

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.

Park, J., Kim, J., Ko, J., Kwak, N., and Rhee, W. When confidence misleads: Suffix anchoring and anchor-proximity confidence modulation for diffusion language models.

arXiv preprint arXiv:2605.28181, 2026. URL https://arxiv.org/abs/2605.28181.

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.

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.

Shi, J., Han, K., Wang, Z., Doucet, A., and Titsias, M. Simplified and generalized masked diffusion for discrete data. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=xcqSOfHt4g.

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 2024. URL https://proceedings.mlr.press/v235/wang24ct.html.

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.

Wang, Z., Shi, J., Heess, N., Gretton, A., and Titsias, M. K. Learning-order autoregressive models with application to molecular graph generation. *arXiv preprint arXiv:2503.05979*, 2025c. URL https://arxiv.org/abs/2503.05979.

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, L., Tian, Y., Li, B., Zhang, X., Shen, K., Tong, Y., and Wang, M. MMaDA: Multimodal large diffusion language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=wczmXLuLGd.

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*, 2025a.

Ye, J., Xie, Z., Zheng, L., Gao, J., Wu, Z., Jiang, X., Li, Z., and Kong, L. Dream 7B: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025b.

You, Z., Nie, S., Zhang, X., Hu, J., Zhou, J., Lu, Z., Wen, J.-R., and Li, C. LLaDA-V: Large language diffusion models with visual instruction tuning. *arXiv preprint arXiv:2505.16933*, 2025.

Zhang, H., Yuan, G.-H., Yuan, C., Xu, T., Bian, T., Cheng, H., Huang, W., Zhao, D., and Rong, Y. Lingshu-Cell: A generative cellular world model for transcriptome modeling toward virtual cells. *arXiv preprint arXiv:2603.25240*, 2026. URL https://arxiv.org/abs/2603.25240.

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. Extended Related Work

Train–test alignment and token ordering in masked diffusion. General masked-diffusion frameworks show that state-dependent masking schedules can be treated within a unified discrete-data diffusion objective (Shi et al., 2024). Recent work has also made clear that the corresponding token order 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. Park et al. (2026) identify another concrete low-dimensional failure mode: high-confidence EOT tokens can be committed too early in fully non-autoregressive decoding, while suffix anchors mitigate incompleteness but create anchor-proximity overconfidence. 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), learned denoising-order planning (Israel et al., 2026), and suffix-anchored confidence modulation (Park 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. (2026) 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.

Multimodal masked diffusion and visual-token ordering. Masked discrete generation has a long visual precedent in vector-quantized image models and masked image-token generators (Austin et al., 2021; Gu et al., 2022; Esser et al., 2021; Chang et al., 2022). In these systems, the sampler is not only deciding what token value to predict; it is also deciding which still-masked visual-codebook positions should be committed first. MaskGIT-style confidence schedules make this design choice explicit by repeatedly committing high-confidence image tokens, and later masked visual-token systems inherit the same ordering sensitivity. This is the image-generation analogue of the language-side confidence-ordering problem studied in the main text, except that the candidate positions are spatial visual-codebook sites rather than response-token positions.

Recent multimodal diffusion language models extend masked discrete generation from a single visual or textual vocabulary to joint text–image or any-to-any token spaces. MMaDA studies diffusion language modeling over mixed visual and textual tokens (Yang et al., 2025). Omni-Diffusion builds a unified masked discrete diffusion model over text, speech, and image tokens, so text-to-image generation exposes an explicit visual-token ordering policy (Li et al., 2026). LLaDA-V adapts diffusion language models to visual instruction following, where an image conditions the denoiser logits but the generated object in VQA is still a text-token sequence (You et al., 2025). These two host types differ in what the ordering action means: in visual generation, the controller chooses image-codebook positions; in image-conditioned VQA, it chooses answer-token positions whose logits are conditioned on visual features.

Our multimodal extension therefore uses exactly two hosts, one from each category. DPRM-Omni tests whether reward-bucket ordering can improve visual-codebook text-to-image generation without changing the visual tokenizer, diffusion backbone, or official sampler budget. DPRM-LLaDA-V tests whether the same bucketized process-reward rule can improve image-conditioned text decoding without changing the image encoder, prompt template, language diffusion backbone, or lmms-eval task. MMaDA and MaskGIT-style models are cited as related context rather than evaluated tasks, and other image-diffusion hosts are intentionally outside the current nine-task scope. This keeps the multimodal claim narrow: the paper evaluates DPRM as an ordering-only plug-in for one image-generation host and one image-conditioned text-generation host, not as a new multimodal architecture or a broad visual-generation benchmark.

Protein diffusion and 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., 2024). DPLM established diffusion language models as strong protein learners, and DPLM-2 extended this direction to sequence–structure modeling (Wang et al., 2025b). Hsieh et al. (2025) further show that performance in 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 sequence–structure architecture, tokenizer, structure abstraction, or denoising loss, and instead isolate the effect of changing only the sequence ordering controller in DPLM-2 Bit.

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. The importance of ordering is not unique to diffusion: learned-order autoregressive models also learn state-dependent generation orders for molecular graphs (Wang et al., 2025c). Our DPRM-GenMol experiment 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 experiment is deliberately narrow and ordering-focused: we keep the reward-optimization objective, oracle suite, data preprocessing, optimizer and denoising loss fixed, and test whether alternative reveal-order controllers improve the resulting DNA design metrics.

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. Experiment Details and Mechanism Controls

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 5. 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	Self-supervised molecular-token utility used for ordering: average confidence assigned to the selected revealed SAFE tokens under the masked molecular denoiser. Molecular validity, QED/SA, fragment validity, and task retention are evaluation metrics, not the bucket-update signal in the reported runs.	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
DPRM-Omni	Text-to-image utility for completed visual-token rollouts: CLIP-L/14 image–text cosine on the generated image and its prompt, used to populate the visual-token bucket table in the reported comparison.	1.0
DPRM-LLaDA-V	Image-conditioned VQA utility: target-normalized correctness for the generated answer, keyed by order, task, and document id before bucket aggregation.	1.0

The utility in Tab. 5 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 controller settings in Tab. 6.

Table 6. 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: checkpoint-local phase table with 39 phase buckets and 16 confidence bins.	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.	Confidence-only DPRM limit: $(0, 0, 256)$. Reward-tilted DPRM variants: $(2000, 20000, 256)$.	Train: $m_t = \lceil M_t/P_t \rceil$. Decode: native DPLM linear reparameterized schedule over $T = 500$ steps, roughly $L/500$ positions per modality per step. The confidence-collapse threshold is 0.	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.	Confidence-only DPRM limit: $(0, 0, 32)$. Reward-tilted variants: $(100, 500, 32)$.	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.	Confidence-only DPRM limit: $(0, 0, 128)$. Reward-tilted 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. GenMol V2's sampler temperatures and randomness settings are fixed.	Sampled Soft-BoN with $N_t = \min\{64, \max(8, 4m_t)\}$.
DPRM-SDPO	Phase sweep over $P \in \{1, 4, 8\}$ reveal phases and 10 confidence bins over DNA sequence tokens; the reported table uses the coverage-supported $P = 1$ setting.	Confidence-only schedule for the progressive limit. Reward-tilted 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 $N_t = 64$.
DPRM-Omni	8 phases, 16 visual-token confidence bins, and 16 auxiliary confidence bins.	$(0, 64, 8)$ with guidance 2.0.	Official text-to-image evaluation uses the host's 260-step visual-token generation path with 256 generated image tokens. The ordering action selects visual codebook-token positions.	Full candidate scoring in the hook-backed comparison; no extra denoiser forward passes.
DPRM-LLaDA-V	8 text-token decode phases, 16 confidence bins, and 16 auxiliary bins under image-conditioned logits.	$(0, 4, 16)$ with guidance 2.0.	LLaDA-V VQA generation keeps the image encoder, language diffusion model, and lmms-eval tasks fixed. The ordering action selects generated answer-token positions.	Full candidate scoring in the reported comparison; no extra denoiser forward passes.

What is actually conditioned on at inference. The implementation never uses an exact partial-sequence lookup table at inference. Each inference-time candidate first computes its host score under the current partial sequence, maps the decision to a phase/confidence bucket, and then uses the checkpoint-local or online bucket statistic for that cell. Therefore "unseen

Table 7. How many positions are updated in parallel when the order controller is queried. L denotes the current generated length, M_t the remaining masked positions, and P_t the remaining phases.

Host	Ordered item	Decode steps	Positions per update
PUMA	text token	task-specific	2 or 3
DMPO	text token	128	about 2 per block step
Prism	text token	32	1
DPLM-2 Bit	protein token	500	$\approx L/500$ per modality
DCM	gene bin	32	$\lceil M_t/P_t \rceil$
GenMol V2	SAFE token	8 phases	$\lceil M_t/P_t \rceil$
SDPO-DNA	DNA token	P=1 reported	M_t
Omni-Diffusion	visual token	260	≈ 1
LLaDA-V reported VQA	answer token	4	1

partial sequence" does not mean "no DPRM value" unless the bucket is also unseen. If the exact partial sequence is new but its bucket was populated during training or earlier test-time search, DPRM uses the historical log-moment terminal utility of that bucket. If the bucket is empty, the safe value and local gate are both zero, so the controller reduces to the host base order.

B.2. Inference Procedure and Computational Cost

We next spell out the inference-time computation performed by the practical bucketized controller. This is the procedure used by the reported DPRM implementations whenever a valid checkpoint-local or online DPRM state is available. It also matches the multimodal extension experiments: for image generation the candidate positions are visual codebook tokens, while for image-caption or VQA decoding they are generated text-token positions conditioned on the image. For a partial masked state s_t, let M_t be the set of still masked positions. These positions can be text tokens, protein residues, SAFE molecular tokens, DNA bases, gene-expression bins, or visual codebook tokens. The host model first computes logits on s_t. For each masked position $i \in M_t$, the host provides a provisional token prediction $\hat{x}_i$ and a base ordering score $\psi_i(s_t)$, usually the probability assigned to the provisional token or the task-native confidence score. Thus confidence is used to rank candidate positions to reveal; it does not by itself replace the denoiser's predicted token distribution.

The exact DPRM process reward for revealing position i in state s_t is

$$R_t^\star(i; s_t) = \frac{1}{\beta} \log \mathbb{E}[\exp(\beta R(X)) \mid S_t = s_t,\ I_t = i]\,,$$

where $R(X)$ is the completed-trajectory utility. The practical implementation replaces this exact conditional expectation by a bucket projection. Let

$$\phi_t = \mathrm{Phase}(t, s_t), \qquad b_i = \mathrm{Bin}(\psi_i(s_t)).$$

The phase ϕ_t is a coarse progress index, for example the current reveal phase, block-local denoising step, or a discretization of the fraction of tokens already revealed. The confidence bin b_i is a coarse bin of the current host confidence for position i. The stored estimator is therefore

$$\widehat{R}_{\phi,b} = \frac{1}{\beta} \log \frac{\sum_{j \in \mathcal{H}_{\phi,b}} \exp(\beta R_j)}{|\mathcal{H}_{\phi,b}|} \approx \frac{1}{\beta} \log \mathbb{E}[\exp(\beta R(X)) \mid \Phi_t = \phi,\ B_t = b]\,.$$

This is a coarsened approximation to $R_t^\star(i; s_t)$: it forgets the exact partial sequence but keeps the stage of generation and the host model's current confidence about the candidate position.

At inference, DPRM performs the following steps at each denoising step.

1. Run the host denoiser once on the current partial state s_t, obtain provisional predictions and base ordering scores $\psi_i(s_t)$ for all $i \in M_t$.
2. Map each candidate position to (ϕ_t, b_i). If an auxiliary bucket is implemented, such as a structural bucket in protein design, the key is extended to (ϕ_t, b_i, a_i).
3. Read the bucket count and log-moment value. If the bucket is empty or below the readiness threshold, the local gate is zero or small, so the candidate falls back toward the base score.

4. Form the ordering score

$$g_i(s_t) = \log \psi_i(s_t) + \eta_i(s_t)\, \beta \widehat{R}_{\phi_t, b_i},$$

where $\eta_i(s_t)$ combines the global warmup/switch schedule with the local bucket-readiness gate.

5. Select the next reveal set A_t using the host reveal budget m_t. In the deterministic version this is the top m_t positions under g_i. In the sampled Soft-BoN version, the controller first samples a shortlist from the base proposal and then ranks that shortlist by g_i.

6. Reveal the selected positions using the host's provisional token predictions or its native sampling rule, then continue to the next masked state. If the controller is online at test time, the completed sample's utility is used to update the visited buckets after the trajectory ends.

Tab. 8 summarizes the cost. The bucketized DPRM step is scalar bookkeeping: bin candidates, look up a small table, apply readiness gates, and mix scores. It does not add denoiser forward passes unless the host method itself chooses extra verifier, reward, or search calls.

Table 8. Per-step overhead of bucketized DPRM for a remaining masked set M_t, P phases, B confidence bins, optional auxiliary bins A, and Soft-BoN shortlist size N_t.

Component	Cost	Comment
Bucket key construction	$O(\lvert M_t \rvert)$	phase/confidence/auxiliary binning
Lookup and gate	$O(\lvert M_t \rvert)$	counts, readiness, $\widehat{R}_{\phi,b}$
Score mixing	$O(\lvert M_t \rvert)$	$\log \psi_i + \eta_i \beta \widehat{R}_{\phi,b}$
Soft-BoN ranking	$O(N_t \log N_t)$ or $O(N_t)$	shortlist ranking or top-m_t selection
Table memory	$O(PBA)$	usually 10^2–10^3 scalar entries
Denoiser calls	unchanged	same host forward path unless the host adds search/verifier calls

The measured multimodal costs match this accounting. Omni-Diffusion uses the same 260-step text-to-image denoising path for both DPRM orders; the measured generation times are 21.04 seconds per prompt for DPRM-confidence and 20.72 seconds for DPRM-random, before image decode/evaluation bookkeeping. In LLaDA-V, bucket loading and score mixing occur inside the existing image-conditioned decode loop: AI2D takes $1055/974/981/982$ seconds for random/progressive/DPRM-confidence/DPRM-random, and RealWorldQA takes $2177/2148/2127/2166$ seconds.

Thus DPRM changes reveal order, not the host model or denoising objective. Large overheads reported elsewhere come from host-level choices such as Prism NFE, verifier calls, or task-native evaluation loops. If a bucket is empty or under-ready, the local gate reduces that decision to the corresponding confidence-only order; this is why selected-decision coverage and fallback rates are part of the reported diagnostics.

The bucket diagnostics separate three effects that would otherwise be conflated. First, confidence-progressive ordering is the reward-blind DPRM limit and already removes much of the random-mask mismatch. Second, reward-tilted DPRM can move probability mass toward low-confidence buckets whose historical terminal utility is favorable. Third, empty or under-ready buckets should not be trusted, so the readiness gate deliberately reverts those decisions toward the confidence-only order.

Countdown illustrates the second effect. The diagnostics log phase, confidence bin, selected-token indicator, terminal reward, DPRM value, DPRM score, and schedule mix. Late in training, confidence-only Progressive DMPO selects only 3.1% of tokens from bins 0–5, while DPRM selects 4.5%, 4.2%, and 3.9% for $\beta = 0.5, 1.0, 2.0$, respectively. The selected DPRM value is positive and increases with β. This is the intended use of a populated bucket estimator for a never-seen exact partial sequence: it transfers historical terminal-reward information from the bucket, not from an identical sequence state. The entropy and ablation controls in Sec. B.15 show that this evidence is strongest when read together with task-level outcomes and with the SDPO bucket sweep.

Tab. 11 summarizes the key attribution question: which gains come from moving to the confidence-only ordering limit, and which gains remain after adding reward tilt. 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.

Table 9. Inference-time bucket state used by the reported DPRM experiments. DPRM changes ordering only when a populated bucket contributes a nonzero $\widehat{R}$; otherwise the gate falls back toward the confidence-only order.

Experiment	Bucket key	Bucket utility	Interpretation
DPRM-PUMA	Decode phase $\times$ confidence bin.	Teacher-forced token log-probability.	Self-supervised tilt at the shared checkpoint.
DPRM-DMPO	Local denoising phase $\times$ confidence bin.	Completed reasoning reward.	Terminal-reward tilt evaluated with hard/OOD pass@K.
DPRM-Prism	Search phase $\times$ confidence bin.	Online self-verification outcome.	Reward tilt starts after online buckets populate.
DPRM-DPLM	Protein phase $\times$ amino-acid confidence $\times$ structural bin.	Provisional amino-acid recovery.	Task utility tilt, but reported metrics are near-tied or progressive-favorable.
DPRM-DCM	Gene reveal phase $\times$ expression confidence bin.	Selected-token reconstruction utility.	Self-supervised reconstruction tilt.
DPRM-GenMol	SAFE-token phase $\times$ molecular-token confidence bin.	Self-supervised molecular-token confidence.	Coverage is high; utility alignment is the bottleneck.
DPRM-SDPO	DNA-token phase $\times$ confidence bin; reported setting uses $P = 1$.	GOSAI regulatory utility.	Positive reward tilt only for the phase-independent abstraction.
Omni-Diffusion	Visual-token phase $\times$ confidence bin ($\times$ auxiliary bin).	CLIP image–text cosine.	Small DPRM-confidence gain with populated visual buckets.
LLaDA-V	Text-token phase $\times$ confidence bin under image conditioning.	Target-normalized VQA reward.	AI2D-positive; RealWorldQA needs richer bins.

Table 10. Bucket coverage for DPRM controller states. Nonzero means at least one selected event populated the bucket; ready means the count reached the controller's local threshold.

Experiment	Coverage	Fallback	Interpretation
DPRM-PUMA	$605/624$ nonzero; $571/624$ ready at decode threshold 16.	Under-ready buckets use confidence.	Broad enough for checkpoint-local self-supervised tilt.
DPRM-DMPO	$112/128$ nonzero; $106/128$ ready; phase 7 empty.	Late phase-7 decisions use confidence.	Reward tilt is supported where populated; late-phase coverage is limited.
DPRM-Prism	Online cold start.	Early search uses confidence/threshold fallback.	Cost-quality and gate coverage are reported because earliest steps are not reward-tilted.
DPRM-DPLM	Coverage not available for the compact protein comparison.	Empty buckets use confidence.	Protein result is conservative: the AAR-style utility is not DPRM-positive on the reported metrics.
DPRM-DCM	$99/128$ nonzero and ready.	Remaining buckets use confidence.	Adequate for the synthetic single-cell setting.
DPRM-GenMol	$128/128$ nonzero and ready.	Coverage is not the bottleneck.	Main issue is utility semantics, not bucket sparsity.
DPRM-SDPO	$P = 1$: $10/10$ ready and selected ready rate 1.000; $P = 4/8$ degrade.	$P = 1$ has no selected fallback.	Positive only for phase-independent reward buckets.
DPRM-Omni	16640 selected events; 0.880 nonzero; selected ready $0.977/0.966$.	Low-count visual buckets use base score.	Sufficient for the small DPRM-confidence gain.
DPRM-LLaDA-V	$169/2048$ nonzero; 3175 selected trace rows.	Sparse buckets fall back, especially on RealWorldQA.	AI2D transfers; broad VQA needs auxiliary bins.

Table 11. Attribution summary separating the order-alignment effect from the incremental reward-tilt effect. "Base/conf." denotes the strongest non-reward-tilted order available in the matched comparison: random/default when no confidence-only controller is implemented, or confidence-progressive when it is.

Host	Metric	Base/conf. reference	Best DPRM row	Interpretation
PUMA	GSM8K mean at shared 1.53M checkpoint	PUMA confidence 29.34	DPRM-PUMA 34.27	Direct DPRM-over-confidence gain at a shared checkpoint; not a full training-curve claim.
DMPO	Mean pass@K, hard subsets	Progressive MATH Hard 44.3, Countdown Hard 29.6	DPRM 47.9, 33.4	Positive, but current strongest row is an end-to-end train/decode DPRM pipeline.
Prism	GSM8K voted accuracy / mean NFE	Confidence 82.41 / 609	DPRM 83.85 / 1,071	Positive quality shift with substantial NFE overhead.
DPLM-2 Bit	CAMEO and co-generation	Progressive CAMEO TM 0.3066, CoGen TM 0.4043	DPRM CAMEO RMSD 35.41, CoGen TM 0.3675	Near-tied CAMEO and progressive-favorable co-generation; not a DPRM dominance result.
DCM	Token recovery / zero accuracy	Progressive recovery 76.07, random zero acc. 82.83	DPRM(random) zero acc. 99.86	Strong ordering effect; DPRM mainly shifts which metric is best.
GenMol	De novo and fragment metrics	Baseline quality 0.854, progressive diversity 0.853	DPRM(random) validity 0.997, linker validity 0.429	Metric-specific trade-off; reward utility is not aligned with all molecular metrics.
SDPO-DNA	Total / ATAC / k-mer	Baseline total 1.141, progressive 0.384	DPRM(random, $P = 1$) total 2.119	Positive after reducing phase abstraction; multi-phase buckets degrade under current coverage.
Omni-Diffusion	CLIP-L/14 cosine	Progressive 0.24744	DPRM(conf.) 0.24915	Small confidence-warmup gain; DPRM-random is negative.
LLaDA-V	AI2D / RealWorldQA target-normalized accuracy	Progressive 0.658 / 0.46013	DPRM(conf.) 0.692 / 0.41699	AI2D positive, RealWorldQA negative; bucket transfer depends on task homogeneity.

B.3. DPRM-PUMA: Setup, Evaluation, and Interpretation

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 evaluate both methods at the shared EMA checkpoint step 1.53M, keeping the checkpoint source, validation protocol, and reveal budget matched.

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 materially large 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. 12 summarizes the resulting 95% percentile intervals from 5,000 paired bootstrap resamples. The point estimates are 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. The paired-bootstrap deltas 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.

Table 12. 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]

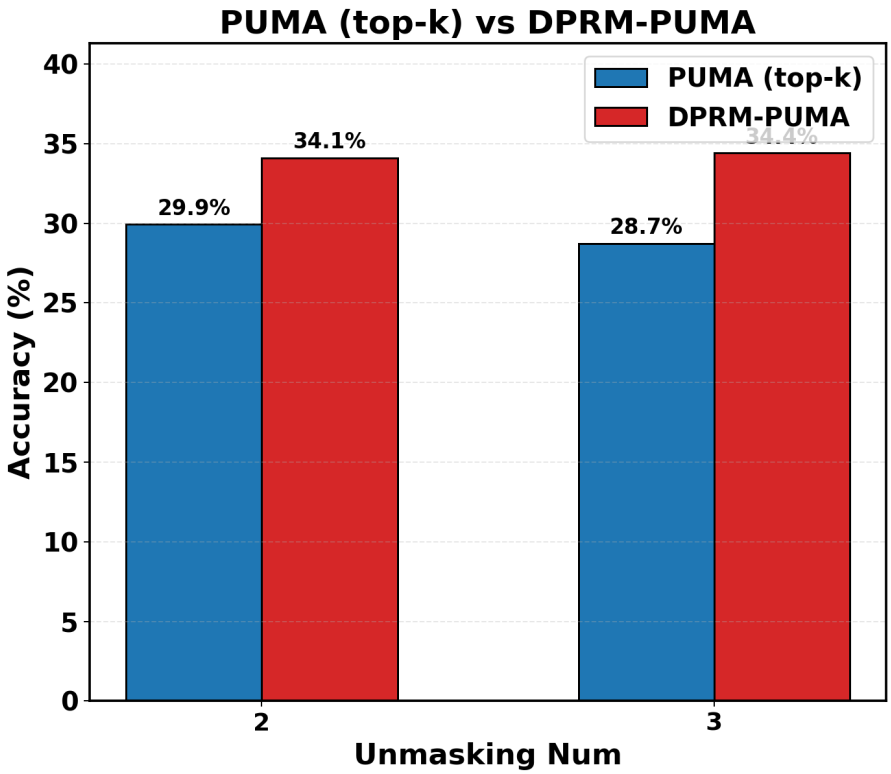

Figure 3. PUMA vs. DPRM-PUMA on GSM8K at the shared 1.53M EMA checkpoint. We use the two official PUMA validation settings, `unmasking_num`$\in \{2, 3\}$. DPRM-PUMA improves both.

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. 3 therefore isolates a narrower question: once train–test alignment is already enforced, is confidence itself the right ordering signal? The answer from this experiment is 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. 12 and Fig. 4 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 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 under matched checkpointing and uncertainty accounting.

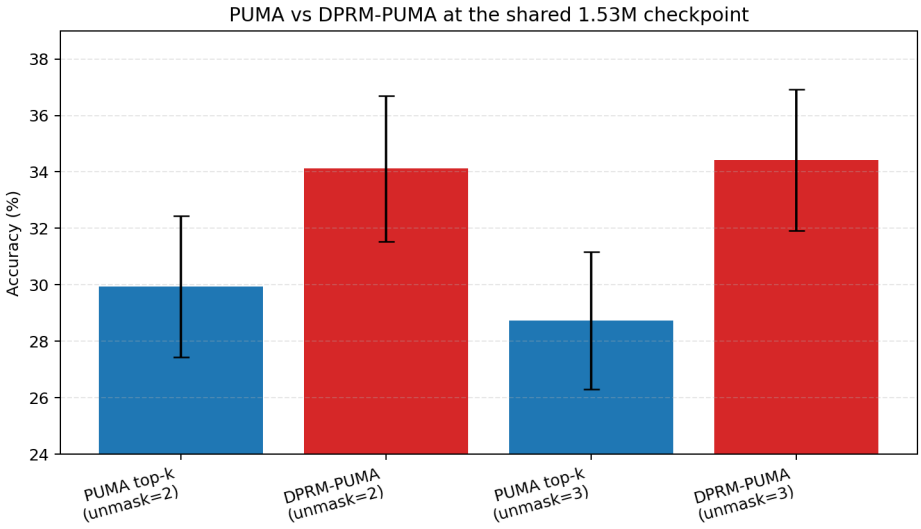

Figure 4. Bootstrap confidence intervals for PUMA and DPRM-PUMA at the shared 1.53M checkpoint. The two official unmasking settings both favor DPRM-PUMA, and the paired-bootstrap deltas in Tab. 12 exclude zero at the 95% level.

B.4. DPRM-DMPO: Setup, Pass@K, and Difficulty Analysis

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.

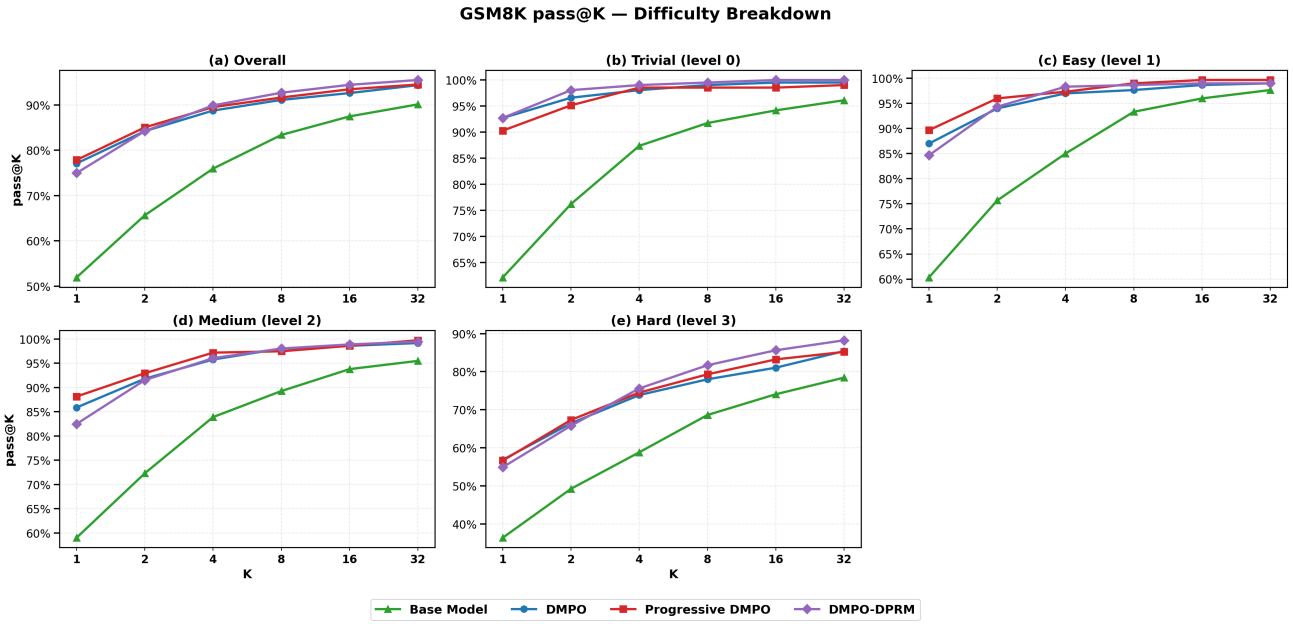

Figure 5. GSM8K pass@K curves by difficulty level (0: trivial, 1: easy, 2: medium, 3: hard). DMPO-DPRM's advantage over Progressive DMPO is most visible on harder levels and at larger K.

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

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 $\{N_{\phi,b}, S_{\phi,b}\}_{\phi=0}^{K-1}{}_{b=0}^{B-1} \leftarrow 0$
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 $\mathcal{L}_{\text{WDCE}}(\theta; x, z^{(u)}, w)$ 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**
        $p_{n,i} \leftarrow \max_v \operatorname{softmax}(\ell_{n,i})_v, \quad b_{n,i} \leftarrow \textsc{Bin}(p_{n,i})$
        $\widehat{R}_{n,i} \leftarrow \frac{1}{\beta} \log\left(\frac{S_{\phi_n, b_{n,i}}}{\max(N_{\phi_n, b_{n,i}}, 1)}\right)$
        $\eta_{n,i} \leftarrow \text{clip}\left(\frac{t - T_{\text{warm}}}{T_{\text{switch}} - T_{\text{warm}}}, 0, 1\right) \cdot \min\left(\frac{N_{\phi_n, b_{n,i}}}{N_{\text{ready}}}, 1\right)$
        $s_{n,i} \leftarrow \log p_{n,i} + \eta_{n,i} \widehat{R}_{n,i}$
      end for
      Optionally sample shortlist $C_n \subseteq M_n$ from proposal $q(i) \propto p_{n,i}$
      Reveal set $A_n \leftarrow \text{TopK}_{i \in C_n \text{ or } M_n}(s_{n,i}, m_n)$
      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 $\nabla_\theta \mathcal{L}_{\text{WDCE}}$
  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 $\hat{x}$

Benchmarks and metrics. We evaluate post-training checkpoints 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**: the confidence-only progressive variant, 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.

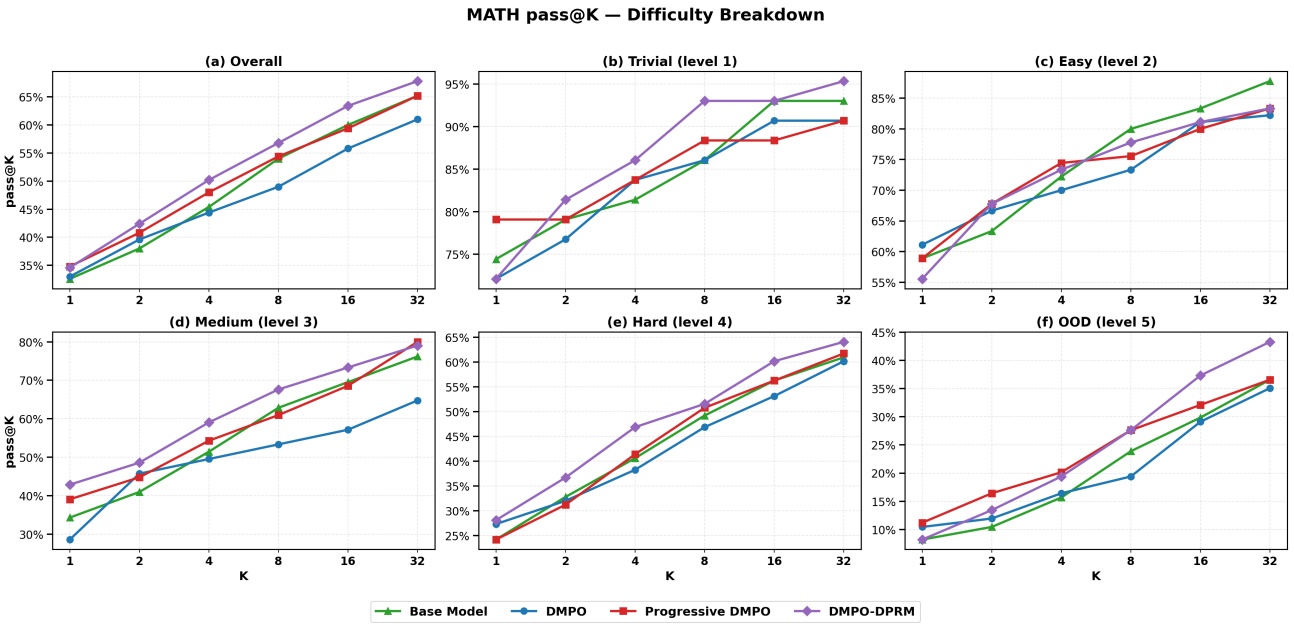

Figure 6. MATH pass@K curves by difficulty level (1: trivial, 2: easy, 3: medium, 4: hard, 5: OOD). DPRM-DMPO provides consistent gains on hard and OOD subsets.

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. Consequently, the strongest DMPO row is best interpreted as an end-to-end DPRM ordering pipeline rather than a clean ablation that separately identifies train-time and decode-time DPRM.

Fairness of the comparison. For each task, DPRM-DMPO is matched against the corresponding confidence-only Progressive DMPO variant rather than against DMPO-random. Thus, the comparison isolates a reward-tilted ordering-pipeline change under an otherwise fixed progressive post-training setup, while decode-matched controls remain a separate ablation target.

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.14, the uncertainty analysis is computed from raw success matrices rather than aggregate pass@K curves. Tab. 13 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.

Table 13. 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	$+45.59$ $[44.44,\ 46.69]$	$+1.67$ $[0.84,\ 2.52]$

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. 5 to 7 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.

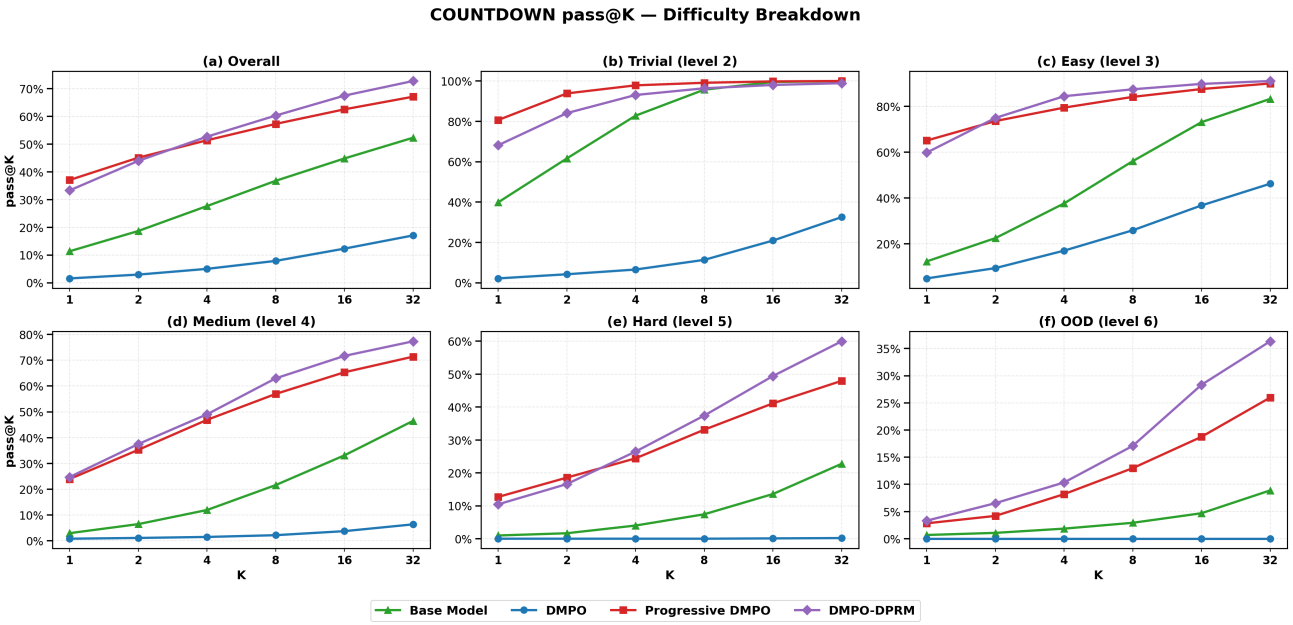

Figure 7. Countdown pass@K curves by number of target operands (2–6). Vanilla DMPO collapses on this task, falling below the base model at every difficulty level. DPRM-DMPO achieves the strongest performance across all levels.

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

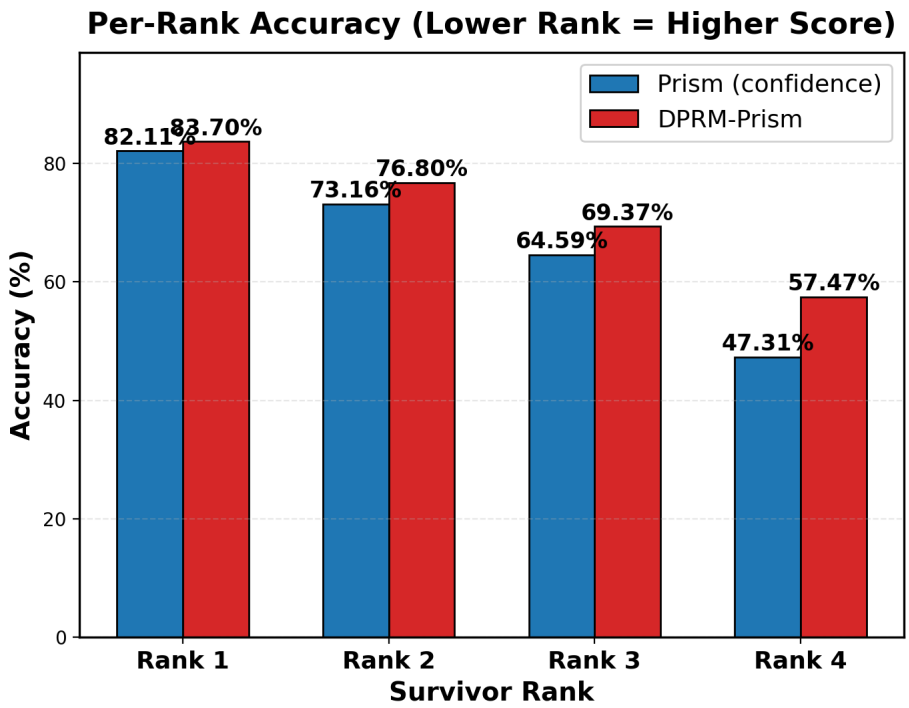

Figure 8. Per-rank accuracy comparison on GSM8K. Rank 1 is the highest-scored survivor after pruning. DPRM-Prism improves at every rank position.

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.5. DPRM-Prism: Test-Time Search and Cost-Quality Trade-off

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 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. 8 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. 9 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). DPRM-Prism improves accuracy relative to the matched confidence-order Prism baseline, but it also shifts the method rightward on the cost axis. The right panel shows the per-sample NFE distributions: DPRM-Prism's distribution is shifted rightward by the ordering overhead, while the two distributions have comparable spread.

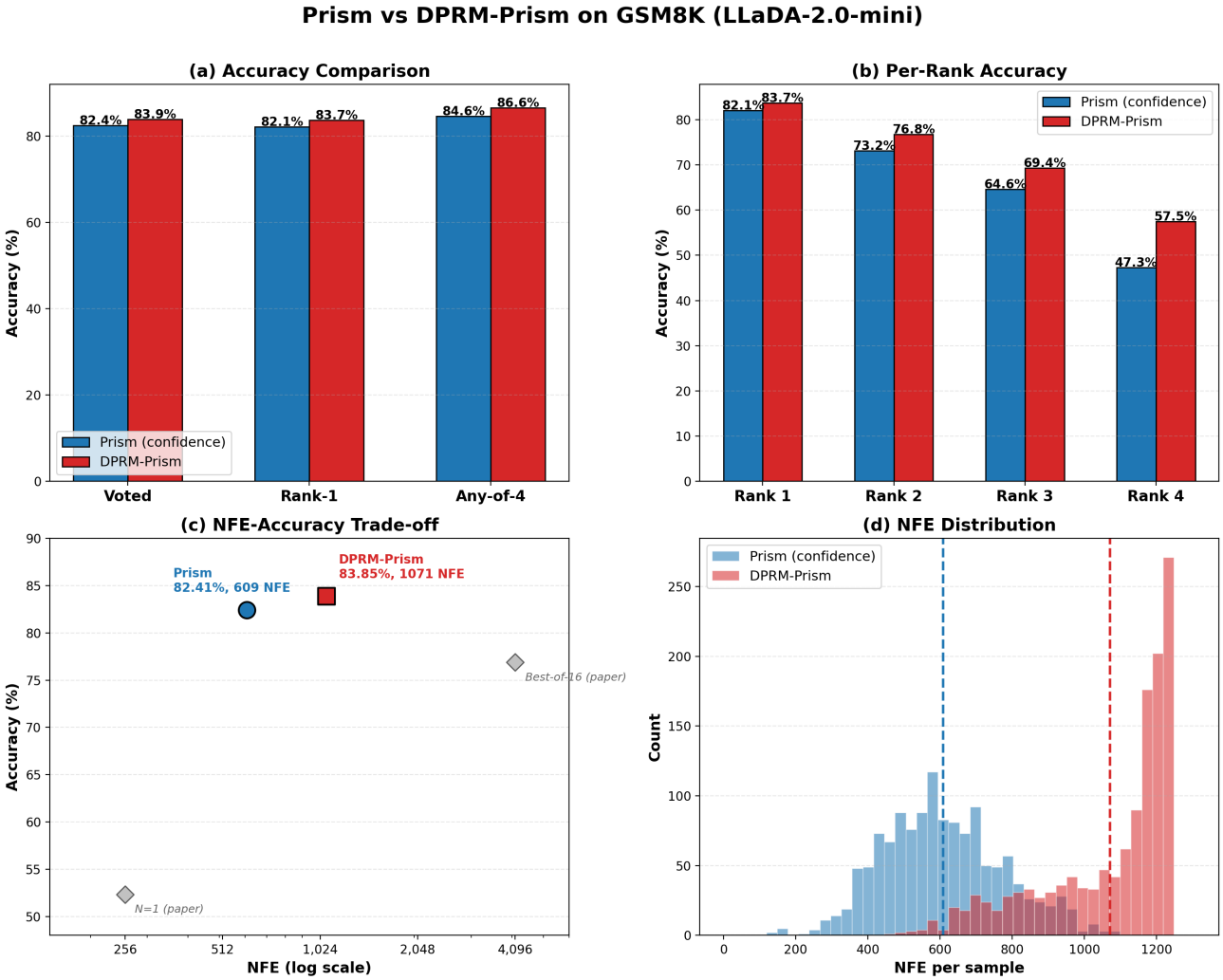

Figure 9. Left: NFE–accuracy trade-off. The diamond markers show reference baselines from the Prism paper. Right: per-sample NFE distributions.

NFE overhead and compact cost-quality sweep. The $\times 1.76$ NFE increase of DPRM-Prism over the full GSM8K baseline originates in 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 unchanged, and the NFE per SVF call remains identical (29), so the comparison isolates a quality–cost trade-off for the ordering controller rather than a free accuracy gain. We also include a higher-budget confidence-only control on the same 250-example diagnostic split. It improves over the default confidence row but remains below DPRM-Prism on both rank-1 and any-of-4 accuracy, while using less NFE than DPRM.

Table 14. Compact Prism cost-quality sweep on 250 GSM8K examples. The higher-budget confidence-only row is a reward-blind compute control; DPRM gives the strongest accuracy but uses the most NFE.

Policy	Rank-1	Any-of-4	Mean NFE
Confidence Prism	0.828	0.856	610.6
Confidence Prism, larger N	0.844	0.872	745.2
DPRM, warmup $0.2T$	**0.860**	**0.896**	1061.1

Interpretation. The DPRM-Prism results show that reward-aware token ordering can improve a fixed test-time search scaffold, but they should be read together with the NFE diagnostics. Because the Prism scaffold handles trajectory-level search while DPRM handles token-level ordering, the two modules compose naturally; however, the online controller cold-starts from confidence/threshold fallback before verifier rewards populate buckets. We therefore report DPRM-Prism as a positive quality–cost point, not as evidence that DPRM is compute-free or uniformly preferable under a matched NFE budget.

B.6. Protein Ordering with DPLM-2 Bit

B.6.1. DPRM-DPLM: Setup and Protein Metrics

This subsection provides the full experimental details for the DPLM-2 Bit ordering ablation reported in Tab. 4a.

We compare four DPLM-2 Bit ordering policies: **DPLM-2 Bit (default)**, **DPLM-2 Bit (confidence-progressive)**, **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 controlled variable is the sequence token-ordering controller. The confidence-progressive row is the reward-blind DPRM limit: it follows confidence-aligned progressive ordering with $(T_{\text{warm}}, T_{\text{switch}}, N_{\text{ready}}) = (0, 0, 256)$ and never uses terminal-reward bucket values. The reward-tilted DPRM variants use 8 phases, 16 confidence bins, one active structural bucket, sampled shortlist $N_t = \min\{32, \max(8, 4m_t)\}$, and $(T_{\text{warm}}, T_{\text{switch}}, N_{\text{ready}}) = (2000, 20000, 256)$. DPRM-DPLM warms up with confidence; DPRM(random)-DPLM warms up with random ordering. During training 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; generation uses the host DPLM linear reparameterized schedule over 500 steps.

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 restored 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 table reports macro means over the five length buckets.

Statistical uncertainty. For forward folding, all variants are evaluated on the same CAMEO2022 target set, so differences should be read as paired target-level comparisons. For co-generation, generated structures are not paired sample-by-sample across methods; we therefore summarize macro means over the length buckets and use unpaired generated-sample uncertainty rather than claiming paired sample-level superiority. When repeated generation seeds are available, we also report seed-to-seed variation.

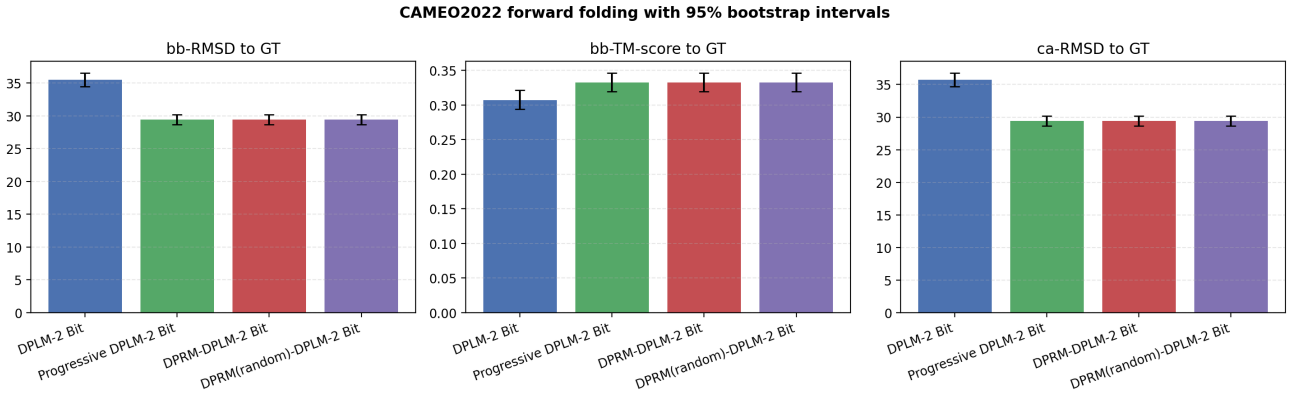

Figure 10. DPLM-2 Bit CAMEO2022 forward-folding comparison with bootstrap intervals. The reward-tilted rows are close to the confidence-progressive row: DPRM-confidence has a small bb-RMSD edge, but not a TM-score edge.

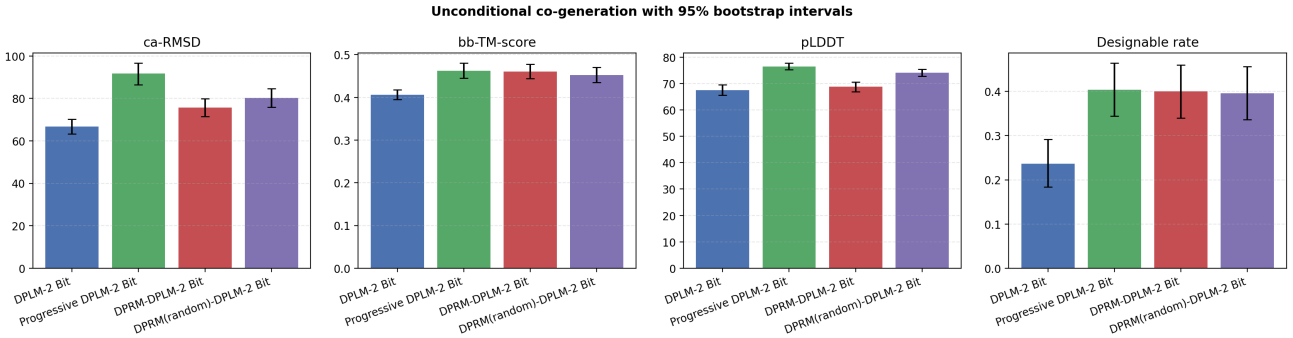

Figure 11. DPLM-2 Bit unconditional co-generation comparison with bootstrap intervals. Confidence-progressive ordering is strongest on pLDDT, and the DPRM rows do not dominate the non-reward-tilted references.

Results and interpretation. The protein comparison is conservative. On CAMEO forward folding, all four rows are close: default gives RMSD 35.47 and TM-score 0.3071, confidence-progressive gives $35.47/0.3066$, DPRM-confidence gives $35.41/0.3053$, and DPRM-random gives $35.46/0.3055$. The small RMSD edge of DPRM-confidence is not accompanied by a TM-score edge. In co-generation, the non-reward-tilted rows are stronger on the reported macro metrics: default gives backbone TM-score 0.4063, and confidence-progressive gives the best pLDDT 70.95. DPRM-confidence gives $0.3565/59.35$, and DPRM-random gives $0.3675/62.59$. We therefore treat DPLM as a mixed or negative DPRM case: ordering remains consequential, but the AAR-style protein utility and schedule do not reliably improve over the default or confidence-progressive orders.

B.7. Multimodal Scope and Shared Controls

The multimodal extension uses two hosts because they expose two different order targets. Omni-Diffusion[1] orders visual codebook-token positions during text-to-image generation. LLaDA-V (You et al., 2025) orders generated answer-token positions under image conditioning. Both hosts use the same four ordering families used elsewhere: random, confidence-progressive, DPRM with confidence warmup, and DPRM with random warmup.

The intervention remains order-only. Token values come from the host denoiser or host sampling rule. DPRM only changes which still-masked positions are revealed next; it does not change the visual tokenizer, image encoder, language diffusion backbone, prompt template, loss, or evaluator. Tab. 15 gives the shared numeric summary; the two task-specific sections below provide the same level of setup, figures, and interpretation as the non-multimodal hosts.

[1] https://github.com/VITA-MLLM/Omni-Diffusion

Table 15. Multimodal extension results. All rows are ordering-only comparisons within the same host, checkpoint/evaluation path, and prompt or task split. Higher is better. Negative DPRM rows are included because they identify bucket-transfer limits.

Task	Metric and order target	Random	Progressive	DPRM(conf.)	DPRM(random)	Interpretation
Omni-Diffusion T2I	CLIP-L/14 image–text cosine over 64 official-step prompts; order acts on visual codebook-token positions.	0.22184	0.24744	**0.24915**	0.21456	Small DPRM-confidence gain; random warmup fails.
LLaDA-V AI2D	Target-normalized accuracy over 500 image-conditioned VQA examples; order acts on generated text-token positions.	0.628	0.658	**0.692**	0.654	Positive structured diagram-QA transfer.
LLaDA-V Real-WorldQA	Target-normalized accuracy over 765 image-conditioned VQA examples; order acts on generated text-token positions.	0.44575	**0.46013**	0.41699	0.43791	Negative transfer under coarse buckets.

B.8. DPRM-Omni: Visual-Token Ordering

Task and ordered object. Omni-Diffusion is the image-generation host in our multimodal suite. The text prompt conditions a masked discrete diffusion process over visual codebook tokens, and the order controller selects image-token positions. The base score $\psi_i(s_t)$ is the host confidence for the provisional visual token at image position i. DPRM-Omni uses 8 denoising phases, 16 visual-token confidence bins, 16 auxiliary confidence bins, and CLIP-L/14 image–text cosine as the completed-rollout utility.

Evaluation protocol. All four orders use the official 260-step text-to-image path, 256 generated visual tokens, and the same 64-prompt split. The comparison keeps the text prompts, denoising backbone, visual tokenizer/decoder, and CLIP evaluator fixed. DPRM-labeled rows require a populated hook-backed visual-token table; there is no silent fallback to a confidence-only DPRM label. The DPRM rows use guidance 2.0 and readiness threshold 8.

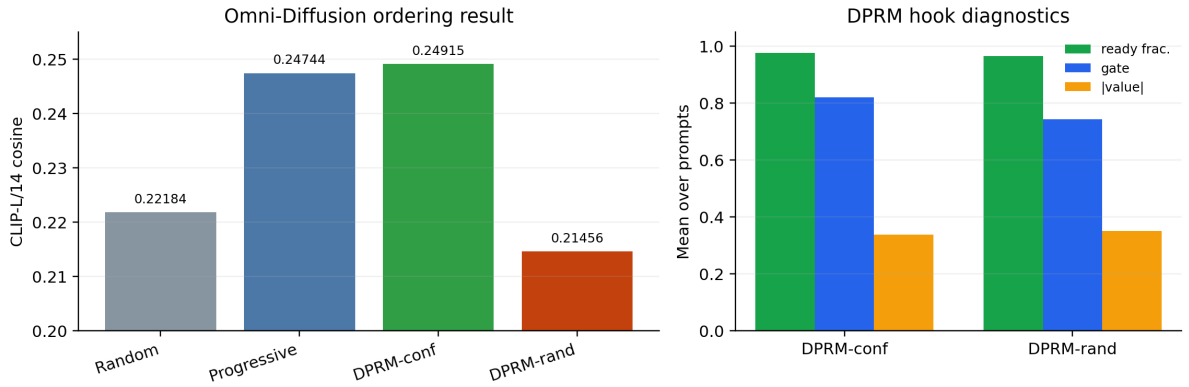

Figure 12. DPRM-Omni quantitative and hook diagnostics. Left: CLIP-L/14 image–text cosine for the four order policies on the same 64-prompt official-step split. Right: the hook-backed DPRM rows have high selected-bucket readiness and nonzero gates, so the DPRM labels correspond to active bucket scoring rather than a confidence fallback.

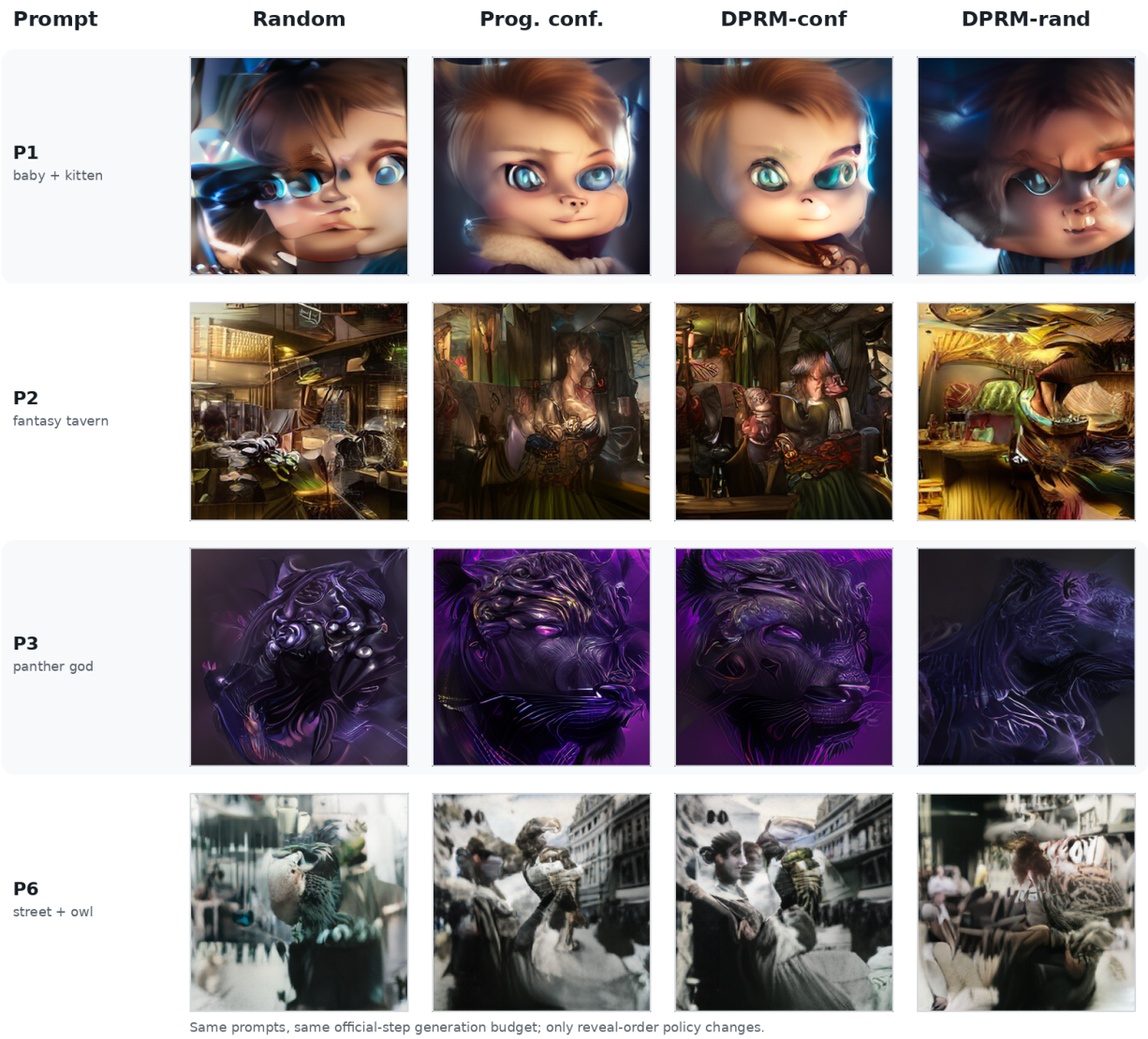

Figure 13. Omni-Diffusion visual examples for selected aligned prompts. Columns correspond to random, confidence-progressive, DPRM-confidence, and DPRM-random ordering. The images are generated by the official 260-step visual-token path; the figure verifies that the reported CLIP comparison is not driven by malformed images.

Interpretation. Figs. 12 and 13 show a small but clean visual-token ordering effect. Confidence-progressive ordering is already strong for masked image generation because high-confidence visual tokens often preserve spatial coherence. DPRM(conf.) improves CLIP-L/14 from 0.24744 to 0.24915 by using historical prompt-level CLIP utility inside phase/confidence buckets. DPRM(random) drops to 0.21456, below random. This failure is useful: random warmup creates poor visual-token traces for a spatially coupled image generator, so the resulting reward table is not a reliable improvement over confidence-guided image-token commitment. We therefore report DPRM-Omni as a confidence-warmup positive case, not as a uniform four-order win.

B.9. DPRM-LLaDA-V: Image-Conditioned Text Ordering

Task and ordered object. LLaDA-V is the image-conditioned text-generation host. The image encoder conditions the language-diffusion logits, but the reveal order is over generated answer-token positions. The base confidence is the probability of the provisional answer token at a masked text position under the current image-conditioned partial answer.

The DPRM table uses text-token decode phase and confidence buckets; the completed-trajectory utility is target-normalized VQA correctness. Rewards are keyed by order, task, and document id before aggregation, so AI2D and RealWorldQA traces do not collapse into one invalid pool.

Evaluation protocol. All four orders are evaluated on the same 500 AI2D examples and 765 RealWorldQA examples. The LLaDA-V image encoder, diffusion language model, prompt formatting, decode budget, and lmms-eval task definitions are fixed. DPRM-labeled decoders must load the populated target-fixed text-token table, and answer extraction is normalized against the task target. We additionally run reward-blind entropy, direct EOT suppression, and suffix-anchor confidence modulation controls to test whether DPRM is merely increasing entropy or delaying end-of-text. Here EOT denotes the end-of-text token. The EOT and suffix-anchor controls follow the failure mode identified by Park et al. (2026): a confidence-ordered DLM can reveal a locally high-confidence EOT token too early, while suffix anchors can trade this incompleteness for anchor-proximity overconfidence. These controls are not DPRM components; they test whether the multimodal text-order gain is explained by this known confidence artifact rather than by reward-conditioned buckets. The reward-blind entropy row uses the same reveal budget but scores each candidate position by the cheap uncertainty proxy

$$u_i(s_t) = 1 - \max_v p_\theta(v \mid s_t),$$

and reveals the largest-u_i positions. Thus it increases exploration without using terminal-reward statistics.

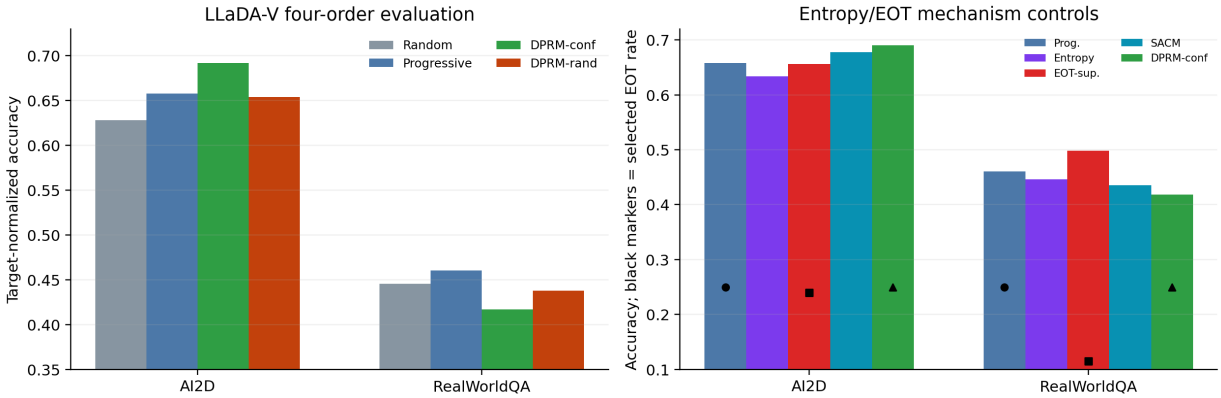

Figure 14. DPRM-LLaDA-V task and mechanism diagnostics. Left: four-order target-normalized accuracy on AI2D and RealWorldQA. Right: entropy/EOT controls on the same target-fixed evaluation; bars show accuracy and black markers show selected-EOT rate for representative policies. DPRM-confidence beats both progressive and entropy-only on AI2D, while RealWorldQA is better explained by direct EOT suppression and remains a negative DPRM transfer case.

Interpretation. LLaDA-V demonstrates why bucketized DPRM can help an unseen exact partial sequence without memorizing that sequence. AI2D has structured diagram-QA examples, short answer strings, and repeated answer-position patterns. Phase/confidence buckets built from image-conditioned text traces therefore transfer useful target-normalized reward information, raising accuracy from 0.658 under confidence-progressive decoding to 0.692 under DPRM(conf.).

RealWorldQA is different. Its visual scenes, question types, and answer strings are more heterogeneous, so the same (ϕ, b) bucket can mix partial states with different future rewards. This increases the abstraction error in (21) and (22). Accordingly, confidence-progressive remains best among the four order policies, and the direct EOT/SACM-style controls are stronger mechanism controls. This is still informative for the bucketized approximation: EOT position, suffix-anchor proximity, decode phase, and confidence are precisely the kind of low-dimensional variables that can organize repeated decoding behavior, but the evaluated LLaDA-V DPRM table only uses coarse phase/confidence buckets. The correct conclusion is task-dependent: DPRM transfers to structured image-conditioned text ordering, but broad open-world VQA needs richer auxiliary bins such as EOT status, answer format, object/region type, or task family before a stronger claim is justified.

B.10. DPRM-DCM: Single-Cell Gene-Token Ordering

This subsection documents the single-cell discrete diffusion experiment reported in Tab. 4b. 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 compact host has fewer than three thousand processed Dentate cells, so the matched training protocol is lightweight. The best teacher-forced validation losses are 1392.50 for DCM-random, 1875.53 for Progressive-DCM, 1519.73 for DPRM(random)-DCM, and 3151.08 for DPRM(conf.)-DCM. These losses are not used to rank methods because each controller induces a different ordered masking distribution; the matched all-mask decoding evaluation below is the fair comparison for reconstruction quality.

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. 15 visualizes the resulting 95% bootstrap intervals.

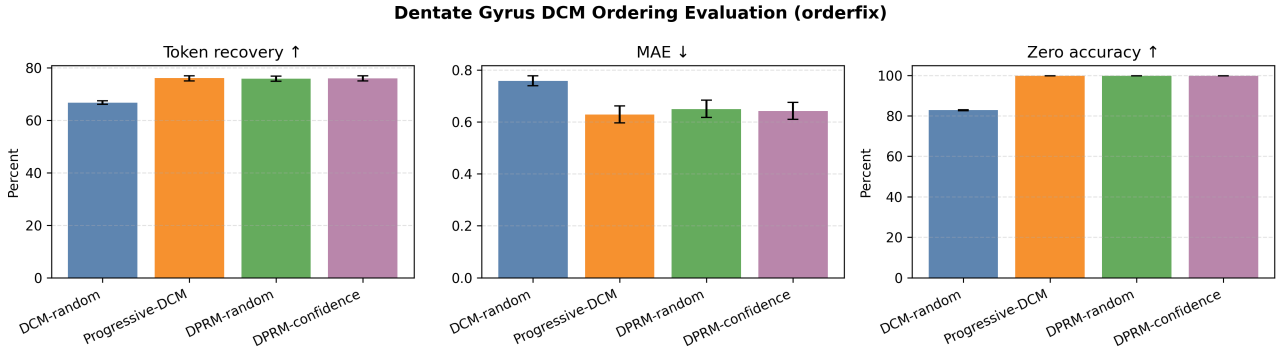

Figure 15. Dentate Gyrus DCM ordering evaluation with 95% bootstrap intervals over 293 validation cells. All ordering-aware variants improve token recovery, MAE, and zero-expression accuracy over random ordered masking. Progressive-DCM is strongest on token recovery and MAE, while DPRM(random)-DCM is strongest on zero-expression accuracy.

Interpretation. The DCM result should be read as evidence for the plug-in nature of ordering control 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 intervention used in language and protein settings transfers 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 biological utilities may be needed for DPRM to separate more strongly from confidence on downstream distributional metrics.

B.11. DPRM-GenMol: Molecular SAFE-Token Ordering

This subsection documents the molecular drug-design ordering comparison reported in Tab. 4c. 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. The fragment evaluation uses the common stable subset that excludes ELIGLUSTAT, ERLOTINIB, and FUTIBATINIB for every method, so every ordering controller is evaluated on the same conditioning fragments. 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.

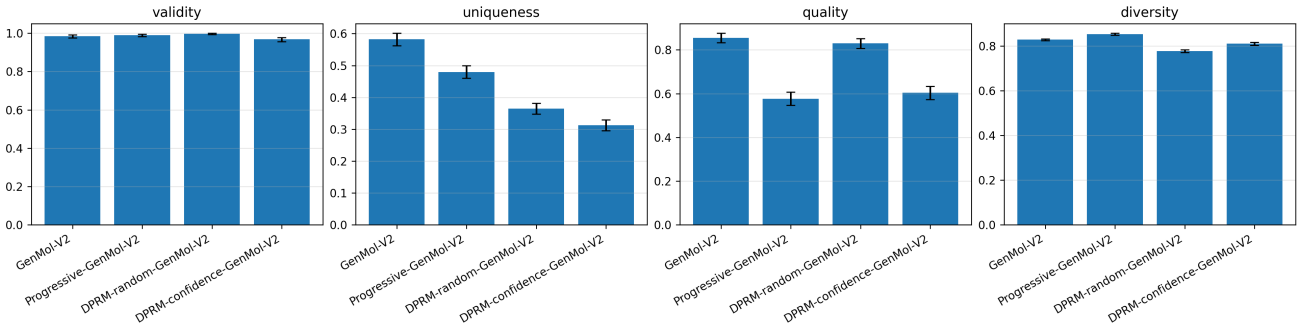

Figure 16. GenMol V2 de novo molecular generation with 95% bootstrap intervals over 1,000 generated molecules per method. GenMol V2 remains strongest on quality and uniqueness; DPRM(random)-GenMol has the highest validity; Progressive-GenMol has the highest diversity.

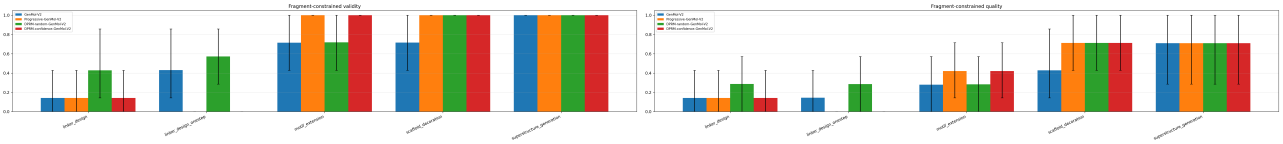

Figure 17. GenMol V2 fragment-constrained generation on the common stable seven-fragment subset. Error bars show 95% bootstrap intervals over fragment-task units. DPRM(random)-GenMol improves linker and linker-onestep validity, while Progressive and DPRM-confidence improve motif-extension and scaffold-decoration quality.

Interpretation. The GenMol comparison 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 self-supervised reconstruction proxy used by the generic DPRM controller 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.12. DPRM-SDPO: DNA Reward-Optimized Ordering

This subsection documents the DNA reward-optimization comparison reported in Tab. 4d. 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 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. To measure the effect of the bucket abstraction itself, we sweep the number of phase buckets $P \in \{1, 4, 8\}$ while keeping 10 confidence bins, $\beta = 1.0$, $(T_{\text{warm}}, T_{\text{switch}}, N_{\text{ready}}) = (100, 400, 64)$, and Soft-BoN shortlist size $N_t = 64$. The main table reports the $P = 1$ abstraction because it has full selected-decision readiness and the best total metric among the phase variants. 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.

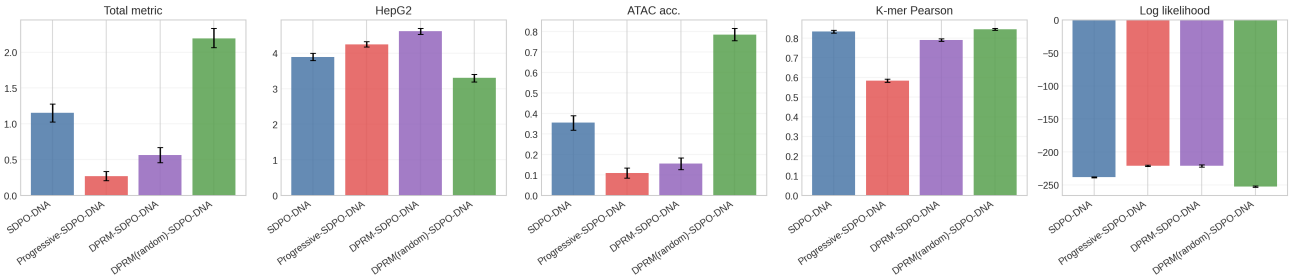

Figure 18. DPRM-SDPO DNA reward-optimization metrics. All rows use the same pretrained DNA diffusion checkpoint, SDPO objective, oracle suite, and sampling budget; only the reveal-order controller changes. DPRM(random)-SDPO gives the strongest total metric, ATAC success, and k-mer Pearson correlation, while DPRM-SDPO gives the strongest HepG2 score.

Table 16. SDPO-DNA phase-granularity audit. Finer phase buckets reduce ready coverage and degrade the total metric; the reported SDPO rows therefore use the phase-independent $P = 1$ abstraction.

Policy	Phases P	Ready buckets	Total metric
DPRM(conf.)	1	$10/10$	1.423
DPRM(conf.)	4	$34/40$	0.592
DPRM(conf.)	8	$60/80$	0.317
DPRM(random)	1	$10/10$	**2.119**
DPRM(random)	4	$35/40$	1.197
DPRM(random)	8	$64/80$	1.032

Interpretation. The SDPO experiment shows both the promise and the sensitivity of bucketized reward tilting. With $P = 1$, DPRM(random)-SDPO produces the strongest product-style total, ATAC success, and k-mer correlation, improving all three over the SDPO baseline, while DPRM(conf.)-SDPO achieves the highest HepG2 expression score. With $P = 4$ or $P = 8$, coverage and selected-decision readiness drop and the total metric becomes worse. Thus the SDPO result

should be read as evidence that a phase-independent reward bucket can help this DNA setting, not as evidence that finer phase discretization is automatically better. Since the pretrained checkpoint, reward optimization setup, sampler budget, and evaluation oracle are held fixed, the controlled variable is still token ordering, but the useful abstraction level is task dependent. Fig. 18 visualizes the same trade-off across all reported SDPO metrics.

B.13. Runtime and Sampling-Cost Analysis

The runtime tables use the clean timing or cost field available for each host. For non-multimodal hosts, this is evaluator wall-clock time for PUMA, mean NFE for Prism, decoded generation time for DPLM, training time for DCM, and sampling time for GenMol and SDPO-DNA. Because these fields have different units, the last column in Tab. 17 first normalizes each entry by the corresponding baseline/random cost and then averages over available hosts.

DMPO pass@K evaluations contain raw correctness matrices but not a clean matched wall-clock trace, so DMPO is omitted from the runtime average. The two multimodal hosts use prompt- or document-level evaluation loops that are not comparable to the scalar fields above; their costs are therefore reported separately in Tab. 18.

Ordering policy	PUMA eval (s)	Prism NFE	DPLM decode (min)	DCM train (min)	GenMol sample (s)	SDPO sample (s)	Avg. relative cost
Baseline/random	387.5	609.0	6.3	30.6	98.3	9.3	1.00×
Progressive/confidence	n/a	n/a	56.1	108.3	102.1	24.5	4.03×
DPRM(conf.)	417.5	1070.5	52.7	32.7	109.8	55.3	3.22×
DPRM(random)	n/a	n/a	53.2	33.0	109.6	50.7	4.02×

Table 17. Runtime and sampling-cost analysis. Lower is better. The average relative cost column first normalizes each available host entry by the corresponding baseline/random cost, then averages over hosts where that ordering policy is implemented with a comparable cost field.

Host and split	Shared generation/evaluation budget	Logged cost field	Interpretation
Omni-Diffusion T2I, 64 prompts	Official 260-step visual-token generation path with 256 generated image tokens per prompt; no extra denoiser forward passes for DPRM scoring.	DPRM(conf.): 21.04 s mean generation time and 39.69 s total prompt time. DPRM(random): 20.72 s mean generation time and 41.96 s total prompt time.	Runtime is dominated by multimodal denoising, visual-token decode, and CLIP/evaluation bookkeeping; bucket lookup is scalar ranking overhead.
LLaDA-V AI2D, 500 documents	Same image-conditioned VQA prompt/decode settings for all four orders; order acts only on generated answer-token positions.	Total evaluation time: random 1055 s, progressive 974 s, DPRM(conf.) 981 s, DPRM(random) 982 s.	DPRM adds negligible wall-clock overhead relative to the image-conditioned language-diffusion evaluation loop.
LLaDA-V RealWorldQA, 765 documents	Same RealWorldQA lmms-eval split and decode settings for all four orders; order acts only on generated answer-token positions.	Total evaluation time: random 2177 s, progressive 2148 s, DPRM(conf.) 2127 s, DPRM(random) 2166 s.	The cost differences are within ordinary evaluation variability; the negative DPRM result on this task is quality/generalization, not runtime.

Table 18. Multimodal runtime and sampling-cost analysis. These rows are reported separately from Tab. 17 because prompt-level text-to-image generation and lmms-eval VQA jobs are not directly comparable to the scalar cost fields used by the other seven tasks.

The analysis shows a task-dependent compute profile rather than a uniform wall-clock speedup over the original baseline. DPRM(conf.) is close to baseline on DCM and much faster than Progressive-DCM, which makes its average relative cost lower than the confidence-progressive row. In SDPO-DNA, Prism, and DPLM, the DPRM controller pays extra cost for reward-guided or shortlist-based scoring; these overheads are the cost side of the same ordering intervention used for the quality gains reported above.

B.14. 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 evaluators 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 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. Both the main-text checkpoint comparison and the paired-bootstrap uncertainty analysis use the same shared EMA checkpoint at step 1.53M, so the point estimates and the uncertainty table are 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 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. 4b 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 comparison, 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 method. The fragment analysis is therefore an ordinary bootstrap over shared task units rather than a paired bootstrap over identical generated molecules. This is appropriate 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. 4d 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.

Omni-Diffusion family. For Omni-Diffusion, the evaluation unit is a text-to-image prompt under the official 260-step visual-token generation path. Each ordering policy generates one image for each of the same 64 prompts. The evaluator records image validity, number of generated visual tokens, CLIP-L/14 image–text cosine, per-prompt generation time, and total decode/evaluation time. Generated images are stochastic and method-specific, so the reported mean CLIP values are method summaries over prompts. Prompt-level deltas are useful diagnostics but are interpreted conservatively because the four policies sample different images for the same prompt.

LLaDA-V family. For LLaDA-V, the evaluation unit is an image-conditioned VQA document. All four orders are evaluated on the same 500 AI2D examples and the same 765 RealWorldQA examples, so target-normalized accuracy deltas can be paired over document ids. The target-fixed table builder keys rewards by order, task, and document id before aggregation. The entropy/EOT diagnostic additionally logs selected confidence, entropy proxy, candidate EOT count, selected EOT count, and EOT confidence for every decode decision.

B.15. Empirical Diagnostics for the Finite-Sample Theory

Sec. C.4 formalizes a finite-sample optimization story with two practical claims. First, confidence is useful early because high-confidence positions are usually easier local denoising decisions. Second, confidence can become too narrow later because hard tasks may require visiting order classes that are not locally most confident but have higher future reward. The theorem calls these late useful classes *residual families*; in experimental terms, they are simply under-selected bucket regions that still lead to better terminal outcomes.

This subsection is organized as a reader's guide to the diagnostics. It answers four questions. Does confidence really identify easier positions? Does DPRM recover some useful low-confidence coverage without reverting to random exploration? Does the online bucket estimator show the empirical-Bernstein shrinkage predicted for the sampling-error term? Is the gain more than a reward-blind entropy increase or a hand-designed EOT fix? The first three questions are probed with instrumented Countdown logs; the last uses SDPO-DNA and LLaDA-V controls summarized in Fig. 2 and Tab. 20.

For Countdown, difficulty labels are the number of operands in the target arithmetic expression. The diagnostic suite includes random aligned-order DMPO, confidence-only Progressive DMPO ($\beta = 0$), and DPRM-DMPO with $\beta \in \{0.5, 1.0, 2.0\}$. Each 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 a CE true-logit gradient proxy. These are proxies for the theorem's forward-KL and gradient quantities, not direct measurements of the theorem constants.

Optimization trace. Fig. 19 is the coarse training-level view. All curves use matched DMPO training logs under their respective ordering policies. The important qualitative pattern is the large separation between random masking and train–test aligned progressive masking. This matches the early-stage theorem intuition: confidence does not need to be globally optimal; it only needs to focus early SGD updates on easier partial states often enough to beat random state sampling. DPRM stays close to the confidence scaffold early because it is explicitly warmed up before its bucket statistics are trusted.

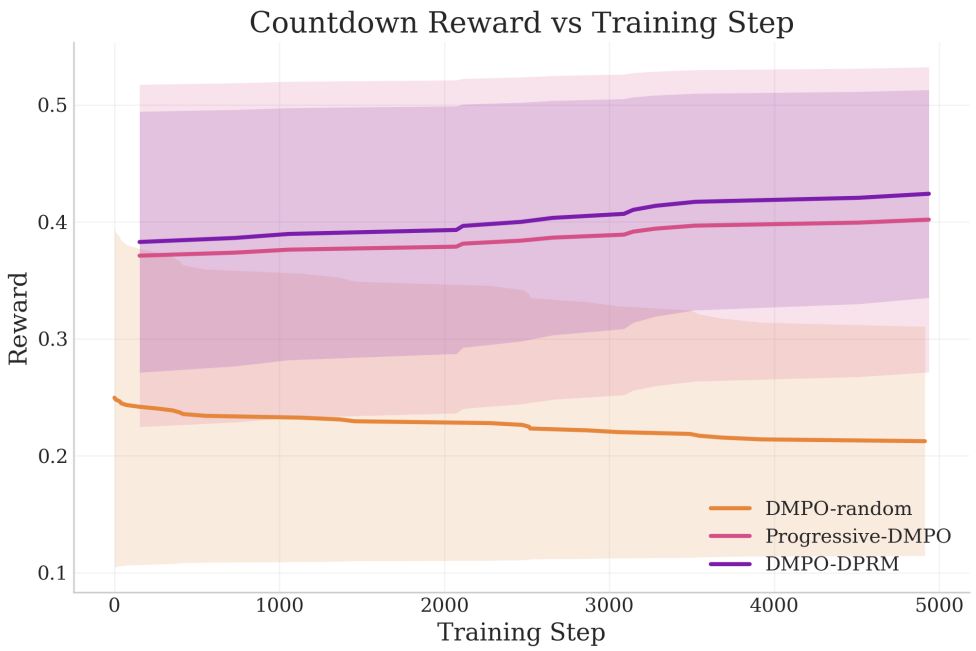

Figure 19. Countdown training reward versus global step from W&B logs. Bands use the logged reward standard deviation. The curves compare matched ordering policies and are used as optimization diagnostics rather than final evaluation metrics.

How to read the token-level Countdown figure. Fig. 20 zooms in from a training curve to individual reveal decisions. Panel (a) asks whether confidence is a useful local proxy: if higher confidence bins have lower CE loss, then confidence-ranked orders select easier denoising decisions. The correlations in Tab. 19 are strongly negative, between -0.94 and -0.97, so this proxy relation is clear. Panel (b) asks what the controller actually selects. Random ordering selects early tokens with CE 0.740, while Progressive DMPO selects 0.477 and DPRM with $\beta = 1$ selects 0.442. Lower selected CE means the policy spends early training budget on easier, more stable decisions. Panel (c) then switches to the late-stage question: how much probability mass remains on low-confidence bins that confidence-only ordering tends to ignore.

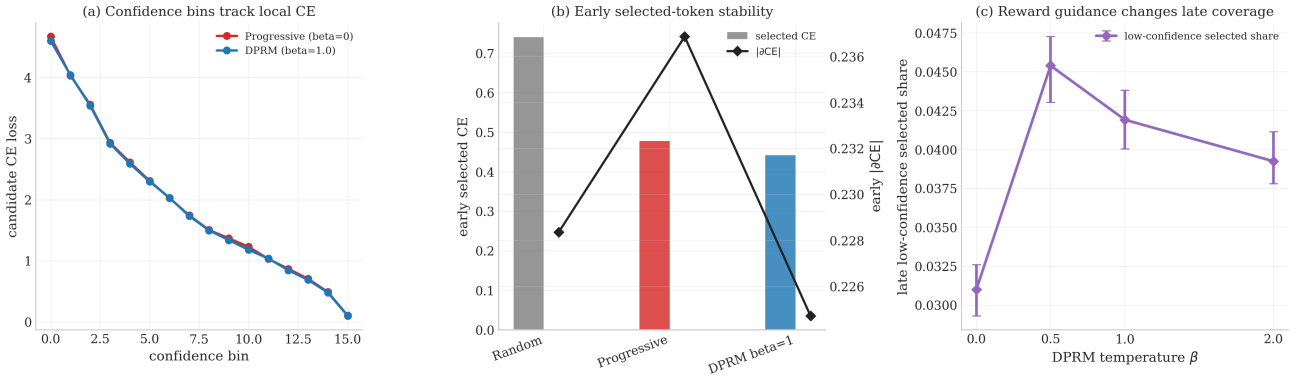

Figure 20. How the Countdown token-level diagnostics should be read. Panel (a): confidence bins are a strong proxy for local CE loss, so confidence is a useful early ordering signal. Panel (b): confidence-aligned controllers select lower-CE tokens than random ordering early, meaning the same training budget is spent on easier partial states. Panel (c): DPRM restores a small amount of late low-confidence coverage relative to confidence-only ordering. Intervals in Panel (c) are bootstrap intervals over logged confidence-bin snapshots.

Table 19. Numerical reading guide for Fig. 20. Early CE is the local token-prediction error of selected positions; lower is easier. Corr(conf., CE) close to -1 means higher confidence bins have lower CE. Late low-conf. selected % measures whether the policy still visits bins 0–5 out of 16. Late DPRM score is the learned bucket score used to tilt the order.

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

How to read the low-confidence coverage figure. Fig. 21 is not saying that low confidence is always good. Random ordering visits many low-confidence tokens, but it also wastes early training budget and has much worse selected CE. The useful pattern is more specific: confidence-only Progressive DMPO almost never visits low-confidence bins late in training, with only 3.1% of selected tokens in bins 0–5. DPRM raises this selected mass modestly to 4.5%, 4.2%, and 3.9% for $\beta = 0.5, 1.0, 2.0$. At the same time, the selected DPRM score becomes positive and grows with β. This is the intended compromise: start from the confidence-aligned scaffold, then add reward-guided visits to historically useful low-confidence buckets once the process-reward estimates are populated.

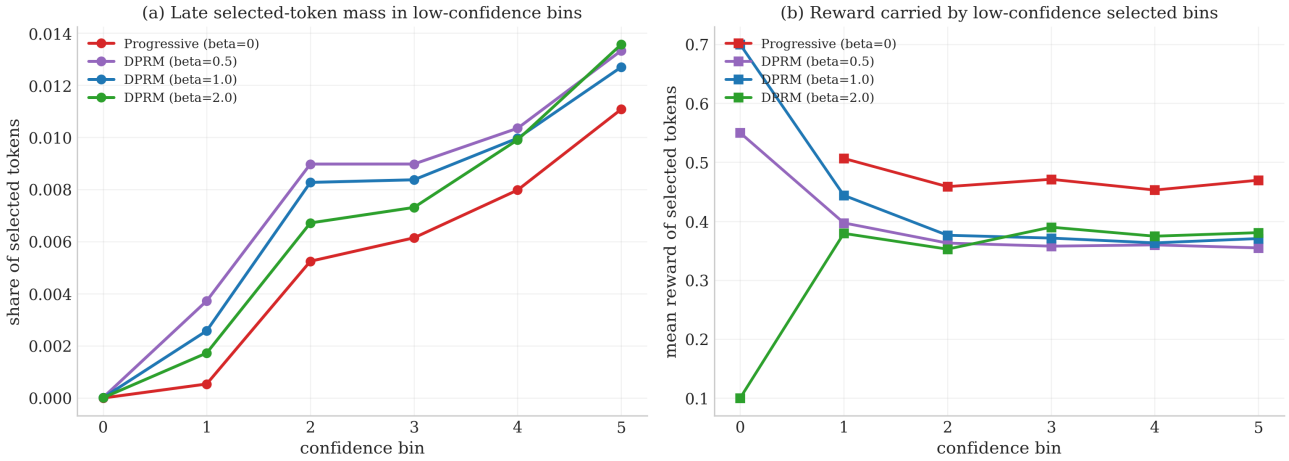

Figure 21. Late-stage coverage inside the low-confidence region on Countdown. Confidence-only Progressive DMPO rarely selects bins 0–5 late in training. DPRM restores a small amount of this coverage and assigns positive learned DPRM scores to the selected buckets. This is the observable analogue of the theorem's late under-coverage assumption.

Empirical-Bernstein rate diagnostic. Theorem 3.2 bounds the online bucketized DPRM score by a sampling-error term plus an abstraction-bias term. The sampling-error term is controlled by an empirical-Bernstein radius, so the diagnostic logger records the variance of $\exp(\beta R)$, the empirical-Bernstein radius, bucket drift, and CE-gradient proxies throughout training. Fig. 22 checks this part of the theory. The empirical-Bernstein radius falls rapidly as bucket counts accumulate, which is the expected finite-sample behavior of the estimator. The bucket-drift panel is deliberately separate: it tracks nonstationarity and coarsening effects, not sampling error. Thus this plot validates that the implemented controller has the predicted radius shrinkage, while the remaining gap to exact DPRM is still represented by the bias term $\mathrm{Bias}_t(s; \delta)$.

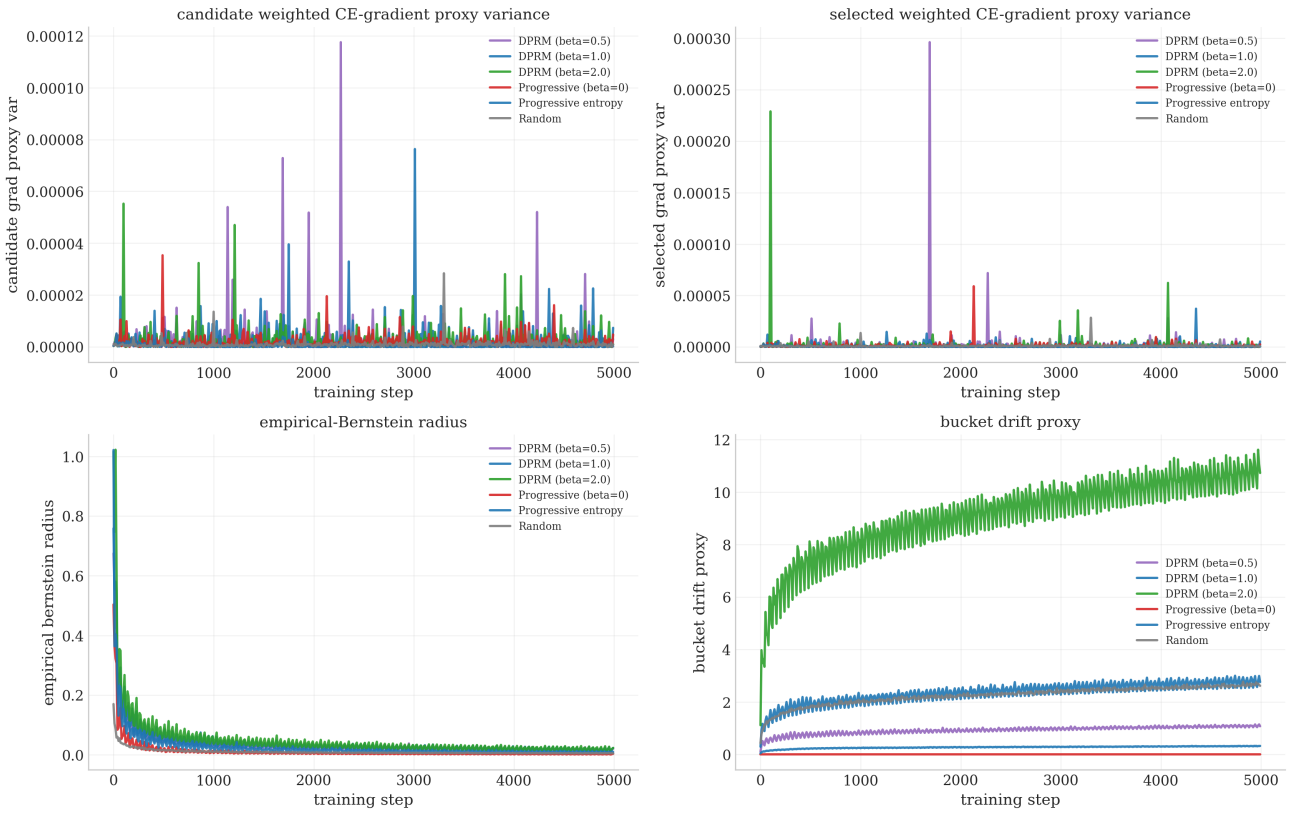

Figure 22. Empirical-Bernstein diagnostics for the online bucketized DPRM estimator on Countdown. The lower-left panel shows the empirical-Bernstein radius shrinking as bucket evidence accumulates. The top panels show the logged CE-gradient proxy variance for candidate and selected decisions, and the lower-right panel separates bucket drift from the sampling-error radius. The figure is a diagnostic for the theorem's finite-sample rate term, not a direct measurement of the exact abstraction bias.

Task-level consequence on Countdown. The Countdown evaluation uses the DPRM-DMPO setup in Sec. B.4: 5,000 post-training steps, $K = 8$ progressive phases, 16 confidence bins, $\beta = 1.0$, the same Fast-dLLM decode budget, and pass@K evaluation for $K \in \{1, 2, 4, 8, 16, 32\}$ on the fixed 5,120-example split. Fig. 23 connects the logged token-level behavior to task outcomes. Panel (a) checks the early proxy assumption: confidence-aligned Progressive DMPO improves pass@1 over random DMPO most on easy operand-count subsets, where local confidence is most likely to identify stable partial states. Panels (b)–(c) check the late residual-family prediction: on hard and OOD subsets, confidence-only ordering leaves useful trajectories under-covered, and DPRM improves at larger sampling budgets. For example, Countdown hard pass@32 increases from 47.9% under Progressive DMPO to 60.0% under DPRM-DMPO, and OOD pass@32 increases from 26.0% to 36.3%. This is the outcome-level pattern predicted by the finite-sample theorem and summarized visually in Fig. 2(a): confidence alignment explains the large random-to-progressive gain, while reward-tilted bucket guidance is most useful when the remaining successful trajectories are not simply the locally most confident ones.

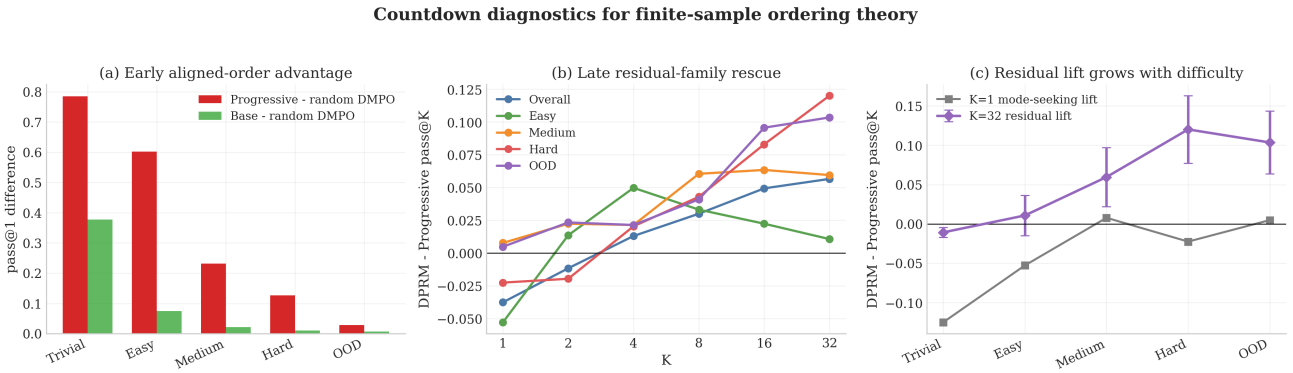

Figure 23. Outcome-level diagnostics for the finite-sample ordering theory on Countdown under the DPRM-DMPO evaluation protocol in Sec. B.4. Panel (a) supports the early-stage assumption that confidence is a useful local optimization proxy: confidence-aligned progressive training dominates random DMPO at pass@1, especially on easier operand-count subsets. Panels (b)–(c) show where the DPRM pipeline improves over confidence-only Progressive DMPO on harder subsets and at larger pass@K.

Entropy and bucket-abstraction controls. The theorem separates useful reward tilt from two simpler effects: generic exploration and bucket readiness. We therefore use reward-blind uncertainty controls and bucket ablations on hosts where the necessary artifacts are matched. The uncertainty score is

$$u_i(s_t) = 1 - \max_v p_\theta(v \mid s_t),$$

and uses the same reveal budget as the corresponding confidence or DPRM policy. Tab. 20 summarizes the compact controls. On SDPO-DNA, entropy-only improves over confidence-progressive but remains far below DPRM(random), and shuffling the learned bucket values reduces the total metric while keeping the same counts and gates. This supports the practical claim that, in the $P = 1$ DNA abstraction, the bucketized terminal-utility estimate is not merely an entropy increase. The LLaDA-V controls test a related but different low-dimensional phenomenon. Park et al. (2026) explicitly design suffix-anchor confidence modulation to handle high-confidence EOT and anchor-proximity failures. DPRM is not designed with an EOT rule; it only sees phase/confidence buckets and terminal utility. Thus, if DPRM improves an image-conditioned text task without explicitly suppressing EOT or applying suffix-anchor modulation, it is evidence that the bucketized process reward is learning a useful ordering regularity rather than merely applying a hand-coded EOT fix. This is what happens on AI2D: DPRM(conf.) improves over confidence-progressive, entropy-only, and SACM while the selected-EOT rate stays at 0.250. RealWorldQA gives the boundary case: direct EOT suppression is strongest, while DPRM(conf.) and SACM are weaker, so the coarse phase/confidence table is missing task-format or EOT-specific auxiliary variables needed for heterogeneous open-world VQA.

Table 20. Compact entropy and bucket controls for the finite-sample diagnostics. Higher task metrics are better; entries of the form a/b report target accuracy and selected-EOT rate. All rows keep the host model, sampler budget, and evaluation split fixed within the row's host.

Host	Diagnostic	Reference	Control	DPRM / conclusion
SDPO-DNA	Total metric, 3 seeds	Progressive 0.634	Entropy 1.141	DPRM(random) 2.129
SDPO-DNA	Bucket value ablation	Normal 2.129	Shuffled 1.866	Utility values contribute beyond counts
SDPO-DNA	Readiness	–	Gate/count 2.070	Selected-ready rate 1.000 at $P = 1$
LLaDA-V AI2D	Target acc. / EOT rate	Progressive 0.658/0.250	Entropy 0.634/0.250; SACM 0.678/0.192	DPRM(conf.) 0.690/0.250
LLaDA-V RealWorldQA	Target acc. / EOT rate	Progressive 0.460/0.250	EOT-supp. 0.498/0.115; SACM 0.435/0.152	Coarse DPRM 0.418/0.250; richer bins needed

Together, these controls give a task-conditional view of the abstraction-bias term in the theorem. For SDPO-DNA with $P = 1$, bucket coverage is complete and the reward-bucket values beat entropy-only and shuffled-value controls, so the practical bucket approximation is accurate enough to guide ordering; the gate/count rows also show that readiness structure contributes. For LLaDA-V AI2D, phase/confidence buckets transfer useful target-normalized reward information beyond entropy-only exploration. For RealWorldQA, the same coarse abstraction misses EOT and answer-format structure, so the bias term is visibly larger and DPRM is not the best policy.

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

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*

$$p_{\theta^\star}(u \mid z, q) = \mathbb{P}(O_i = u \mid Q = q, Z = z) \qquad \textit{for almost every } (q, z, i) \textit{ with } i \in M(z).$$

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

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

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 $Z_1, \ldots, Z_T \sim P$ 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 $U_1, \ldots, U_d$ 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

C.2.1. Exact-state and bucket-state definitions used by the experiments

The proof compares two objects that should not be conflated. The exact DPRM object conditions on the full partial state. For a masked or diffusion host, this state is

$$S_t = s_t = (\text{prompt/condition},\ \text{current visible tokens},\ \text{mask pattern},\ t,\ \text{host side information}),$$

and an action $A_t = i$ is the candidate position, token block, residue, gene bin, SAFE token, or DNA token to update next. The exact Doob process reward is

$$R_t^\star(i; s_t) = \frac{1}{\beta}\log \mathbb{E}_{q_0}[\exp(\beta R(X_T)) \mid S_t = s_t,\ A_t = i]\,. \tag{16}$$

This is the conditional expectation one would use if every partial sequence state could be stored and revisited with enough samples.

It is important that i is an ordering action, not a replacement token. At state s_t, the host model still defines the token kernel

$$K_\theta(y \mid s_t, i) = p_\theta(X_i = y \mid S_t = s_t),$$

or, in teacher-forced training, the revealed value is fixed to the ground-truth token $x_i^\star$. DPRM changes the distribution over the next position I_t, while leaving this host token kernel, architecture, loss, and labels unchanged. The confidence score used by the controller is therefore a position score computed from the current token distribution, typically

$$\psi_i(s_t) = \max_y K_\theta(y \mid s_t, i), \tag{17}$$

with host-specific variants that use the probability of the ground-truth token during teacher-forced training or the probability of the sampled token during stochastic decoding. Thus a phrase such as "low-confidence bucket" means a bucket of positions whose current token distribution is uncertain; it does not mean that DPRM directly chooses a low-probability token.

The reveal order nevertheless changes the final token distribution because future token kernels are conditioned on the newly visible tokens. For a reveal policy ρ, the induced trajectory law has the schematic factorization

$$\mathbb{P}_\rho(x_T) = \sum_{i_{0:T-1},\, y_{0:T-1}} \prod_{t=0}^{T-1} \rho_t(i_t \mid s_t)\, K_\theta(y_t \mid s_t, i_t)\, \mathbf{1}\{s_{t+1} = U(s_t, i_t, y_t)\}, \tag{18}$$

where U is the host state update that reveals y_t at position i_t. Changing ρ_t changes the sequence of partial states s_t, and hence changes the later token distributions $K_\theta(\cdot \mid s_t, i)$, even though K_θ itself is not modified. This is the operational meaning of an ordering-only intervention.

The implemented online DPRM uses a coarser state because exact partial sequences are almost never revisited. For each candidate decision it records

$$Z_t := \zeta_t(S_t, A_t) = \big(\phi_t(S_t),\ b_t(S_t, A_t),\ c_t(S_t, A_t)\big), \tag{19}$$

where ϕ_t is the phase id, b_t is the confidence bin, and c_t is an optional auxiliary bin. Most experiments set c_t to a single dummy value, so the theorem writes buckets as (ϕ, b) for readability. The phase id is a coarse progress label. In DMPO, for example, it is obtained from the local denoising or reveal step inside a block; with 8 phases, early local steps map to small phase ids and late local steps map to large phase ids. In progressive teacher-forced training, the same idea is implemented by the current reveal ratio or remaining-mask schedule. The confidence bin is a discretization of the host model's current score under the actual partial state:

$$b_t(S_t, i) = \min\{B - 1, \lfloor B\, \psi_i(S_t) \rfloor\}, \qquad \psi_i(S_t) \in [0, 1], \tag{20}$$

or the analogous monotone binning of the host proposal score. Thus the full partial sequence affects DPRM through the host score $\psi_i(S_t)$, the mask/progress-dependent phase $\phi_t(S_t)$, and any host-specific auxiliary bin c_t, but it is not stored as a table key.

The bucket target is the log-moment projection of the exact target onto the sigma-field generated by Z_t:

$$\bar{R}_{z,t} := \frac{1}{\beta} \log \mathbb{E}[\exp(\beta R(X_T)) \mid Z_t = z]\,. \tag{21}$$

Equivalently, if $H_t(S_t, i) := \mathbb{E}[\exp(\beta R(X_T)) \mid S_t, A_t = i]$, then by the tower property

$$\exp(\beta \bar{R}_{z,t}) = \mathbb{E}[H_t(S_t, A_t) \mid \zeta_t(S_t, A_t) = z]\,. \tag{22}$$

Therefore bucketized DPRM is not claiming to know the exact value of a never-seen partial sequence. It estimates the average exponentiated future reward of the historical decisions that had the same phase, confidence bin, and optional auxiliary bin. This is reasonable when the missing exact details of the partial sequence are less predictive of the ordering advantage than the coarse progress and uncertainty regime. For example, two Countdown partial responses may contain different numbers and intermediate arithmetic strings, so their exact states will never match, but both may be late-block decisions in a medium-confidence bucket. If historical decisions from that bucket produce higher verified final rewards than neighboring confidence buckets, the bucket value supplies a reusable correction to the position order on a new problem. The same abstraction is used in protein, molecular, gene-expression, DNA, and visual-token settings: individual sequences or images differ, but reveal progress and model uncertainty define repeatedly visited decision regimes. The abstraction error in the theorem is precisely the within-bucket mismatch

$$|R_t^\star(i; s_t) - \bar{R}_{\zeta_t(s_t,i),t}|,$$

which becomes small only when decisions sharing a bucket have similar future log-moment reward.

Given $N_{z,t}$ observed decisions in bucket z, the empirical estimator is

$$\widehat{R}_{z,t} = \frac{1}{\beta} \log \left(\frac{1}{N_{z,t}} \sum_{j:\, Z_j = z} \exp(\beta R_j) \right), \qquad N_{z,t} > 0. \tag{23}$$

The implementation uses the safe convention $\widehat{R}_{z,t} = 0$ when $N_{z,t} = 0$, together with a readiness gate proportional to $N_{z,t}/N_{\text{ready}}$. Consequently, an unseen exact partial sequence has three possible behaviors at inference. If it falls in a populated bucket, DPRM tilts its order by the historical conditional terminal reward of that bucket. If it falls in an under-populated bucket, the same tilt is downweighted by the local readiness gate. If the bucket has never been observed, the DPRM term is zero and the ordering reduces to the host base score, typically confidence ordering.

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

$$\mathrm{rad}_t(s;\delta) := \max_{i \in M(z)} \frac{1}{\beta \underline{\mu}} \left[\sqrt{\frac{2\widehat{v}_{\phi,b_i(s),t} \log(3KBT^2/\delta)}{N_{\phi,b_i(s),t}}} + \frac{3(e^\beta - 1)\log(3KBT^2/\delta)}{N_{\phi,b_i(s),t}} \right] \tag{24}$$

and

$$\varepsilon_t(s;\delta) := \beta \varepsilon_{\mathrm{abs},t}(s) + \beta \eta_t(s)\, \mathrm{rad}_t(s;\delta) + \beta(1-\eta_t(s))\, B_t(s) + \frac{2}{\underline{\mu}} D_t(s). \tag{25}$$

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 $t \le T$,

$$\sup_{i \in M(z)} |\widehat{g}_t(i;s) - g_t^\star(i;s)| \le \varepsilon_t(s;\delta). \tag{26}$$

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

$$0 \le \sum_{i \in A_t^\star(s)} g_t^\star(i;s) - \sum_{i \in \widehat{A}_t(s)} g_t^\star(i;s) \le 2\, m(s)\, \varepsilon_t(s;\delta). \tag{27}$$

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.2.2. 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 $1-\delta$,*

$$|\widehat{\mu}_{\phi,b,t} - \mu_{\phi,b,t}| \le \sqrt{\frac{2\widehat{v}_{\phi,b,t}\log(3/\delta)}{N_{\phi,b,t}}} + \frac{3(e^{\beta}-1)\log(3/\delta)}{N_{\phi,b,t}}. \tag{28}$$

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

Lemma C.13 (Bucket reward concentration)**.** *Assume also that $\mu_{\phi,b,t} \ge \underline{\mu} > 0$. Then, on the event in (28),*

$$|\widehat{R}_{\phi,b,t} - \bar{R}_{\phi,b,t}| \le \frac{1}{\beta\underline{\mu}}\left[\sqrt{\frac{2\widehat{v}_{\phi,b,t}\log(3/\delta)}{N_{\phi,b,t}}} + \frac{3(e^{\beta}-1)\log(3/\delta)}{N_{\phi,b,t}}\right]. \tag{29}$$

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 $\mu_{\phi,b,t} \ge \underline{\mu}$, the mean-value theorem gives

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

Substituting (28) proves (29). □

In the online setting, the bucket mean is generally nonstationary. This nonstationarity is not only due to changing visitation frequencies, but also due to updates of the host denoiser. At training update u, the base proposal $q_{0,u}$ is induced by the current score ψ_{θ_u}, and therefore the exact conditional log-moment reward $R^{\star}_{u,t}$ and its bucket projection $\bar{R}_{\phi,b,u}$ may change with (u). We absorb this moving-target effect into the variation budget

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

Thus the online tracking theorem should be read as a guarantee for slowly drifting bucket means; when the host model changes rapidly, the drift term becomes large, and the guarantee correspondingly weakens.

C.2.4. 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 $t \le T$,*

$$|\widehat{R}_{\phi,b,t} - \bar{R}_{\phi,b,t}| \le \mathrm{rad}_{\phi,b,t}(\delta) + \frac{2}{\underline{\mu}}D_{\phi,b}(t), \tag{30}$$

where

$$\mathrm{rad}_{\phi,b,t}(\delta) := \frac{1}{\beta\underline{\mu}}\left[\sqrt{\frac{2\widehat{v}_{\phi,b,t}\log(3KBT^2/\delta)}{N_{\phi,b,t}}} + \frac{3(e^{\beta}-1)\log(3KBT^2/\delta)}{N_{\phi,b,t}}\right]. \tag{31}$$

Proof. Apply Thm. C.13 to every bucket and every time $t \le T$ 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 $\mu_{\phi,b,u} \ge \underline{\mu}$, the mean-value theorem yields

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

Summing over $u \le t$ gives a cumulative drift guidance proportional to $D_{\phi,b}(t)$. Using the slightly looser coefficient $2/\underline{\mu}$ yields (30). □

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

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

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

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

where $\varepsilon_t(s; \delta)$ is the quantity in (25).

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

$$\begin{aligned}|\widehat{g}_t(i; s) - g^\star_t(i; s)| &= \beta \left|\eta_t(\phi, b_i(s))\widehat{R}_{\phi,b_i(s),t} - R^\star_t(i; s)\right| \\ &\le \beta|R^\star_t(i; s) - \bar{R}_{\phi,b_i(s),t}| + \beta\eta_t(\phi, b_i(s))|\widehat{R}_{\phi,b_i(s),t} - \bar{R}_{\phi,b_i(s),t}| \\ &\quad + \beta(1 - \eta_t(\phi, b_i(s)))|R^\star_t(i; s)|. \end{aligned} \tag{32}$$

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 (25). □

C.2.5. Step 4: score error implies reveal regret

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

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

and let

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

If

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

then

$$0 \le J^\star_t(A^\star_t(s); s) - J^\star_t(\widehat{A}_t(s); s) \le 2m(s)\varepsilon. \tag{33}$$

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

$$\begin{aligned} J_t^\star(A_t^\star(s);s) &\le \widehat{J}_t(A_t^\star(s);s) + m(s)\varepsilon \\ &\le \widehat{J}_t(\widehat{A}_t(s);s) + m(s)\varepsilon \\ &\le J_t^\star(\widehat{A}_t(s);s) + 2m(s)\varepsilon. \end{aligned} \tag{34}$$

This proves (33). □

Proof of Theorem C.11. The score bound (26) is exactly Thm. C.15. Applying Thm. C.16 with $\varepsilon = \varepsilon_t(s;\delta)$ gives (27). □

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. (2026). Second, the stagewise Soft-BoN approximation follows from the main KL approximation theorem of Verdun et al. (2025).

Setup. Let

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

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. (2026). 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*

$$\mathbb{E}[\mathrm{KL}(\pi_t^\star(\cdot \mid s) \,\|\, \widehat{\pi}_{t,N}(\cdot \mid s; A_{t,1:N}))] \le \frac{\sinh(\beta/2)^2}{N}. \tag{35}$$

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 (35). □

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,*

$$\mathrm{KL}\Big(\mathbb{P}_\beta \,\Big\|\, \widehat{\mathbb{P}}_{\{N_t\}}\Big) = \mathbb{E}_{\mathbb{P}_\beta}\left[\sum_{t=0}^{T-1} \mathrm{KL}\big(\pi_t^\star(\cdot \mid S_t) \,\big\|\, \widehat{\pi}_{t,N_t(S_t)}(\cdot \mid S_t)\big)\right]. \tag{36}$$

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 (36). □

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

$$\mathbb{E}\Big[\mathrm{KL}\Big(\mathbb{P}_\beta \,\Big\|\, \widehat{\mathbb{P}}_{\{N_t\}}\Big)\Big] \le \sinh(\beta/2)^2 \, \mathbb{E}_{\mathbb{P}_\beta}\left[\sum_{t=0}^{T-1} \frac{1}{N_t(S_t)}\right]. \tag{37}$$

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

$$\mathbb{E}\Big[\mathrm{KL}\Big(\mathbb{P}_\beta \,\Big\|\, \widehat{\mathbb{P}}_N\Big)\Big] \le \frac{T \, \sinh(\beta/2)^2}{N}. \tag{38}$$

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

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

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

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

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

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

Substituting this bound gives (37). If $N_t(s) \equiv N$, then

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

so (38) 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 can be exponentially faster than random aligned-order sampling in the assumed latent-family regime. This view mirrors curriculum learning (Bu et al., 2025) 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 sufficiently accurate DPRM guidance yields a conditional exponential separation 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

$$\mathcal{K}_t(\theta) := \mathbb{E}_{x\sim\widehat{P}_n}\mathbb{E}_{o\sim u_t(\cdot|x)}\Big[\mathrm{KL}\big(p_t^\star(\cdot \mid x, o) \,\|\, p_\theta(\cdot \mid x, o)\big)\Big]. \tag{39}$$

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 (39), 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

$$\widehat{G}_t^{(p)} := \frac{u_t(O_t \mid X_t)}{p_t(O_t \mid X_t)}\, g_t(\theta_t; X_t, O_t, \xi_t), \tag{40}$$

with conditional second moment

$$\mathcal{M}_t(p) := \mathbb{E}\Big[\|\widehat{G}_t^{(p)}\|_2^2 \mid \theta_t\Big]. \tag{41}$$

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*

$$p_t^{\mathrm{opt}}(o \mid x) \propto u_t(o \mid x)\, \nu_t(\theta_t; x, o). \tag{42}$$

Proof. Condition on x. By (40) and (41),

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

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

$$p_t^{\mathrm{conf}}(o \mid x) := \frac{u_t(o \mid x)c_t(x, o)}{\sum_{o'} u_t(o' \mid x)c_t(x, o')}, \tag{43}$$

and the random aligned proposal

$$p_t^{\mathrm{rand}}(o \mid x) := \frac{1}{|\mathcal{O}_t(x)|}. \tag{44}$$

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

$$\mathcal{M}_t\big(p_t^{\mathrm{conf}}\big) \le \kappa_t^2\, \mathcal{M}_t\big(p_t^{\mathrm{opt}}\big), \tag{45}$$

and

$$\mathcal{M}_t\big(p_t^{\mathrm{rand}}\big) \ge (1-\rho_t)^2 e^{a_t d_t}\, \mathcal{M}_t\big(p_t^{\mathrm{opt}}\big). \tag{46}$$

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 (41) yields (45).

For the second bound, condition on x and write

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

Then

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

By Cauchy–Schwarz,

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

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 (46). □

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* (40)*.*

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

$$T_t^{(\mathrm{conf})}(\varepsilon) = O\left(\frac{L_t \kappa_t^2\, \mathcal{M}_t(p_t^{\mathrm{opt}})}{\mu_t^2\, \varepsilon} \log \frac{\mathcal{K}_t(\theta_0) - \mathcal{K}_t^\star}{\varepsilon}\right). \tag{47}$$

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

$$T_t^{(\mathrm{rand})}(\varepsilon) = \Omega\left(\frac{L_t (1-\rho_t)^2 e^{a_t d_t}\, \mathcal{M}_t(p_t^{\mathrm{opt}})}{\mu_t^2\, \varepsilon} \log \frac{\mathcal{K}_t(\theta_0) - \mathcal{K}_t^\star}{\varepsilon}\right). \tag{48}$$

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 (45) gives (47).

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 (46) yields (48). □

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*

$$H(X) \le L h_V(\delta), \qquad B_{\mathrm{eff}} := \exp(H(X)/L) \le \exp(h_V(\delta)), \tag{49}$$

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*
$$\widehat{\pi}_t(\mathcal{R}_t(s) \mid s) \ge \frac{|\mathcal{R}_t(s)|\, e^{\Delta_t^{\mathrm{res}}(s) - 2\epsilon_t(s)}}{|\mathcal{R}_t(s)|\, e^{\Delta_t^{\mathrm{res}}(s) - 2\epsilon_t(s)} + |\mathcal{O}_t(s) \setminus \mathcal{R}_t(s)|}; \tag{50}$$

2. *if, in addition,*
$$\Delta_t^{\mathrm{res}}(s) - 2\epsilon_t(s) \ge b_t h_t + \log \frac{|\mathcal{O}_t(s) \setminus \mathcal{R}_t(s)|}{|\mathcal{R}_t(s)|} + c_t$$
for some $c_t > 0$, then
$$\widehat{\pi}_t(\mathcal{R}_t(s) \mid s) \ge \frac{1}{1 + e^{-c_t}} =: p_t^{\mathrm{DPRM}}, \tag{51}$$
whereas
$$\pi_t^{\mathrm{conf}}(\mathcal{R}_t(s) \mid s) \le C_t e^{-b_t h_t}.$$

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
$$T_{t,\mathrm{late}}^{(\mathrm{conf})}(\varepsilon) = \Omega\left(\frac{e^{b_t h_t}}{\gamma_t C_t} \log \frac{\mathcal{K}_t^{\mathrm{res}}(\theta_{T_{\mathrm{warm}}})}{\varepsilon}\right), \tag{52}$$
whereas
$$T_{t,\mathrm{late}}^{(\mathrm{DPRM})}(\varepsilon) = O\left(\frac{1}{\gamma_t p_t^{\mathrm{DPRM}}} \log \frac{\mathcal{K}_t^{\mathrm{res}}(\theta_{T_{\mathrm{warm}}})}{\varepsilon}\right). \tag{53}$$
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 (50). If the stronger gap condition holds, the right-hand side is at least

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

which proves (51).

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 (52). Under DPRM, $p \geq p_t^{\mathrm{DPRM}}$, which yields (53). □

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 (52) *and* (53) *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.