SAMPLE THE ORDER, NOT THE TOKEN: TEST-TIME SCALING FOR MASKED DIFFUSION LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Standard test-time scaling methods like temperature-sampled self-consistency are highly effective for autoregressive models, but we show it is not true for masked diffusion language models (dLLMs). Attempting to scale dLLMs via temperature sampling collapses generation length and plummets accuracy (e.g., from 35% to 6% on MATH500), failing to yield the diversity required for ensembling. Motivated by this misalignment, we propose Order-Space Exploration (OSE), a novel decoding strategy demonstrating that dLLM diversity should originate from the order of token generation, not the token distribution itself. We first observe that a perfect model's answer distribution is invariant to unmasking order, thus shuffling this schedule yields "free" diversity driven entirely by the model's internal disagreement across generation paths, avoiding the accuracy penalty of temperature scaling. OSE introduces a one-parameter Plackett–Luce perturbation of the sampler's position ranking and aggregates the N draws with a majority vote over their answers. Empirically, OSE raises answer diversity by 2–5$\times$ while leaving single-draw accuracy unchanged. This unlocks test-time scaling for dLLMs, beating standard self-consistency on `Dream` by up to 45.5 points on a variety of downstream tasks. Additionally, OSE rollouts restore the learning signal in GRPO, removing nearly all zero-gradient steps on generation tasks and about half on multiple choice. It improves the trained policy where the base model has room to learn (+7.6 on MATH500, +8.5 on COUNTDOWN), with no change to the model or the inference cost.

1 INTRODUCTION

Test-time scaling is the practice of buying accuracy with inference compute instead of training. Its simplest and most widely used form is self-consistency (Wang et al., 2023): sample several reasoning chains, parse each final answer, and return the plurality. Its premise is that samples differ. For auto-regressive models this is arranged by sampling tokens at a temperature above zero, which trades a little per-sample quality for the diversity that voting needs.

Unlike auto-regressive causal models, masked diffusion language models (dLLMs) (Li et al., 2022; He et al., 2023; Nie et al., 2025b; Ye et al., 2025b; Shi et al., 2024; Sahoo et al., 2024; Ou et al., 2025) generate text by iteratively converting noise or masks into tokens and can therefore 'generate' multiple tokens in parallel. This leads to an additional consideration: *which positions should be sampled (or unmasked) and when?* The most accepted approach is often to rank such positions using confidence scores and unmasked based on greater confidence. In this work, we challenge this orthodoxy. We first observe the settings under which DREAM (Ye et al., 2025b) reports its headline accuracies (temperature 0.1, nucleus 0.9), where the confidence score is exactly zero at almost every masked position, as the nucleus contains a single token. The ranking is therefore decided by tie-breaking, and the tie-break commits the leftmost masked position. On GSM8K, 98.1% of DREAM's commits fall exclusively at the leftmost masked position. The "any-order" sampler is auto-regressive in its realized order and thus deterministic. On 43.7% of GSM8K prompts, eight seeds return eight identical outputs; on COUNTDOWN, sixteen seeds return one answer. Thus self-consistency is exactly flat, while raising the temperature remains unhelpful. At 0.5, DREAM's MATH500 accuracy falls from 35% to 6% because the sampler, its ranking disturbed, commits the padding tail before the reasoning.

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

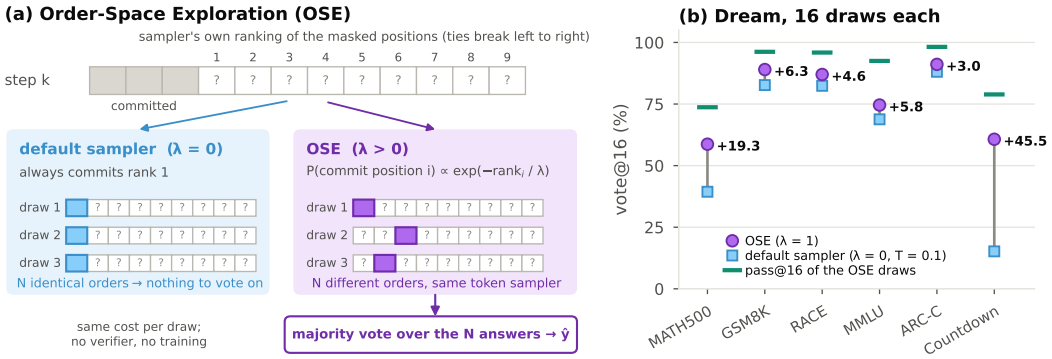

Figure 1: **(a)** OSE: at every step the sampler's own ranking of the masked positions is perturbed (Plackett–Luce, strength λ), so N draws commit in different orders while the token sampler is unchanged; the answers are majority-voted. At $\lambda = 0$ every draw repeats the default order. **(b)** Dream, vote@16 on full test sets: OSE($\lambda = 1$) against the default sampler ($\lambda = 0$, $T = 0.1$), with pass@16 of the OSE draws.

Thus the overarching question becomes: *where should a dLLM's diversity come from?* Our hypothesis is that one additional degree of freedom which dLLMs possess over auto-regressive models can be utilized: the order in which tokens are committed. This is based on two core arguments. First, for a model whose conditionals are the conditionals of a single joint distribution, every commit order is a different factorization of the same distribution, thus perturbing the order changes nothing (Ben-Hamu et al., 2025). Temperature, by contrast, changes the distribution of every conditional, and degrades even a perfect model. Second, when the model is not exactly order-consistent, the disagreement between orders is precisely the mutual information between order and output, which upper-bounds the quality cost of perturbation and is at the same time the quantity an ensemble aggregates over. Order perturbation is exploration whose cost is bounded by its own benefit.

To test this hypothesis, we introduce a new inference-time approach, **Order-Space Exploration** (OSE), which draws completions under a Plackett–Luce perturbation of the sampler's own ranking, controlled by a parameter λ that recovers the default sampler at $\lambda = 0$, and aggregates them by plurality over canonicalised answers (Figure 1). Its training-time form, Order-Diverse GRPO, uses the same draws as rollouts and needs no importance correction, because order perturbation changes which canvases are visited, not which tokens are drawn at them. Using this approach, results on six tasks and two models demonstrate that $\lambda = 1$ raises answer diversity 2–5$\times$ with no single-draw cost; OSE beats self-consistency by 19.3 points on MATH500, 6.3 on GSM8K, 4.6 on RACE, 3.0 on ARC-C and 45.5 on COUNTDOWN. Furthermore, it beats the schedule ensemble HEX (Lee et al., 2025) at matched temperature on all three settings tested and is best or tied at HEX's own temperature; order-perturbed rollouts remove the half to nine tenths of GRPO steps that carry no gradient. Overall we summarize our contributions as:

- We analyze dLLM decoding, revealing that under standard settings, DREAM's confidence sampler is deterministic and strictly left-to-right, causing self-consistency to flatline and temperature to collapse outputs. Furthermore, the untempered confidence rules of Kim et al. (2025) prematurely commit the end-of-text tail.
- We demonstrate that order perturbation incurs zero cost for order-consistent models, bounded strictly by the order–output mutual information (unlike temperature, which is never free).
- We introduce OSE, a single-parameter sampler perturbation. We explain why likelihood-based selection fails on order draws, provide a span-restricted fix for small N, and formulate an RL rollout sampler that strictly eliminates importance weights.
- We demonstrate empirically that OSE significantly improves output diversity and self-consistency across multiple benchmarks, requiring fewer rollouts than comparable methods (Lee et al., 2025).

2 BACKGROUND

Masked diffusion language models (dLLMs) denoise a sequence iteratively. The input is a prompt q and a *canvas* s of L positions, each holding either a token or a mask symbol, and for every masked position i the model outputs a distribution over the vocabulary, $p_\theta(y_i \mid x)$. Training hides a random subset of the tokens of a text and minimizes the cross-entropy of the hidden tokens given the visible ones. Since any subset can be the visible one, the model learns to predict any position from any context; up to the weighting of the subsets this is an any-order auto-regressive objective (Nie et al., 2025a; Kim et al., 2025).

Generation starts from L masks and fills them in T steps, never re-masking. At step k, with canvas s_k and masked set M_k, the sampler makes three separate decisions: *1) how many* positions to fill, n_k, given by a schedule; *2) which* positions, the commit set $C_k \subseteq M_k$, chosen by an ordering rule; and *3) what* token goes at each, a draw from $p_\theta(y_i \mid s_k, q)$ for every $i \in C_k$. We call $\sigma = (C_1, \ldots, C_T)$ the *order*; since nothing is re-masked, y and σ together determine every s_k. OSE changes **2)** only.

Samplers and self-consistency. Throughout, $p_\theta(\cdot \mid s, q)$ is the distribution the sampler draws from: the model's prediction after temperature τ (logits divided by τ), top-k and top-p. Dream's sampler (Ye et al., 2025a) runs at $\tau = 0.1$, top-p 0.9, top-k 50, $T = L = 256$; it scores each masked position by the negative entropy of $p_\theta(\cdot \mid s_k, q)$ computed after these transforms, commits the n_k highest, and breaks ties in favour of the leftmost position. LLaDA's sampler is d1's `generate` (Zhao et al., 2025): left-to-right *blocks* of 32 positions, ranked inside the block by the probability of the most likely token, with greedy token draws ($\tau = 0$). Self-consistency (Wang et al., 2023) draws N completions, parses the final answer of each, and returns the *plurality*; the draws differ only because tokens are sampled at a temperature above zero, which for an auto-regressive model is the only source of variety.

3 ANALYSIS OF EXISTING DECODING STRATEGIES

We first ask how existing samplers behave, using `Dream-7B-Instruct` (Ye et al., 2025b) at its default settings; `LLaDA` is covered at the end of this section and in Appendix B. Four facts stand out.

1. *The confidence score is almost always zero.* The nucleus usually holds a single token, so the entropy is exactly 0: at step 0 of a 256-token canvas 97% of positions are tied at the top score, and ties persist to the last step (Table 2).
2. *The tie-break makes the order left to right.* CUDA `topk` returns the lowest index, so 98.1% of commits on GSM8K and 94.9% on COUNTDOWN land at the leftmost masked position, with Spearman 1.000 between step and position. Moving away from that order costs accuracy: 8.8 points at $\lambda = 4$ and 44 points for a uniform order (Table 1).
3. *Only batching has an impact.* Across seeds, 91–100% of (prompt, draw) pairs are identical; what varies comes from batch composition, because padding length and `bf16` rounding flip near-ties. That noise is all self-consistency has to work with. At $N = 16$ it gains nothing on GSM8K (82.8 → 82.7) or Countdown, and 1.8–5.2 points elsewhere (Table 4).
4. *Raising the temperature breaks the output.* The padding tail is where the model is most confident, so at higher T it is committed before the reasoning and the answer is cut short. GSM8K falls 81.1 → 43.4, MATH500 35.6 → 15.7, and mean content length 223 → 27 tokens (Table 5).

The last row of Table 1 matters for the design. The untempered confidence rules of Kim et al. (2025) commit the end-of-text tail *first* (Spearman −0.93) and leave 61–68% of outputs empty. Any perturbation that lets the most confident positions win first fixes the length before the content, which §4.1 is designed to avoid. `LLaDA` differs in two ways. Its default sampler is not deterministic across draws (it attends to left padding, so batch composition changes the output), and it tolerates temperature 0.9 on short-answer tasks, so we also compare against a tuned temperature (§5.4).

4 ORDER-SPACE EXPLORATION

OSE changes one decision of the sampler: *which* positions are filled next. The model, the token distribution, the temperature and the number of steps all stay as they are. We define the perturbation

Table 1: Realized order and single-draw accuracy of Dream on GSM8K.

order rule	leftmost commits	Spearman(step, position)	accuracy
Entropy (Ye et al., 2025b)	98.1%	1.000	82.7
Uniform Random	2.6%	0.00	38.5
Untempered top-Probability / Margin (Kim et al., 2025)	2.7% / 2.4%	−0.93 / −0.92	23.9 / 25.0

Algorithm 1 Order-Ensemble Decoding (OSE)

Require: prompt q, length L, steps T, schedule $n_1..n_T$, base sampler S, perturbation λ, draws N

for $n = 1..N$ **in parallel** (one batch; draws are independent) **do**
  $x \leftarrow [\mathrm{MASK}]^L$; seed the order noise with (n, seed)
  for $k = 1..T$ **do**
    logits $\leftarrow f_\theta(q, x)$; $M \leftarrow$ masked positions of x ▷ one forward pass, as in S
    $\mathrm{rank}_i \leftarrow$ rank of $i \in M$ by $s(\mathrm{logits}_i)$, ties → leftmost ▷ S's own order
    $C \leftarrow$ top-n_k of $(-\mathrm{rank}_i + \lambda\, g_i)$, $g_i \sim \mathrm{Gumbel}(0,1)$ ▷ $\lambda = 0 \Rightarrow S$'s commit set
    $x_i \leftarrow S.\mathrm{sample_token}(\mathrm{logits}_i)$ for $i \in C$ ▷ S's token rule, unchanged
  end for
  $a^{(n)} \leftarrow \mathrm{canonical}(\mathrm{extract_answer}(x))$ ▷ None if no answer parses
end for
return $\arg\max_a \#\{n : a^{(n)} = a\}$ ▷ majority vote; ties → earliest draw

(§4.1), show why the order axis is cheap while the temperature axis is not (§4.2), turn many draws into one answer (§4.3), and reuse the draws for RL (§4.4). All proofs are in Appendix C.

4.1 The order distribution

The perturbation. The default sampler scores every masked position, sorts the positions by that score, breaks ties in favour of the leftmost position, and fills the first n_k. We keep this sorted list and write $\mathrm{rank}_i \in \{1, \ldots, |M_k|\}$ for the place of position i in it, so rank 1 is the position the sampler fills next. OSE walks down the same list, but in a slightly random order: each position receives a weight that falls exponentially with its rank, and the commit set is drawn with these weights,

$$w_i(\lambda) = \exp\big(-\mathrm{rank}_i/\lambda\big), \qquad C_k \sim \mathrm{PL}\big(w(\lambda),\, n_k\big), \tag{1}$$

where PL is the Plackett–Luce distribution, that is, sampling without replacement. We implement equation 1 with the Gumbel-top-k trick (Kool et al., 2019). We draw independent $\mathrm{Gumbel}(0,1)$ noise g_i at every masked position and commit the n_k positions with the largest $-\mathrm{rank}_i + \lambda\, g_i$. For every $\lambda > 0$ this is an exact draw from equation 1 (Appendix C.1). λ is actually the size of the noise added to the ranking. At $\lambda = 0$ there is no noise and OSE reproduces the default sampler token for token; as $\lambda \to \infty$ the order becomes uniformly random. λ is the only hyperparameter of OSE.

How large is the change? With one commit per step the rank of the filled position follows a geometric law with $\gamma = e^{-1/\lambda}$, so the default choice is kept with probability $1 - e^{-1/\lambda}$, the mean rank is about $\lambda + \frac{1}{2}$, and the chance of reaching past rank r is at most $e^{-r/\lambda}$ (Appendix C.2). At $\lambda = 1$, the perturbation is highly localized. The baseline choice is retained 63% of the time, roughly $1/3$ of commits advance by one or two positions, and the overall left-to-right order is strictly preserved (Spearman $\rho = 1.000$, Table 1). We perturb the *ranks* and not the *scores*. With 97% of the scores tied (§3), perturbing scores would commit the padding tail early and collapse the output (Appendix C.3).

The output distribution. Fix an order σ. Every token is drawn from the sampler's own token distribution at its position, given q and the canvas s_k at that step:

$$p_\theta(y \mid q, \sigma) = \prod_{k=1}^{T} \prod_{i \in C_k} p_\theta\big(y_i \mid s_k, q\big), \qquad p_\lambda(y \mid q) := \mathbb{E}_{\sigma \sim \pi_\lambda}\big[\, p_\theta(y \mid q, \sigma)\,\big]. \tag{2}$$

Here $p_\theta(\cdot \mid s_k, q)$ is the distribution the default sampler uses, the model's prediction after temperature 0.1 and the nucleus cut, and π_λ is the distribution over orders from equation 1; nothing is remasked, so y and σ together fix every canvas. We call p_λ the *order mixture* where each order gives one member, all members share the same token distributions, and at $\lambda = 0$ the single member is the default sampler. OSE changes only which positions are drawn, never how a token is drawn.

Remark 4.1 (interleaving and parallel commits). Order and tokens are drawn in turn (Appendix C.4). With one commit per step, Equation 2 is a chain-rule factorization; with several, the tokens in a commit set are drawn independently given s_k, the usual parallel-decoding error (Appendix C.5). The statements below assume $n_k = 1$.

4.2 Why order is free and temperature is not

Definition 4.2 (Order-Consistency). p_θ is *order-consistent* on q if one joint distribution $P(\cdot \mid q)$ over outputs satisfies $p_\theta(y \mid q, \sigma) = P(y \mid q)$ for every order σ; equivalently, if its conditionals are the conditionals of a single joint.

Proposition 4.3 (Order is Free but Temperature is Not). *If p_θ is order-consistent on q, then $p_\lambda(\cdot \mid q) = p_0(\cdot \mid q) = P(\cdot \mid q)$ for every λ, including $\lambda = \infty$. By contrast, for $p^{(\tau)}(x) \propto p(x)^{1/\tau}$ and $\tau \neq 1$, $p^{(\tau)} = p$ only if p is uniform on its support or a point mass.*

The first half states, for our sampler, that an exact denoiser samples the target under any unmasking order (Ben-Hamu et al., 2025). The *distribution* is unchanged; draws under two orders are still unique. The second half states that temperature moves every conditional that is neither flat nor certain, so the order axis is free at *every* λ and the temperature axis only at $\tau = 1$.

The published conditionals are tempered, and tempered conditionals are in general not the conditionals of any joint (Example C.4), so even an exact model is not order-consistent at $\tau = 0.1$. Different orders reach different outputs and the model is confident about each. OSE turns that disagreement into diversity, measured by the *factorization inconsistency*

$$\iota_\lambda(q) := \mathbb{E}_{\sigma \sim \pi_\lambda}\Big[D_{\mathrm{KL}}\big(p_\theta(\cdot \mid q, \sigma) \,\big\|\, p_\lambda(\cdot \mid q)\big)\Big] = I_\lambda(\sigma;\, y \mid q), \tag{3}$$

the average divergence of one member from the mixture, which is the mutual information between the order and the output. It is zero exactly when p_θ is order-consistent, and quantifying the diversity inherent to the mixture.

Proposition 4.4 (Diversity Cost Bound). *For reward with $|R| \le \|R\|_\infty$ ($\|R\|_\infty = 1$ for accuracy),*

$$\mathbb{E}_{\sigma \sim \pi_\lambda}\Big|\, \mathbb{E}_{p_\lambda}[R] - \mathbb{E}_{p_\theta(\cdot \mid q, \sigma)}[R] \,\Big| \le \|R\|_\infty \sqrt{2\, \iota_\lambda(q)}, \tag{4}$$

and $\big|\mathbb{E}_{p_\lambda}[R] - \mathbb{E}_{p_0}[R]\big| \le \|R\|_\infty \sqrt{2\, D_{\mathrm{KL}}(p_0 \| p_\lambda)}$ for the default sampler itself.

A typical order draw, and the default sampler, are therefore within $\sqrt{2\iota_\lambda}$ of the ensemble in expected reward. The quantity that creates the diversity is the quantity that bounds its cost. However, temperature comes with no such guarantee.

4.3 From many draws to one answer

When does the vote pay? Let $\bar{p}$ be the answer distribution of the N draws with answers $a_1, \ldots, a_N$, so the majority vote is its peak. Assume $e_\star$ the one-hot vector at the correct answer. The ambiguity decomposition (Krogh & Vedelsby, 1994) makes the trade exact:

$$\underbrace{\|\bar{p} - e_\star\|^2}_{\text{aggregate error}} = \underbrace{\tfrac{1}{N}\textstyle\sum_n \|e_{a_n} - e_\star\|^2}_{\bar{A},\ \text{average member error}} - \underbrace{\tfrac{1}{N}\textstyle\sum_n \|e_{a_n} - \bar{p}\|^2}_{D,\ \text{diversity}}, \tag{5}$$

with $\bar{A} = 2\,(1 - \text{single-draw accuracy})$ the average member error and $D = 1 - \sum_a \bar{p}(a)^2$ the Gini–Simpson diversity. For a perturbation parameter ρ (λ or τ), the aggregate error therefore falls with ρ exactly when $\mathrm{d}D/\mathrm{d}\rho > \mathrm{d}\bar{A}/\mathrm{d}\rho$, and we call $\eta(\rho) := (\mathrm{d}D/\mathrm{d}\rho)/(\mathrm{d}\bar{A}/\mathrm{d}\rho)$ the *exploration efficiency* of a perturbation family (Appendix C.8). For an exact model $\mathrm{d}\bar{A}/\mathrm{d}\lambda = 0$ by Proposition 4.3, so order perturbation pays as soon as D grows; for a real model Proposition 4.4 bounds $\mathrm{d}\bar{A}/\mathrm{d}\lambda$. Temperature instead moves every conditional, so $\bar{A}$ moves with it, and on Dream it rises steeply (§3). We therefore predict $\eta_{\text{order}} \gg \eta_{\text{temperature}}$ near the published setting (§5.2).

4.4 Training: order-diverse rollouts for GRPO

GRPO (Shao et al., 2024) normalizes rewards within a group of G samples per prompt, and a group whose rewards are all equal contributes zero gradient. Under a nearly deterministic sampler most groups are all-equal, which is why 54–90% of steps on `Dream` carry no gradient (Figure 11).

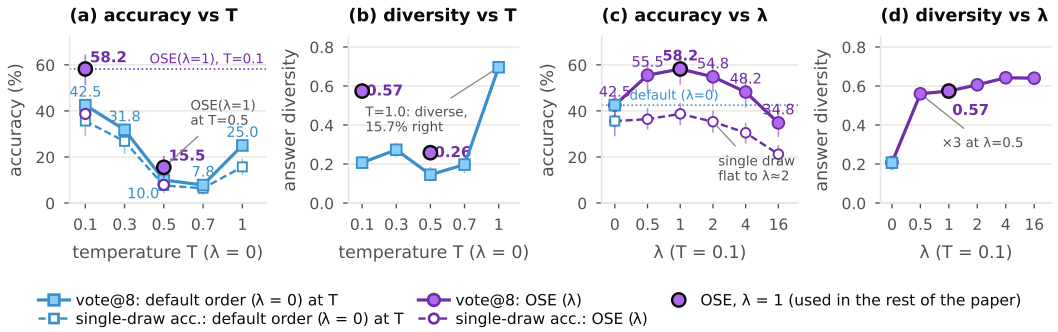

Figure 2: `Dream` on MATH500 over temperature (**a-b**, $\lambda = 0$) and λ (**c-d**, $T = 0.1$): vote@8 (filled), single-draw accuracy (dashed), and answer diversity (200 prompts, 8 draws, 3 seeds, 95% CIs). LLaDA and other tasks are in Appendix J.

Proposition 4.5. *Let* $J_\lambda(\theta) := \mathbb{E}_{\sigma\sim\pi_\lambda}\mathbb{E}_{y\sim p_\theta(\cdot|q,\sigma)}[R(y)]$*, with the order rule treated as part of the environment (as every dLLM policy-gradient method treats its confidence ranking). Then*

$$\nabla_\theta J_\lambda = \mathbb{E}_{\sigma\sim\pi_\lambda}\mathbb{E}_{y\sim p_\theta(\cdot|q,\sigma)}\Big[R(y)\sum_i \nabla_\theta \log p_\theta\big(y_i \mid s_{b(i)}, q\big)\Big], \tag{6}$$

where $b(i)$ *denotes the step committing position* i*. This recovers the standard on-policy estimator without importance weights. If* p_θ *is order-consistent,* $J_\lambda = J_0$ *for all* λ*.*

OSE rollouts look off-policy, because the policy never chooses the order, but they are not off-policy in the sense that matters for the estimator: the order is drawn by the environment rather than by p_θ, and at every step the token is still drawn from p_θ itself. What λ changes is the objective, J_λ in place of J_0, and for an order-consistent model the two coincide. Temperature has no such property, since it changes the action distribution (Propositions 4.3–4.4, Appendix L, §5.5,).

What OSE is, end to end. *Order-Ensemble Decoding*: draw N completions with Algorithm 1 at $\lambda = 1$, canonicalise, and return the majority vote, or the weighted vote of equation 17 at small N. *Order-Diverse GRPO*: the same sampler for rollouts, uniform credit, no importance weights, deployed at $\lambda = 0$ or with the ensemble. Neither needs a verifier, extra training or a temperature increase, and N draws cost exactly N default-sampler draws.

5 Experiments

5.1 Setup

We use two models with their own default samplers: `Dream-v0-Instruct-7B` (entropy order, $T = 0.1$, top-p 0.9, top-k 50, $T = L = 256$) and `LLaDA-8B-Instruct` under the d1 protocol (greedy, blocks of 32, $L/2$ steps), which our sampler reproduces token for token before the perturbation. The tasks are GSM8K, MATH500, ARC-Challenge, RACE, MMLU and Countdown. We compare a single draw, self-consistency (majority vote over N draws of the default sampler), HEX (Lee et al., 2025, block size cycled over $\{8, 16, 32, 64, 128\}$, at its own $T = 0.9$ and at ours), and OSE at $\lambda = 1$ with a majority vote over canonicalised answers. All results use three seeds; compared methods share the random numbers used for token draws, and every confidence interval is a 95% paired bootstrap over prompts. From the same set of draws we report *single-draw accuracy* (the member error A of Proposition C.5, not a vote), *answer diversity* (the Gini–Simpson index over the N answers), *parse rate*, *vote@*N and pass@N. Other details are in Appendix D.

5.2 How base models behave across temperature and λ

Question. What happens to accuracy and answer diversity when the temperature or λ is raised?

Result. The two knobs behave in opposite ways (Figure 2). Raising T costs accuracy before it buys diversity. The single draw falls from 35.6% to 7.6% by $T = 0.5$, because the padding tail

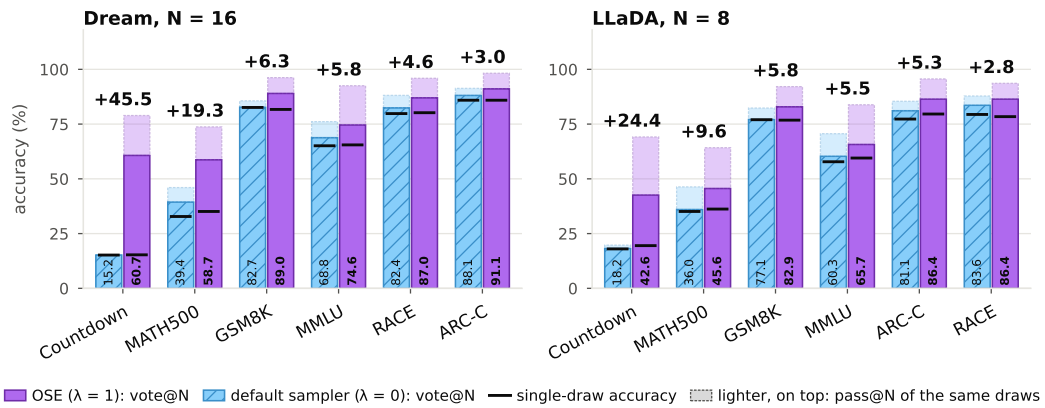

Figure 3: OSE ($\lambda = 1$) outperforms baselines ($\lambda = 0$; $T = 0.1$ for Dream, $T = 0$ for LLaDA) on full test sets (3 seeds). Solid bars denote vote@N (values inset), black dashes mark single-draw accuracy, and lighter top extensions show pass@N. Paired gains appear above each pair (Table 12).

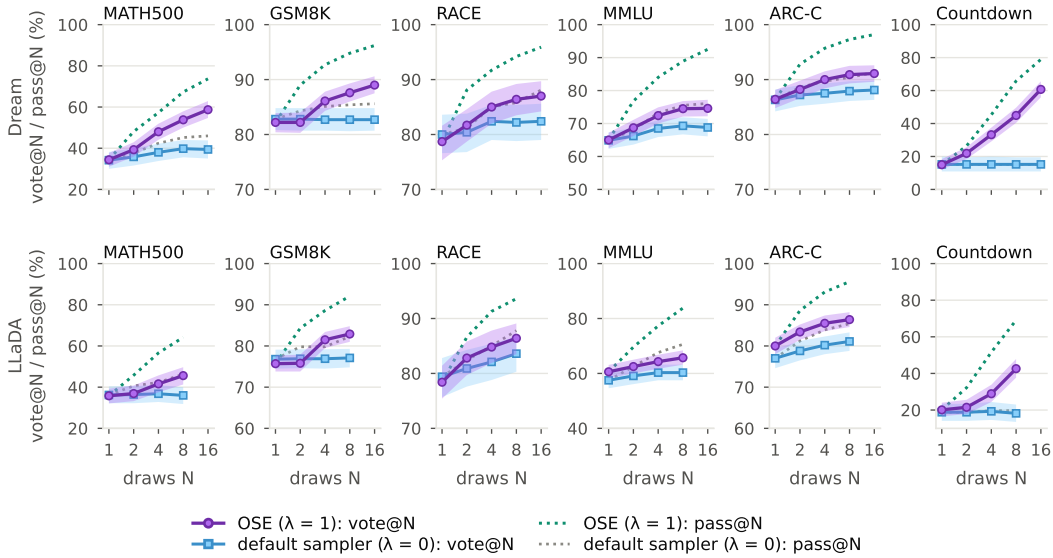

Figure 4: Scaling curves for vote@N (solid, 95% bands) and pass@N (dotted) on test sets. It compares OSE ($\lambda = 1$) against baselines ($\lambda = 0$; $T = 0.1$ for Dream, $T = 0$ for LLaDA) across Dream (top) and LLaDA (bottom).

is committed before the reasoning (§3), and the only temperature with real diversity, $T = 1.0$, is right 15.7% of the time. No T makes the default sampler's vote better than at $T = 0.1$ on all the tasks (Table 14). Raising λ buys diversity; it triples at $\lambda = 0.5$ with the single draw unchanged, vote@8 peaks at $\lambda = 1$ (58.2 against 42.5 for the default sampler), and the single draw stays flat until $\lambda \approx 2$. The same shape holds on every Dream task. OSE costs accuracy only from $\lambda = 2$–4 on the generation tasks and never up to $\lambda = 16$ on multiple choice, and on LLaDA it is free as well, although there temperature is usable too (Appendix J). We also studied whether order diversity still help once the temperature is raised. We found the two knobs are substitutes, not complements. Once T is raised, OSE adds a fraction of what it adds at the default T, so they should not be stacked (Table 16).

Thus the prediction of §4.3 holds: on Dream, only OSE(λ) is in the $\eta = \infty$ regime up to $\lambda \approx 1$–2.

5.3 OSE VERSUS THE DEFAULT SAMPLER

Question. Fixing the number of draws and the vote, does OSE beat reseeding the $\lambda = 0$ sampler?

Result. OSE wins on all the tasks for both Dream and LLaDA (Figure 3). The gain grows with N without saturating at $N = 16$ (Figure 4). At $N = 1$ nothing is lost; the single-draw differences run

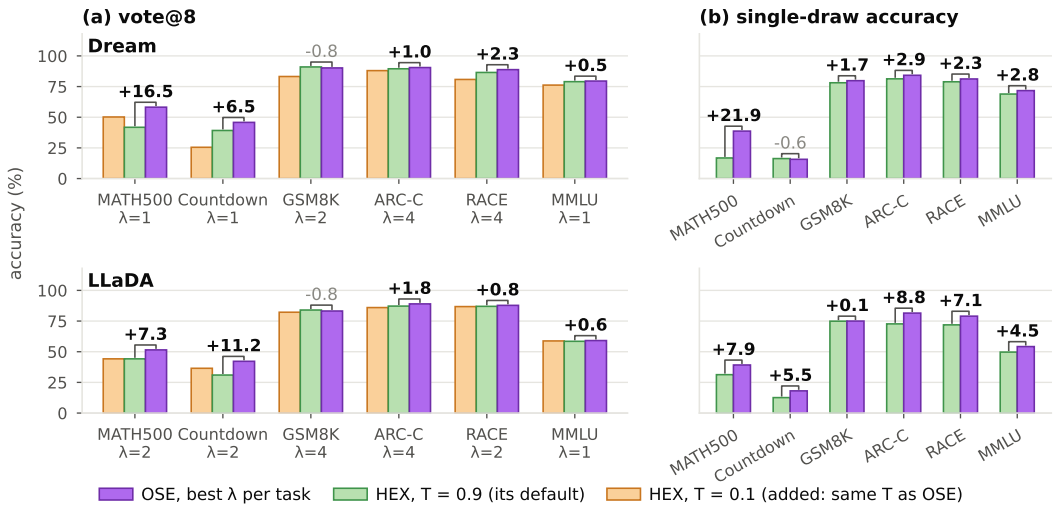

Figure 5: OSE at its best λ (under each task) against HEX at its default $T = 0.9$ and at $T = 0.1$, an arm we add so that HEX and OSE share the token sampler: **(a)** vote@8 and **(b)** single-draw accuracy on 200 prompts with three seeds. Numbers: OSE minus HEX($T = 0.9$); all results are in Table 15.

from -1.0 to $+2.3$ (Table 12). The pass@N ticks show where the gain comes from: the OSE pool holds the right answer far more often (MATH500 73.7 vs 46.0), and the vote takes most of that room. To determine the optimal aggregation strategy for N draws pool, we compared majority voting against the model's sequence likelihoods. Voting remains definitively superior as we found selecting or weighting candidates by likelihood underperforms the majority vote by 2–11 points (Appendix G). On Countdown the $\lambda = 0$ sampler returns one answer at every N, so its vote cannot move at all. Part of OSE's gain here stems from the vote's parsability filter (Appendix I).

Sampling the order beats reseeding the token sampler on every task with a checkable answer, on both models, at equal cost per draw.

5.4 Comparison with HEX

Question. HEX (Lee et al., 2025) resembles OSE in that it uses a vote over draws decoded with different block sizes. How does OSE compare and why may it be preferable?

Result. We compared HEX with $T = 0.9$ as its own setting and $T = 0.1$ to be matched with our temperature which gives HEX the same token sampler as OSE. So that any difference left between the two is due to the order axis alone (block cycling against the Plackett–Luce perturbation) and not to the temperature. The results show OSE never loses to HEX (Figure 5), with strong gains when generation is long or the answer format is brittle, MATH500 and Countdown on both models, by 6.5 to 16.5 points, and ties HEX on the other tasks, where every difference is within 2.3 points. Two facts explain this. HEX's diversity comes from its temperature, not from its block schedule, at $T = 0.1$ it is never ahead of OSE (Figure 5a). This makes HEX less accurate than the OSE (Figure 5b). On short tasks the vote closes the gap, not on MATH500 and Countdown because the right answer has left the pool. The same occurs at $N = 16$ (Figure 10).

Why OSE rather than HEX. The ties are not a draw between two equal methods. *(i) Cost.* An OSE draw costs what a default draw costs; a HEX draw is 3–3.5× slower on Dream in our implementation of its block decoding (Appendix M). At equal wall-clock, OSE gets three draws for each HEX draw. *(ii) Members.* OSE members are as good as the default sampler's, HEX members are worse (by 22 points on Dream MATH500, panel b), so OSE works at $N = 1$, its pass@N is higher, and its vote keeps rising with N; HEX needs the vote to get back to where it started. *(iii) One knob, no temperature.* HEX depends on $T = 0.9$, the knob that §5.2 shows breaks Dream; OSE has one parameter, keeps the default temperature, and $\lambda = 1$ works on every task. *(iv) Where it matters.* The wins are on the tasks with headroom (long generation); the ties are on tasks whose vote@8 is already 87–91% for both methods. *(v) The same draws train the model.* OSE rollouts feed GRPO without importance weights (§5.5); a block-size ensemble has no such use.

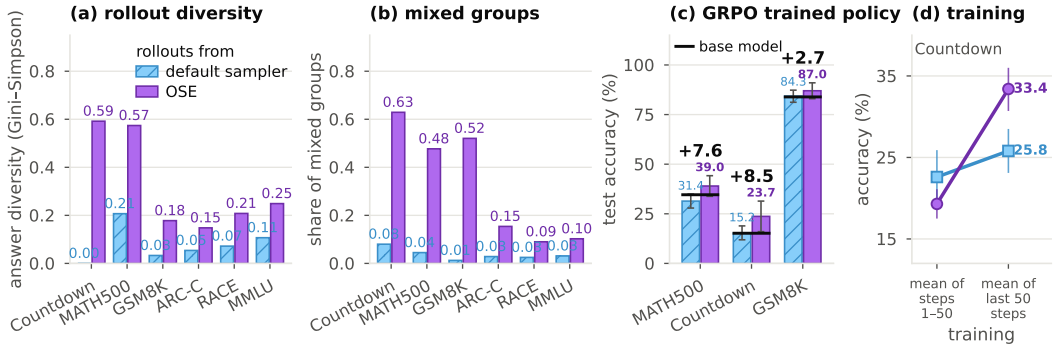

Figure 6: GRPO on `Dream` with default-sampler ($\lambda = 0$) vs. OSE($\lambda = 1$) rollouts, $T = 0.1$. **(a)** 8-rollout group answer diversity. **(b)** Share of mixed groups (not all rewards equal; the only ones with a gradient). **(c)** Test accuracy of the trained policies, decoded with the default sampler; dash is base model; numbers show the paired gain. **(d)** COUNTDOWN: accuracy of the training rollouts, averaged over the first and over the last 50 steps. Means over seeds, 95% CIs.

Therefore one per-task constant λ can beat the block-schedule ensemble at lower cost per draw. On `LLaDA`, whose sampler tolerates temperature, a tuned temperature is also competitive (Appendix K).

5.5 OSE ROLLOUTS REVIVE GRPO

Question. GRPO learns only from groups whose rollouts disagree on the reward, which is not the case at `Dream`'s default where most groups agree. Do OSE rollouts fix this? At what cost?

Result. Figure 6 shows OSE can fix do so at no cost in reward. With default-sampling, a group is nearly always unanimous, with answer diversity 0.00–0.21 but only 1–9% of groups mixed, thus very few steps carry gradient (Figure 11). OSE rollouts raise the mixed share to up to 36–64% on the generation tasks, removing this ineffectiveness. Higher rollout temperatures meanwhile mix groups but harm rollouts; at $T = 0.7$, 84% of groups are mixed, but accuracy has collapsed.

Our first note here is that extra disagreement is not harmful. A mixed group contains, by definition, at least one rollout with positive reward, so every group that OSE revives is a group with a correct rollout in it. OSE changes where the correct rollouts sit, not how many there are; OSE leaves the mean reward per rollout, the single-draw accuracy, unchanged (Table 12) but spreads them over more prompts. The share of groups with at least one correct rollout thus rises (Table 17), with each now carrying a gradient. Next, the rollouts are on-policy for J_λ, because the order is drawn by the environment and every token by p_θ (§4.4); the only question is whether a policy trained under J_λ is better under the default decoding, J_0. Figure 6c shows the policies trained on OSE rollouts are decoded with the default sampler with noticable gain. Furthermore, with OSE, MATH500 and COUNTDOWN training rollout accuracy rises by 12–14 points, compared to at most 3 with default-sampler rollouts, with the trained policies are better on test prompts when decoded with the default sampler (Figure 6c), showing the importance of live gradients.

6 CONCLUSION

The sampler driving DREAM's reported accuracy is nearly deterministic and strictly left-to-right, rendering self-consistency ineffective and temperature destructive. Commit order is an exploration axis that dLLMs have and autoregressive models do not. It is free for an order-consistent model, and for a real model its cost is bounded by the mutual information between order and answer. OSE uses it with a one-parameter Plackett–Luce perturbation of the sampler's own ranking and a majority vote. This gives verifier-free test-time scaling that never loses to the block-schedule ensemble HEX. It beats HEX where generation is long and acts as a substitute for temperature. OSE revives dead GRPO groups with a held-out gain where the base model has room to learn. An OSE draw costs exactly what a $\lambda = 0$ draw costs, the same forward passes with a different commit set (3.1 s per rollout on Dream at batch 64; Appendix M), while a HEX draw is 3–3.5× slower.

AI USE STATEMENT

In this work, we used generative AI tools for interpret results, help for conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, provide feedback on research methodology or experiments, help in implementing the method and evaluations, and generate synthetic data sets, assist with translation, support qualitative and thematic data analysis, clean and reformat dataset are not applicable to this work. Additionally, we used generative AI tools for edit the research paper to improve readability, identify relevant literature, polish figures. We have reviewed all the AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies a decoding method and a rollout sampler for existing open-weight masked diffusion language models (Dream and LLaDA). It uses only public benchmark datasets (GSM8K, MATH500, ARC-Challenge, RACE, MMLU and Countdown), involves no human subjects and no personal data, and trains no new base model. OSE changes only the order in which a model fills its output; it does not change the model's weights at inference time or add new capabilities, so it carries the same risks and limitations as the underlying models, including their potential to produce incorrect or biased text. The method makes test-time scaling cheaper for diffusion LMs, which can lower the compute cost of reaching a given accuracy. We see no specific ethical concern beyond those that apply to language models in general.

REPRODUCIBILITY STATEMENT

To ensure reproducibility, the Appendix provides comprehensive proofs and detailed empirical results. Furthermore, we will open-source our codebase to facilitate future research.

REFERENCES

Hikaru Asano, Tadashi Kozuno, Kuniaki Saito, and Yukino Baba. Where-to-unmask: Ground-truth-guided unmasking order learning for masked diffusion language models. *CoRR*, abs/2602.09501, 2026. doi: 10.48550/ARXIV.2602.09501. URL https://doi.org/10.48550/arXiv.2602.09501.

Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas Nolte, and Brian Karrer. Accelerated sampling from masked diffusion models via entropy bounded unmasking. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/510e22f4a2da5212a64f1591736e2eaf-Abstract-Conference.html.

Daehoon Gwak, Minseo Jung, Junwoo Park, Minho Park, ChaeHun Park, Junha Hyung, and Jaegul Choo. Reward-weighted sampling: Enhancing non-autoregressive characteristics in masked diffusion llms. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 34574–34594. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.EMNLP-MAIN.1754. URL https://doi.org/10.18653/v1/2025.emnlp-main.1754.

Zhengfu He, Tianxiang Sun, Qiong Tang, Kuanning Wang, Xuanjing Huang, and Xipeng Qiu. Diffusionbert: Improving generative masked language models with diffusion models. In Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023,*

Toronto, Canada, July 9-14, 2023, pp. 4521–4534. Association for Computational Linguistics, 2023. URL https://doi.org/10.18653/v1/2023.acl-long.248.

Yuki Ichihara, Naoto Iwase, Mohammad Atif Quamar, and Junpei Komiyama. Prefix-denoising consistency: Test-time verification for diffusion language models. *CoRR*, abs/2608.25311, 2026. doi: 10.48550/ARXIV.2608.25311. URL https://doi.org/10.48550/arXiv.2608.25311.

Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham M. Kakade, and Sitan Chen. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/kim25ah.html.

Wouter Kool, Herke van Hoof, and Max Welling. Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3499–3508. PMLR, 2019. URL http://proceedings.mlr.press/v97/kool19a.html.

Anders Krogh and Jesper Vedelsby. Neural network ensembles, cross validation, and active learning. In Gerald Tesauro, David S. Touretzky, and Todd K. Leen (eds.), *Advances in Neural Information Processing Systems 7, [NIPS Conference, Denver, Colorado, USA, 1994]*, pp. 231–238. MIT Press, 1994. URL https://proceedings.neurips.cc/paper_files/paper/1994/hash/b8c37e33defde51cf91e1e03e51657da-Abstract.html.

Jihoon Lee, Hoyeon Moon, Kevin Zhai, Arun Kumar Chithanar, Anit Kumar Sahu, Soummya Kar, Chul Lee, Souradip Chakraborty, and Amrit Singh Bedi. Test-time scaling in diffusion llms via hidden semi-autoregressive experts, 2025. URL https://doi.org/10.48550/arXiv.2510.05040.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B. Hashimoto. Diffusion-lm improves controllable text generation. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL https://openreview.net/forum?id=3s9IrEsjLyk.

Enshu Liu, Xuefei Ning, Yu Wang, and Zinan Lin. NI sampling: Accelerating discrete diffusion sampling by token order optimization. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=rrD1U0Izt5.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *CoRR*, abs/2502.09992, 2025a. doi: 10.48550/ARXIV.2502.09992. URL https://doi.org/10.48550/arXiv.2502.09992.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025b. URL https://openreview.net/forum?id=KnqiC0znVF.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=sMyXP8Tanm.

Zhangzhi Peng, Zachary Bezemek, Sawan Patel, Jarrid Rector-Brooks, Sherwood Yao, Alexander Tong, and Pranam Chatterjee. Path planning for masked diffusion model sampling. *CoRR*,

abs/2502.03540, 2025. doi: 10.48550/ARXIV.2502.03540. URL https://doi.org/10.48550/arXiv.2502.03540.

Subham S. Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL https://openreview.net/forum?id=L4uaAR4ArM.

Subham Sekhar Sahoo, Zhihan Yang, Yash Akhauri, Johnna Liu, Deepansha Singh, Zhoujun Cheng, Zhengzhong Liu, Eric P. Xing, John Thickstun, and Arash Vahdat. Esoteric language models: A family of any-order diffusion LLMs. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=4C5lHAIWiF.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL http://arxiv.org/abs/2402.03300.

Yangyi Shen, Tianjian Feng, Jiaqi Han, Wen Wang, Tianlang Chen, Chunhua Shen, Jure Leskovec, and Stefano Ermon. Improving diffusion language model decoding through joint search in generation order and token space, 2026. URL https://doi.org/10.48550/arXiv.2601.20339.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and generalized masked diffusion for discrete data. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL https://openreview.net/forum?id=xcqSOfHt4g.

Xiaohang Tang, Rares Dolga, Sangwoong Yoon, and Ilija Bogunovic. wd1: Weighted policy optimization for reasoning in diffusion language models. *CoRR*, abs/2507.08838, 2025. doi: 10.48550/ARXIV.2507.08838. URL https://doi.org/10.48550/arXiv.2507.08838.

Wen Wang, Bozhen Fang, Chenchen Jing, Yongliang Shen, Yangyi Shen, Qiuyu Wang, Hao Ouyang, Hao Chen, and Chunhua Shen. Time is a feature: Exploiting temporal dynamics in diffusion language models. *CoRR*, abs/2508.09138, 2025a. doi: 10.48550/ARXIV.2508.09138. URL https://doi.org/10.48550/arXiv.2508.09138.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Yinjie Wang, Ling Yang, Bowen Li, Ye Tian, Ke Shen, and Mengdi Wang. Revolutionizing reinforcement learning framework for diffusion large language models. *CoRR*, abs/2509.06949, 2025b. doi: 10.48550/ARXIV.2509.06949. URL https://doi.org/10.48550/arXiv.2509.06949.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models. *CoRR*, abs/2508.15487, 2025a. doi: 10.48550/ARXIV.2508.15487. URL https://doi.org/10.48550/arXiv.2508.15487.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models, 2025b. URL http://arxiv.org/abs/2508.15487.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao

Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: an open-source LLM reinforcement learning system at scale. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/a4277440d50f1f15d2cb4c14f7e0c0d2-Abstract-Conference.html.

Siyan Zhao, Devaansh Gupta, Qinqing Zheng, and Aditya Grover. d1: Scaling reasoning in diffusion large language models via reinforcement learning. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/52190a0362148d179f1cbd9080956872-Abstract-Conference.html.

Yuyan Zhou, Kai Syun Hou, Weiyu Chen, and James Kwok. Attention-based sampler for diffusion language models, 2026. URL https://doi.org/10.48550/arXiv.2604.08564.

A Related Work

Samplers for masked diffusion LMs. Masked diffusion language models (Shi et al., 2024; Sahoo et al., 2024; Ou et al., 2025; Nie et al., 2025b; Ye et al., 2025b) fill a masked canvas over many steps, and the sampler decides which positions to fill next. Kim et al. (2025) show that a confidence-based order avoids hard infilling subproblems and beats a random order; we compare against the confidence sampler people actually run and find that it is left-to-right. Ben-Hamu et al. (2025) note that an exact denoiser gives the same distribution under any order (our Proposition 4.3) and use entropy bounds to unmask faster. Other work picks or learns a single better order, for accuracy or for speed: reward-weighted sampling (Gwak et al., 2025), path planning (Peng et al., 2025), NI sampling (Liu et al., 2026), attention-based samplers (Zhou et al., 2026), learned unmasking orders (Asano et al., 2026), and any-order training (Sahoo et al., 2026). All of these choose *one* order. OSE keeps the sampler's own order as the centre and samples a distribution around it, so that N draws differ.

Test-time scaling for dLLMs. Self-consistency (Wang et al., 2023) votes over samples that differ because of token temperature. For dLLMs, HEX (Lee et al., 2025) votes over draws decoded with different block sizes at $T = 0.9$; prefix-denoising consistency (Ichihara et al., 2026) regenerates from a kept prefix and votes; temporal self-consistency (Wang et al., 2025a) votes over the intermediate steps of one trajectory; and OTS (Shen et al., 2026) runs a beam search over order and tokens with likelihood selection. None of these analyses why the default sampler is deterministic, or separates the order axis from the temperature axis. OSE does both: its draws differ only in order, its members are as accurate as the default sampler, and it compares against HEX at both temperatures with pass@N (§5.4). Its likelihood results (Appendix G) agree with OTS, where likelihood selection does not beat a small vote.

Reinforcement learning for dLLMs. d1 (Zhao et al., 2025), wd1 (Tang et al., 2025) and TraceRL (Wang et al., 2025b) train dLLMs with group-based objectives such as GRPO (Shao et al., 2024). For autoregressive models, groups with equal rewards are handled by dynamic sampling, which drops them and samples more prompts (Yu et al., 2025). OSE instead makes the groups mixed at the source, at the default rollout temperature, and Proposition 4.5 shows why no importance correction is needed. To our knowledge, no prior work samples commit orders at low token temperature for ensembling, or uses order diversity for RL exploration.

B Diagnosis details (Dream, default sampler)

B.1 The ties

Table 2: Masked positions tied at the top confidence score, GSM8K, 256-token canvas (mean over prompts). Under top-p 0.9 at temperature 0.1 the nucleus usually holds one token, so the entropy is exactly 0.

step (of 256)	0	25	50	100	150	200	250
masked positions	256	232	207	157	107	57	7
tied at the top score	248.3	157.3	161.7	137.7	99.8	55.3	7.0

CUDA `torch.topk` returns the lowest index among ties. Table 3 gives the realised-order statistics for the default sampler, for OSE at three values of λ and for uniform random order; the "content" statistics that exclude the end-of-text tail are 97.2% leftmost / Spearman 1.000 for the default sampler, 62.1% / 0.999 at $\lambda = 1$, 23.0% / 0.982 at $\lambda = 4$, 8.9% / 0.76 at $\lambda = 16$, 6.3% / 0.27 at $\lambda = 64$ and 5.4% / 0.007 for uniform random order.

B.2 What varies between draws of the default sampler

Changing the seed barely changes the output; changing the batch a prompt is decoded in does (padding length and bf16 arithmetic flip near-ties). Consequences: (1) the default sampler's vote gains on ARC-C (86.3 $\rightarrow$ 88.1 at $N = 16$), RACE (80.0 $\rightarrow$ 82.4), MATH500 (34.2 $\rightarrow$ 39.4) and MMLU (65.5 $\rightarrow$ 70.6) come entirely from batch noise, and it gains nothing on GSM8K (82.8 $\rightarrow$ 82.7) or Countdown (15.2); (2) the $\lambda = 0$ seeds are nearly the same data, so its CIs overstate independence,

Table 3: Realised order and single-draw accuracy on GSM8K and Dream as λ grows: the graded version of Table 1.

order rule	leftmost commits	Spearman(step, position)	accuracy
entropy (Dream default; $\lambda = 0$)	98.1%	1.000	82.7
OSE, $\lambda = 1$	62.5%	1.000	81.7
OSE, $\lambda = 4$	22.7%	0.995	73.9
OSE, $\lambda = 16$	7.4%	0.93	55.1
uniform random ($\lambda \to \infty$)	2.6%	0.00	38.5

Table 4: Share of prompts with identical output.

task	seed 0 vs seed 1, same (prompt, draw)	within seed 0, draw 0 vs draw 1
GSM8K	90.8%	62.3%
MATH500	98.2%	66.0%
Countdown	97.2%	96.5%
ARC-C	99.7%	53.5%
RACE	97.1%	78.6%

while OSE runs are genuinely seeded; (3) a $\lambda = 0$ run that decoded every prompt in the same batch would be exactly deterministic; the measured sampler is what a practitioner gets, and we compare against it.

B.3 Temperature

Table 5: Dream at raised T ($\lambda = 0$), 200 prompts, 8 draws, three seeds. It shows parse rate, content tokens per draw (ARC-C) and vote@8. Single-draw accuracy and answer diversity of the same methods are in Table 14.

T	GSM8K parse	MATH500 parse	ARC-C content tokens	GSM8K vote@8	MATH500 vote@8
0.1 (default)	100%	77%	223	80.5	42.5
0.3	98%	67%	177	76.5	31.8
0.5	75%	29%	39	38.8	10.0
0.7	45%	30%	9	24.8	7.8
1.0	81%	76%	27	55.8	25.0
OSE $\lambda = 1$, $T = 0.1$	100%	80%	217	**87.8**	**58.2**

Temperature 0.5–0.7 is the worst region; the partial recovery at $T = 1.0$ is real in the data and not explained. Block decoding partly protects short-answer tasks. With HEX's block schedule at $T = 0.9$ the single draw is 76.8% on GSM8K (23.7% unblocked at $T = 0.7$) but 14.5% on MATH500.

B.4 The multiple-choice anomaly

On ARC-C, RACE and MMLU, Dream at $T = 1.0$ emits a short answer (27 content tokens vs 223 at $T = 0.1$) that still contains the option letter, and its single-draw accuracy is *above* the $T = 0.1$ setting (ARC-C 85.8, RACE 87.6, MMLU 69.1 on 200 prompts) with low diversity (0.120, 0.072, 0.237).

B.5 Confidence rules without tempering

Kim et al. (2025) propose top-probability and top-margin ordering on the untempered distribution. On Dream these rules are strongly anti-left-to-right (Spearman -0.93). The highest-probability positions are the end-of-text tail, which is committed first, and 61–68% of outputs are empty (accuracy 23.9 / 25.0 on GSM8K). Dream's own order-softening knob, `alg_temp`, samples uniformly among tied positions, which again includes the tail. HEX reports the same tail-first collapse for LLaDA under top-k margin (55.5% of runs). Any order perturbation that lets the most confident positions win first

commits the length before the content. The rank-based perturbation equation 1 avoids this because the tail's ranks are large.

B.6 THE TAG-PROMPT STRESS CASE

Under a prompt that requires `<reasoning>` and `<answer>` tags, Dream writes the opening tag and then commits the entire padding tail at once, giving an empty output on 41.8% of prompts and 27.7% accuracy against 82.7% under the default sampler. A perfectly stable, confidently committed length decision destroys 55 points. We use this regime only as a stress test.

B.7 LLADA

Our LLaDA sampler reproduces d1's `generate` token for token on 8 GSM8K prompts before the order perturbation and common-random-number token draws are added (4 CPU tests and a GPU parity job). LLaDA's default sampler attends to left padding, so its output depends on batch composition: draws 0 and 1 differ on 655 of 1,319 GSM8K prompts, while draw 0 of two different seeds is identical on all 1,319 with the same batches. Its self-consistency is therefore measured from real draws (Table 13). With block decoding LLaDA tolerates temperature. On GSM8K, $T = 0.9$ with block 32, it gives single draw 75.4 (vs 76.8 at $T = 0$) and vote 82.5 / 83.6 / 85.0 at $N = 4/8/16$ (pass@16 94.4).

C PROOFS

C.1 GUMBEL-TOP-k AND THE NOISE FORM

Lemma C.1. *(Kool et al., 2019) Let $a_1, \dots, a_m$ be real numbers and $g_1, \dots, g_m$ independent* $\mathrm{Gumbel}(0, 1)$ *variables. Sort the positions by $a_i + g_i$, largest first. The first n positions are a Plackett–Luce draw of size n with weights $w_i = \exp(a_i)$.*

Proof. If $g \sim \mathrm{Gumbel}(0, 1)$ then $e^{-g} \sim \mathrm{Exp}(1)$, so $E_i := e^{-(a_i+g_i)}$ is exponential with rate $w_i = e^{a_i}$, and the E_i are independent. Sorting $a_i + g_i$ largest first is the same as sorting E_i smallest first, that is, the order in which m independent exponential clocks with rates $w_1, \dots, w_m$ ring. The first clock to ring is clock i with probability $w_i / \sum_l w_l$. By the memory less property, once a clock has rung the others start afresh, so the next to ring is clock l with probability $w_l / \sum_{l' \text{ remaining}} w_{l'}$. Repeating this n times is exactly a Plackett–Luce draw of size n.

Noise form. OSE commits the n_k positions with the largest value of

$$-\mathrm{rank}_i + \lambda\, g_i, \qquad g_i \sim \mathrm{Gumbel}(0, 1). \tag{7}$$

Take $a_i = -\mathrm{rank}_i/\lambda$ with $\lambda > 0$. Multiplying every value $a_i + g_i$ by the same positive number λ does not change their order, so the top n_k of $-\mathrm{rank}_i/\lambda + g_i$ are the top n_k of $-\mathrm{rank}_i + \lambda\, g_i$. Hence equation 7 samples from equation 1. At $\lambda = 0$, equation 7 has no noise and returns ranks $1, \dots, n_k$, which is the published commit set. In code, Gumbel noise is drawn as $g = -\log(-\log u)$ with u uniform on $(0, 1)$; form equation 7 needs no division by λ and so needs no special case at $\lambda = 0$.

C.2 LAW OF THE CHOSEN RANK

Consider one draw from m masked positions and write $\gamma = e^{-1/\lambda}$, so that rank r has weight γ^r. With the geometric sum $\sum_{j=1}^{m} \gamma^j = \gamma\,(1 - \gamma^m)/(1 - \gamma)$,

$$P(\mathrm{rank} = r) = \frac{\gamma^r}{\sum_{j=1}^{m} \gamma^j} = \frac{(1-\gamma)\,\gamma^{r-1}}{1 - \gamma^m}, \qquad r = 1, \dots, m. \tag{8}$$

Four consequences:

(a) $P(\mathrm{rank} = 1) = (1 - \gamma)/(1 - \gamma^m)$, which tends to $1 - e^{-1/\lambda}$ as $m \to \infty$.

(b) $P(\mathrm{rank} > r) = (\gamma^r - \gamma^m)/(1 - \gamma^m) \le \gamma^r = e^{-r/\lambda}$.

(c) For large m the mean rank is $1/(1 - \gamma) = \lambda + \frac{1}{2} + \frac{1}{12\lambda} + O(\lambda^{-3})$.

(d) With several draws per step, the j-th draw satisfies $P(\text{rank} > r) \leq \gamma^{r+1-j}/(1-\gamma)$. The weight beyond rank r is at most $\gamma^{r+1}/(1-\gamma)$, and the total remaining weight is at least the weight of the best remaining position, whose rank is at most j.

λ	0.5	1	2	4	16
$P(\text{rank} = 1)$	86.5%	63.2%	39.3%	22.1%	6.1%
mean rank	1.16	1.58	2.54	4.52	16.5
$P(\text{rank} > 4)$	0.03%	1.8%	13.5%	36.8%	77.9%

Table 6: Consequences of equation 8 for one draw per step and large m.

C.3 Predicted leftmost-commit share

Assume (i) one commit per step, so that the number of masked positions runs through $L, L-1, \ldots, 1$, and (ii) rank 1 is the leftmost masked position. By equation 8(a), the expected share of commits that land on the leftmost masked position is

$$F(\lambda, L) = \frac{1}{L} \sum_{m=1}^{L} \frac{1-\gamma}{1-\gamma^m}. \tag{9}$$

$F \to 1$ as $\lambda \to 0$, and $F \to H_L/L$ as $\lambda \to \infty$, where $H_L = 1 + \frac{1}{2} + \cdots + \frac{1}{L}$; the second limit is the value for a uniformly random order. For $L = 256$:

	$\lambda = 1$	$\lambda = 4$	$\lambda = 16$	$\lambda = 64$	uniform
predicted, $F(\lambda, 256)$	63.4%	22.8%	7.3%	3.4%	2.4%
measured (Table 1, "all")	62.5%	22.7%	7.4%	3.5%	2.6%

Table 7: Leftmost-commit share on GSM8K. It shows closed form equation 9 against the measurement of 1.

Assumption (ii) holds in 98.1% of steps under the default sampler (Table 1). At small λ this scales the prediction ($0.981 \times 63.4 = 62.2$ at $\lambda = 1$); at large λ the draw is nearly uniform and the assumption no longer matters.

Why ranks and not scores. The usual way to soften a confidence sampler is to sample positions from a softmax over the scores; Dream's `alg_temp` is a knob of this kind. Under the published settings it fails because of the ties of §3. At step 0, 248 of the 256 positions have exactly the same score, so a softmax over scores gives all of them the same probability at every softmax temperature above zero, and this includes the padding tail. The tail is then filled early, the output length is fixed before the reasoning is written, and the output collapses (§3). There is also no setting in between: the sampler jumps from "leftmost first" at zero to "uniform over the tied positions" at any positive value. Ranks avoid both problems. Ties are broken before the perturbation, so every position has its own place in the list, and its weight depends only on that place. At $\lambda = 1$ a padding position more than 100 places down the list is reached with probability at most e^{-100} per step; it is filled after the content to its left, as in the published sampler.

C.4 The exact sampling process

The ranking at step k is computed from the canvas s_k, so the sampler cannot draw the whole order first. It draws C_1, then the tokens in C_1, then C_2, and so on. Below is the exact output distribution of this process. It equals the mixture equation 2 whenever the ranking depends only on which positions are masked, and not on the tokens already revealed. For Dream under the default settings this is nearly true, because the ranking is decided almost entirely by position (§3), and we use form equation 2 in the analysis.

The sampler alternates two moves. It draws $C_k \sim \pi_\lambda(\cdot \mid s_k, q)$, the Plackett–Luce draw on the ranking computed from s_k; then it draws $y_i \sim p_\theta(\cdot \mid s_k, q)$ for every $i \in C_k$. The joint probability of

an output and an order is

$$P_\lambda(y, \sigma \mid q) = \prod_{k=1}^{T} \Big[\pi_\lambda(C_k \mid s_k, q) \prod_{i \in C_k} p_\theta(y_i \mid s_k, q) \Big], \tag{10}$$

and the output distribution is $p_\lambda(y \mid q) = \sum_\sigma P_\lambda(y, \sigma \mid q)$.

Lemma C.2. *If $\pi_\lambda(C_k \mid s_k, q)$ depends on s_k only through the set of masked positions, then p_λ equals the mixture equation 2.*

Proof. Under the assumption, the masked set at step k is determined by $C_1, \ldots, C_{k-1}$, so the first factors in equation 10 multiply to a probability $\pi_\lambda(\sigma)$ that does not involve y. The second factors multiply to $p_\theta(y \mid q, \sigma)$ by equation 2. Summing over σ gives $\sum_\sigma \pi_\lambda(\sigma)\, p_\theta(y \mid q, \sigma)$, which is equation 2.

C.5 One commit per step, and what changes with parallel commits

Lemma C.3. *Assume $n_k = 1$ at every step. Assume the token distributions are the conditionals of one joint distribution $P(\cdot \mid q)$: $p_\theta(y_i \mid s, q) = P(y_i \mid y_U, q)$ for every canvas s with revealed set U and revealed tokens y_U. Assume also that the rule that picks the next position may use the canvas and its own random noise, but not the token about to be drawn (this holds for the entropy ranking and for equation 7). Then the output of the sampler has distribution $P(\cdot \mid q)$. This covers every fixed order, the default sampler, and OSE at every λ, including the step-by-step process of Appendix C.4.*

Proof. Fix y. A run of the sampler is a path $(i_1, \ldots, i_L)$ of filled positions. The probability of this path together with output y is

$$\prod_{k=1}^{L} \pi(i_k \mid s_k)\; P\big(y_{i_k} \mid y_{U_k}, q\big),$$

where $U_k = \{i_1, \ldots, i_{k-1}\}$. By the chain rule the P factors multiply to $P(y \mid q)$ for every path, so the probability is $P(y \mid q) \prod_k \pi(i_k \mid s_k)$. For fixed y the factors $\pi(i_k \mid s_k)$ define a probability distribution over paths, because each step picks the next position given a canvas that y and the earlier picks determine. So they sum to 1 over all paths, and the probability of output y is $P(y \mid q)$.

C.6 Tempered conditionals are not order-consistent

Example C.4. Let P on $(y_1, y_2) \in \{0,1\}^2$ have $P(0,0) = 0.35$, $P(0,1) = 0.25$, $P(1,0) = 0$, $P(1,1) = 0.40$, and decode greedily ($\tau \to 0$) from the exact conditionals. Order $(1,2)$: $P(y_1 = 0) = 0.60$, so $y_1 = 0$; $P(y_2 = 0 \mid y_1 = 0) = 0.58$, so $y_2 = 0$. Order $(2,1)$: $P(y_2 = 1) = 0.65$, so $y_2 = 1$; $P(y_1 = 1 \mid y_2 = 1) = 0.62$, so $y_1 = 1$. The two orders return $(0,0)$ and $(1,1)$ with certainty; at $\tau = 0.1$ they return them with probability 0.95 and 0.99. So the tempered conditionals of an exact model are not the conditionals of any joint: $\iota_\lambda > 0$ for every $\lambda > 0$, and the mixture over orders is more diverse than any one member.

C.7 Proofs for §4.2

Parallel commits affect only MATH500 among our Dream runs ($n_k = 2$), where §5.3 measures no single-draw cost at $\lambda = 1$; every other run uses $n_k = 1$.

Proof of Proposition 4.3. *Order.* Every member of the mixture equation 2 equals P, so their average does too. Lemma C.3 extends this to the interleaved sampler of Remark 4.1, in which the order and the tokens are drawn in turn. *Temperature.* $p^{(\tau)}(x) = p(x)$ on the support means that $p(x)^{1/\tau - 1}$ is constant there, so p is constant on its support.

Proof of Proposition 4.4. For bounded R, $|\mathbb{E}_p R - \mathbb{E}_{p'} R| \le \|R\|_\infty \sqrt{2 D_{\mathrm{KL}}(p \| p')}$ by Pinsker's inequality. For the first bound take $p = p_\theta(\cdot \mid q, \sigma)$ and $p' = p_\lambda$, average over σ, and apply Jensen's inequality to the concave square root, which gives $\mathbb{E}_\sigma \sqrt{2\, D_{\mathrm{KL}}(\cdot)} \le \sqrt{2\, \mathbb{E}_\sigma D_{\mathrm{KL}}(\cdot)} = \sqrt{2\, \iota_\lambda(q)}$. For the second take $p = p_0$, the member at $\lambda = 0$.

The identity $\iota_\lambda(q) = I_\lambda(\sigma; y \mid q)$ in equation 3 is the standard variational form of the mutual information: for a mixture $p_\lambda = \mathbb{E}_\sigma[p_\theta(\cdot \mid q, \sigma)]$, the average D_{KL} of the components from the mixture is the information that σ carries about y.

C.8 AGGREGATION: THE AMBIGUITY DECOMPOSITION IN FULL

Let a range over the possible canonicalised answers to a prompt, let $a_\star$ be the correct one, let e_a be the one-hot vector with a 1 at answer a, and write $e_\star := e_{a_\star}$. Take N draws with answers $a_1, \dots, a_N$ and let $\bar{p} = \frac{1}{N}\sum_n e_{a_n}$ be their answer distribution, so that $\bar{p}(a)$ is the fraction of draws that answered a; the plurality vote is its peak. For squared error the ambiguity decomposition (Krogh & Vedelsby, 1994) is an identity:

$$\underbrace{\|\bar{p} - e_\star\|^2}_{\text{aggregate error}} \;=\; \underbrace{\tfrac{1}{N}\textstyle\sum_n \|e_{a_n} - e_\star\|^2}_{\bar{A},\ \text{average member error}} \;-\; \underbrace{\tfrac{1}{N}\textstyle\sum_n \|e_{a_n} - \bar{p}\|^2}_{D,\ \text{diversity}}, \tag{11}$$

with $\bar{A} = 2\,(1 - \text{single-draw accuracy})$ and $D = 1 - \sum_a \bar{p}(a)^2$, the Gini–Simpson index. To study a perturbation parameter ρ (λ for order, τ for temperature), let $\bar{p}_\rho(a)$ be the probability that one draw at setting ρ answers a, which is the $N \to \infty$ limit of $\bar{p}$; for OSE, $\bar{p}_\lambda(a) = p_\lambda\big(a(y) = a \mid q\big)$. Then $\bar{A}(\rho) = 2\big(1 - \bar{p}_\rho(a_\star)\big)$ and $D(\rho) = 1 - \sum_a \bar{p}_\rho(a)^2$, and equation 11 reads $\|\bar{p}_\rho - e_\star\|^2 = \bar{A}(\rho) - D(\rho)$.

Proposition C.5 (profitable exploration). *The aggregate error $\|\bar{p}_\rho - e_\star\|^2$ falls with ρ if and only if $\mathrm{d}D/\mathrm{d}\rho > \mathrm{d}\bar{A}/\mathrm{d}\rho$.*

Proof. Differentiate $\bar{A}(\rho) - D(\rho)$ with respect to ρ.

The ratio $\eta(\rho) = (\mathrm{d}D/\mathrm{d}\rho)/(\mathrm{d}\bar{A}/\mathrm{d}\rho)$ of §4.3 is the exploration efficiency of a perturbation family, and Proposition C.5 says that exploration pays exactly when $\eta > 1$. Two caveats are worth stating. First, $\bar{A}$ and D are defined on the answer distribution, not on the token distribution, so a perturbation that changes the text without changing the parsed answer moves neither. Second, plurality voting is not the squared-error minimiser; equation 11 describes the aggregate distribution $\bar{p}$, and the vote reads off its peak. The condition is therefore a clean guide rather than an exact criterion for the vote, and §5.2 measures both sides directly.

C.9 SELECTION: WHY FULL LIKELIHOOD FAILS, AND THE SPAN FIX

Given N candidates $y^{(1..N)} \sim p_\lambda$, the any-order log-likelihood splits into the answer span $a(y)$ and the rest $r(y)$, that is, the reasoning and the padding:

$$\mathbb{E}_\sigma\big[\log p_\theta(y \mid q, \sigma)\big] = \mathbb{E}_\sigma\big[\log p_\theta(a \mid q, r, \sigma)\big] + \mathbb{E}_\sigma\big[\log p_\theta(r \mid q, \sigma)\big]. \tag{12}$$

The second term is two orders of magnitude larger than the first and carries no information about correctness. On short-answer tasks it is the padding tail, which every order assigns probability near one and which is largest for the emptiest drafts. On Countdown it is long, predictable reasoning that never reaches an answer. Full-response likelihood therefore picked an unparsable draft 49–75% of the time, above the 53% base rate, and normalising by length did not help. The model endorses its own errors, and a score that sums this endorsement over every token is the worst rule we tested (§G).

The span score equation 17 scores only the tokens that carry the outcome, given everything else, at a cost of one masked forward pass over the answer positions per candidate. It is still the model's own infilling objective, restricted to the span where the orders disagree about the outcome. It beats voting at $N \le 8$ and loses to it at $N = 16$. The likelihood-weighted vote in equation 17 combines the two: it is the span rule when all answers differ, and it approaches plurality as N grows. Since it gains at most $+2.3$ points at $N = 4$ and nothing by $N = 16$, plurality is OSE's default and the weighted vote its small-N variant.

Canonicalisation. Answers are canonicalised before they are tallied. The numeric value for mathematics, the option letter for multiple choice, and, for Countdown, the evaluated value together with the multiset of operands used, which encodes the use-each-number constraint without reference to the target. Dividing the span score by an unconditional prior over the span (PMI-style normalisation) lowered Countdown accuracy by 9 points.

C.10 WHERE TO SPEND THE PERTURBATION: ADAPTIVE λ

A constant λ spends the same perturbation at every step, which need not be optimal: some steps carry a real choice between positions, and others do not. Let $D_k(\lambda_k)$ be the diversity gained and $C_k(\lambda_k)$

the quality cost at step k. Maximising $\sum_k D_k$ under a budget $\sum_k C_k \le B$ gives the water-filling condition $\mathrm{d}D_k/\mathrm{d}\lambda_k = \mu\,\mathrm{d}C_k/\mathrm{d}\lambda_k$ at every step with $\lambda_k > 0$: spend the perturbation where diversity is cheap. A closed-form instantiation reads both quantities off the logits,

$$\lambda_k = \mathrm{clip}\left(c \cdot \frac{\mathrm{spread}_k}{\mathrm{gap}_k + 0.05},\ 0,\ 16\right), \tag{13}$$

where spread_k measures how much the position scores differ at step k and gap_k is the distance between the top two. So λ_k is large when many positions are nearly equally good, and small when one position clearly leads. Setting $c = 0$ reproduces the default sampler exactly, which makes equation 13 a strict generalisation of the baseline. In our runs a per-task constant λ matched or beat this rule (§H), so OSE keeps a single hyperparameter, and $\lambda = 1$, fixed before any result was seen, is what every headline number uses.

C.11 Order-diverse GRPO

Written out, the gradient of Proposition 4.5 is

$$\nabla_\theta J_\lambda = \mathbb{E}_{\sigma\sim\pi_\lambda}\mathbb{E}_{y\sim p_\theta(\cdot|q,\sigma)}\Big[R(y)\sum_i \nabla_\theta \log p_\theta\big(y_i \mid s_{b(i)}, q\big)\Big], \tag{14}$$

with $b(i)$ the step at which position i was committed.

Proof of Proposition 4.5. Treat the order rule as part of the environment, so that $\sigma \sim \pi_\lambda$ does not depend on θ. Then $J_\lambda(\theta) = \mathbb{E}_{\sigma\sim\pi_\lambda}\sum_y p_\theta(y \mid q,\sigma)R(y)$, and differentiating under the expectation over σ gives $\nabla_\theta J_\lambda = \mathbb{E}_\sigma \sum_y R(y)\,\nabla_\theta p_\theta(y \mid q,\sigma)$. The likelihood-ratio identity $\nabla_\theta p_\theta = p_\theta \nabla_\theta \log p_\theta$ and the factorization $\log p_\theta(y \mid q,\sigma) = \sum_i \log p_\theta(y_i \mid s_{b(i)}, q)$, which holds because nothing is remasked and y together with σ fixes every canvas, give equation 14. If p_θ is order-consistent then $p_\theta(\cdot \mid q,\sigma) = P(\cdot \mid q)$ for every σ by Proposition 4.3, so the inner expectation does not depend on σ and $J_\lambda = J_0$.

Why no importance weights are needed. OSE rollouts look off-policy: the policy never chooses the order, and the order distribution π_λ is not the one used at deployment. They are not off-policy in the sense that matters for the estimator. The order σ is drawn by the environment, not by p_θ, exactly as the confidence ranking is in every dLLM policy-gradient method, and at every step the token is drawn from p_θ itself. The rollouts are therefore on-policy for J_λ, the reward under OSE decoding, and the plain estimator equation 14 is unbiased for $\nabla_\theta J_\lambda$ with no ratio. What λ changes is the objective, J_λ in place of J_0. For an order-consistent model the two coincide; for a real model they can differ, which is why §5.5 decodes the trained policy with the default sampler ($\lambda = 0$) and checks that the gain transfers. Temperature does not have this property: it changes the action distribution, so an unbiased estimator of J_0 needs a per-token ratio.

Why dead groups come back. A GRPO group contributes a gradient only if its G rewards are not all equal. Under a nearly deterministic sampler the G rollouts are nearly identical (§3), so the group is almost always all-correct or all-wrong. The probability that a group is mixed rises with the rate at which two rollouts give different answers, and by Propositions 4.3–4.4 raising λ raises that rate without lowering the mean reward: the diversity is created by the order mixture, and its cost is bounded by the same quantity that creates it. §5.5 measures how many dead groups come back and whether the policy improves.

D Experimental details

We emphasize two experimental details. First, the answer of each draw is read from the finished text, and the vote needs no extra forward pass. Draw n uses the same token-noise seed in every evaluated method (common random numbers). So two methods differ only where their commit sets differ, and every confidence interval is a paired bootstrap over prompts.

Answer extraction and canonical keys. GSM8K: last boxed expression or last number, compared as a numeric value. MATH500: last boxed expression, math equivalence. ARC-C, RACE, MMLU: the option letter. Countdown: a valid expression over the given numbers equal to the target; the vote key is the evaluated value together with the multiset of operands used, which encodes the use-each-number

constraint without reference to the target (value-only key: 56.4 vs 59.2; raw string: 46.3, on random 16-draw subsets). Draws with no parsable answer get no vote; if every draw is unparsable the prompt is wrong.

Protocol. Three seeds are three independent draw sets (seed bases 0, 1 and 2). Draw n of prompt p uses the same token-noise seed in every method (common random numbers). So methods differ only where their commit sets differ. Sweeps use the 200 prompts of each task with 8 draws per seed. CIs are 95% bootstrap over prompts, paired across methods.

HEX on Dream. Dream has no block mode; we add `block_length` so that the response is decoded in L/b blocks left to right, T/n_{blocks} steps each, with Dream's own schedule inside a block and only the current block committable; `block_length` $= L$ is byte-identical to the unblocked sampler (test). The block size is cycled over draws, $\{8, 16, 32, 64, 128\}[d \bmod 5]$, so $N = 8$ and $N = 16$ cycle all five sizes, at $T = 0.9$ (the paper's setting) and $T = 0.1$. LLaDA decodes in blocks naively. The runner sets the block size per draw and never mixes block sizes in a batch.

E Answer diversity: the Gini–Simpson index

Definition. Fix a prompt q and N draws with canonicalised answers $a^{(1)}, \ldots, a^{(N)}$ (Algorithm 1; draws with no parsable answer all receive the same key $\varnothing$). Let n_a be the number of draws whose answer is a and $p_a = n_a/N$ its share. The answer diversity of the draw set is

$$D \;=\; 1 - \sum_a p_a^2, \tag{15}$$

the Gini–Simpson index of the answer distribution which the probability that two draws picked at random (with replacement) give different answers. $D = 0$ when all N draws agree and $D = 1 - 1/N$ when all differ (0.875 for $N = 8$, 0.9375 for $N = 16$). We report the mean of D over prompts, each seed's draws forming their own set, with 95% bootstrap CIs over prompts.

Worked example. Eight draws with answers $\{42, 42, 42, 42, 42, 17, 17, 9\}$ give $p = (5/8, 2/8, 1/8)$, $\sum_a p_a^2 = (25 + 4 + 1)/64 = 30/64$ and $D = 1 - 30/64 = 0.531$; the majority vote returns 42 with share $5/8$. Table 8 lists D for the common splits of eight draws. It is the key to reading Table 14: GSM8K's $\lambda = 0$ value 0.03 means that almost every prompt has eight identical answers; $\lambda = 1$'s 0.18 is on average a 7–1 split; MATH500's 0.57 at $\lambda = 1$ is about a 4–2–2 split.

Table 8: Gini–Simpson index D for splits of $N = 8$ draws over answers (counts of the distinct answers).

split	8	7–1	6–2	6–1–1	5–3	4–4	5–2–1	4–2–2	3–3–2	2–2–2–2	eight distinct
D	0	0.219	0.375	0.406	0.469	0.500	0.531	0.625	0.656	0.750	0.875

Why this index. It is the empirical form of the diversity term in the ambiguity decomposition equation 11. Each draw commits to one answer, so its member vector is one-hot, $f_n = e_{a^{(n)}}$, and the mixture over the N draws is $\bar{f} = (p_a)_a$. With $\|e_a - \bar{f}\|^2 = (1-p_a)^2 + \sum_{b \neq a} p_b^2 = 1 - 2p_a + \sum_b p_b^2$ and $\frac{1}{N}\sum_n p_{a^{(n)}} = \sum_a p_a^2$,

$$\frac{1}{N}\sum_{n=1}^{N} \left\| e_{a^{(n)}} - \bar{f} \right\|^2 \;=\; 1 - 2\sum_a p_a^2 + \sum_b p_b^2 \;=\; 1 - \sum_a p_a^2 \;=\; D. \tag{16}$$

The same plug-in turns the average member error of equation 11 into $\frac{1}{N}\sum_n \|e_{a^{(n)}} - e_{y^*}\|^2 = 2\,(1 - \bar{A})$, with $\bar{A}$ the single-draw accuracy. So the two numbers in every cell of Table 14 are the two terms of equation 11 estimated from the same draws, which is what Proposition C.5 and the efficiency η is needed. Compared with the entropy of the answer distribution, D is bounded, needs no logarithm of small counts, and has the direct meaning above.

What it does not measure. D is blind to correctness: eight different wrong answers give $D = 0.875$, eight copies of the right answer give $D = 0$. That is why D always appears next to the single-draw accuracy (Table 14), and why the vote (Table 12) and pass@N are reported on their own.

F IMPLEMENTATION NOTES

Gumbel-top-k. Adding i.i.d. Gumbel(0,1) noise to $-\mathrm{rank}_i/\lambda$ and taking the top n_k is equivalent to sampling n_k positions without replacement from the Plackett–Luce distribution with weights $\exp(-\mathrm{rank}_i/\lambda)$. Scores are $-\mathrm{rank}_i + \lambda g_i$, so λ can also vary per position (used by the spatial variant, Appendix C.10). At $\lambda = 0$ the noise vanishes and the commit set is the default sampler's; this is a regression test ($\lambda = 0$, $c = 0$ and block $= L$ all reproduce Dream token for token; 73 CPU tests).

Span scoring equation 17. After generation, run the grader's extraction rule on the finished text to get the answer's character range; map it to token positions with the tokenizer's offset mapping; put masks back on those positions only, keep the prompt, the reasoning, the wrapper tokens (`\boxed{`, `<answer>`) and the padding tail visible; one forward pass; sum the log-probabilities of the original answer tokens (for a one-token letter this is exact; for a multi-token answer it is the product of marginals under a joint mask, which we use because spans are short). A draw with no parsable answer is ranked last. Nothing is scored during generation; every candidate is a complete canvas.

G SELECTION RULES

Question. Given the same N OSE draws, is the majority vote the right way to pick the answer, or does the model's own likelihood do better?

The natural score for N candidates $y^{(1..N)} \sim p_\lambda$ is the model's own any-order log-likelihood, and it fails for an instructive reason. Split a candidate into its answer span $a(y)$ and the rest $r(y)$, the reasoning and the padding: the score splits the same way, and the term for the rest is two orders of magnitude larger while carrying no information about correctness. Full-response likelihood picked an unparsable draft 49–75% of the time, above the 53% base rate (Appendix G). The fix is to score only the tokens that carry the outcome, given everything else:

$$S_a(y) := \mathbb{E}_{\sigma\sim\pi_\lambda}\big[\log p_\theta\big(a(y) \mid q, r(y), \sigma\big)\big], \qquad \hat{y} = \operatorname*{argmax}_a \sum_{n:\, a(y^{(n)})=a} \exp\big(S_a(y^{(n)})\big). \quad (17)$$

The span score on the left costs one masked forward pass per candidate and beats voting at $N \le 8$; the weighted vote on the right reduces to it when all answers differ and approaches plurality as N grows. Since it gains at most +2.3 points at $N = 4$ and nothing by $N = 16$, plurality over canonicalised answers is OSE's default and the weighted vote its small-N variant; canonicalisation in Appendix G). A per-step λ did not beat a constant (Appendix C.10), so $\lambda = 1$, fixed before any result was seen, is what every headline number uses.

Table 9: Selection over the same 16 OSE draws, Dream, full test sets, three seeds. Weighted votes and every N: Table 10.

task	vote@16	answer-span likelihood equation 17	full-response likelihood	pass@16
MATH500	**58.7**	48.1	39.2	73.7
GSM8K	**89.0**	86.3	84.5	96.2
ARC-C	**91.1**	88.7	87.0	98.2
RACE	**87.0**	85.0	80.7	95.9
Countdown	**60.7**	49.2	22.3	78.9

Result. The vote wins on every task and both models (Table 9). Full-response likelihood is a bad selector everywhere, and on Countdown it is worse than a random draw, because it prefers unparsable drafts. The answer-span score removes that pathology but still trails the vote by 2–11 points at $N = 16$; as a vote weight it helps only at small N on Countdown and MATH500 (+5 at $N = 4$–8) and nowhere by $N = 16$. Picking the single most likely draw is always the worst option.

Why full-response likelihood fails (Countdown, 256 prompts $\times$ 3 seeds). Full-response rules pick a draft with no parsable answer 49% (sequential log-probability), 64% (any-order $K = 4$) and 75% (pseudo-likelihood) of the time, against a 53% base rate; content-only normalisation changes nothing (22.3 vs 22.3). The drafts they prefer are long, truncated, highly predictable reasoning (mean 245 content tokens). Random-subset controls at $N = 16$: majority vote with the canonical key 59.2 [54.4,

Table 10: Likelihood-weighted vote equation 17, Δ versus the majority vote on the same random N-subsets (32 per prompt), Dream, three seeds, and the single most likely draw.

task	N	weights = any-order ($K = 2$)	weights = answer span	argmax (single most likely draw)
Countdown	4 / 8 / 16	+0.8 / +0.9 / −1.8	**+4.7** / **+5.3** / −0.8	−15.6 / −26.9 / −44.5
MATH500	4 / 8 / 16	+1.0 / +0.4 / +0.0	**+2.3** / +1.0 / −0.2	−10.9 / −18.2 / −23.3
GSM8K	4 / 8 / 16	+0.3 / +0.2 / −0.2	−0.1 / −0.0 / −0.3	−2.6 / −3.9 / −5.1
RACE	4 / 8 / 16	+0.1 / +0.0 / +0.1	+0.4 / +0.1 / +0.0	−5.4 / −6.9 / −7.9

64.0]; answer-span likelihood 49.2 [44.3, 53.9] (43.5 at $N = 8$, 34.5 at $N = 4$, never unparsable); span + PMI-style normaliser 40.4; random among parsable draws 29.5; random among all draws 15.1; full-response likelihood 22.3; pass@16 78.9. The majority vote is the best rule in every cell at $N = 16$, and the answer-span rule ties it only at $N \leq 8$ on Countdown and RACE.

H A CONSTANT λ SUFFICES: ADAPTIVE AND SPATIAL λ

Closed-form schedule. In equation 13, spread_k is the standard deviation of the untempered entropies of the top-8 candidate positions and gap_k is how much more certain the top-ranked position is than the alternatives; $c = 0$ reproduces the $\lambda = 0$ sampler.

Question. Does the closed-form adaptive schedule of §C.10 beat a constant λ?

Result. No, Figure 7 shows that. The schedule matches the best constant on GSM8K, ARC-C and RACE and loses on MATH500, Countdown and MMLU; other values of c do worse (Appendix C.10). The best constant differs by task, $\lambda \approx 1$ where generation is long and 2–4 on short answers, and $\lambda = 1$ is within noise of it everywhere.

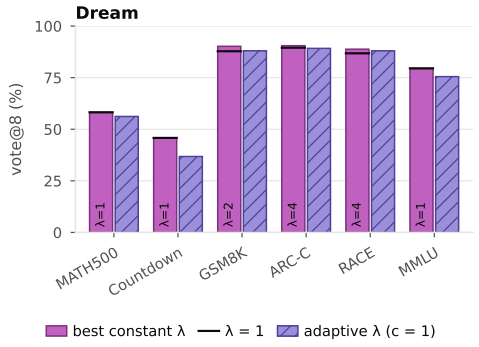

Figure 7: vote@8 on Dream, 200 prompts, three seeds: the best constant λ (value in the bar; black line = $\lambda = 1$) against the adaptive schedule, $c = 1$. All λ and c: Table 11.

Table 11: Vote@8, 200 prompts, three seeds.

task	$\lambda = 0$	$\lambda = 0.5$	$\lambda = 1$	$\lambda = 2$	$\lambda = 4$	adaptive $c = 0.5$	$c = 1$	$c = 2$
GSM8K	80.5	86.2	87.8	**90.2**	87.0	85.0	88.0	87.0
MATH500	42.5	55.5	**58.2**	54.8	48.2	54.2	56.2	50.0
Countdown	13.0	41.0	**45.8**	39.2	33.2	38.0	36.8	30.8
ARC-C	83.0	89.5	89.5	90.2	**90.5**	86.2	89.2	89.2
RACE	82.0	86.5	86.8	88.0	**88.8**	85.8	88.0	87.5
MMLU	74.5	75.0	**79.5**	74.8	75.0	75.2	75.5	73.8

Spatial λ. Rank-softening is $-\mathrm{rank} + \lambda \cdot \mathrm{Gumbel}$, so λ can vary per position. Keeping $\lambda = 0$ on positions whose proposed token is format skeleton (<, >, {, }, \boxed, specials, the tag payloads) and, in a second variant, on digits and operators, did not raise Countdown's parse rate: $\lambda = 0$ 42.6%, uniform $\lambda = 1$ 46.9%, variants 41.7% and 45.4%; vote@8 45.3 and 46.5 vs 44.9 (+0.4 [−3.5, +4.3], +1.6 [−3.9, +7.2]). The unparsable half is a length-limit effect present in the $\lambda = 0$ sampler (55% of its outputs hit 256 tokens).

Table 12: vote@N on full test sets, three seeds, paired 95% CIs. OSE($\lambda = 1$) against the default sampler ($\lambda = 0$; $T = 0.1$ on Dream, $T = 0$ on LLaDA), whose vote is self-consistency. pass@N and single-draw accuracy over the same draws. Dream $N = 16$, LLaDA $N = 8$.

model · task	N	OSE($\lambda = 1$)	default ($\lambda = 0$)	Δ [95% CI]	pass@N OSE / default	single-draw OSE / default
Dream · MATH500	16	**58.7**	39.4	**+19.3 [+15.4, +23.2]**	73.7 / 46.0	35.1 / 32.8
Dream · GSM8K	16	**89.0**	82.7	**+6.3 [+4.9, +7.8]**	96.2 / 85.6	81.7 / 82.6
Dream · MMLU	16	**74.6**	68.8	**+5.8 [+4.0, +7.7]**	92.5 / 76.1	65.5 / 65.1
Dream · RACE	16	**87.0**	82.4	**+4.6 [+2.2, +7.1]**	95.9 / 88.1	80.2 / 79.8
Dream · ARC-C	16	**91.1**	88.1	**+3.0 [+1.7, +4.4]**	98.2 / 91.3	85.9 / 85.9
Dream · Countdown	16	**60.7**	15.2	**+45.5 [+39.3, +51.8]**	78.9 / 15.2	15.3 / 15.2
LLaDA · MATH500	8	**45.6**	36.0	**+9.6 [+6.4, +12.6]**	64.2 / 46.3	36.2 / 35.1
LLaDA · GSM8K	8	**82.9**	77.1	**+5.8 [+4.2, +7.4]**	92.1 / 82.3	76.8 / 77.0
LLaDA · MMLU	8	**65.7**	60.3	**+5.5 [+3.6, +7.3]**	83.8 / 70.6	59.5 / 57.8
LLaDA · RACE	8	**86.4**	83.6	**+2.8 [+0.6, +5.0]**	93.6 / 87.8	78.4 / 79.4
LLaDA · ARC-C	8	**86.4**	81.1	**+5.3 [+3.7, +7.0]**	95.6 / 85.4	79.6 / 77.3
LLaDA · Countdown	8	**42.6**	18.2	**+24.4 [+18.8, +30.3]**	69.1 / 19.7	19.5 / 18.0

Table 13: OSE vote / default-sampler ($\lambda = 0$) vote / OSE pass@N.

Dream	$N = 1$	$N = 2$	$N = 4$	$N = 8$	$N = 16$
GSM8K	82.2 / 82.8 / 82.2	82.2 / 82.8 / 89.0	86.1 / 82.7 / 92.7	87.6 / 82.7 / 94.8	89.0 / 82.7 / 96.2
MATH500	34.4 / 34.2 / 34.4	39.3 / 35.8 / 48.1	48.0 / 38.0 / 57.3	53.8 / 39.8 / 67.3	58.7 / 39.4 / 73.7
ARC-C	86.4 / 86.3 / 86.4	88.2 / 87.2 / 92.8	90.0 / 87.5 / 95.7	90.9 / 87.9 / 97.3	91.1 / 88.1 / 98.2
RACE	78.7 / 80.0 / 78.7	81.7 / 80.4 / 88.1	85.0 / 82.4 / 91.7	86.4 / 82.2 / 94.2	87.0 / 82.4 / 95.9
MMLU	65.0 / 64.9 / 65.0	68.7 / 66.2 / 76.8	72.4 / 68.5 / 84.0	74.5 / 69.3 / 88.9	74.6 / 68.8 / 92.5
Countdown	15.0 / 15.2 / 15.0	21.9 / 15.2 / 26.4	33.2 / 15.2 / 45.3	44.9 / 15.2 / 65.8	60.7 / 15.2 / 78.9
LLaDA	$N = 1$	$N = 2$	$N = 4$	$N = 8$	
GSM8K	75.7 / 76.8 / 75.7	75.8 / 76.9 / 84.3	81.5 / 76.9 / 88.6	82.9 / 77.1 / 92.1	
MATH500	35.8 / 36.2 / 35.8	36.9 / 36.4 / 45.8	41.6 / 36.8 / 56.6	45.6 / 36.0 / 64.2	
ARC-C	80.0 / 77.0 / 80.0	83.4 / 78.8 / 88.6	85.5 / 80.2 / 93.1	86.4 / 81.1 / 95.6	
RACE	78.4 / 79.4 / 78.4	82.8 / 80.9 / 86.6	84.8 / 82.1 / 91.3	86.4 / 83.6 / 93.6	
MMLU	60.6 / 57.5 / 60.6	62.5 / 59.1 / 69.6	64.3 / 60.3 / 77.3	65.7 / 60.3 / 83.8	
Countdown	20.1 / 18.8 / 20.1	21.5 / 18.8 / 32.2	28.9 / 19.3 / 51.6	42.6 / 18.2 / 69.1	

I MAIN-RESULT DETAILS

Countdown. Only 47% of OSE draws contain a parsable expression (the $\lambda = 0$ sampler parses 42.6%; 55% of its outputs hit the 256-token limit). A random pick among parsable draws scores 29.5 against 15.1 among all draws, so about a third of the +45.5 is "pick a draw that answered". With $L = 512$ the model reasons longer and parses less (21.9% at $\lambda = 0$, 27.1% at $\lambda = 1$; 75.8% / 66.8% length-limited), and the OSE gain survives (+37.1 [+30.9, +43.4] at $N = 8$). Counting unparsable draws as votes gives 6.1 at $N = 16$ (2.3 at $L = 512$).

Screening experiment (GSM8K, 1,319 prompts $\times$ 8 draws). Best-of-8: reseeded $\lambda = 0$ draws 85.4; $\lambda = 1$ 94.5 ($G_8 = +9.0$ [+7.4, +10.6]); $\lambda = 4$ 95.3; $\lambda = 16$ 89.3; $\lambda = 64$ 81.8; uniform random order 74.5; untempered top-probability / margin 26.4 / 27.1. Perturbing the order near the $\lambda = 0$ ranking is what works.

J EXPLORATION EFFICIENCY, λ SWEEP, AND ORDER $\times$ TEMPERATURE

Every task. Figure 8 plots the whole grid for Dream, Figure 9 for LLaDA, and Table 14 lists it with the temperature sweep. On every Dream task the OSE path climbs first and bends only at $\lambda = 2$–4 (never on multiple choice up to $\lambda = 16$), while the T path slides left before it climbs; Countdown's higher single-draw accuracy at $T = 0.3$–0.5 comes with almost no diversity, so its vote gains nothing. On LLaDA, OSE($\lambda = 1$) buys more diversity than on Dream at no single-draw cost, and the T path lies on the same curve, because its block sampler tolerates temperature.

The one single-draw difference at $\lambda = 1$ that is not within ± 2 points of $\lambda = 0$ is LLaDA RACE (79.2 vs 81.6, -2.4; the CIs [75.0, 83.2] and [76.8, 86.4] overlap). On LLaDA, temperature is a usable knob: $T = 0.9$ with block decoding costs 2.1 points on GSM8K for diversity 0.238, about $\lambda = 1$'s

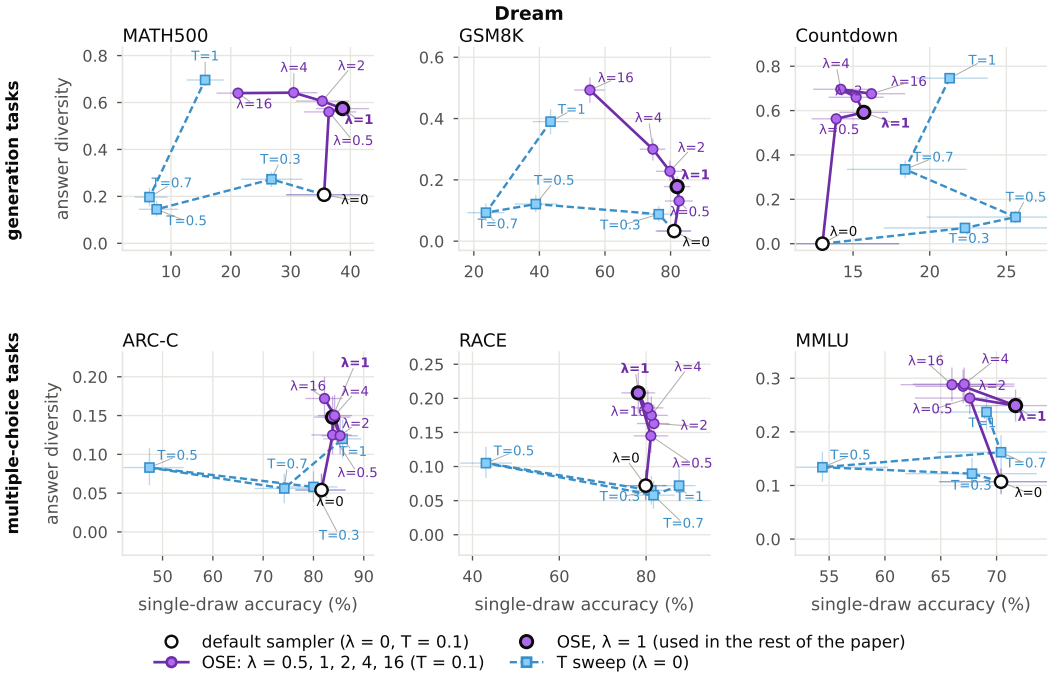

Figure 8: Dream: answer diversity against single-draw accuracy, 200 prompts, 8 draws, three seeds, 95% CIs. Both sweeps start at $\lambda = 0$, $T = 0.1$ (open circle): OSE(λ) at $T = 0.1$ (violet) and the T sweep at $\lambda = 0$ (light blue). A free knob climbs straight up. Top row: generation tasks; bottom: multiple choice.

Table 14: Single-draw accuracy / answer diversity, 200 prompts, 8 draws, three seeds. OSE(λ) at the default T; the temperature sweep at $\lambda = 0$. For LLaDA the last column is $T = 1.0$ ($T = 0.9$ on GSM8K: 75.2 / 0.238).

Dream	$\lambda = 0$	$\lambda = 0.5$	$\lambda = 1$	$\lambda = 2$	$\lambda = 4$	$\lambda = 16$	$T = 0.3$	$T = 0.5$	$T = 0.7$	$T = 1.0$
GSM8K	81.1 / 0.033	82.6 / 0.131	82.0 / 0.178	79.8 / 0.228	74.6 / 0.300	55.4 / 0.493	76.4 / 0.088	38.9 / 0.121	23.7 / 0.093	43.4 / 0.390
MATH500	35.6 / 0.207	36.4 / 0.560	38.7 / 0.574	35.3 / 0.606	30.5 / 0.642	21.2 / 0.640	26.8 / 0.273	7.6 / 0.145	6.4 / 0.197	15.7 / 0.696
ARC-C	81.6 / 0.054	83.8 / 0.125	83.8 / 0.148	85.3 / 0.124	84.2 / 0.150	82.2 / 0.172	80.0 / 0.058	47.4 / 0.083	74.2 / 0.056	85.8 / 0.120
RACE	79.9 / 0.072	81.1 / 0.145	78.2 / 0.208	81.8 / 0.163	81.2 / 0.175	80.4 / 0.186	81.2 / 0.065	43.1 / 0.105	81.7 / 0.058	87.6 / 0.072
MMLU	70.4 / 0.107	67.6 / 0.263	71.7 / 0.249	67.0 / 0.285	67.1 / 0.289	66.0 / 0.288	67.8 / 0.122	54.4 / 0.134	70.4 / 0.162	69.1 / 0.237
Countdown	13.0 / 0.000	13.9 / 0.563	15.7 / 0.592	15.2 / 0.660	14.2 / 0.696	16.2 / 0.676	22.3 / 0.071	25.6 / 0.120	18.4 / 0.335	21.3 / 0.746
LLaDA ($T = 0$)	$\lambda = 0$	$\lambda = 0.5$	$\lambda = 1$	$\lambda = 2$	$\lambda = 4$	$\lambda = 16$	$T = 0.2$	$T = 0.4$	$T = 0.7$	$T = 1.0$
GSM8K	77.3 / 0.057	76.5 / 0.175	76.2 / 0.218	76.6 / 0.226	75.0 / 0.251	70.8 / 0.325	77.2 / 0.145	75.1 / 0.200	75.2 / 0.223	75.0 / 0.245
MATH500	37.7 / 0.233	38.9 / 0.529	38.9 / 0.549	39.2 / 0.548	36.4 / 0.582	28.9 / 0.680	38.7 / 0.523	38.2 / 0.553	37.7 / 0.560	36.3 / 0.610
ARC-C	75.4 / 0.101	78.8 / 0.186	79.2 / 0.195	81.2 / 0.178	81.5 / 0.189	80.0 / 0.212	80.1 / 0.174	78.2 / 0.211	78.9 / 0.206	80.1 / 0.207
RACE	81.6 / 0.064	78.3 / 0.164	79.2 / 0.184	79.0 / 0.189	77.2 / 0.220	70.8 / 0.291	78.9 / 0.156	79.0 / 0.175	78.1 / 0.196	74.6 / 0.249
MMLU	59.3 / 0.150	61.2 / 0.284	59.9 / 0.306	61.6 / 0.280	60.6 / 0.291	58.2 / 0.333	61.3 / 0.268	61.1 / 0.287	60.6 / 0.303	58.9 / 0.332
Countdown	15.8 / 0.054	19.0 / 0.788	19.6 / 0.795	18.1 / 0.809	16.2 / 0.813	11.7 / 0.831	16.4 / 0.799	16.2 / 0.812	17.7 / 0.813	15.8 / 0.817

0.218. Figure 9 plots the LLaDA grid in the layout of Figure 8: the two families trace the same curve, and single-draw accuracy moves by less than its CI on every task up to $\lambda = 4$ and $T = 1.0$.

HEX's single draw at $T = 0.9$ is below OSE's on every setting (e.g. Dream MATH500 16.8 vs 38.7, LLaDA Countdown 12.6 vs 18.1), which is why it loses where the vote cannot repair the pool.

The interaction is negative on every generation task of both models and on every LLaDA task (Table 16), once the temperature is up, OSE adds a fraction of what it adds at the default T (Countdown +7.4 vs +32.8). Dream's three multiple-choice tasks are the exception, for a clear reason that $T = 0.5$ wrecks the single draw there and OSE repairs most of the vote, but the repaired level never beats OSE at $T = 0.1$. On LLaDA the cell "$\lambda = 0$, $T = 0.4$" equals the cell "$\lambda = 1$, $T = 0$" on every task: two ways to buy the same diversity.

Composition Method (OSE $\lambda = 1$ inside HEX's block cycling at $T = 0.9$) Table 15. No gain over either parent: Dream GSM8K +0.0 [−4.7, +4.7] vs OSE at $N = 4$ (128 prompts, seed 0);

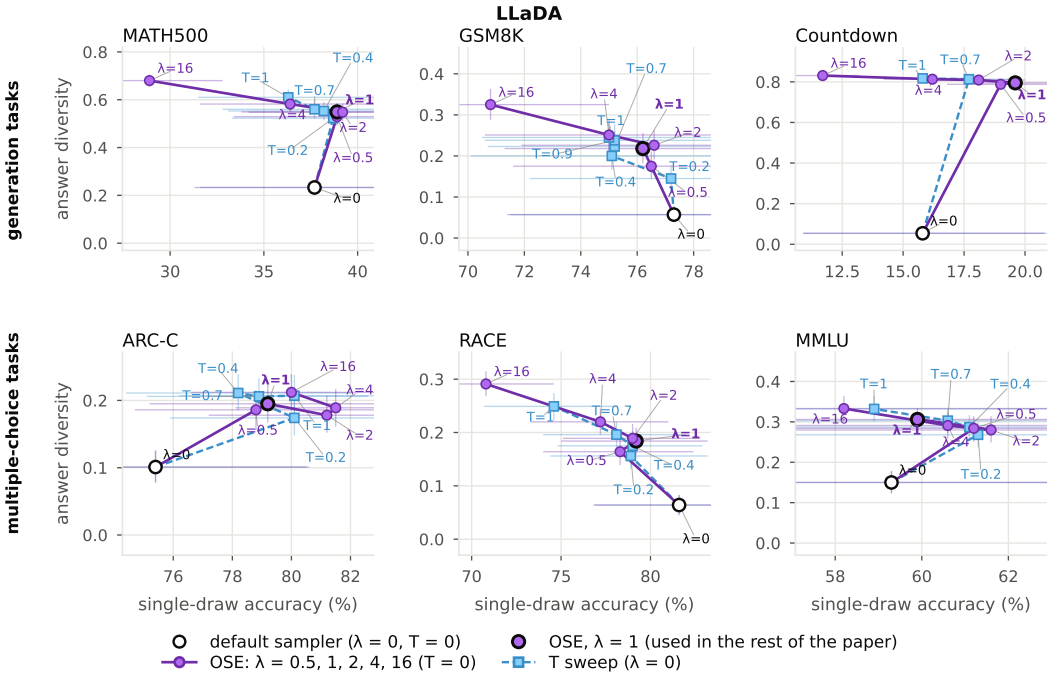

Figure 9: LLaDA: nswer diversity against single-draw accuracy. The open circle is $\lambda = 0$, $T = 0$, and the T sweep is $\{0.2, 0.4, 0.7, 1.0\}$. Here both knobs climb at little cost and lie on one curve. The x-axes are narrow, so the horizontal CIs look wide.

Table 15: λ sweep vs HEX: vote@8 (pass@8), 200 prompts, three seeds unless marked. The $\lambda = 0$ column is the default sampler. The LLaDA MMLU row uses the 170 prompts that all arms share.

setting	$\lambda = 0$	$\lambda = 0.5$	$\lambda = 1$	$\lambda = 2$	$\lambda = 4$	$\lambda = 16$	HEX $T = 0.9$	HEX $T = 0.1$
Dream · GSM8K	80.5 (85.0)	86.2 (92.0)	87.8 (95.5)	**90.2** (96.5)	87.0 (94.5)	73.2 (88.5)	91.0 (96.0)	83.2 (87.2)
Dream · MATH500	42.5 (48.5)	55.5 (66.5)	**58.2** (70.0)	54.8 (68.5)	48.2 (65.2)	34.8 (52.2)	41.8 (51.0)	50.2 (60.5)
Dream · Countdown	13.0 (13.0)	41.0 (59.5)	**45.8** (67.0)	39.2 (65.5)	33.2 (61.0)	36.0 (59.2)	39.2 (67.5)	25.5 (35.2)
Dream · ARC-C	83.0 (87.0)	89.5 (94.0)	89.5 (96.2)	90.2 (95.0)	**90.5** (97.5)	87.2 (97.2)	89.5 (98.0)	88.0 (93.5)
Dream · MMLU	74.5 (80.8)	75.0 (90.5)	**79.5** (90.8)	74.8 (92.2)	75.0 (92.0)	74.8 (91.2)	79.0 (93.8)	76.2 (86.5)
Dream · RACE	82.0 (87.5)	86.5 (93.8)	86.8 (95.0)	88.0 (95.0)	**88.8** (95.5)	86.5 (95.5)	86.5 (97.2)	80.8 (89.0)
LLaDA · GSM8K	78.0 (82.5)	80.8 (88.8)	81.0 (92.8)	83.0 (93.2)	**83.2** (92.8)	83.0 (93.2)	84.0 (93.5)	82.2 (89.0)
LLaDA · MATH500	38.2 (47.8)	47.2 (62.5)	49.0 (64.2)	**51.5** (65.0)	47.2 (65.2)	43.2 (62.3)	44.2 (63.0)	44.2 (58.8)
LLaDA · Countdown	15.5 (16.8)	39.2 (68.0)	42.0 (69.0)	**42.2** (69.8)	35.2 (64.2)	29.0 (57.0)	31.0 (58.0)	36.5 (63.0)
LLaDA · ARC-C	79.0 (84.8)	85.0 (94.5)	86.0 (95.5)	87.0 (98.0)	**89.0** (97.2)	88.2 (96.8)	87.2 (97.5)	86.0 (94.0)
LLaDA · MMLU	52.4 (65.3)	57.5 (80.8)	**59.1** (81.8)	58.1 (81.2)	59.1 (79.9)	58.3 (77.3)	58.5 (79.7)	58.8 (76.2)
LLaDA · RACE	85.5 (88.0)	84.5 (93.0)	87.5 (93.2)	**87.8** (94.2)	87.2 (94.8)	86.0 (94.2)	87.0 (95.0)	86.8 (93.8)

LLaDA GSM8K +1.0 [−0.2, +2.2] vs OSE and +0.9 [−0.2, +2.2] vs HEX at $N = 4$ (1,319 prompts). It shows again the two sources of diversity are substitutes.

K GENERALITY: LLADA AND HEX

Does the picture change on a second model whose sampler is not left-to-right and not deterministic? This section collects what is different on LLaDA; its main-table results are in Table 12 and Figure 5.

The gains transfer (Table 12, Figure 4), and three things differ, as the theory predicts. (i) OSE($\lambda = 1$) buys *more* diversity on LLaDA than on Dream at no single-draw cost (Table 14), because its realised order is less left-to-right and its factorization inconsistency ι is larger (Proposition 4.4). (ii) Its block sampler tolerates temperature, so order and temperature lie on the same efficiency curve (Figure 9) and are interchangeable (Table 16: every LLaDA interaction is negative). (iii) Consequently a tuned temperature is competitive on the short tasks: HEX at $T = 0.9$ ties OSE on GSM8K and the default

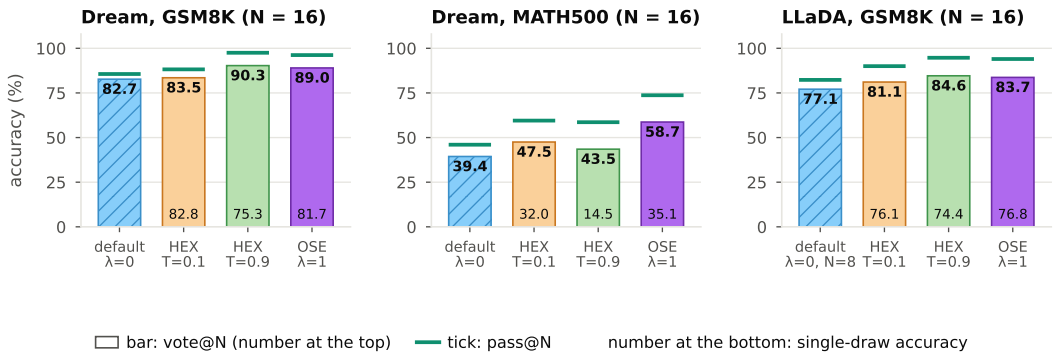

Figure 10: Full test sets at $N = 16$, three seeds: OSE($\lambda = 1$) against HEX at both temperatures and the default sampler (LLaDA default sampler at $N = 8$). Bars: vote@N; ticks: pass@N; bottom numbers: single-draw accuracy. No λ sweep was run at $N = 16$; on the 200-prompt sweep $\lambda = 2$ adds +2.5 [+0.5, +4.8] on Dream GSM8K.

Table 16: The 2×2 grid $\lambda \in \{0, 1\} \times T \in \{\text{default}, \text{raised}\}$ on 14 settings ($N = 8$, 200 prompts; raised $T = 0.5$ on Dream, 0.4 on LLaDA). It shows what OSE($\lambda = 1$) adds to vote@8 at each T, and the interaction (raised − default). Negative interaction means the two knobs are substitutes. Full grids are in Table 17.

setting	OSE adds at default T	OSE adds at raised T	interaction
Dream · Countdown	+32.8	+7.4	−25.4
Dream · MATH500	+15.7	+5.5	−10.2
Dream · GSM8K	+7.3	+3.4	−3.9
Dream · ARC-C / RACE / MMLU	+6.5 / +4.8 / +5.0	+27.0 / +27.0 / +11.0	+20.5 / +22.2 / +6.0
LLaDA · Countdown ($N = 4$)	+13.5	+1.4	−12.1
LLaDA · MATH500	+10.8	+0.8	−10.0
LLaDA · ARC-C / MMLU	+7.0 / +4.8	+1.5 / −0.2	−5.5 / −5.0
LLaDA · GSM8K / RACE	+3.0 / +2.0	+2.2 / +0.3	−0.8 / −1.7

order ($\lambda = 0$) at $T = 0.9$ is 1.2 points ahead at $N = 16$ (Figure 10); OSE's clear wins on LLaDA are MATH500 and Countdown (Figure 5).

In general, on LLaDA order is *a* free axis, not the only good one; it beats the default sampler everywhere and a tuned temperature where generation is long.

L GRPO DETAILS

Dead steps. With default-sampler rollouts, 54–90% of the training steps carry no gradient: every group in the step is all-correct or all-wrong (Figure 11). With OSE rollouts the share falls to 0–2% on GSM8K, MATH500 and Countdown, and to 29–49% on ARC-C, RACE and MMLU. On multiple choice the answer space is four letters, so unanimous groups remain common even with order diversity; this is why the test accuracy there does not move (§5.5).

GRPO setting is: Dream, LoRA $r = 128$ ($\alpha = 64$), learning rate 3×10^{-6}, 8 prompts × 8 rollouts per step, uniform sequence-level credit, no importance weights, three seeds. Rollouts at $T = 0.1$ with top-k 50 and no top-p, under the default order ($\lambda = 0$) or under OSE($\lambda = 1$); the held-out evaluation is one draw per prompt on the evaluation set with the default sampler ($\lambda = 0$, $T = 0.1$, top-p 0.9), which is why the training accuracy starts below the held-out accuracy. MATH500 300 steps (micro-batch 2); Countdown 190; GSM8K, ARC-C, MMLU, RACE 150. A mixed group is one whose 8 rollouts do not all get the same reward. At the last step the paired held-out differences (OSE minus default-sampler rollouts, mean over seeds, 95% CIs over prompts) are MATH500 +7.6 [+3.8, +11.4], Countdown +8.5 [+3.6, +13.4], GSM8K +2.7 [+0.8, +4.7], RACE +1.0 [−1.8, +3.8], ARC-C +0.4 [−1.4, +2.2], MMLU −0.4 [−4.8, +4.2]. Accuracy of the training rollouts, averaged over the first 50 and over the last 50 steps (default-sampler / OSE rollouts): MATH500 26.6 → 25.8 / 25.0 →

Table 17: Order × temperature, cells. It shows how increasing λ on top of T affects Dream ($N = 8$, 200 prompts, three seeds, with $T = 0.5$): vote@8 (single draw; pass@8).

task	$\lambda = 0, T = 0.1$	$\lambda = 1, T = 0.1$	$\lambda = 0, T = 0.5$	$\lambda = 1, T = 0.5$
Countdown	13.0	**45.8**	26.8	34.2
MATH500	42.5	**58.2**	10.0	15.5
GSM8K	80.5	**87.8**	38.8	42.2
ARC-C	83.0	**89.5**	56.0	83.0
RACE	82.0	**86.8**	55.0	82.0
MMLU	74.5	**79.5**	62.5	73.5

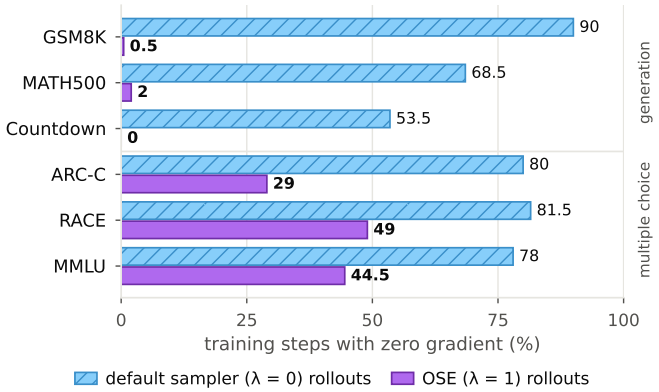

Figure 11: Share of GRPO training steps with zero gradient on Dream, default-sampler ($\lambda = 0$) rollouts against OSE($\lambda = 1$) rollouts, both at $T = 0.1$; mean over seeds. A step has zero gradient when all 8 of its groups are unanimous. OSE removes almost every dead step on the generation tasks and about half of them on multiple choice.

36.8; Countdown 22.6 → 25.8 / 19.3 → 33.4; GSM8K 36.2 → 38.9 / 48.6 → 50.2; ARC-C 90.3 → 90.0 / 89.6 → 89.9; RACE 88.2 → 88.5 / 87.0 → 88.4; MMLU 85.2 → 86.4 / 85.5 → 86.0. Control at rollout temperature 0.7: the mixed-group fraction is 0.84 with either order on Countdown, so order diversity has nothing to add there; the effect belongs to the deterministic regime in which these models are deployed.

M Cost

An OSE draw costs the same as a default draw, the same forward passes, with a different commit set. On one H100 at batch 64 ($L = T = 256$, ARC-C), a Dream rollout takes about 3.1 s with the default sampler and with OSE($\lambda = 1$) (3.16 / 3.13 s against 3.13 / 3.12 s), while a HEX draw takes 9.4–11.0 s at either temperature. That is, 3–3.5× more in our implementation. Dream at $T = 0.5$–1.0 runs in the same time per rollout but emits 9–48 content tokens instead of about 220 (the collapse of §3). As for scoring, the answer-span score costs 0.08 s per candidate, and the full-response scores 0.03–0.37 s (1–12% of a draw). The majority vote is free.

N The credit-assignment study that preceded this work

Before OSE we tried to turn the same trajectories into per-token credit for RL, with a counterfactual oracle (CRN-coupled re-rollouts from intermediate canvases) to validate the free signals. The oracle has a resolution floor $z\sqrt{2\sigma^2(1-\rho)/m}$ (0.30 at $m = 8$ re-rollouts). The free signals (stability, entropy, flip counts) located consequential commits but not their sign, and every reweighting scheme, including a published one, lost to uniform credit; a rewind-and-repair variant also failed. The trajectory tells you *where* a decision is made, not *which way* it should go; OSE uses it for the former.