S^3: STRATIFIED SCALING SEARCH FOR TEST-TIME SCALING IN DIFFUSION LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

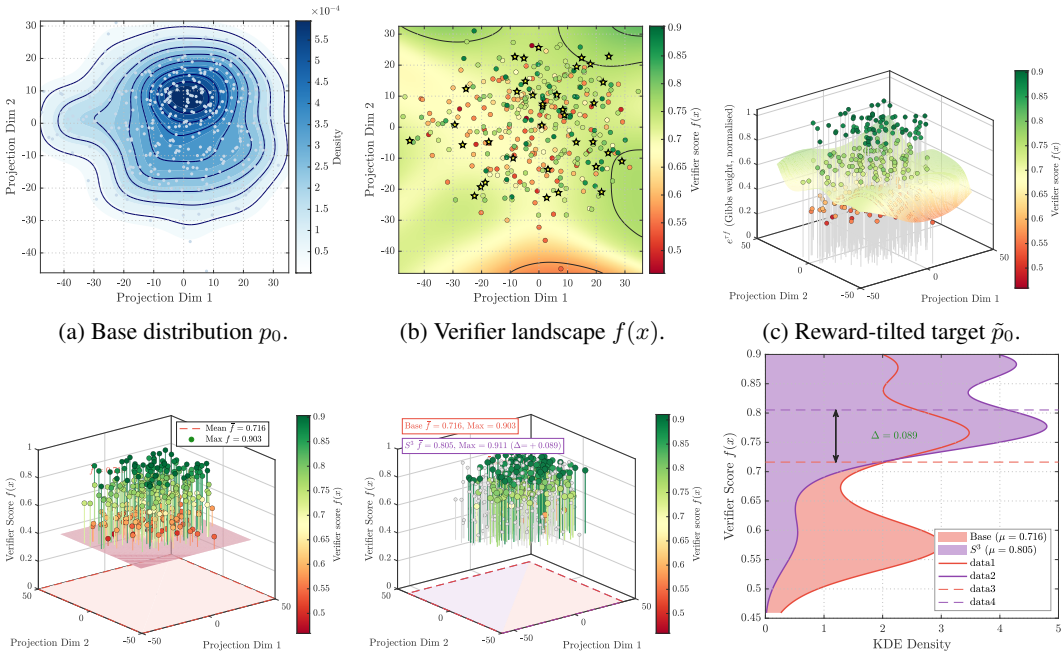
Test-time scaling asks how a fixed diffusion language model (DLM) can use additional inference compute to improve generation without retraining. Standard Best-of-K sampling spends this compute on generating independent complete outputs and selects only after decoding, leaving the generation process itself unchanged. We introduce S^3 **(Stratified Scaling Search)**, which instead allocates compute *during* denoising. At each denoising step, S^3 expands multiple partial trajectories, evaluates their one-step clean predictions with a black-box verifier, and resamples promising trajectories while preserving diversity. We derive this procedure from a KL-regularized reward-tilted target distribution and use incremental child-parent potentials whose product telescopes to the desired terminal reward tilt, making the ideal search objective invariant to the number of denoising steps. Our default verifier is training-free and requires neither a learned reward model nor ground-truth labels. Across all six benchmarks, S^3 improves over standard diffusion decoding at every tested inference budget, with performance relative to other test-time scaling methods depending on the reliability of intermediate predictions. Its gains depend on the quality of intermediate signals, while costly or unreliable PRM scoring can favor output-level selection.

1 INTRODUCTION

Test-time scaling addresses a fundamental question: given a fixed model and additional compute at inference, how much can performance improve? For autoregressive language models, the answer depends strongly on how compute is spent: chain-of-thought reasoning, Best-of-K (BoK) sampling, and tree search have all been shown to significantly enhance performance (Brown et al., 2024; Snell et al., 2024). Diffusion language models (DLMs) offer a structurally distinct opportunity. Recent diffusion LLMs already perform competitively with, and sometimes better than, similarly sized autoregressive models (Zhao et al., 2025).

For autoregressive models, inference-time compute is commonly allocated through token-level strategies such as beam search, Monte Carlo tree search, and BoK sampling (Brown et al., 2024; Snell et al., 2024), leveraging the sequential left-to-right generation process. DLMs offer a parallel opportunity: iterative denoising over T steps progressively refines partial states and induces multiple possible trajectories at each step (Kim et al., 2025a). However, standard diffusion decoding typically follows a single trajectory (Kim et al., 2025a; Nie et al., 2025a), while BoK only selects among independent samples from the unchanged base distribution and therefore cannot redirect generation before decoding is complete. Existing trajectory-level methods introduce different trade-offs. FK Steering performs particle resampling within a single denoising pass but can be sensitive to noisy intermediate scores and multinomial-resampling-induced diversity collapse (Singhal et al., 2025). Its original difference potential also telescopes along a trajectory; S^3 instead differs in its scheduled temperature, explicit $N \to Nb \to N$ branching, and Srinivasan Sampling Process (SSP) based resampling. PG-DLM instead iteratively refines complete trajectories through conditional sequential Monte Carlo (SMC), introducing additional full denoising passes and a separate refinement-iteration scaling axis (Dang et al., 2025). Amortized approaches require additional training, while BFS/DFS search incurs rapidly growing computational cost (Eyring et al., 2025; Zhang et al., 2025a). PRISM uses hierarchical self-verification, whose guidance may be unreliable for incomplete generations (Bai et al., 2026). To address these limitations, we analyze the inference-time objective from a

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.



(a) Base distribution p_0. (b) Verifier landscape $f(x)$. (c) Reward-tilted target $\tilde{p}_0$.

(d) Direct sampling from p_0. (e) S^3 trajectory search. (f) Score distribution shift.

Figure 1: Density-quality mismatch and S^3 trajectory search on MATH-500.

distributional perspective and derive a principled target distribution under a KL constraint relative to the model prior. Motivated by this objective, we introduce S^3 **(Stratified Scaling Search)**, which reallocates inference-time compute across multiple denoising trajectories through an $N \to Nb \to N$ search cycle within a single denoising pass.

Contributions. First, we identify a **density-quality mismatch** in DLMs, where high-probability regions of $p_0(x)$ are misaligned with verifier rewards, limiting naive BoK sampling. Second, building on the standard KL-regularized objective, we derive a **reward-tilted sequential target** for DLM denoising using incremental potentials that preserve the same terminal reward tilt across denoising schedules. Third, we propose a **training-free, verifier-guided particle search** that uses any black-box scalar verifier. S^3 uses a lightweight label-free verifier, optionally supports Process Reward Model (PRM) scoring, and preserves trajectory diversity via the SSP. Across six benchmarks under matched compute, it consistently improves over standard diffusion decoding and remains competitive with strong test-time scaling methods, with gains driven by the quality of intermediate predictions.

2 Preliminaries

Unlike autoregressive models that generate left-to-right, DLMs start from a fully masked state x_T and iteratively denoise through stochastic transitions $x_T \to x_{T-1} \to \cdots \to x_0$. These transitions induce multiple possible trajectories; standard decoding follows one, while S^3 searches over multiple trajectories during denoising.

2.1 The Density-Quality Mismatch

Let $p_0(x)$ be the base output distribution, $f(x)$ the verifier score, and $\mathcal{Q}(p_0) = \mathbb{E}_{x\sim p_0}[f(x)]$ the expected quality under direct sampling. Figure 1 visualizes the difficulty on MATH-500: the high-probability regions of p_0 (Figure 1a) are not the high-scoring regions of f (Figure 1b), so the mass that would raise $\mathcal{Q}(p_0)$ sits in a sparse part of the model's own distribution, and the target that corrects this (Figure 1c) has to move mass toward where f is high. Such misalignment has been observed in autoregressive settings (Snell et al., 2024; Brown et al., 2024); we formalize it for DLMs as the *density-quality mismatch*. A natural remedy is BoK sampling, whose cost we analyze first.

Remark 1 (BoK under an explicit tail profile). *Draw $x_1, \dots, x_K \overset{iid}{\sim} p_0$, define the i.i.d. scores $Y_k = f(x_k)$, and return $x_{k^\star}$ with $k^\star \in \arg\max_{k \le K} Y_k$ (ties broken arbitrarily). Assume the score has a two-sided Weibull tail under p_0 with index $\alpha \ge 1$ and scale σ, i.e. $\mathbb{P}_{p_0}[f - \mathcal{Q}(p_0) \ge u] \asymp e^{-(u/\sigma)^\alpha}$: the upper bound for all u, the lower bound on a finite window, which is the most a bounded f permits. Appendix A.2 states the constants and shows that, over the range of K where the window binds,*

$$\mathbb{E}\left[\max_{k \le K} f(x_k)\right] - \mathcal{Q}(p_0) = \Theta\Big(\sigma(\ln K)^{1/\alpha}\Big), \qquad \text{so a surplus } \varepsilon \text{ costs } \; K = \exp\{\Theta((\varepsilon/\sigma)^\alpha)\}. \tag{1}$$

Upper and lower bounds hold under the same *assumption, so the rate reflects BoK itself and not one-sided bounding; α=2 recovers the usual $\sigma\sqrt{2 \ln K}$ maximal inequality (Boucheron et al., 2013). The exponent is tail-sensitive: an exponential tail (α=1) costs $\exp\{\Theta(\varepsilon/\sigma)\}$ instead of $\exp\{\Theta(\varepsilon^2/\sigma^2)\}$, but the cost stays exponential for every α, and grows as the verifier score concentrates. The reason is structural: BoK never changes the sampling distribution. Every candidate is still a draw from p_0, so selection can only wait for a rare high-scoring sample rather than make one more likely. Improving on (1) therefore requires* tilting *the distribution rather than* filtering *its samples, which Section 2.2 makes precise.*

2.2 The Optimal Inference-Time Target

Tilting is made precise by asking which distribution maximizes expected verifier reward while staying close to the base model. We write this in KL-regularized form, $\max_{\tilde{p}}\{\mathbb{E}_{\tilde{p}}[f(x)] - \frac{1}{\tau} D_{\mathrm{KL}}(\tilde{p} \,\|\, p_0)\}$, rather than the constrained form $\max_{\tilde{p}} \mathbb{E}_{\tilde{p}}[f]$ subject to $D_{\mathrm{KL}}(\tilde{p} \,\|\, p_0) \le \delta$. The following solves both and relates τ to δ, giving τ a principled interpretation as the parameter that determines the allowable KL deviation from the base model and thus the reward-fidelity trade-off.

Proposition 1 (Optimal reward-tilted distribution). *Let $\mathcal{X}^L$ be finite, $f : \mathcal{X}^L \to [0, 1]$, and optimize over $\tilde{p} \ll p_0$. For every $\tau > 0$ the regularized objective is strictly concave, and its* unique *maximizer is the Gibbs distribution*

$$\tilde{p}_0(x_0; \tau) \;\propto\; p_0(x_0)\, e^{\tau f(x_0)}. \tag{2}$$

Write $\delta(\tau) = D_{\mathrm{KL}}(\tilde{p}_0(\cdot;\tau) \,\|\, p_0)$. Then $\delta'(\tau) = \tau \operatorname{Var}_{\tilde{p}_0(\cdot;\tau)}[f]$, which is positive whenever f is non-constant on $\operatorname{supp}(p_0)$. In that case δ rises continuously from 0 to $\delta_{\max} = -\ln p_0(M)$, where M is the set on which f attains its maximum over $\operatorname{supp}(p_0)$, so every radius $\delta \in (0, \delta_{\max})$ corresponds to exactly one τ. At that τ the constraint is active and the constrained maximizer is the same distribution. If f is constant on $\operatorname{supp}(p_0)$, the regularized optimizer remains uniquely p_0 for every $\tau > 0$, while $\delta(\tau) \equiv 0$, so the temperature-to-KL-radius map degenerates. For the constrained reward-only problem, uniqueness may instead fail because multiple feasible distributions can attain the same reward. Proof in Appendix A.1.

Equation (2) is the standard variational solution of KL-constrained optimization and exponential tilting (Donsker & Varadhan, 1975). It also settles what BoK can achieve.

Corollary 1 (BoK is weakly dominated at matched KL). *Let π_K be the law of the BoK output $x_{k^\star}$ defined in Remark 1 under p_0, with any tie-breaking rule. The returned output is one of the K draws, so $\pi_K \le K\, p_0$ pointwise and $D_{\mathrm{KL}}(\pi_K \,\|\, p_0) \le \ln K$. BoK is therefore a feasible point of the same problem, and writing τ_K for the temperature Proposition 1 assigns to its KL radius,*

$$\mathbb{E}_{\tilde{p}_0(\cdot;\tau_K)}[f] - \mathbb{E}_{\pi_K}[f] = \tfrac{1}{\tau_K} D_{\mathrm{KL}}(\pi_K \,\|\, \tilde{p}_0(\cdot;\tau_K)) \ge 0. \tag{3}$$

Thus the reward-tilted distribution weakly dominates BoK at matched KL. Equality holds if and only if $\pi_K = \tilde{p}_0(\cdot;\tau_K)$; otherwise the inequality is strict. Proof in Appendix A.3.

Two consequences guide the rest of the paper. First, (3) holds for any sampler at its matched temperature, so divergence to $\tilde{p}_0$, not achieved reward, is the informative diagnostic: BoK attains the highest mean reward yet the worst match to $\tilde{p}_0$ in Appendix E.3, buying reward by spending KL rather than allocating it well. Second, approximating $\tilde{p}_0$ requires the sequential denoising structure that output-level selection ignores. Section 3 lifts (2) to the denoising path through per-step weights whose product recovers the terminal reward tilt. Figures 1d-1f illustrate this behavior, while Appendix E.3 confirms that S^3 matches $\tilde{p}_0$ more closely than direct sampling, BoK, and FK-Steering (Figure 11).

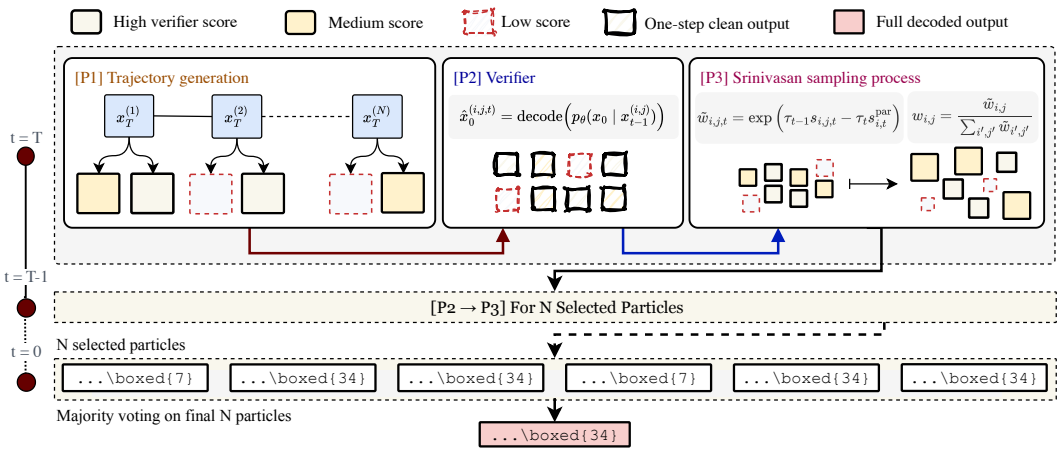


Figure 2: S^3 **pipeline.** Trajectory expansion [P1], look-ahead verifier scoring [P2], and SSP resampling [P3] at each denoising step, followed by majority voting at t=0.

3 METHODOLOGY

Section 2 derives the reward-tilted distribution in Equation (2) as the optimal inference-time target. S^3 approximates this target via training-free trajectory search under a fixed decoding schedule (Figure 2; Appendix B).

3.1 DIFFUSION DECODING AS A PATH-SPACE PROBLEM

Let x be the prompt and L the output length. A fixed decoding schedule $e = (\mathcal{K}, \pi, T)$, comprising block length $\mathcal{K}$, masking update policy π, and $T = \lceil L/\mathcal{K} \rceil$ steps defines a reverse chain from the fully masked state $x_T \in \mathcal{X}^L$ to the decoded sequence $x_0 \in \mathcal{X}^L$. At step t the schedule specifies $\mathcal{K}$ updatable positions $U_t \subseteq \{1, \dots, L\}$: the model samples $x_{t-1} \sim p_\theta(\cdot \mid x_t, e)$ and holds all positions outside U_t fixed. Chaining these transitions gives a *base path measure* over trajectories $x_{0:T}$, $p(x_{0:T}) = p_T(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t, e)$, whose terminal marginal $p_0(x_0) = \sum_{x_{1:T}} p(x_{0:T})$ is the base output distribution of Section 2 (Nie et al., 2025b). The formulation needs only a stochastic reverse transition, so it applies to discrete (Dat et al., 2025), continuous, or energy-based (Xu et al., 2025) instantiations alike.

Level 1: the exact terminal target. Extending the tilt in Equation (2) from outputs to trajectories gives the *target path measure* $\tilde{p}(x_{0:T}) \propto p(x_{0:T})\, e^{\tau f(x_0)}$, whose terminal marginal is exactly $\tilde{p}_0$. Here $f : \mathcal{X}^L \to [0, 1]$ is the verifier and $\tau > 0$ is the reward temperature. We refer to $\tilde{p}_0$ throughout as the *reward-tilted target*.

Level 2: the exact sequential realization. Sampling from $\tilde{p}$ requires *twisting* the reverse kernels. Define the *backward information function* $h_t(x_t) := \mathbb{E}_p[\, e^{\tau f(x_0)} \mid x_t \,]$, the expected future reward reachable from the partial state x_t. Its boundary values are $h_0(x_0) = e^{\tau f(x_0)}$ at the clean state and $h_T(x_T) = \mathbb{E}_p[e^{\tau f(x_0)}] = Z$ at the masked state, where Z is a *constant*: every trajectory starts from the same x_T, so h_T carries no state dependence. The density ratio of the target is therefore the clean-state h divided by that constant, $d\tilde{p}_0/dp_0 = h_0/h_T$. The step-$t$ marginal factorizes as $\tilde{p}_t(x_t) \propto p_t(x_t) h_t(x_t)$, and the twisted kernel realizing $\tilde{p}$ sequentially is

$$\tilde{p}_\theta(x_{t-1} \mid x_t) \;=\; p_\theta(x_{t-1} \mid x_t,\, e) \cdot \frac{h_{t-1}(x_{t-1})}{h_t(x_t)}. \tag{4}$$

This is the Doob h-transform of the base kernel (Whiteley & Lee, 2014), with incremental potential $G_t(x_t, x_{t-1}) := h_{t-1}(x_{t-1})/h_t(x_t)$. These potentials define the ideal Feynman–Kac path measure with terminal marginal $\tilde{p}_0$, while a finite-N SMC sampler provides an approximation that becomes consistent as N increases under standard assumptions (Doucet et al., 2001; Neal, 2001). The auxiliary particle filter (Pitt & Shephard, 1999) provides a finite-particle approximation to this weighting.

Level 3: the S^3 approximation. Computing h_t exactly requires marginalizing over all trajectories from step t to 0, which is as expensive as running the full reverse process from every candidate state. S^3 replaces it with a surrogate built from the model's own one-step clean prediction.

3.2 Trajectory search with look-ahead scoring

The reverse chain induces a search tree over partial states. We maintain N trajectories $\{x_t^{(i)}\}_{i=1}^N$ and expand each into b successors, $x_{t-1}^{(i,j)} \sim p_\theta(\cdot \mid x_t^{(i)}, e)$, yielding Nb candidates. Since x_0 is unavailable at intermediate steps, we approximate h_{t-1} using the one-step clean prediction $\hat{x}_0^{(i,j,t)} = \mathrm{decode}(p_\theta(x_0 \mid x_{t-1}^{(i,j)}))$ and define the look-ahead score $s_{i,j,t} = f(\hat{x}_0^{(i,j,t)}) \in [0,1]$.

A scheduled surrogate for h_t. Let $0 = \tau_T \le \tau_{T-1} \le \cdots \le \tau_0 = \tau$ be a non-decreasing tilt schedule and define

$$\hat{h}_t(x_t) := \begin{cases} \exp\{\tau_t f(\hat{x}_0(x_t))\}, & t \ge 1, \\ \exp\{\tau f(x_0)\}, & t = 0. \end{cases} \tag{5}$$

For $t \ge 1$, $\hat{h}_t$ reuses the look-ahead prediction already computed by the search, while $\hat{h}_0 = h_0$ exactly and $\hat{h}_T \equiv 1$ because $\tau_T = 0$. Thus the surrogate preserves the exact boundary values while providing a tractable approximation at intermediate steps, following the look-ahead principle of auxiliary and twisted particle filters (Pitt & Shephard, 1999; Whiteley & Lee, 2014). Appendix E.1 shows that the look-ahead score becomes more predictive of final correctness as denoising progresses (Figure 7).

3.3 Verifier-weighted resampling with Srinivasan Sampling Process (SSP)

Following Equation (4), we weight each expansion by the incremental potential of the surrogate rather than by the surrogate itself: for each child in the frontier,

$$\tilde{w}_{i,j,t} \;=\; G_t\Big(x_t^{(i)}, x_{t-1}^{(i,j)}\Big) \;=\; \frac{\hat{h}_{t-1}\Big(x_{t-1}^{(i,j)}\Big)}{\hat{h}_t\Big(x_t^{(i)}\Big)} \;=\; \exp\big\{\tau_{t-1}\, s_{i,j,t} \;-\; \tau_t\, s_{i,t}^{\mathrm{par}}\big\}, \tag{6}$$

normalized as $w_{i,j,t} = \tilde{w}_{i,j,t} / \sum_{i',j'} \tilde{w}_{i',j',t}$. The parent score $s_{i,t}^{\mathrm{par}} = f(\hat{x}_0(x_t^{(i)}))$ is cached from the previous expansion and carried with its lineage through resampling, so Equation (6) adds no model or verifier evaluation. A larger increment $\tau_{t-1} - \tau_t$ more strongly favors high-scoring children. A small increment alone does not ensure near-uniform weights when τ_t is nonzero; in our geometric schedule, early weights are near-uniform because both τ_t and τ_{t-1} are close to zero.

Proposition 2 (The per-step potentials telescope to the reward tilt)**.** *Let $0 = \tau_T \le \cdots \le \tau_0 = \tau$ be any valid tilt schedule, and let G_t denote the S^3 incremental potential in Equation (6). Then, along every denoising lineage $x_T \to \cdots \to x_0$,*

$$\prod_{t=1}^{T} G_t(x_t, x_{t-1}) = \frac{\hat{h}_0(x_0)}{\hat{h}_T(x_T)} = e^{\tau f(x_0)}, \tag{7}$$

so the accumulated reward tilt is independent of the number of denoising steps T and block length $\mathcal{K}$.

Because each G_t is a ratio across consecutive states, all intermediate factors telescope, leaving only $\hat{h}_T \equiv 1$ and $\hat{h}_0 = e^{\tau f(x_0)}$. Thus, any valid schedule yields the same ideal reward-tilted terminal target as Equation (2).

Remark 2 (The surrogate controls variance, not the target)**.** *Equation (7) depends only on the boundary values of $\hat{h}$ and does not require $\hat{h}_t \approx h_t$ at intermediate steps. Hence, the schedule determines how the fixed total tilt is distributed across denoising steps and, at finite N, can affect weight variance, resampling error, particle genealogy, and approximation quality (Whiteley & Lee, 2014; Pitt & Shephard, 1999). In contrast, a stand-alone weight $\exp\{\lambda s_{i,j,t}\}$ does not telescope: its factors accumulate as $\exp\{\lambda \sum_t s_t\}$, causing the effective terminal tilt to grow with T. This comparison concerns absolute-score weighting; the FK-Steering difference potential also telescopes. Appendix E.4 verifies Equation (7) on logged lineages.*

Choice of schedule. We use a fixed geometric schedule $\tau_t = ((1+\gamma)^{T-t} - 1)\tau_{\text{base}}$, where γ controls how quickly the tilt grows toward later denoising steps. This applies little guidance early and concentrates it near termination, where the look-ahead score is more informative (Appendix E.1). In contrast, the Effective Sample Size (ESS)-based adaptive schedule selects each temperature increment from the current frontier weights using $\text{ESS}_t = 1/\sum_{i,j} w_{i,j,t}^2$, increasing the tilt while maintaining particle diversity. In our experiments, however, it allocates most of the tilt at highly masked states where look-ahead scores are least reliable, so we retain the geometric/fixed schedule as the default; Appendix C provides the adaptive construction and ablation.

From weights to offspring counts. The expected allocation $\xi_{i,j,t} = N\, w_{i,j,t}$, with $\sum_{i,j} \xi_{i,j,t} = N$, is converted into integer counts $n_{i,j,t} \in \mathbb{Z}_{\geq 0}$ summing to N by the **Srinivasan Sampling Process (SSP)** (Srinivasan, 2001), a low-variance dependent-rounding procedure. Retaining stochasticity here is critical: deterministic top-k pruning collapses modes within the search tree, whereas stochastic resampling preserves multiple plausible trajectories. On identical logged frontier weights, SSP attains minimal offspring-count variance while keeping all N founder lineages, while multinomial and top-N resampling collapse to a few (Appendix E.2, Figures 9 and 10). Each step therefore applies one *expand–score–resample* update: expansion proposes Nb continuations, scoring evaluates their clean predictions, and SSP resampling reallocates the particle budget. Repeating over T steps shifts the population along $\tilde{p}$, and by Proposition 2 the weight a lineage accumulates over the full pass is exactly the reward tilt of Equation (2); Appendix D illustrates this empirically, and Algorithm 1 in Appendix B states the full procedure.

3.4 Verifier, compute budget, and output selection

S^3 requires only a normalized scalar score per candidate, allowing any bounded verifier to be substituted without changing the search. By default, we use a training-free composite verifier combining structural completeness, arithmetic consistency, answer reachability, model confidence, non-degeneracy, and task-specific constraint satisfaction, $f(x) = \sum_{k=1}^{5} \alpha_k s_k(x) + \alpha_c s_{\text{constraint}}(x)$, with $\sum_{k=1}^{5} \alpha_k + \alpha_c = 1$, using only lightweight deterministic operations on generated text and existing model logits, with no external models, labels, API calls, or additional DLM forward passes. The optional S^3+PRM variant replaces this score with a PRM. Appendix H details the verifier components and weights, and Appendix H.6 analyzes their correlation with correctness. We measure compute by NFE, with $\text{NFE} = TNb$ for N particles, branching factor b, and T denoising steps, versus $\text{NFE} = T$ for single-trajectory decoding (Table 11). For both S^3 and S^3+PRM, the final answer $y^\star$ is selected by majority vote over surviving particles, breaking ties by lowest base-model NLL. Computational and selection ablations are reported in Appendices I and E.5.

4 Evaluation and Results

We evaluate S^3 on GSM8K (Cobbe et al., 2021), MATH-500 (Hendrycks et al., 2021), TruthfulQA (Lin et al., 2022), and ARC-Challenge (Clark et al., 2018), as well as MBPP (Austin et al., 2021b) and HumanEval (Chen et al., 2021) for code generation, using LLaDA-8B-Instruct (Nie et al., 2025b) with its default denoising schedule. We follow Lee et al. (2025) for preprocessing and answer normalization. At matched inference budgets, we compare against the baseline, BoK, and FK-Steering (Singhal et al., 2025).[1] Full experimental details and the configuration of all budgets and grids are provided in Appendix G (Table 11), with the FK-Steering implementation in Appendix G.1; throughout, **bold** and underlined denote the best and second-best results, respectively.

4.1 Verifier Configurations and Benchmark Performance

Default S^3 and FK-Steering use the training-free composite verifier (Appendix H). Theoretical BoK in Section 2.1 selects the highest-verifier-scoring sample, while our default experimental BoK uses majority voting to match the output rule across S^3, BoK, and FK-Steering. BoK+PRM instead selects the highest-scoring completed sample using Qwen2.5-Math-PRM-7B (Zhang et al., 2025b). S^3+PRM

[1] FK-Steering is the closest matched baseline. DAS (Kim et al., 2025b) targets continuous diffusion, PG-DLM (Dang et al., 2025) uses multi-pass refinement, and PRISM (Bai et al., 2026) changes the search hierarchy and verifier.

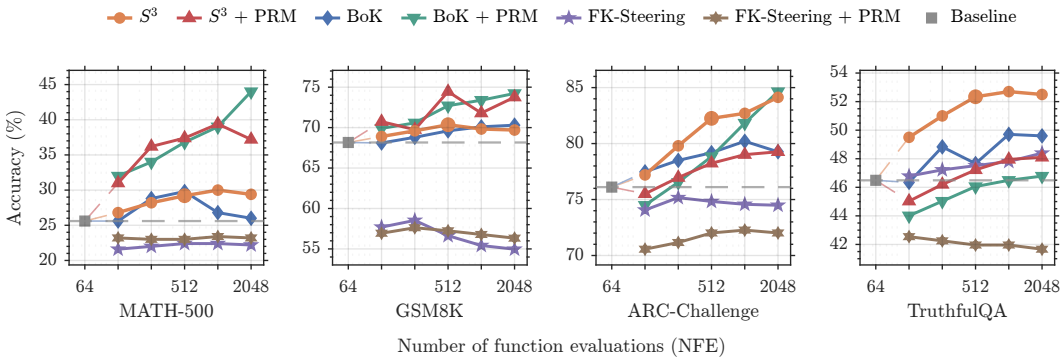


Figure 3: **Inference-time accuracy versus NFE** across the budgets in Table 11; BoK comparisons are in Appendix F, and PRM results under total FLOPs are in Figure 4.

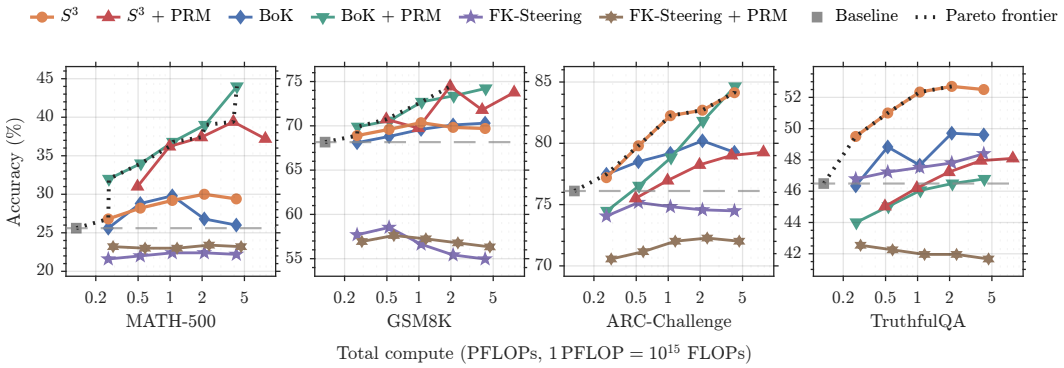


Figure 4: **Accuracy versus matched total inference FLOPs**, including verifier costs.

and FK-Steering+PRM replace their intermediate verifier with the same PRM, while retaining their respective output rules. Training-free methods are matched by NFE, whereas PRM variants are compared by total FLOPs including verifier cost (Table 11, Table 17, Appendix F.4).

As shown in Figure 3, S^3 improves over single-trajectory decoding on all four benchmarks and at every tested budget. At the default budget, the largest gains are on ARC-Challenge (+6.14 pp) and TruthfulQA (+5.85 pp), with consistent improvements on MATH-500 and GSM8K. In contrast, FK-Steering remains below the single-trajectory baseline on three of four benchmarks despite using the same model, decoding schedule, verifier, and majority-vote output rule, highlighting the benefit of S^3's branching and SSP-based resampling (Appendices E.2 and E.5). Against matched BoK, the advantage of S^3 depends on the informativeness of intermediate predictions. On LLaDA, S^3 leads BoK at the default budget on TruthfulQA, ARC-Challenge, and GSM8K, while trailing by only 0.60 pp on MATH-500; at the largest budget, it outperforms BoK on ARC-Challenge, MATH-500, and TruthfulQA, while BoK remains stronger on GSM8K (Table 5, Appendix F). This behavior reflects the structural distinction in Corollary 1: BoK spends additional compute on independent complete samples from the unchanged base distribution, whereas S^3 can redirect generation during denoising. The saturation of S^3 at larger budgets suggests an exploration–exploitation trade-off: stronger verifier-guided resampling improves exploitation but can reduce trajectory diversity. This

Table 1: Code-generation test-time scaling using LLaDA.

Method	B=2	B=4	B=8	B=16	B=32
	MBPP	*Baseline: 24.40*			
BoK	30.60	30.60	26.40	29.40	31.00
FK-Steering	29.20	30.60	**35.40**	33.40	34.20
S^3	**30.80**	**33.00**	35.20	**34.60**	**34.80**
	HumanEval	*Baseline: 19.51*			
BoK	20.73	23.17	24.39	25.00	25.30
FK-Steering	20.73	**31.71**	32.32	**32.62**	**32.78**
S^3	**21.95**	27.44	**32.62**	32.02	**32.78**

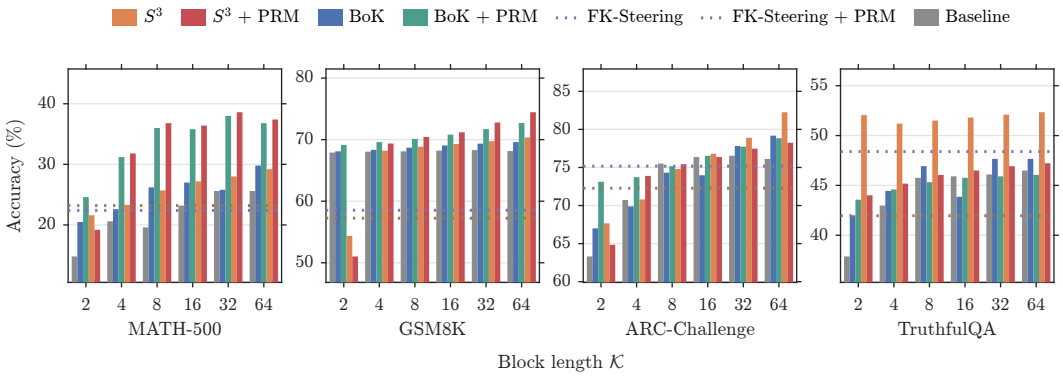


Figure 5: **Accuracy across block lengths** at the matched default budget of Table 11. Dotted lines denote the block-independent FK-Steering variants. Per-block deltas and the two crossings are analyzed in Appendix F.

motivates adaptive control of branching and resampling based on verifier confidence to preserve exploration as compute increases.

The broader robustness of this trajectory-level compute allocation is more evident across models and task domains. On MBPP and HumanEval, S^3 improves over single-trajectory decoding by $+10.8$ and $+13.1$ pp at the default budget, consistently outperforms output-level scaling across the full budget sweep, and remains competitive with FK-Steering, achieving the best performance or tying for best at the largest budget (Table 1). More importantly, the Dream-v0-Instruct-7B results provide particularly clear evidence of cross-model generalization: using the same default configuration ($\mathcal{K}$=64, N=4, b=2) without model-specific retuning, S^3 achieves the best accuracy on all four benchmarks, improves over single-trajectory decoding by up to 6.40 pp, and consistently matches or outperforms BoK-8 (Table 13, Appendix G.2). Together, the code-generation and Dream experiments show that the benefits of trajectory-level compute allocation extend beyond the original LLaDA reasoning setting and transfer across both task domains and independently trained DLMs.

4.2 Where the Extra Inference Compute Pays Off

With the model fixed, performance depends on three factors: inference budget, block length, and verifier choice. Their effect is governed by how well the one-step clean prediction reflects the final answer. When this signal is informative, S^3 can use compute to steer generation effectively; when it is weak, generating more complete samples is preferable. The look-ahead AUROC in Appendix E.1 identifies these regimes without labels, while Appendix F analyzes the corresponding trends across all three axes.

Budget and block length. The scaling behavior reflects the same structural distinction as Corollary 1: BoK expands only its pool of independent complete samples, whereas S^3 reallocates compute during denoising. This is why S^3 leads BoK across essentially the whole budget range on the general-domain benchmarks (Figures 3 and 5, Tables 5 and 6). The exceptions are exactly the low-AUROC regions of Appendix E.1: GSM8K, and the two ends of the block-length range, where the clean prediction is either reconstructed from a nearly fully masked state or read too rarely to redirect the positions a step commits (Appendices F.1 and F.2). The claim is supported by consistent trends across the grid rather than any single result (Appendix F.5), with population-level evidence that the advantage emerges during denoising in Appendix F.6 and qualitative matched-budget trajectory examples in Appendix J.

Verifier. Substituting a PRM holds the search and the budget fixed and so isolates the scorer; its effect tracks the PRM's training domain rather than either search procedure (Appendix H.7). Because the methods query it at very different frequencies, matched diffusion-model NFE is not matched total compute for them, so we re-price every configuration in FLOPs with all verifier calls charged (Figure 4, Table 17). That re-pricing changes the answer and we report it rather than restrict the comparison: at equal total FLOPs, BoK+PRM is ahead at every matched point on MATH-500 and ARC-Challenge, S^3+PRM never falls behind on TruthfulQA, and GSM8K alternates (Appendix F.4). This is the same criterion read on the scorer instead of the schedule: a PRM is calibrated on complete

reasoning, whereas S^3+PRM must query it on partial states and TNb times against K, so this is a real boundary of the method rather than a detail of tuning: *trajectory-level search pays off when the intermediate signal is cheap and reasonably informative, and output-level selection is the better buy when the scorer is expensive and only trustworthy on complete text.* It is also why the training-free composite verifier is the default, and why the reward-model-free S^3 remains the strongest of all seven configurations on both general-domain benchmarks.

Which mechanism drives the gain? Look-ahead (previewing candidate outcomes) and reward tilting (favoring higher-scoring trajectories) are complementary: each gives only limited gains alone, while together they recover nearly the full benefit of S^3 at matched compute (Table 10, Appendix F.7). This matches the twisted-SMC view, where look-ahead estimates trajectory quality and tilting converts that estimate into particle allocation, while Table 3 shows that the benefit of ESS-adaptive scheduling varies with problem difficulty, supporting the fixed schedule as the more consistent default.

5 Related Work

Diffusion models for generation. Diffusion models generate samples by reversing a noising process (Song et al., 2021), with masked diffusion extending this framework to discrete data (Austin et al., 2021a). Diffusion-LM introduced diffusion-based text generation (Li et al., 2022), while recent DLMs scale masked denoising to competitive language modeling (Nie et al., 2025b).

Inference-time control and test-time scaling. Effective allocation of inference-time compute can substantially improve model performance (Snell et al., 2024), motivating adaptive verification strategies (Chen et al., 2025a). Exponential tilting defines a principled reward-weighted distribution anchored to the model prior (Donsker & Varadhan, 1975), while DAS approximates this target using diversity-preserving tempered SMC for continuous diffusion models (Kim et al., 2025b). Related control methods include training-free guidance (Ye et al., 2024), derivative-free value guidance (Li et al., 2024), controlled Langevin diffusion (Chen et al., 2025b), and verifier-guided scaling for image and video diffusion (Ma et al., 2025; Baraldi et al., 2025). These methods mainly modify transitions or proposals rather than search over populations of discrete masked trajectories.

Population and search-based diffusion inference. Population-based methods maintain and resample multiple trajectories, building on sequential Monte Carlo (Doucet et al., 2001), annealed importance sampling (Neal, 2001), auxiliary particle filters (Pitt & Shephard, 1999), and twisted particle filters (Whiteley & Lee, 2014). FK-Steering is the closest prior method, already using Feynman-Kac potentials, look-ahead scoring, particle resampling, and telescoping difference potentials (Singhal et al., 2025). S^3 instead differs through scheduled reward tilting, explicit trajectory branching, and SSP-based diversity-preserving resampling. Twisted Diffusion Samplers construct twisted proposals for conditional generation (Wu et al., 2023); soft value-based decoding uses derivative-free value estimates (Li et al., 2024); and twisted SMC applies learned twisting functions to autoregressive language models (Zhao et al., 2024). PG-DLM applies particle Gibbs and conditional SMC to iteratively refine complete DLM trajectories across multiple denoising passes, introducing the number of refinement iterations as an additional scaling axis (Dang et al., 2025). In contrast to these multi-pass approaches, S^3 performs verifier-guided branching and diversity-preserving resampling within a single denoising pass. Related approaches include amortized test-time computation (Eyring et al., 2025), classical BFS/DFS search over diffusion trajectories (Zhang et al., 2025a), and hierarchical self-verifying search for discrete DLMs (Bai et al., 2026).

6 Conclusions

The consistent gains across reasoning, code generation, and an independently trained DLM suggest that S^3 captures a robust principle for test-time scaling: inference compute is more useful when it is allocated adaptively across promising trajectories rather than spent uniformly on independent samples. A key limitation, however, is that this benefit depends on intermediate predictions being informative and cheap to verify; when these signals are weak, the search can saturate or concentrate too strongly on a small set of trajectories. This limitation motivates future methods that adapt branching, resampling strength, and particle diversity based on verifier confidence. More broadly, S^3 points toward test-time inference as a dynamic resource-allocation problem, where compute should be spent where it is most likely to improve the final output.

AI USE STATEMENT

`ChatGPT` (OpenAI) and `Claude` (Anthropic) were used for editing, proofreading, LaTeX formatting, and checking references; `Codex` (OpenAI) was used for plotting, table generation, and code refactoring. AI was not used to develop the theory or proofs, design the method or experiments, implement S^3 or the baselines, or interpret results; synthetic data, translation, dataset processing, and qualitative analysis are not applicable. The authors verified all AI-assisted content and take full responsibility for the paper.

ETHICS STATEMENT

This work studies inference-time compute allocation for DLMs and does not introduce new datasets, human subjects, or deployment systems that raise additional ethical concerns beyond those associated with large language models.

REPRODUCIBILITY STATEMENT

All experiments use public benchmarks (GSM8K, MATH-500, ARC-Challenge, TruthfulQA, MBPP, HumanEval) and publicly released models (LLaDA-8B-Instruct, Dream-v0-Instruct-7B, Qwen2.5-Math-PRM-7B). The full S^3 procedure is given in Algorithm 1 (Appendix B), and its ESS-adaptive variant in Algorithm 2 (Appendix C). Complete proofs of all theoretical results are in Appendix A. Table 11 and Appendix G list every model, budget, schedule, hyperparameter, and preprocessing step. Appendix G.1 details the FK-Steering baseline, and Appendix H specifies every verifier component and weight (Table 14). Compute accounting is in Appendix I.2 (Table 17).

REFERENCES

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 17981–17993, 2021a.

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

Jinbin Bai, Yixuan Li, Yuchen Zhu, Yi Xin, Qingyu Shi, Aosong Feng, Xiaohong Liu, Molei Tao, Jianru Xue, Xiangtai Li, and Ming-Hsuan Yang. Prism: Efficient test-time scaling via hierarchical search and self-verification for discrete diffusion language models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=QrC8OgQyOI.

Lorenzo Baraldi, Davide Bucciarelli, Zifan Zeng, Chongzhe Zhang, Qunli Zhang, Marcella Cornia, Lorenzo Baraldi, Feng Liu, Zheng Hu, and Rita Cucchiara. Verifier matters: Enhancing inference-time scaling for video diffusion models. In *Proceedings of the 36th British Machine Vision Conference (BMVC)*, 2025.

Ahmad Beirami, Alekh Agarwal, Jonathan Berant, Alexander D'Amour, Jacob Eisenstein, Chirag Nagpal, and Ananda Theertha Suresh. Theoretical guarantees on the best-of-n alignment policy. *arXiv preprint arXiv:2401.01879*, 2024.

Alexandros Beskos, Ajay Jasra, Nikolas Kantas, and Alexandre Thiery. On the convergence of adaptive sequential Monte Carlo methods. *The Annals of Applied Probability*, 26(2):1111–1146, 2016.

Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford University Press, 2013. ISBN 9780199535255. doi: 10.1093/acprof:oso/9780199535255.001.0001.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

Hao Mark Chen, Guanxi Lu, Yasuyuki Okoshi, Zhiwen Mo, Masato Motomura, and Hongxiang Fan. Rethinking optimal verification granularity for compute-efficient test-time scaling. *arXiv preprint arXiv:2505.11730*, 2025a.

Junhua Chen, Lorenz Richter, Julius Berner, Denis Blessing, Gerhard Neumann, and Anima Anandkumar. Sequential controlled Langevin diffusions. In *International Conference on Learning Representations (ICLR)*, 2025b.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

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

Meihua Dang, Jiaqi Han, Minkai Xu, Kai Xu, Akash Srivastava, and Stefano Ermon. Inference-time scaling of diffusion language models with particle Gibbs sampling. *arXiv preprint arXiv:2507.08390*, 2025.

Do Huu Dat, Duc Anh Do, Anh Tuan Luu, and Wray Buntine. Discrete diffusion language model for efficient text summarization. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 6293–6305, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi: 10.18653/v1/2025.findings-naacl.352. URL https://aclanthology.org/2025.findings-naacl.352/.

Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. An adaptive sequential Monte Carlo method for approximate Bayesian computation. *Statistics and Computing*, 22(5):1009–1020, 2012.

Monroe D. Donsker and S. R. Srinivasa Varadhan. Asymptotic evaluation of certain Markov process expectations for large time, I. *Communications on Pure and Applied Mathematics*, 28(1):1–47, 1975.

Arnaud Doucet, Nando de Freitas, and Neil Gordon (eds.). *Sequential Monte Carlo Methods in Practice*. Information Science and Statistics. Springer, New York, 2001. doi: 10.1007/978-1-4757-3437-9.

Luca Eyring, Shyamgopal Karthik, Alexey Dosovitskiy, Nataniel Ruiz, and Zeynep Akata. Noise hypernetworks: Amortizing test-time compute in diffusion models. *arXiv preprint arXiv:2508.09968*, 2025.

Mathieu Gerber, Nicolas Chopin, and Nick Whiteley. Negative association, ordering and convergence of resampling methods. *The Annals of Statistics*, 47(4):2236–2260, 2019.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2021.

Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham Kakade, and Sitan Chen. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, 2025a.

Sunwoo Kim, Minkyu Kim, and Dongmin Park. Test-time alignment of diffusion models without reward over-optimization. In *International Conference on Learning Representations (ICLR)*, 2025b.

Augustine Kong, Jun S. Liu, and Wing Hung Wong. Sequential imputations and Bayesian missing data problems. *Journal of the American Statistical Association*, 89(425):278–288, 1994.

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. *arXiv preprint arXiv:2510.05040*, 2025.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S. Liang, and Tatsunori B. Hashimoto. Diffusion-LM improves controllable text generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 4328–4343, 2022.

Xiner Li, Yulai Zhao, Chenyu Wang, Gabriele Scalia, Gokcen Eraslan, Surag Nair, Tommaso Biancalani, Shuiwang Ji, Aviv Regev, Sergey Levine, and Masatoshi Uehara. Derivative-free guidance in continuous and discrete diffusion models with soft value-based decoding, 2024. URL https://arxiv.org/abs/2408.08252.

Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3214–3252, 2022.

Nanye Ma, Shangyuan Tong, Haolin Jia, Hexiang Hu, Yu-Chuan Su, Mingda Zhang, Xuan Yang, Yandong Li, Tommi Jaakkola, Xuhui Jia, and Saining Xie. Scaling inference time compute for diffusion models, 2025. URL https://arxiv.org/abs/2501.09732.

Radford M. Neal. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, 2001.

Shen Nie, Fengqi Zhu, Chao Du, Tianyu Pang, Qian Liu, Guangtao Zeng, Min Lin, and Chongxuan Li. Scaling up masked diffusion models on text. In *International Conference on Learning Representations (ICLR)*, 2025a.

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

Michael K. Pitt and Neil Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94(446):590–599, 1999.

Raghav Singhal, Zachary Horvitz, Ryan Teehan, Mengye Ren, Zhou Yu, Kathleen McKeown, and Rajesh Ranganath. A general framework for inference-time scaling and steering of diffusion models. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, 2025.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling model parameters, 2024. URL https://arxiv.org/abs/2408.03314.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021.

A. Srinivasan. Distributions on level-sets with applications to approximation algorithms. In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*, pp. 588–597, 2001. doi: 10.1109/SFCS.2001.959935.

Nick Whiteley and Anthony Lee. Twisted particle filters. *The Annals of Statistics*, 42(1):115–141, 2014.

Luhuan Wu, Brian Trippe, Christian Naesseth, David Blei, and John P. Cunningham. Practical and asymptotically exact conditional sampling in diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 31372–31403, 2023.

Minkai Xu, Tomas Geffner, Karsten Kreis, Weili Nie, Yilun Xu, Jure Leskovec, Stefano Ermon, and Arash Vahdat. Energy-based diffusion language models for text generation. In *International Conference on Learning Representations (ICLR)*, 2025.

Haotian Ye, Haowei Lin, Jiaqi Han, Minkai Xu, Sheng Liu, Yitao Liang, Jianzhu Ma, James Y. Zou, and Stefano Ermon. TFG: Unified training-free guidance for diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 22370–22417, 2024.

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

Xiangcheng Zhang, Haowei Lin, Haotian Ye, James Zou, Jianzhu Ma, Yitao Liang, and Yilun Du. Inference-time scaling of diffusion models through classical search. *arXiv preprint arXiv:2505.23614*, 2025a.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*, 2025b.

Siyan Zhao, Devaansh Gupta, Qinqing Zheng, and Aditya Grover. d1: Scaling reasoning in diffusion large language models via reinforcement learning. *arXiv preprint arXiv:2504.12216*, 2025.

Stephen Zhao, Rob Brekelmans, Alireza Makhzani, and Roger Grosse. Probabilistic inference in language models via twisted sequential Monte Carlo. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, 2024.

Appendix Contents

A Proofs for Section 2 **15**

A.1 Proof of Proposition 1 . . . 15

A.2 Tail profile and proof of Remark 1 . . . 15

A.3 Proof of Corollary 1 . . . 16

B The S^3 Algorithm **16**

C ESS-Adaptive Tilt Increments **16**

C.1 The increment rule . . . 17

C.2 What the adaptive schedule preserves, and what it does not . . . 18

C.3 Accuracy with and without ESS adaptation . . . 19

D Distributional Shift of Verifier Scores **20**

E Diagnostic Validation of the S^3 Mechanism **20**

E.1 Does the look-ahead score predict final correctness? (validating $\hat{h}_t \approx h_t$) . . . 21

E.2 Is SSP resampling necessary? (diversity preservation) . . . 21

E.3 Does S^3 approximate the Gibbs target? (exact validation) . . . 22

E.4 Do the per-step potentials multiply to the reward tilt? (identity check) . . . 23

E.5 Is majority voting responsible for robustness? (selection ablation) . . . 24

F Regime Analysis and Mechanism Ablation: When Trajectory Search Beats Terminal Sampling **26**

F.1 The budget axis . . . 26

F.2 The block-length axis . . . 27

F.3 The verifier axis . . . 27

F.4 The PRM variants at matched total compute . . . 27

F.5 What these differences resolve . . . 29

F.6 Where the gain appears during denoising . . . 30

F.7 Mechanism ablation: tilted reward vs. look-ahead . . . 30

G Experimental Details **31**

G.1 FK-Steering Implementation . . . 31

G.2 Transfer to a Second DLM: Dream-v0-Instruct-7B . . . 32

H Verifier Design **33**

H.1 Composite Score Formulation . . . 33

H.2 Verifier Weight Profiles . . . 33

H.3 Component Scores . . . 34

H.4 Task-Specific Constraint Term . . . 36

H.5 Computational Complexity . . . 36

H.6 Verifier–Correctness Rank Correlation . . . 36

H.7 Optional Verifier Substitution: Process Reward Model 37

I Ablation Study: Computational Cost and Output Conciseness **38**

I.1 Component Ablation at the Default Budget 39

I.2 Compute Budget and FLOPs Analysis 40

I.3 Effect of Block Size 40

J Qualitative Trajectory-Level Comparison **40**

A Proofs for Section 2

All three results follow from one identity, which we record first. Comparing any $\tilde{p} \ll p_0$ against the tilt $\tilde{p}_0(\cdot;\tau)$ of Equation (2), and writing $Z(\tau) = \mathbb{E}_{p_0}[e^{\tau f}]$ for its normalizer,

$$D_{\mathrm{KL}}(\tilde{p} \,\|\, \tilde{p}_0(\cdot;\tau)) \;=\; D_{\mathrm{KL}}(\tilde{p} \,\|\, p_0) \;-\; \tau\, \mathbb{E}_{\tilde{p}}[f] \;+\; \ln Z(\tau), \tag{8}$$

obtained from $\ln \frac{\tilde{p}}{\tilde{p}_0} = \ln \frac{\tilde{p}}{p_0} - \tau f + \ln Z(\tau)$ by taking expectations under $\tilde{p}$. In words: a distribution's reward and its distance from the prior are linked, and the slack in that link is exactly how far it sits from the tilt. Every claim below is this sentence applied to a different $\tilde{p}$.

A.1 Proof of Proposition 1

Uniqueness. Rearranging (8), the regularized objective satisfies $\tau\, \mathcal{J}_\tau(\tilde{p}) = \ln Z(\tau) - D_{\mathrm{KL}}(\tilde{p} \| \tilde{p}_0(\cdot;\tau))$. The only $\tilde{p}$-dependent term is a KL divergence, which is non-negative and vanishes at one point, so the maximizer is $\tilde{p}_0(\cdot;\tau)$ and nothing else. Nothing beyond finiteness of $\mathcal{X}^L$ is needed: there $\mathbb{E}_{\tilde{p}}[f]$ is linear and $D_{\mathrm{KL}}(\tilde{p} \| p_0)$ strictly convex, so $\mathcal{J}_\tau$ is strictly concave.

Temperature versus KL radius. Let $\psi(\tau) = \ln Z(\tau)$. Standard exponential-family calculus gives $\psi'(\tau) = \mathbb{E}_{\tilde{p}_0(\cdot;\tau)}[f]$ and $\psi''(\tau) = \mathrm{Var}_{\tilde{p}_0(\cdot;\tau)}[f]$, the mean and variance of the score at temperature τ. Putting $\tilde{p} = \tilde{p}_0(\cdot;\tau)$ into (8) gives $\delta(\tau) = \tau\psi'(\tau) - \psi(\tau)$, hence $\delta'(\tau) = \tau\psi''(\tau)$: raising the temperature buys KL at a rate set by how much the score still varies. If f is non-constant this rate is positive, so δ increases strictly from 0. As $\tau \to \infty$ the tilt concentrates on the top-scoring set M, approaching $p_0(\cdot \mid M)$, whose divergence from p_0 is $-\ln p_0(M) = \delta_{\max}$ approached but never attained at finite τ, which is why the range is the open interval $(0, \delta_{\max})$.

The constrained problem. Fix $\delta \in (0, \delta_{\max})$ and $\tau = \tau(\delta)$. Solving (8) for the reward writes $\mathbb{E}_{\tilde{p}}[f]$ as $\frac{1}{\tau}$ times $D_{\mathrm{KL}}(\tilde{p}\|p_0) + \ln Z(\tau) - D_{\mathrm{KL}}(\tilde{p}\|\tilde{p}_0(\cdot;\tau))$. The first term is at most δ and the last is at least 0, so $\mathbb{E}_{\tilde{p}}[f] \le \frac{1}{\tau}[\delta + \ln Z(\tau)] = \mathbb{E}_{\tilde{p}_0(\cdot;\tau)}[f]$. Reaching the bound needs both slacks to close at once: the candidate must spend its full KL budget *and* coincide with the tilt. That gives a single maximizer and an active constraint.

Degenerate cases. If f is constant on $\mathrm{supp}(p_0)$, say $f(x) = c$, then the reward term is the same for every distribution. The regularized objective therefore becomes $c - \frac{1}{\tau} D_{\mathrm{KL}}(\tilde{p}\|p_0)$, which is uniquely maximized by $\tilde{p} = p_0$. Hence, $\delta(\tau) = 0$ for every τ, meaning that different values of τ no longer correspond to different KL radii. In contrast, for the constrained reward-only problem, every feasible distribution achieves the same reward when f is constant, so the solution may be non-unique for $\delta > 0$. If f is non-constant and $\delta \ge \delta_{\max}$, the KL budget is large enough to place all probability mass on the highest-scoring set M. At $\delta = \delta_{\max}$, $p_0(\cdot \mid M)$ is the unique minimum-divergence solution, while for $\delta > \delta_{\max}$ multiple optimal solutions may exist when $|M| \ge 2$.

A.2 Tail profile and proof of Remark 1

The assumption. Write $f^\star = \max_{x \in \mathrm{supp}(p_0)} f(x)$. We say f has *tail profile* $(\sigma, \alpha, u_\star)$, with $\alpha \ge 1$ and $0 < u_\star \le f^\star - \mathcal{Q}(p_0)$, if for constants $0 < c_- \le 1 \le c_+$ we have $\mathbb{P}_{p_0}[f - \mathcal{Q}(p_0) \ge u] \le c_+ e^{-(u/\sigma)^\alpha}$ for all $u > 0$, and $\mathbb{P}_{p_0}[f - \mathcal{Q}(p_0) \ge u] \ge c_- e^{-(u/\sigma)^\alpha}$ for all $u \in (0, u_\star]$. This is the standard sub-Weibull(α) condition, equivalently a finite Orlicz ψ_α-norm with $\psi_\alpha(x) = e^{x^\alpha} - 1$,

imposed two-sidedly so both bounds below rest on the same hypothesis. The asymmetry is deliberate. The upper inequality is asked for at all u, which costs nothing: $f \in [0,1]$ already forces the tail to vanish past $f^\star - \mathcal{Q}(p_0)$. The lower one is asked for only on $(0, u_\star]$, because a bounded score cannot have a genuine tail beyond its own maximum; requiring one everywhere would be an assumption no verifier could meet. Set $K_\star = \exp\{(u_\star/\sigma)^\alpha\}$; the claim holds for $K \in [e/c_-,\ K_\star/c_-]$, the budgets for which the window binds. Write $Z_k = f(x_k) - \mathcal{Q}(p_0)$, so $\mathbb{E}[Z_1] = 0$, and $M_K = \max_{k \le K} Z_k$ with tail $\bar{F}(u) = \mathbb{P}[Z_1 \ge u]$.

Upper bound. The expected maximum is set by the level at which the union bound stops being informative. At $a = \sigma(\ln(c_+ K))^{1/\alpha}$ we have $K\bar{F}(a) = 1$, so splitting $\mathbb{E}[M_K] \le \int_0^\infty \min\{1, K\bar{F}(u)\}\,\mathrm{d}u$ at a leaves a tail integral bounded by $C_\alpha \sigma$ with C_α depending on α alone, giving $\mathbb{E}[M_K] \le a + C_\alpha \sigma$. This is the classical ψ_α maximal inequality; α=2 is the familiar sub-Gaussian case $\sigma\sqrt{2 \ln K}$ (Boucheron et al., 2013).

Lower bound. An upper bound alone cannot show BoK is inefficient, so we match it. At $u_K = \sigma(\ln(c_- K))^{1/\alpha}$, which lies inside the window on the stated range of K, the lower tail gives $\bar{F}(u_K) \ge 1/K$. At least one draw clears u_K with probability $1 - (1 - \bar{F}(u_K))^K \ge 1 - e^{-1}$, so $\mathbb{E}[M_K^+] \ge (1 - e^{-1}) u_K$. The negative part is small because $M_K \ge Z_1$ and Z_1 is centered, so $\mathbb{E}[M_K^-] \le \mathbb{E}[Z_1^+] \le c_+ \sigma \Gamma(1 + \frac{1}{\alpha})$. Subtracting gives a lower bound of the same order $\sigma(\ln K)^{1/\alpha}$; this is the finite-K counterpart of the classical fact that maxima of i.i.d. variables with Weibull-type tails grow at this rate.

Conclusion. Both bounds are $\sigma(\ln K)^{1/\alpha}$ up to constants, which is (1); solving $\sigma(\ln K)^{1/\alpha} \asymp \varepsilon$ gives the sample complexity. Since they hold under the same hypothesis, the rate is a property of BoK rather than of the proof technique.

A.3 Proof of Corollary 1

BoK stays close to the prior. The returned output must be one of the K draws, so a union bound gives $\pi_K(x) \le K\, p_0(x)$ for every x: the likelihood ratio is capped at K regardless of f or the tie-breaking rule, hence $D_{\mathrm{KL}}(\pi_K \| p_0) \le \ln K$. Sharper analyses place this divergence strictly below $\ln K - (K-1)/K$ (Beirami et al., 2024); a smaller value only strengthens what follows.

Why that settles the comparison. Having finite divergence from p_0, π_K is a feasible point of the very problem Proposition 1 solves at radius $\delta_K = D_{\mathrm{KL}}(\pi_K \| p_0)$, its own divergence. Applying (8) at $\tau_K = \tau(\delta_K)$ to π_K and to $\tilde{p}_0(\cdot; \tau_K)$ and subtracting, the $D_{\mathrm{KL}}(\cdot \| p_0)$ and $\ln Z$ terms cancel and (3) follows.

What matters is the form. The reward gap is exactly $\frac{1}{\tau_K} D_{\mathrm{KL}}(\pi_K \| \tilde{p}_0(\cdot; \tau_K))$, so it is always non-negative. The gap becomes zero only when $\pi_K = \tilde{p}_0(\cdot; \tau_K)$; otherwise, the Gibbs-tilted distribution achieves higher expected reward at the same KL distance from p_0. Therefore, the result shows weak domination rather than guaranteed strict domination. This result does not depend on the tail assumption in Remark 1: that assumption describes how many samples BoK needs, while this result compares its reward with the KL-optimal Gibbs tilt. The same argument also applies to any $q \ll p_0$ at the same KL radius.

B The S^3 Algorithm

Algorithm 1 states the full procedure of Section 3. Expansion and scoring are those of Section 3.2; the weights are the incremental potentials of Equation (6), and resampling is the SSP dependent rounding of Section 3.3.

C ESS-Adaptive Tilt Increments

Proposition 2 fixes only the endpoints of the tilt schedule, $\tau_T = 0$ and $\tau_0 = \tau$. The temperatures between these endpoints can be chosen in different ways. By Remark 2, changing this schedule affects the variance of the particle weights but does not change the final target distribution. Here, instead of using the fixed geometric schedule of Section 3.3, we choose each temperature increment

Algorithm 1 Stratified Scaling Search (S^3): verifier-guided particle search under a single decoding schedule

Input: Diffusion model p_θ, verifier f, prompt x, fixed schedule $e = (\mathcal{K}, \pi, T)$, particle count N, branching factor b, tilt schedule $0 = \tau_T \le \cdots \le \tau_0 = \tau$
Output: Best decoded output $y^\star$

Initialize: $x_T^{(i)} \leftarrow \textsc{FullMask}(x)$ and $s_i^{\mathrm{par}} \leftarrow 0$ for $i = 1, \dots, N$ ▷ all output positions masked; $\hat{h}_T \equiv 1$
for $t = T, T-1, \dots, 1$ **do**
  Determine updatable positions $U_t \subseteq \{1, \dots, L\}$ from schedule policy π ▷ $|\,U_t\,| = \mathcal{K}$
  for each particle $i \in \{1, \dots, N\}$ **do**
    for each child $j \in \{1, \dots, b\}$ **do**
      $x_{t-1}^{(i,j)} \sim p_\theta(\cdot \mid x_t^{(i)}, e)$ ▷ update only $\mathcal{K}$ positions in U_t
      if $x_{t-1}^{(i,j)}$ has no masked positions **then**
        $s_{i,j,t} \leftarrow f\left(x_{t-1}^{(i,j)}\right)$ ▷ score the realized output: $\hat{h}_0 = h_0$ exactly
      else
        $\hat{x}_0^{(i,j,t)} \leftarrow \mathrm{decode}\left(p_\theta(x_0 \mid x_{t-1}^{(i,j)})\right)$ ▷ one-step clean prediction
        $s_{i,j,t} \leftarrow f\left(\hat{x}_0^{(i,j,t)}\right)$ ▷ look-ahead verifier score
      end if
    end for
  end for
  $\tilde{w}_{i,j,t} \leftarrow \exp\{\tau_{t-1}\, s_{i,j,t} - \tau_t\, s_i^{\mathrm{par}}\}$ for all i, j ▷ incremental potential G_t, Equation (6)
  $w_{i,j,t} \leftarrow \tilde{w}_{i,j,t} \big/ \sum_{i',j'} \tilde{w}_{i',j',t}$ ▷ normalize over all $N \cdot b$ children
  $\xi_{i,j,t} \leftarrow N\, w_{i,j,t}$ for all i, j ▷ expected offspring counts
  $\{x_{t-1}^{(i)}\}_{i=1}^N, \mathcal{I} \leftarrow \textsc{SSPResample}\left(\{x_{t-1}^{(i,j)}\}, \{\xi_{i,j,t}\}, N\right)$ ▷ low-variance dependent rounding; $\mathcal{I}$ are the survivor indices
  $s^{\mathrm{par}} \leftarrow \{s_{i,j,t}\}_{\mathcal{I}}$ ▷ carry each survivor's own score along its lineage
end for
return $y^\star \leftarrow \mathrm{MajorityVote}\left(\{x_0^{(i)}\}_{i=1}^N\right)$ ▷ break ties by lowest NLL under the base model

online using the effective sample size (ESS) of the current frontier. We first define the adaptive rule, then discuss what it preserves, and finally compare it with the fixed geometric schedule. Writing $\Delta_t := \tau_{t-1} - \tau_t \ge 0$, the increments satisfy $\sum_{t=1}^T \Delta_t = \tau_0 - \tau_T = \tau$. Thus, the schedule determines how the fixed total tilt τ is distributed across the T denoising steps. All other notation follows Section 3.

C.1 The increment rule

At step t, the current temperature τ_t and the child and parent scores are already known. The only quantity we need to choose is the new increment Δ. Substituting $\tau_{t-1} = \tau_t + \Delta$ into Equation (6) gives

$$\log \tilde{w}_{i,j,t}(\Delta) \;=\; \underbrace{\tau_t\big(s_{i,j,t} - s_{i,t}^{\mathrm{par}}\big)}_{\text{offset, independent of } \Delta} \;+\; \Delta\, s_{i,j,t}, \tag{9}$$

This equation shows how the particle weights change as Δ increases. From these normalized weights, we compute the effective sample size (ESS) (Kong et al., 1994):

$$\mathrm{ESS}_t(\Delta) \;=\; \Big(\textstyle\sum_{i,j} w_{i,j,t}(\Delta)^2\Big)^{-1} \;\in\; [1,\, Nb]. \tag{10}$$

The ESS is largest, $\mathrm{ESS}_t = Nb$, when all frontier candidates have equal weights. It decreases as the weights become more concentrated and reaches 1 when a single candidate receives all the weight.

Algorithm 2 ESS-adaptive tilt increment (replaces line 15-16 of Algorithm 1)

Input: Child scores $\{s_{i,j,t}\}$ and parent scores $\{s^{\mathrm{par}}_{i,t}\}$ over the Nb frontier, current temperature τ_t, terminal temperature τ, target fraction η, tolerance ϵ, iteration cap m, terminal flag LAST
Output: Increment Δ_t and normalized weights $\{w_{i,j,t}\}$

$\ell(\Delta) \leftarrow \big\{\tau_t(s_{i,j,t} - s^{\mathrm{par}}_{i,t}) + \Delta s_{i,j,t}\big\}_{i,j}$ ▷ Equation (9), in log-space
$\mathrm{ESS}(\Delta) \leftarrow \big(\sum_{i,j} \mathrm{softmax}(\ell(\Delta))^2_{i,j}\big)^{-1}$ ▷ Equation (10)
$\Delta_{\max} \leftarrow \max\{\tau - \tau_t,\, 0\}$ ▷ unspent budget
if LAST **then**
  $\Delta_t \leftarrow \Delta_{\max}$ ▷ force the residual so $\tau_0 = \tau$ exactly
else if $\mathrm{ESS}(\Delta_{\max}) \geq \eta\, Nb$ **then**
  $\Delta_t \leftarrow \Delta_{\max}$ ▷ full remaining budget is feasible
else if $\mathrm{ESS}(0) \leq \eta\, Nb$ **then**
  $\Delta_t \leftarrow 0$ ▷ target already missed; add no tilt
else
  $(\mathrm{lo}, \mathrm{hi}) \leftarrow (0, \Delta_{\max})$ ▷ $\mathrm{ESS}(\mathrm{lo}) \geq \eta Nb > \mathrm{ESS}(\mathrm{hi})$
  for $r = 1, \ldots, m$ **while** $\mathrm{hi} - \mathrm{lo} > \epsilon$ **do**
    $\mathrm{mid} \leftarrow \frac{1}{2}(\mathrm{lo} + \mathrm{hi})$
    if $\mathrm{ESS}(\mathrm{mid}) \geq \eta Nb$ **then** $\mathrm{lo} \leftarrow \mathrm{mid}$ **else** $\mathrm{hi} \leftarrow \mathrm{mid}$
  end for
  $\Delta_t \leftarrow \mathrm{lo}$ ▷ largest feasible increment found by bisection
end if
return Δ_t, $\{w_{i,j,t}\} \leftarrow \mathrm{softmax}\big(\ell(\Delta_t)\big)$

Lemma 1 (Monotone weight spread)**.** *For fixed τ_t and verifier scores, increasing Δ gives more weight to higher-scoring candidates. As a result, the weighted average score does not decrease, while $\mathrm{ESS}_t(\Delta)$ does not increase. These changes are strict unless all look-ahead scores are equal.*

Proof From Equation (9), $\partial_\Delta \mathbb{E}_{w(\Delta)}[s] = \mathrm{Var}_{w(\Delta)}(s) \geq 0$. Therefore, increasing Δ shifts more probability toward candidates with larger scores. This makes the weight distribution more concentrated, so $\sum_{i,j} w^2_{i,j,t}$ increases and ESS_t decreases. If all scores are equal, changing Δ does not change the weights or the ESS.

A larger Δ applies stronger verifier guidance, but it also makes the particle weights more concentrated. We therefore choose the largest increment that keeps the ESS above a chosen threshold. For a target fraction $\eta \in (0, 1]$, we set $\Delta_t = \sup\big\{\Delta \in [0,\ \tau - \tau_t] : \mathrm{ESS}_t(\Delta) \geq \eta\, Nb\big\}$ and $\tau_{t-1} = \tau_t + \Delta_t$. Here, $\tau - \tau_t$ is the remaining tilt budget, so the schedule cannot exceed the final temperature τ. The parameter η controls how much weight concentration is allowed. Smaller η allows more aggressive tilting, while larger η favors more uniform particle weights.

Algorithm 2 selects Δ_t by one-dimensional bisection: if the full remaining tilt budget satisfies the ESS constraint, it is used entirely; if the constraint is already violated at $\Delta = 0$, no additional tilt is applied; otherwise, bisection returns the largest feasible increment. Any residual tilt budget at the final step $t = 1$ is assigned to Δ_1, ensuring $\tau_0 = \tau$ and preserving the telescoping property in Proposition 2; no residual budget occurred in our experiments. The adaptive schedule requires no additional model or verifier evaluations, since bisection only recomputes normalized weights over the existing Nb candidates, so the overall $O(TNb)$ evaluation cost and the FLOP accounting in Table 17 remain unchanged.

C.2 WHAT THE ADAPTIVE SCHEDULE PRESERVES, AND WHAT IT DOES NOT

Proposition 3 (Adaptive schedules keep the terminal target)**.** *Let $\{\Delta_t\}$ be any non-negative increments, possibly depending on the realized particle system, with $\sum_{t=1}^{T} \Delta_t = \tau$ and $\tau_T = 0$. Then Equation (7) holds pathwise:*

$$\prod_{t=1}^{T} G_t(x_t, x_{t-1}) = e^{\tau f(x_0)}$$

along every lineage, for every T and every block length $\mathcal{K}$.

Table 2: **Fixed geometric vs. ESS-adaptive schedules.** Independent ablation runs; $\Delta = \mathrm{ESS}-\mathrm{Fixed}$, n is the paired sample size, and p is the exact McNemar test.

Verifier	Dataset	(N, b)	$B{=}Nb$	Fixed schedule	ESS-adaptive	Δ	n	p
Composite	MATH-500	$(4, 2)$	8	28.60	**29.20**	$+0.60$	500	0.798
	MATH-500	$(8, 2)$	16	**30.00**	28.40	-1.60	500	0.382
	MATH-500	$(8, 4)$	32	**29.40**	28.40	-1.00	500	0.640
	GSM8K	$(4, 2)$	8	**70.36**	68.76	-1.59	1319	0.160
	GSM8K	$(8, 2)$	16	69.32	**70.68**	$+1.37$	805	0.400
	ARC-C	$(4, 2)$	8	**82.25**	81.66	-0.60	1172	0.641
	TruthfulQA	$(4, 2)$	8	**52.34**	51.46	-0.88	684	0.685
PRM	MATH-500	$(4, 2)$	8	**30.80**	28.80	-2.00	500	0.282
	MATH-500	$(8, 2)$	16	**32.40**	28.40	-4.00	500	0.027
	MATH-500	$(8, 4)$	32	**34.20**	29.20	-5.00	500	0.005
	GSM8K	$(4, 2)$	8	**74.47**	69.85	-4.62	1038	< 0.001

Proposition 2 depends only on the two endpoints, $\hat{h}_T \equiv 1$ and $\hat{h}_0 = e^{\tau f(x_0)}$. It does not depend on how the intermediate temperatures are chosen. Therefore, the temperatures may be selected adaptively from the current particle system without changing the final telescoping identity, as long as the total increment remains τ.

However, three practical qualifications are important. First, because Δ_t is selected from the current particle frontier, the intermediate targets are data-dependent; at finite N, we therefore do not claim exact unbiasedness of intermediate normalizing-constant estimates, but instead rely on standard consistency and central-limit results for adaptive SMC methods (Del Moral et al., 2012; Beskos et al., 2016). This issue does not arise for the fixed geometric schedule. Second, SSP produces negatively associated rather than independent offspring counts while preserving the desired allocation in expectation, $\mathbb{E}[n_{i,j,t}] = N w_{i,j,t}$, so results for negatively associated resampling are more appropriate than standard independent-resampling analyses (Gerber et al., 2019). Third, telescoping applies along individual realized lineages: although weights are reset after each resampling step, Propositions 2 and 3 characterize the cumulative potential along each lineage, whereas the final majority-vote output over the N survivors is not itself a sample from $\tilde{p}_0$ (Appendix E.5). The ESS rule and SSP play complementary roles: ESS limits weight concentration, while SSP converts the resulting weights into low-variance integer offspring counts; consequently, SSP preserves diversity only when the input weights are sufficiently dispersed and will correctly replicate a dominant particle when its weight is highly concentrated.

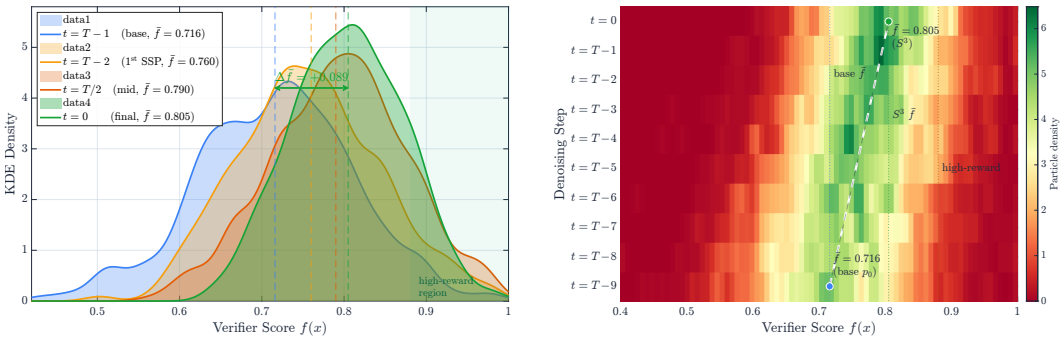
C.3 Accuracy with and without ESS adaptation

Table 2 compares the fixed geometric and ESS-adaptive schedules under matched settings; they differ only in how the total tilt is distributed across denoising steps. By Proposition 3, both preserve the same ideal terminal target, so differences reflect finite-particle effects and tilt timing. For compute efficiency, two GSM8K configurations use fixed-seed random subsets, with both schedules evaluated on the same questions; random sampling preserves an unbiased estimate of Δ, while pairing controls question-set difficulty. The full-set results lead to the same conclusions (GSM8K: $p{=}0.16$; all PRM comparisons: $\Delta{<}0$). Overall, ESS adaptation does not consistently improve accuracy: with the composite verifier, changes range from -1.60 to $+1.37$ points and none is significant ($p \geq 0.16$). With the PRM, it underperforms by 2.00, 4.00, and 5.00 points on MATH-500 at $B = 8$, 16, and 32, respectively, and by 4.62 points on GSM8K.

Table 3 further shows that, at $(N, b) = (8, 2)$, the composite schedules are similar across most MATH-500 difficulty levels except level 4, where ESS is substantially worse; with the PRM, ESS underperforms at every level. This likely reflects a limitation of ESS as a scheduling criterion: it controls weight concentration rather than verifier reliability and can allocate tilt while intermediate scores are still weak. The fixed geometric schedule instead delays stronger guidance to later denoising stages, motivating delayed ESS adaptation as a natural extension.

Table 3: **MATH-500 accuracy (%) by difficulty level** at N=8, b=2, and η=0.5; n is the number of questions.

Level	n	Composite			PRM		
		Fix.	ESS	Δ	Fix.	ESS	Δ
1	43	72.09	72.09	+0.00	72.09	69.77	−2.33
2	90	54.44	55.56	+1.11	58.89	52.22	−6.67
3	105	30.48	31.43	+0.95	36.19	32.38	−3.81
4	128	22.66	15.62	−7.03	22.66	17.19	−5.47
5	134	6.72	5.97	−0.75	8.21	6.72	−1.49
All	500	30.00	28.40	−1.60	32.40	28.40	−4.00



(a) KDE of verifier scores across denoising steps.

(b) Particle score density over denoising steps.

Figure 6: **Distributional shift of verifier scores** under S^3 on MATH-500.

D Distributional Shift of Verifier Scores

Figure 6 illustrates the distributional shift of verifier scores across denoising steps under S^3 search on MATH-500. Sequential resampling progressively concentrates particles in higher-reward regions compared to direct sampling from p_0. The KDE plot (Figure 6a) shows the score distribution shifting rightward across steps, while the heatmap (Figure 6b) shows the particle score density concentrating toward higher-reward regions as denoising proceeds, confirming that the expand–score–resample updates progressively move the particle population away from the unguided base path measure p and toward the target path measure $\tilde{p}$, whose terminal marginal is the reward-tilted distribution in Equation (2).

E Diagnostic Validation of the S^3 Mechanism

Beyond downstream accuracy, we use five controlled diagnostics to validate the key assumptions behind S^3. Unless noted otherwise, results come from a single instrumented S^3 pass on MATH-500 using Table 11 settings with N=8, reusing search-time logs with only a few additional roll-outs. The Gibbs diagnostic (Appendix E.3) is an exact standalone experiment on an enumerable problem. We compare the default composite verifier (S^3) with the PRM variant (S^3+PRM; Appendix H.7) to separate effects of the search procedure from those of the verifier.

Look-ahead scores become more predictive as denoising progresses but weaken at very fine and coarse block lengths (Appendix E.1); SSP preserves particle lineages with low resampling variance (Appendix E.2); S^3 more closely matches the Gibbs target than direct sampling, BoK, and FK-Steering on an enumerable diffusion (Appendix E.3); logged incremental potentials telescope to the terminal reward tilt up to floating-point precision, validating Proposition 2 in the implementation (Appendix E.4); and majority voting matches (within 0.5 points) or exceeds arg-max selection on every logged benchmark and is decisive on the multiple-choice ones, while FK-Steering's deficit persists under the same rule (Appendix E.5). Appendix C further shows that replacing the fixed geometric schedule with ESS-adaptive increments preserves the same terminal target but changes

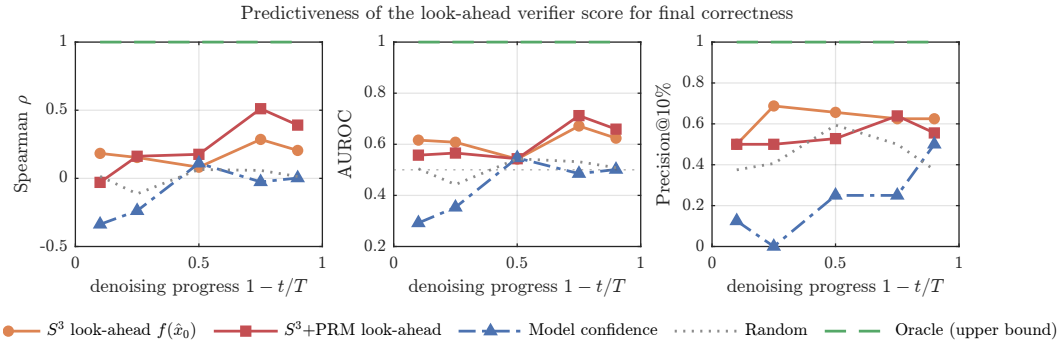


Figure 7: **Predictiveness of the look-ahead score for final correctness.** Spearman ρ, AUROC, and precision@10% of $s_{i,j,t}$ vs. denoising progress $1 - t/T$, with model-confidence, random, and oracle reference series.

accuracy by -5.00 to $+1.37$ points (Table 2), confirming that finite-particle performance depends on where the tilt is allocated.

Notation. We follow Section 3: $f{:}\mathcal{X}^L{\to}[0,1]$ is the verifier; $x_{t-1}^{(i,j)}$ is child j of particle i at step t; $\hat{x}_0^{(i,j,t)}$ is its one-step clean prediction; and $s_{i,j,t} = f(\hat{x}_0^{(i,j,t)})$ is the corresponding look-ahead score. The exact and approximate backward information functions are h_t and $\hat{h}_t$ (Equation (5)); $s_{i,t}^{\mathrm{par}}$ is the parent score; G_t is the incremental potential (Equation (6)); $w_{i,j,t}$ and $\xi_{i,j,t}{=}Nw_{i,j,t}$ are the normalized weight and expected offspring count; and $\tilde{p}_0 \propto p_0 e^{\tau f}$ is the reward-tilted Gibbs target (Equation (2)). The symbols $N, b, \mathcal{K}, T, \{\tau_t\}$, and τ retain their definitions. Appendix C additionally uses $\Delta_t = \tau_{t-1} - \tau_t$, ESS_t for the effective sample size of the Nb frontier, and $\eta \in (0,1]$ for the target ESS fraction. We denote normalized denoising time by t/T and progress by $1 - t/T$, from 0 at the fully masked state to 1 at the clean output, as used in Figure 7.

E.1 Does the look-ahead score predict final correctness? (validating $\hat{h}_t \approx h_t$)

The practical approximation in S^3 relies on the look-ahead score $s_{i,j,t} = f(\hat{x}_0^{(i,j,t)})$ providing useful information about the intractable backward term h_{t-1} in Equation (5). To evaluate this, at $t/T \in \{0.9, 0.75, 0.5, 0.25, 0.1\}$ we record candidate look-ahead scores and run R independent continuations from each state to estimate its probability of final correctness. At each stage, we measure Spearman correlation ρ, AUROC, and precision among the top-10% scored candidates, with model confidence, a random score, and an oracle based on terminal correctness as references.

Figure 7 shows that the look-ahead signal becomes more informative as denoising progresses: AUROC rises from ≈ 0.50 at $t/T{=}0.9$ to ≈ 0.68 near termination, while Spearman correlation increases from ≈ 0 to ≈ 0.25–0.31. By Proposition 2, this affects the efficiency of the particle approximation rather than the terminal target: early scores are nearly uninformative and receive small tilt increments, keeping $G_t \approx 1$, whereas later predictions provide a stronger signal for resampling. The same pattern appears across block lengths $\mathcal{K}$, with predictiveness highest at intermediate values and weaker at both extremes; fine blocks evaluate highly masked states, while coarse blocks provide too few look-ahead updates to redirect many committed positions. Appendix F uses this behavior to explain the block-length crossings in Section 4.2. Consistently, Figure 8 shows that selecting the highest-scored child incurs low regret (≤ 0.13), falling to $\approx$ 0-0.07 for $t/T \leq 0.25$ and remaining below the random baseline (0.13–0.30). The complementary component-wise correlation analysis of the default composite verifier is reported in Appendix H.6.

E.2 Is SSP resampling necessary? (diversity preservation)

S^3 turns the expected offspring counts $\xi_{i,j,t} = Nw_{i,j,t}$ into integer counts using SSP dependent rounding (Section 3.3). To show that this specific choice matters, we apply seven resamplers: SSP, multinomial, systematic, stratified, residual, deterministic top-N, and no resampling to the *identical* logged frontier weights $\{w_{i,j,t}\}$, so any difference is due to the resampler alone. We compare them

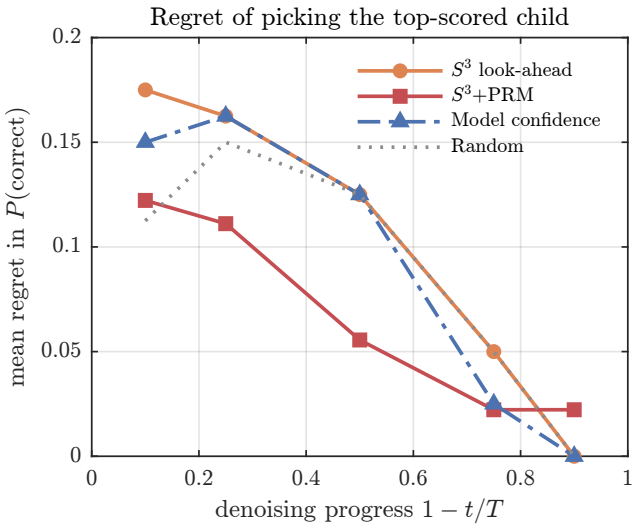


Figure 8: **Regret of trusting the top-scored child.** Mean drop in $P(\text{correct})$ from keeping only the highest-scored child instead of the best available one, across denoising progress $1 - t/T$.

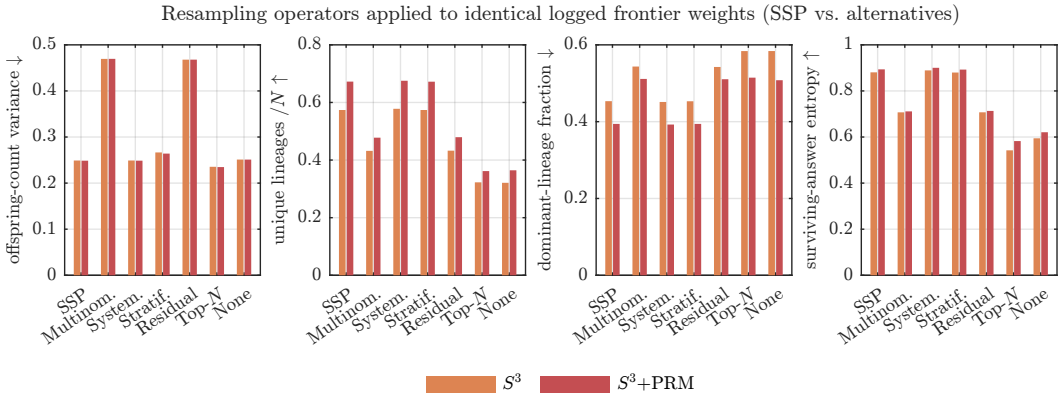


Figure 9: **Resampling operators applied to identical logged frontier weights.** Offspring-count variance, unique founder lineages $/N$, dominant-lineage fraction, and surviving-answer entropy for seven resamplers on the same weights $\{w_{i,j,t}\}$. Panel titles mark the preferred direction (↓ / ↑).

on the offspring-count variance (lower is better), the fraction of the N founder lineages that survive, the dominant-lineage fraction (diversity), and the entropy of the surviving answers.

Figure 9 shows that SSP attains the minimal offspring-count variance (0.25, matched only by systematic and stratified resampling) while retaining *all* N founder lineages (unique-lineage fraction 1.0, dominant-lineage fraction 0.125, answer entropy 1.11). Multinomial and residual resampling inflate the variance to ≈ 0.47 and collapse to $\approx 66\%$ of lineages; deterministic top-N and no-resampling keep the variance low but collapse diversity (founder fraction 0.5, answer entropy 0.77). The genealogy in Figure 10 makes the mechanism explicit over denoising: across all T=64 steps, SSP (like systematic/stratified) keeps every one of the N=8 lineages alive, multinomial decays to ≈ 5, and top-N/no-resampling collapse immediately to 4. SSP therefore combines low-variance rounding with full diversity retention, the property that Section 3.3 claims prevents mode collapse in the search tree.

E.3 Does S^3 approximate the Gibbs target? (exact validation)

The distributional-shift figure (Appendix D) shows verifier scores moving rightward but does not prove that S^3 approaches the reward-tilted target $\tilde{p}_0 \propto p_0\, e^{\tau f}$. We therefore construct a small

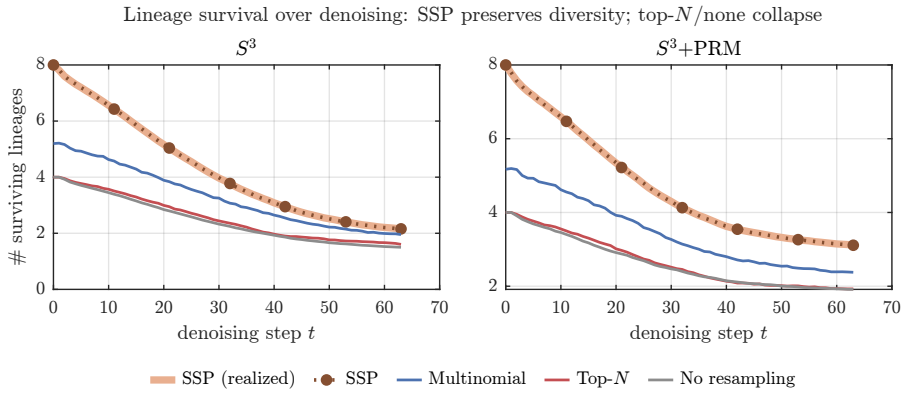


Figure 10: **Founder-lineage survival over denoising.** Number of the N=8 founder lineages still represented at each denoising step t.

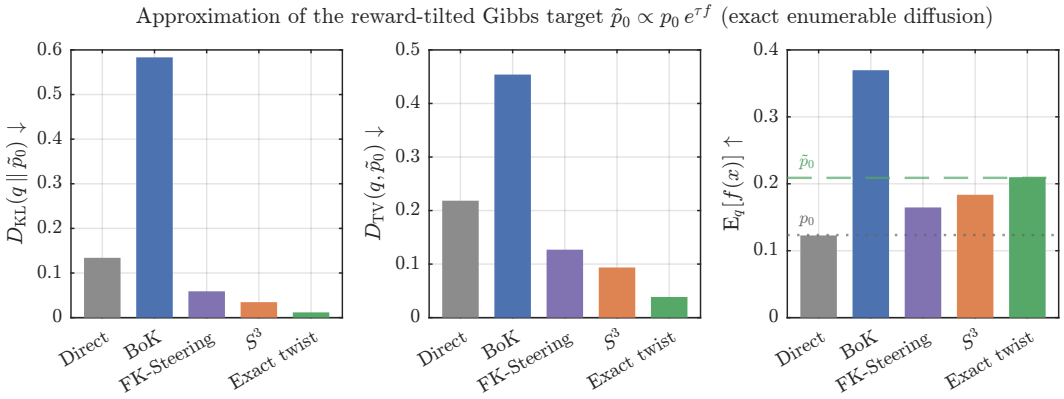


Figure 11: **Approximation of the reward-tilted Gibbs target $\tilde{p}_0 \propto p_0\, e^{\tau f}$ on an exactly enumerable diffusion.** $D_{\mathrm{KL}}(q\|\tilde{p}_0)$, $D_{\mathrm{TV}}(q,\tilde{p}_0)$, and mean reward $\mathbb{E}_q[f]$ for the five samplers. Dashed/dotted lines in the reward panel mark $\mathbb{E}[f]$ under $\tilde{p}_0$ and p_0.

masked-diffusion problem whose terminal sequences can be *fully enumerated* (small vocabulary and length, known transition probabilities, and an explicit terminal reward), so that p_0, $\tilde{p}_0$, the exact backward functions h_t, and the Doob-twisted kernel are all computable in closed form. We then draw equal-budget samples from direct sampling, BoK, FK-Steering, S^3, and exact twisted sampling, and compare each empirical distribution q to the target.

Figure 11 reports the KL divergence $D_{\mathrm{KL}}(q\|\tilde{p}_0)$, the total-variation distance $D_{\mathrm{TV}}(q,\tilde{p}_0)$, and the achieved mean reward $\mathbb{E}_q[f]$. Among the practical methods S^3 is closest to the target (D_{KL}=0.035, D_{TV}=0.094), improving over FK-Steering (0.059/0.127) and direct sampling (0.134/0.218); exact twisting recovers the target ($D_{\mathrm{KL}}\to$0.012), confirming the implementation. Notably, BoK attains the *highest* reward ($\mathbb{E}_q[f]$=0.37) yet the *worst* distributional match (D_{KL}=0.58): it over-optimizes reward and drifts far from p_0—exactly the density-quality failure mode of Section 2.1 and Remark 1. S^3 instead lifts $\mathbb{E}_q[f]$ from the base value 0.12 toward the target 0.21 while staying anchored to p_0, directly confirming the reward-tilted interpretation of Section 2.2.

E.4 Do the per-step potentials multiply to the reward tilt? (identity check)

Proposition 2 is exact algebraically, but we also verify that the implementation preserves it during resampling. For each surviving particle, we track the cumulative log-potential $\sum_t \log G_t$ along its actual lineage and compare it with the terminal tilt $\tau f(x_0)$. This requires each child to use its own parent score and that score to remain attached to the particle through SSP resampling.

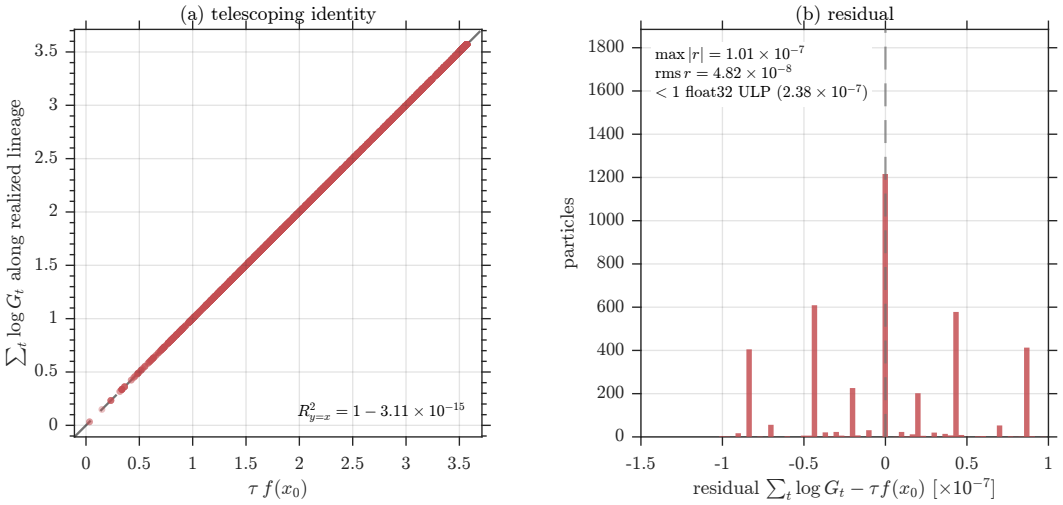


Figure 12: **Telescoping validation.** **(a)** Cumulative log potential $\sum_t \log G_t$ versus terminal tilt $\tau f(x_0)$. **(b)** Residual $\sum_t \log G_t - \tau f(x_0)$, confirming telescoping up to floating-point precision.

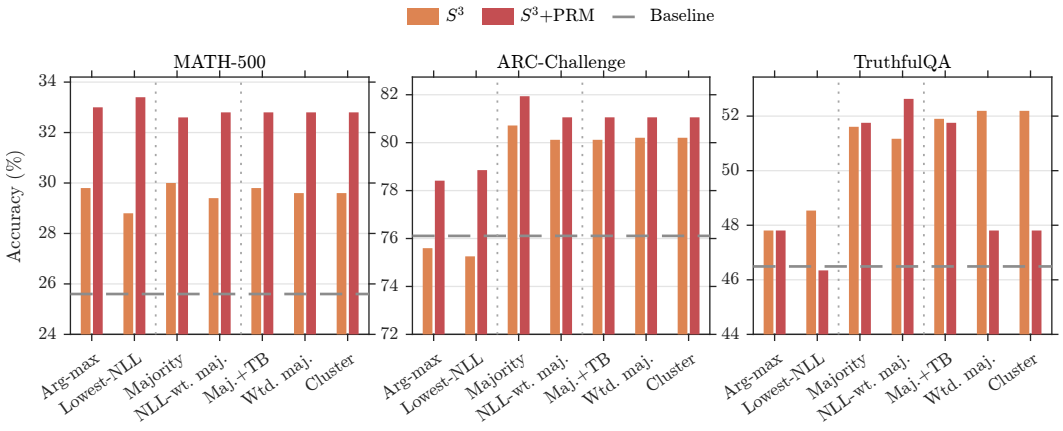


Figure 13: **Final-answer selection ablation.** Accuracy of alternative selection rules on identical final particle sets for S^3 and S^3+PRM. Dashed lines denote single-trajectory baselines.

Figure 12 confirms this behavior: panel (a) aligns the two quantities along $y = x$, while panel (b) shows residuals concentrated near zero, with maximum absolute error 1.01×10^{-7} and root-mean-square error 4.82×10^{-8}. Thus, the implemented potentials telescope to the terminal reward tilt up to floating-point precision. Importantly, this verification must be performed along individual realized lineages; averaging potentials across surviving particles mixes different ancestry paths and can create an artificial residual even when the identity is satisfied exactly.

E.5 Is majority voting responsible for robustness? (selection ablation)

Because S^3, BoK, and FK-Steering all use majority voting at output, we explicitly separate the contribution of *trajectory search* from that of *final-answer selection*. On identical logged S^3 particle sets (N=8, b=2, B=16), we compare single-particle selection (verifier arg-max and lowest NLL) against several consensus rules, including unweighted and verifier-weighted majority voting. We additionally perturb verifier scores with zero-mean Gaussian noise to test whether the conclusions depend on precise verifier calibration. Selection rules are compared only on identical particles; S^3+PRM is reported where matching logs are available.

Control 1: FK-Steering with matched majority voting. FK-Steering is not penalized by a weaker output rule: the reported FK results already use the same majority vote as S^3. Nevertheless, FK

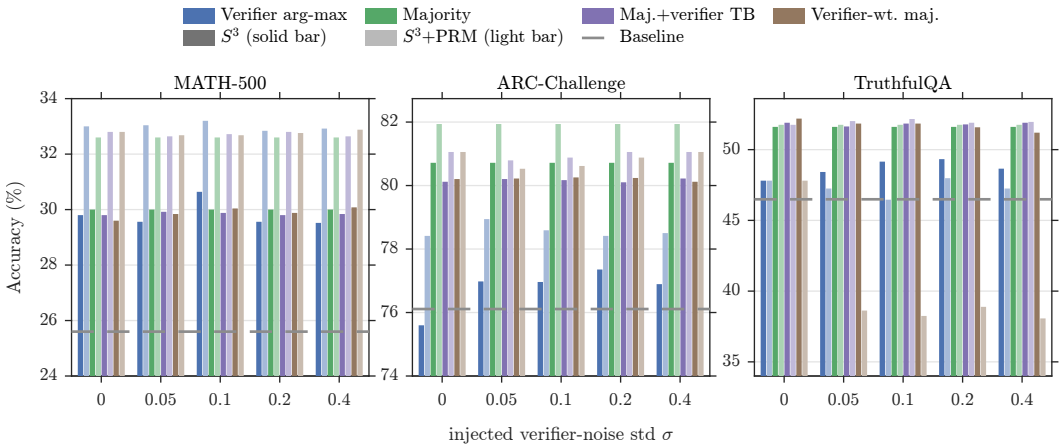


Figure 14: **Selection robustness to verifier noise.** Accuracy under increasing Gaussian perturbations to verifier scores for S^3 and S^3+PRM. Dashed lines denote single-trajectory baselines.

Table 4: **Matched output-selection controls at B=16.** Accuracy differences for FK-Steering with majority voting and for S^3 majority voting relative to verifier arg-max.

Dataset	FK+Maj. − Base	S^3 Maj. − S^3 Arg-max
MATH-500	−3.20	+0.20
GSM8K	−13.86	−0.50
ARC-C	−3.71	+5.12
TruthfulQA	+1.31	+3.80

remains below single-trajectory decoding on three of the four benchmarks at the matched budget, with GSM8K providing the clearest anchor (54.30% vs. 68.16%). Even selecting FK's best logged hyperparameter configuration does not remove this deficit on these benchmarks. Thus, we do not use FK's below-baseline performance itself as evidence for S^3; rather, the matched-output control shows that the difference cannot be explained by final-answer selection alone.

Control 2: S^3 without majority voting. Replacing majority voting by verifier arg-max on the *same* S^3 particles reveals a task-dependent effect. On MATH-500 and GSM8K the difference is at most 0.5 points, so the improvement there is not driven by consensus. In contrast, majority voting contributes materially on ARC-Challenge and TruthfulQA, improving accuracy by roughly 4–5 points. The same qualitative pattern appears with the PRM. We therefore do not attribute the entire gain of S^3 to trajectory search: consensus is an important component on multiple-choice tasks, whereas trajectory quality is the dominant factor on the numeric benchmarks.

Search beyond output-level consensus. BoK provides a complementary control because it applies majority voting to independent complete samples without modifying generation. It explains a substantial fraction of the improvement over single-trajectory decoding on ARC-Challenge, TruthfulQA, and GSM8K, but substantially less on MATH-500. Crucially, under the *same* majority-vote output rule, S^3 still improves over BoK by approximately 2.5–3.2 points on MATH-500, ARC-Challenge, and TruthfulQA, while no measurable advantage is observed on GSM8K. This separates the two effects: consensus provides robustness at selection time, while trajectory search contributes additional gains when intermediate predictions provide a useful steering signal.

Robustness to verifier noise. The noise experiment further distinguishes robust aggregation from dependence on precise verifier scores. Rules that completely ignore the verifier are invariant to injected noise by construction and therefore are not, by themselves, evidence of robustness. Among rules that do use verifier information, majority voting with verifier tie-breaking varies by at most 0.41 points across the tested noise range, compared with up to 1.76 points for verifier arg-max. More aggressively score-weighted consensus can degrade when the verifier is poorly calibrated, particularly for out-of-domain PRM scoring. These results motivate our default rule: use the verifier to guide

Table 5: **Budget axis.** Accuracy difference $\Delta = S^3 - \mathrm{BoK}$ (points) across the budgets of Table 11; positive values favor S^3.

	NFE				
Dataset	128	256	512	1024	2048
Composite verifier					
MATH-500	1.20	–0.60	–0.60	3.20	3.40
GSM8K	0.80	0.80	0.76	–0.27	–0.60
ARC-C	–0.27	1.30	3.07	2.50	4.86
TruthfulQA	3.16	2.17	4.68	2.99	2.90
PRM verifier					
MATH-500	–1.00	2.20	0.60	0.40	–6.80
GSM8K	0.83	–0.83	1.74	–1.59	–0.45
ARC-C	1.02	0.43	–0.60	–2.82	–5.38
TruthfulQA	1.02	1.17	1.17	1.46	1.32

trajectory allocation during denoising, but rely primarily on population consensus for final selection rather than on a single potentially noisy score.

The ablation supports a more precise interpretation of S^3: its robustness does not arise from a single mechanism. Majority voting provides an important final aggregation effect on tasks with recurring discrete answers, while verifier-guided trajectory search improves the quality of the particle population itself. The matched FK and BoK controls show that neither majority voting nor repeated sampling alone explains the full improvement. Conversely, the GSM8K result makes clear that trajectory-level steering is not universally beneficial, consistent with our broader finding that S^3 is most effective when intermediate clean predictions are sufficiently informative.

F Regime Analysis and Mechanism Ablation: When Trajectory Search Beats Terminal Sampling

This appendix provides the detailed evidence for the regime analysis in Section 4.2, with all configurations following Table 11. We compare S^3 with BoK using the signed accuracy difference $\Delta = \mathrm{acc}(S^3) - \mathrm{acc}(\mathrm{BoK})$ at matched inference budget, so $\Delta > 0$ indicates that allocating compute to trajectory-level search is more effective than spending the same diffusion-model NFE on terminal sampling. Tables 5 and 6 examine this comparison along the inference-budget and block-length axes, respectively, while Appendix H.7 analyzes the verifier axis. Together, these experiments instantiate the criterion developed in Section 4.2: the relative advantage of trajectory search is governed by how informative the one-step clean prediction is about the eventual decoded answer. Positive Δ therefore tends to occur when intermediate predictions provide a reliable steering signal, whereas the negative cases concentrate in regimes where this signal is weak or unreliable. Appendix F.4 re-prices the comparison under matched total FLOPs by explicitly charging every verifier call, Appendix F.5 quantifies which observed accuracy differences are statistically resolvable at the corresponding test set sizes, Appendix F.6 identifies when during denoising the advantage emerges, and Appendix F.7 separates the contributions of look-ahead prediction and reward tilting. Importantly, the upper block of Table 5 and all of Table 6 involve no learned reward-model calls, so matching NFE also matches the dominant model-compute cost; only the PRM block of Table 5 has unequal verifier overhead and is therefore superseded by the matched-total-compute results in Table 8.

F.1 The budget axis

Across budgets, S^3 consistently outperforms BoK on ARC-Challenge and TruthfulQA, and becomes stronger than BoK at larger budgets on MATH-500 (Table 5). On MATH-500, S^3 at the default budget already exceeds BoK at its largest budget, using $4\times$ fewer diffusion-model evaluations. The main difference is in scaling behavior: BoK can be non-monotonic because it continues sampling from the unchanged base distribution, whereas S^3 generally improves and then plateaus. This matches the structural distinction underlying Corollary 1 without requiring the experimental majority-vote rule to coincide with the theoretical BoK definition. GSM8K is the main exception, where terminal majority

Table 6: **Block-length axis.** Accuracy difference $\Delta = S^3 - \text{BoK}$ (points) across block lengths $\mathcal{K}$ at the matched default budget; positive values favor S^3.

	Block length $\mathcal{K}$					
Dataset	**2**	**4**	**8**	**16**	**32**	**64**
MATH-500	+1.10	+0.70	-0.50	+0.20	+2.20	-0.60
GSM8K	-13.74	-0.15	+0.15	+0.25	+0.40	+0.76
ARC-C	+0.66	+0.92	+0.48	+2.82	+1.08	+3.07
TruthfulQA	+10.05	+6.76	+4.57	+7.94	+4.44	+4.68

voting is already strong and intermediate steering provides limited additional benefit, allowing BoK to overtake S^3 at the largest budgets.

F.2 The block-length axis

Figure 5 evaluates different block lengths $\mathcal{K}$ at the matched default budget in Table 11. FK-Steering does not depend on $\mathcal{K}$ and is therefore shown as a block-independent reference. As summarized in Table 6, S^3 outperforms BoK at every block length on ARC-Challenge and TruthfulQA, and at four of six settings on MATH-500 and GSM8K. Performance degrades when the look-ahead signal becomes unreliable. With very small blocks, such as $\mathcal{K} = 2$ on GSM8K, intermediate states remain highly masked, making one-step clean predictions noisy; consequently, S^3 scores 54.36 versus 68.10 for BoK. With larger blocks, many positions are committed before the next verifier evaluation, reducing the opportunity for trajectory correction; on MATH-500, BoK slightly exceeds S^3 at $\mathcal{K} = 8$ and $\mathcal{K} = 64$ by only 0.50 and 0.60 points, respectively. These weaker regimes align with the low-AUROC regions identified in Appendix E.1.

F.3 The verifier axis

The lower block of Table 5 repeats the comparison after substituting a PRM into both methods. At the default budget, S^3+PRM performs better on MATH-500, GSM8K, and TruthfulQA, whereas BoK+PRM performs better on ARC-Challenge. At the largest budget, however, BoK+PRM overtakes S^3+PRM on MATH-500 (44.00 vs. 37.20) and ARC-Challenge (84.64 vs. 79.27), while S^3+PRM maintains a positive margin on TruthfulQA at every budget. As shown in Appendix H.7, the effect of PRM substitution follows the PRM's training domain more closely than the choice of search procedure. The large-budget crossings are explained by the verifier–state mismatch analyzed there: S^3+PRM evaluates the PRM TNb times on partially resolved states, whereas BoK+PRM evaluates it only K times on completed outputs. Trajectory-level search is therefore more exposed to errors from a PRM that is less reliable on intermediate states. FK-Steering+PRM, meanwhile, closely follows FK-Steering, suggesting that its main limitation arises from resampling rather than from verifier quality or output selection (Appendix E.5). The PRM variants also differ substantially in verifier cost, so matching diffusion-model NFE does not imply matching total FLOPs. The PRM overhead ranges from only $+1.4\%$ for BoK+PRM to $+87.4\%$ for S^3+PRM (Table 17, derived in Appendix I.2). We therefore interpret the NFE-matched PRM results as accuracy-oriented controls rather than equal-total-compute comparisons. Appendix F.4 completes this comparison by re-pricing all configurations on a matched-total-FLOP axis with every verifier call included. Under this accounting, the crossings occur earlier, and BoK+PRM outperforms S^3+PRM on MATH-500 and ARC-Challenge at every matched budget. We report this as an empirical boundary of trajectory-level search rather than a tuning artifact (Appendix F.4), while Appendix F.5 quantifies which differences of these magnitudes are resolvable given the corresponding test-set sizes.

F.4 The PRM variants at matched total compute

Table 5 matches methods on diffusion-model NFE. That is the correct axis for the three training-free configurations, which issue no verifier-model call and therefore consume identical FLOPs at identical NFE, but it is the wrong axis for the "+PRM" variants, which differ by nearly two orders of magnitude in how often they query the reward model. This subsection repeats the comparison on total FLOPs

Table 7: **Total inference cost (PFLOPs per question)** across the budget grid, with all verifier calls charged. The training-free row is shared by S^3, BoK, and FK-Steering. The rightmost column gives the multiplier over the training-free cost at the same B, i.e. the horizontal shift each curve undergoes between Figure 3 and Figure 4.

Configuration	**PRM calls per question**	**B=2**	**4**	**8**	**16**	**32**	**Shift**
S^3 / BoK / FK (training-free)	0	0.262	0.524	1.049	2.097	4.194	1.00×
BoK + PRM	B	0.266	0.531	1.063	2.126	4.252	1.01×
FK-Steering + PRM	$8B$	0.291	0.582	1.163	2.327	4.653	1.11×
S^3 + PRM	$64B$	0.491	0.983	1.966	3.932	7.864	1.88×

Table 8: **PRM comparison at matched total compute.** Δ_{FLOP} compares S^3+PRM at budget B with BoK+PRM at $2B$; positive values favor S^3+PRM. Bracketed values show the corresponding NFE-matched differences from Table 5.

Dataset **Budget pair** $(B \mid 2B)$	**≈ 0.5** 2 | 4	**≈ 1.0** 4 | 8	**≈ 2.0** 8 | 16	**≈ 4.0** 16 | 32	**Sign**
MATH-500	−3.00	−0.60	−1.60	−4.60	0/4
	[−1.00]	[+2.20]	[+0.60]	[+0.40]	[3/4]
GSM8K	+0.15	−2.96	+1.06	−2.43	2/4
	[+0.83]	[−0.83]	[+1.74]	[−1.59]	[2/4]
ARC-C	−1.02	−1.88	−3.58	−5.63	0/4
	[+1.02]	[+0.43]	[−0.60]	[−2.82]	[2/4]
TruthfulQA	±0.00	+0.15	+0.73	+1.17	3/4
	[+1.02]	[+1.17]	[+1.17]	[+1.46]	[4/4]

with every verifier call charged, using the cost model of Appendix I.2: $2PL$ FLOPs per forward pass with $P = 8$B for the DLM, $P = 7$B for the PRM, and $L = 128$.

The shift is one budget doubling. S^3+PRM costs $1.88\times$ the training-free run at the same B, while BoK+PRM costs $1.01\times$. Equalizing total FLOPs therefore compares S^3+PRM at per-step budget B against BoK+PRM at $2B$, and the residual mismatch of this pairing gives BoK+PRM about 8% *more* total compute than S^3+PRM ($2.03\times$ versus $1.88\times$ the training-free cost). Every entry of Table 8 is thus conservative against S^3+PRM, which strengthens the positive entries and leaves the negative ones to be read as upper bounds on the gap.

What changes, and what does not. Re-pricing leaves the primary training-free results unchanged: the corresponding curves in Figure 4 are identical, so the conclusions in Appendices F.1 and F.2 already hold under matched total FLOPs. The main change is in the PRM comparison. On MATH-500, S^3+PRM is ahead at three of four budgets when matched only in NFE, but falls behind BoK+PRM at all four matched-FLOP points. On ARC-Challenge, the NFE-matched comparison shows BoK+PRM overtaking only at the two largest budgets, whereas under matched total FLOPs BoK+PRM leads throughout. TruthfulQA does not change sign: S^3+PRM remains ahead on both axes, while GSM8K continues to alternate between the two methods. Considering all seven configurations as a Pareto frontier in Figure 4, training-free S^3 remains on the frontier for TruthfulQA at every budget and for ARC-Challenge from $B = 4$ up to 4.19 PFLOPs. At the next point, BoK+PRM exceeds it by only 0.51 points while using 4.25 PFLOPs and an additional 7B reward model. On MATH-500, BoK+PRM forms the frontier above 0.27 PFLOPs, whereas on GSM8K the frontier alternates between BoK+PRM and S^3+PRM.

Why the direction is the informative part. This change has a clear mechanistic interpretation and identifies a practical boundary of trajectory-level search rather than a tuning artifact. A PRM is itself a predictor of final correctness, and its usefulness depends on the state on which it is evaluated. As discussed in Appendix E.1, such signals are most reliable on complete, sequentially written reasoning, which matches the PRM's training regime, and less reliable on one-step reconstructions

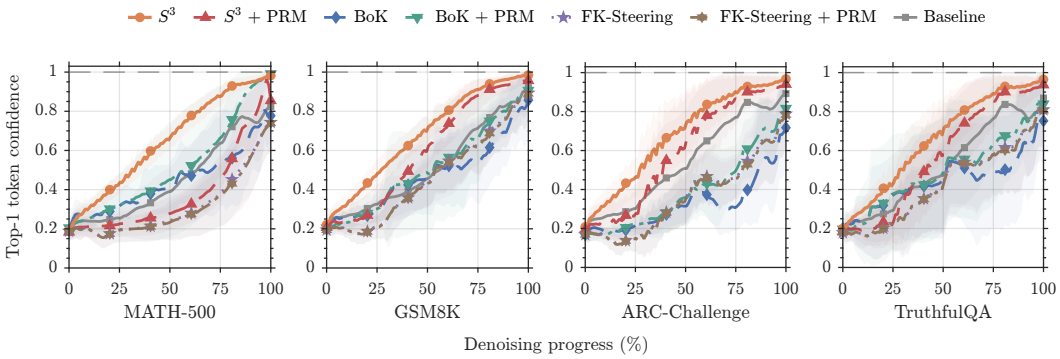


Figure 15: Top-1 token confidence over denoising progress.

from partially masked states, which is the signal S^3 must use during denoising. BoK+PRM therefore evaluates the reward model only K times on completed outputs, while S^3+PRM evaluates it TNb times on intermediate states and incurs roughly $64\times$ as many PRM calls. These two effects, lower intermediate-state reliability and much higher verifier cost, act in the same direction. Consequently, the matched-FLOP crossing appears most clearly on benchmarks where the PRM itself is strongest: in those regimes, the scorer contributes most of the gain, and terminal selection is the cheaper way to exploit it. The opposite pattern is also informative. On TruthfulQA, where the PRM is out of domain and the composite verifier is more effective, terminal PRM selection does not close the gap, and training-free S^3 remains the strongest of all seven configurations at every budget. Overall, trajectory-level allocation is most effective when intermediate signals are both inexpensive and sufficiently informative, whereas output-level selection is preferable when the verifier is costly and reliable mainly on complete generations. A verifier calibrated for partially denoised states, or cheap enough to query at every denoising step, would shift this boundary and represents a direct extension of S^3.

F.5 What these differences resolve

Table 9 provides a reference scale for interpreting the accuracy differences reported in this paper. Because all methods are evaluated on the same test examples, paired comparisons are the most relevant. For a paired difference, the 95% half-width is $1.96\sqrt{\pi_d/n}$, where π_d is the fraction of examples on which the two methods disagree. At $\pi_d = 0.10$, differences below 1.71 points on GSM8K, 1.81 on ARC-Challenge, 2.37 on TruthfulQA, and 2.77 on MATH-500 should be interpreted cautiously; at $\pi_d = 0.20$, these reference thresholds increase to 2.41–3.92 points. These values provide the appropriate scale for reading Tables 5 and 6, as discussed in Section 4.2.

Training-free comparison. The strongest evidence is not any single cell, but the consistency of the pattern across budgets and block lengths. In the upper block of Table 5, several differences exceed the paired reference scale: on MATH-500, the two largest-budget gains are $+3.20$ and $+3.40$ points; on ARC-Challenge, the gains reach $+3.07$, $+2.50$, and $+4.86$ points; and TruthfulQA shows gains above the reference scale at most budgets. By contrast, every budget-level difference on GSM8K is below one point. The same qualitative ordering persists across the block-length sweep in Table 6: across the ten distinct budget/block settings (the default point is shared), S^3 exceeds BoK in $10/10$ settings on TruthfulQA, $9/10$ on ARC-Challenge, $7/10$ on MATH-500, and $6/10$ on GSM8K. We treat these counts as a descriptive measure of consistency rather than independent statistical trials. Importantly, this ordering agrees with the look-ahead predictiveness analysis in Appendix E.1: the largest and most consistent gains occur on benchmarks where intermediate predictions provide the strongest signal about final correctness.

PRM comparison. The matched-compute results in Table 8 show a similarly consistent directional pattern. BoK+PRM outperforms S^3+PRM at all four matched-compute points on MATH-500 and ARC-Challenge, whereas S^3+PRM is ahead at three of four points on TruthfulQA, with one tie; GSM8K remains mixed. Some of the larger differences, such as the -5.63-point gap on ARC-Challenge, exceed the reference half-widths in Table 9. However, because the exact paired uncertainty depends on the observed per-example discordance rate, we use these results primarily to establish the

Table 9: **Benchmark resolution.** Standard error (SE) and 95% half-widths in accuracy points. Paired bounds are shown for discordance rates $\pi_d = 0.10$ and 0.20.

Dataset	n	SE	Unpaired 95%	Paired 95%	
				$\pi_d = 0.10$	$\pi_d = 0.20$
MATH-500	500	2.03	5.64	2.77	3.92
GSM8K	1319	1.26	3.49	1.71	2.41
ARC-C	1172	1.12	3.09	1.81	2.56
TruthfulQA	684	1.91	5.29	2.37	3.35

Table 10: Ablation of look-ahead and Gibbs tilting in S^3. Accuracy (%) at the default operating point of Table 11.

Method	Look-ahead	Tilt	MATH 500	GSM8K	ARC-C	Truthful QA
Baseline	×	×	25.60	68.16	76.11	46.49
BoK	×	×	**29.80**	69.60	<u>79.18</u>	47.66
Tilting only	×	✓	26.40	69.54	76.30	<u>48.10</u>
Look-ahead only	✓	×	26.20	<u>69.69</u>	76.71	46.49
S^3 (full)	✓	✓	<u>29.20</u>	**70.36**	**82.25**	**52.34**

direction and consistency of the effect rather than to make formal significance claims. The pattern is also consistent with the mechanism discussed above: PRMs are most effective on complete reasoning, which favors terminal selection, whereas S^3 relies on repeated evaluation of partially denoised states.

Future statistical validation. Table 9 provides a reference scale for interpreting the observed accuracy differences using benchmark-level uncertainty and paired reference bounds. Future work will extend this analysis to the primary S^3-BoK and S^3-FK-Steering comparisons by retaining per-example predictions and reporting observed discordance rates, exact McNemar tests, and comparison-specific confidence intervals.

F.6 Where the gain appears during denoising

Figure 15 shows how top-1 token confidence changes throughout denoising for all seven configurations. S^3 improves confidence early and maintains this advantage through later steps, while BoK stays close to the single-trajectory baseline because its samples are generated independently and compared only after decoding is complete. This indicates that the gains of S^3 come from steering partial trajectories during generation, rather than from final-output selection alone. This behavior is consistent with the path-space formulation in Section 3 and also explains why the budget- and block-length results depend on the quality of the one-step clean prediction.

F.7 Mechanism ablation: tilted reward vs. look-ahead

Table 10 isolates the two mechanisms of S^3 at the matched budget $K = N \cdot b$, comparing single-trajectory decoding, BoK, look-ahead with uniform pruning, and terminal-only tilting. Neither mechanism is sufficient alone: on MATH-500, look-ahead only (26.20) and tilting only (26.40) improve only slightly over the baseline (25.60), whereas their joint use in S^3 reaches 29.20. The same complementary pattern appears on ARC-Challenge and TruthfulQA, where S^3 exceeds both single-mechanism variants by at least 5.5 and 4.2 points, while on GSM8K the three variants overlap near the model's ceiling. Read against BoK, the table repeats the regime picture of Section 4.2 at a single operating point, and it adds one thing: the two mechanisms are complementary *regardless* of which side of that boundary a task falls on, since on MATH-500 their joint use still recovers most of the gap to BoK that neither part closes alone.

Table 11: **Evaluation configuration.** Every symbol and grid used in the paper, with defaults in bold. All results are averaged over two random seeds. Mentions of "budget", "NFE", "block length", and the "default operating point" throughout the main text and appendices refer to this table.

Symbol	Meaning	Value(s)
Fixed across all experiments		
–	model	LLaDA-8B-Instruct
L	generation length	128
T	denoising steps	64
τ	terminal tilt	3.57
Search budget (matched across methods)		
N	particles	**4**
b	branching factor	**2**
$B{=}Nb$	per-step budget	$2, 4, \mathbf{8}, 16, 32$
K	BoK / FK samples	$K{=}B$
NFE	$= TB$ (baseline: T)	$128, 256, \mathbf{512}, 1024, 2048$
Decoding schedule		
$\mathcal{K}$	block length	$2, 4, 8, 16, 32, \mathbf{64}$

These results support the twisted-SMC interpretation of Section 3: look-ahead supplies the scheduled surrogate $\hat{h}$, and the incremental potential converts it into per-step weights that accumulate along a lineage to the reward tilt of Equation (2).

G Experimental Details

Evaluation setup. Table 11 lists the model and every budget, schedule, and block-length value used in the paper; these are not restated elsewhere. We follow Lee et al. (2025) for dataset preprocessing and answer normalization. For GSM8K and MATH-500, numerical and symbolic answers are extracted using LaTeX canonicalization; for ARC-Challenge and TruthfulQA, predictions are mapped to the corresponding option sets. The evaluated test sets contain 1,319 GSM8K problems, 500 MATH-500 problems, 684 TruthfulQA questions, and 1,172 ARC-Challenge questions. All reported accuracies, latencies, and wall-clock times are averaged over two random seeds. We additionally evaluate 500 MBPP problems and 164 HumanEval problems under the same matched budget grid $B \in \{2, 4, 8, 16, 32\}$; aggregate code-generation results are reported in Table 1. At the default operating point, the geometric tilt schedule is $\tau_t = \big((1+\gamma)^{T-t} - 1\big)\tau_{\text{base}}$ with $\gamma{=}0.024$ and $\tau_{\text{base}}{=}1.0$, giving the terminal tilt parameter $\tau = \tau_0 \approx 3.57$. The parameter τ controls the overall strength of verifier guidance in S^3, while the schedule $\{\tau_t\}_{t=T}^{0}$ determines how this fixed total tilt is distributed across denoising steps. Hence, no separate inverse-temperature parameter is required in S^3; the symbol λ below denotes only the potential-strength parameter used by FK-Steering.

G.1 FK-Steering Implementation

We implement FK-Steering following the original Feynman-Kac particle controller of Singhal et al. (2025). The controller maintains K_{FK} particles, converts their rewards into Feynman-Kac potentials, normalizes the resulting weights, and periodically resamples the particle population. We retain this control procedure and replace only the task-specific model and reward components. Specifically, FK-Steering uses the same LLaDA-8B-Instruct generator, masked-denoising schedule, one-step clean prediction, and verifier as S^3. At resampling round ℓ, particle k produces a clean prediction $\hat{x}_0^{(k,\ell)}$ and receives the reward $r_\ell^{(k)} = f\big(\hat{x}_0^{(k,\ell)}\big) \in [0, 1]$. Consequently, FK-Steering and S^3 use the same model and scoring signal and differ only in their trajectory-expansion and resampling procedures.

Potential and temperature. All reported FK-Steering results use the original difference potential (`pot=diff`), $\widetilde{w}_\ell^{(k)} = \exp\big[\lambda(r_\ell^{(k)} - r_{\ell-1}^{(k)})\big]$, normalized as $w_\ell^{(k)} = \widetilde{w}_\ell^{(k)} \big/ \sum_{k'=1}^{K_{\text{FK}}} \widetilde{w}_\ell^{(k')}$. We retain this original formulation to ensure a faithful comparison with FK-Steering rather than introducing a

Table 12: FK-Steering accuracy (%) across inverse temperatures λ using the composite and PRM verifiers. All runs use the original difference potential (`pot=diff`). No matched S^3 run exists for K_{FK}=6. † denotes rows constant across λ.

			FK-Steering inverse temperature λ						Matched S^3 results		
Verifier	**Dataset**	K_{FK}	**1**	**2**	**4**	**6**	**8**	**10**	**NFE**	S^3	S^3+PRM
Composite	MATH-500	2	21.20	21.40	21.60	21.80	21.80	22.00	128	26.80	31.00
		4	21.60	21.60	21.80	21.80	22.00	22.00	256	28.20	36.20
		6	21.40	21.00	21.80	21.60	21.40	21.40	384	–	–
		8	20.00	20.20	20.60	20.80	20.80	22.40	512	29.20	37.40
	GSM8K	4	56.63	56.10	55.50	55.12	54.89	54.66	256	69.60	69.75
		6	58.53	52.62	52.92	53.22	53.60	54.13	384	–	–
		8	53.22	54.21	53.37	53.75	54.06	54.28	512	70.36	74.45
	ARC-Challenge	4	71.93	71.84	71.76	71.67	71.67	71.59	256	79.80	76.96
		6	75.17	73.04	72.70	72.87	73.04	72.95	384	–	–
		8	72.35	72.27	72.35	72.27	72.10	71.76	512	82.25	78.24
	TruthfulQA	4	47.81	47.95	48.10	48.25	48.39	48.54	256	51.00	46.20
		6	46.49	46.20	46.78	46.93	47.22	47.81	384	–	–
		8	45.03	45.61	45.76	46.64	47.51	48.39	512	52.34	47.22
PRM	MATH-500	2	20.20	20.60	21.20	21.80	22.60	23.20	128	26.80	31.00
		4	22.00	22.00	21.80	21.80	21.60	21.60	256	28.20	36.20
		8†	23.00	23.00	23.00	23.00	23.00	23.00	512	29.20	37.40
	GSM8K	8†	57.24	57.24	57.24	57.24	57.24	57.24	512	70.36	74.45
	ARC-Challenge	8†	72.01	72.01	72.01	72.01	72.01	72.01	512	82.25	78.24
	TruthfulQA	8†	41.96	41.96	41.96	41.96	41.96	41.96	512	52.34	47.22

modified baseline. The potential rewards improvements in predicted quality across successive scoring rounds, while λ controls the strength of this preference. This difference potential telescopes along a trajectory; the distinction from S^3 therefore lies in the scheduled temperature, per-step branching, and SSP-based resampling rather than telescoping itself.

Hyperparameters and compute matching. We evaluate $K_{\mathrm{FK}} \in \{2, 4, 6, 8\}$ where reported and sweep $\lambda \in \{1, 2, 4, 6, 8, 10\}$, averaging over two random seeds and using the same periodic resampling schedule across all runs. The PRM experiments use the subset of configurations reported in Table 12. Because changing λ only modifies the resampling weights, each temperature sweep is performed at a fixed inference budget. For compute matching, an FK-Steering population of size K_{FK} is compared with an S^3 search budget $B = Nb = K_{\mathrm{FK}}$. Budgets and their NFE equivalents follow Table 11.

G.2 Transfer to a Second DLM: Dream-v0-Instruct-7B

To test whether S^3 generalizes beyond LLaDA-8B-Instruct, we repeat the default operating point with the independently trained Dream-v0-Instruct-7B (Ye et al., 2025), changing only the generator. We use $\mathcal{K}$=64, N=4, and b=2 (B=Nb=8; Table 11); BoK uses eight complete samples, while FK-Steering uses its original difference potential with $\lambda \in \{1, 10\}$ (Appendix G.1). The verifier and evaluation protocol remain unchanged (Appendix H).

Table 13 shows that S^3 achieves the highest accuracy on all four benchmarks with Dream. Relative to single-trajectory decoding, gains reach $+6.40$ pp on MATH-500 and exceed $+3$ pp on GSM8K and TruthfulQA; the MATH-500 and GSM8K improvements exceed the paired reference scale even at π_d=0.20 (Table 9). ARC-Challenge is already near saturation at 84.73%, leaving less than 0.6 pp separation among all methods.

At matched compute, S^3 also consistently matches or outperforms BoK-8, with the largest margin of $+2.33$ pp on TruthfulQA; these BoK margins are interpreted as directionally consistent rather than individually significant (Appendix F.5). FK-Steering improves more consistently with Dream than

Table 13: **Transfer to Dream-v0-Instruct-7B.** Accuracy (%) with block length $\mathcal{K}$=64 and the composite verifier. S^3 uses N=4, b=2 (B=8), matched to BoK-8. FK-diff denotes FK-Steering with the difference potential of Appendix G.1. Ties for second place are both underlined.

Method	MATH 500	GSM8K	ARC-C	Truthful QA
Baseline	25.80	59.82	<u>84.73</u>	53.98
BoK-8	<u>31.80</u>	<u>61.71</u>	<u>84.73</u>	54.83
FK-diff (λ=1)	26.60	60.50	84.56	55.45
FK-diff (λ=10)	26.80	60.73	84.22	<u>56.30</u>
S^3	**32.20**	**62.93**	**84.81**	**57.16**

with LLaDA (Section 4), but remains below S^3 across all four benchmarks. Since the methods share the generator, verifier, and one-step clean predictions, this result further supports the contribution of S^3's trajectory expansion and diversity-preserving population search (Appendices E.2 and E.5). Overall, the Dream experiment provides independent-model evidence that the proposed trajectory-level compute allocation transfers beyond LLaDA at the default operating point.

H Verifier Design

S^3 consumes the verifier as a black box, so this appendix concerns the choice of f rather than the search itself. Appendices H.1–H.5 describe the **default** verifier used by S^3 and FK-Steering when reported without a "+PRM" suffix, and Appendix H.6 measures how well that verifier and its individual components rank final correctness; Appendix H.7 describes the optional PRM substitution used as a control.

The default verifier $f : \mathcal{X}^L \to [0, 1]$ must satisfy two operational requirements: it must produce a meaningful quality signal *without access to ground-truth answers*, and it must be *inexpensive enough to evaluate Nb times per denoising step* throughout the S^3 search procedure. To satisfy both constraints simultaneously, we design a lightweight **composite intrinsic verifier** that decomposes solution quality into five orthogonal dimensions derivable from the generated text alone, plus an optional task-specific constraint term. Meeting these two requirements with no trained components is what allows S^3 to be deployed on a fixed DLM with no additional model in the loop.

H.1 Composite Score Formulation

The verifier score is defined as

$$f(x) \;=\; \sum_{k=1}^{5} \alpha_k \, s_k(x) \;+\; \alpha_c \, s_{\text{constraint}}(x), \tag{11}$$

where $s_1, \ldots, s_5$ measure structural completeness, internal arithmetic consistency, answer reachability, model confidence, and non-degeneracy, respectively; $s_{\text{constraint}}$ denotes a task-specific quality signal (e.g., multiple-choice answer quality or code well-formedness), and the coefficients satisfy

$$\sum_{k=1}^{5} \alpha_k + \alpha_c = 1, \qquad \alpha_k \geq 0.$$

Each component is described in detail below. The five dimensions are intentionally orthogonal: structural completeness captures surface form, arithmetic consistency captures local numerical correctness, answer reachability captures global reasoning coherence, model confidence captures fluency, and non-degeneracy penalizes degenerate failure modes. No single component is sufficient alone; the composite aggregation makes the signal robust to the weaknesses of any individual dimension.

H.2 Verifier Weight Profiles

The coefficients $(\alpha_1, \ldots, \alpha_5, \alpha_c)$ are set according to output structure: numerical reasoning for MATH-500 and GSM8K, option selection for ARC-Challenge and TruthfulQA, and code generation

Table 14: Verifier weights for each benchmark. α_1–α_5 weight the five verifier components, while α_c weights the task-specific quality signal. Rows sum to 1.0.

Dataset	α_1	α_2	α_3	α_4	α_5	α_c
MATH-500	0.25	0.20	0.25	0.10	0.20	0.00
GSM8K	0.20	0.25	0.25	0.10	0.20	0.00
ARC-Challenge	0.20	0.00	0.00	0.15	0.20	0.45
TruthfulQA	0.20	0.00	0.00	0.15	0.20	0.45
MBPP	0.10	0.00	0.00	0.10	0.15	0.65
HumanEval	0.10	0.00	0.00	0.10	0.15	0.65

for MBPP and HumanEval. The verifier components remain fixed, while their mixing weights are chosen by task format rather than learned from labels or training data. Arithmetic consistency α_2 and answer reachability α_3 are emphasized for mathematical reasoning but set to zero for multiple-choice and code-generation tasks, where natural-language reasoning-chain signals are not informative. MBPP and HumanEval use the same verifier configuration, with no dataset-specific retuning.

For **mathematical reasoning** (GSM8K, MATH-500), arithmetic consistency α_2 and answer reachability α_3 together carry the largest combined weight (0.50 for GSM8K; 0.45 for MATH-500). MATH-500 upweights structural completeness ($\alpha_1 = 0.25$) relative to GSM8K ($\alpha_1 = 0.20$) to account for the more elaborate solution formats typical of competition mathematics, while GSM8K slightly upweights arithmetic consistency ($\alpha_2 = 0.25$) to capture the denser arithmetic chains present in grade-school word problems. The constraint term is set to zero for both, since no domain constraints are verifiable without ground truth.

For **multiple-choice tasks** (ARC-Challenge, TruthfulQA), α_2 and α_3 are zeroed out because these benchmarks require answer selection rather than numerical derivation. The task-specific term α_c weights an MC quality score based on answer extraction, reasoning cues, answer dominance, and response completeness.

For **code-generation tasks** (MBPP, HumanEval), α_2 and α_3 are likewise set to zero because natural-language reasoning-chain signals are not informative for function bodies. Both benchmarks use the same verifier configuration, with HumanEval reusing the MBPP verifier without separate tuning. Here, α_c weights a code-specific well-formedness signal rather than multiple-choice answer quality.

H.3 Component Scores

S1: Structural Completeness. $s_{\text{struct}}(x)$ rewards outputs that exhibit the surface form of well-structured solutions. For mathematical reasoning tasks, the score aggregates three sub-signals: (i) a soft-thresholded keyword match $\min(h/3, 1.0)$ over a vocabulary of 14 reasoning keywords (e.g., *step*, *therefore*, *thus*, *compute*), where h is the number of distinct keyword hits; (ii) a binary answer-delimiter check using the priority cascade `\boxed{...}` ≻ `<answer>...</answer>` ≻ `####` ≻ "answer is/=/"; and (iii) an alphanumeric density score $\min(r/0.5, 1.0)$ where r is the fraction of alphanumeric or whitespace characters. For multiple-choice tasks, the score reduces to a binary extraction check for a letter option in $\{\text{A}, \ldots, \text{H}\}$ plus a keyword match and density term.

> **Example.** For the problem *"If* $3x + 5 = 14$*, find* x*"*, the output *"Step 1: subtract 5:* $3x = 9$*. Therefore* $x = 3$*.* $\backslash$*boxed{3}"* receives $s_{\text{struct}} = 1.0$ (keywords present, boxed answer, high alphanumeric density). The degenerate output *"3"* receives $s_{\text{struct}} = 0.25$ (answer present but no reasoning structure).

Limitation. Keyword matching is purely lexical. A response of the form "step step therefore \boxed{42}" scores highly on structure despite being semantically vacuous.

S2: Internal Arithmetic Consistency. $s_{\text{consist}}(x)$ verifies explicit arithmetic equalities of the form $a \text{ op } b = c$ for $\text{op} \in \{+, -, \times, \div\}$ appearing in the generated reasoning trace, in both ASCII and LaTeX notation. A tolerance of $\max(|\hat{c}| \times 10^{-6}, 10^{-4})$ is applied to the stated result $\hat{c}$ to handle floating-point rounding. An additional chain-consistency check verifies that consecutive equality values $\{v_1, v_2, \ldots\}$ extracted via the pattern $= \langle\text{number}\rangle$ have consecutive ratios within $(10^{-3}, 10^{3})$,

penalizing outputs whose intermediate values diverge implausibly. The score is

$$s_{\text{consist}}(x) \;=\; \begin{cases} \dfrac{|\{\text{verified equalities}\}|}{|\{\text{total equalities}\}|} & \text{if any equalities detected,} \\ 0.5 & \text{if no equalities detected (neutral default).} \end{cases}$$

Example. The trace "$12 \times 4 = 48$*, then* $48 + 7 = 55$" yields $s_{\text{consist}} = 1.0$ (both equalities verified). The trace "$12 \times 4 = 50$*, then* $50 + 7 = 57$" yields $s_{\text{consist}} = 0.5$ (one of two equalities incorrect). A response containing no arithmetic expressions yields $s_{\text{consist}} = 0.5$ (neutral default, not penalized).

Limitation. The parser does not handle compound parenthesized expressions (e.g., $(a + b) \times c = d$), multi-line equations, or symbolic variables. Logical errors not expressed as explicit equalities (e.g., incorrect substitutions) are invisible to this component.

S3: Answer Reachability. $s_{\text{reach}}(x)$ checks whether the final answer can be traced back to the preceding reasoning. The final answer is extracted via the same priority cascade as s_{struct}. The reasoning prefix is defined as all text before the detected answer delimiter. For numeric answers, the check verifies the presence of the same floating-point value (to tolerance 10^{-6}) among all numbers in the reasoning prefix; for symbolic answers, case-insensitive substring matching is used. The score is

$$s_{\text{reach}}(x) \;=\; \begin{cases} 1.0 & \text{answer found and traceable to reasoning,} \\ 0.3 & \text{answer found but not traceable,} \\ 0.2 & \text{no answer found.} \end{cases}$$

Example. The output *"We compute* $6 \times 7 = 42$*. The answer is* $\backslash$*boxed{42}"* scores $s_{\text{reach}} = 1.0$. The output "$\backslash$*boxed{42}*" with no preceding reasoning scores $s_{\text{reach}} = 0.3$ (answer present but unreachable from the empty prefix).

Limitation. Reachability is checked by value occurrence, not logical derivation. A response that mentions the correct value incidentally (e.g., "this is not 42") would still receive full reachability credit.

S4: Model Confidence. When token-level log-probabilities are available, we compute

$$s_{\text{conf}}(x) \;=\; \frac{1}{|G|} \sum_{i \in G} p_\theta(x_i \mid x_{<i}),$$

where G denotes the generated token positions (excluding the prompt). Logits are clamped to $[-100, 100]$ before softmax to prevent numerical overflow, and the result is clipped to $[0, 1]$. This serves as a proxy for local fluency and coherence. When logits are unavailable (e.g., black-box APIs), the score defaults to 0.5 to avoid biasing the composite.

Limitation. Model confidence reflects the model's own distribution and not solution correctness. High-confidence outputs can still be factually wrong, particularly when the model is confidently biased toward plausible but incorrect completions. This component is therefore most informative in combination with the other four signals rather than in isolation.

S5: Non-degeneracy. $s_{\text{ndegen}}(x)$ penalizes collapsed or low-information outputs using a cascade of hard and soft thresholds. It returns 0.0 if <|endoftext|> or <|eot_id|> tokens exceed 20% of word positions, 0.05 if <|mdm_mask|> tokens exceed 15% of positions, and 0.2 for outputs shorter than 8 words. For outputs longer than 12 words, the unique bigram ratio $\rho_2 =$ |unique bigrams|/|total bigrams| is used, returning 0.05 if $\rho_2 < 0.15$ and 0.3 if $\rho_2 < 0.30$. For outputs longer than 30 words, a unique trigram ratio $\rho_3 < 0.25$ returns 0.2. Otherwise, the score is 1.0.

Example. The output *"the answer is 42 the answer is 42 the answer is 42 the answer is 42"* (16 words; 15 bigrams, 4 unique) yields $\rho_2 = 4/15 \approx 0.27 < 0.30$, returning $s_{\text{ndegen}} = 0.3$. A fully collapsed output of repeated <|endoftext|> tokens returns $s_{\text{ndegen}} = 0.0$ immediately.

Limitation. Very short outputs (fewer than 8 words) bypass the repetition checks and are penalized only by the length threshold. Outputs consisting of diverse but semantically vacuous tokens score $s_{\text{ndegen}} = 1.0$ despite being uninformative.

Table 15: Spearman correlation between verifier signals and final correctness on MATH-500. All p-values are two-sided; the correlation for s_4 is undefined because its values are constant.

Verifier signal	Spearman ρ	p-value
s_1: Structural completeness	0.1554	0.0005
s_2: Arithmetic consistency	0.0880	0.0492
s_3: Answer reachability	0.1612	0.0003
s_4: Model confidence	–	–
s_5: Non-degeneracy	0.0002	0.9964
f: Composite verifier	**0.1904**	$\mathbf{1.8 \times 10^{-5}}$

H.4 Task-Specific Constraint Term

Multiple-choice (ARC-Challenge, TruthfulQA). The constraint term is replaced by an MC reasoning quality score with four additive components: (i) extraction of a clear letter choice from $\{\mathrm{A}, \ldots, \mathrm{H}\}$ (base $+0.3$); (ii) presence of reasoning keywords such as *because*, *therefore*, *eliminat* (up to $+0.3$, soft-thresholded at 3 hits); (iii) answer dominance, i.e., the chosen letter appears at least as frequently as any other letter in the response ($+0.2$); and (iv) minimum response length of 30 characters ($+0.2$). The score is capped at 1.0.

H.5 Computational Complexity

All five base components and the multiple-choice constraint verifier run in $O(|x|)$ time with respect to the output length $|x|$: regex pattern matching, bigram and trigram computation, arithmetic expression parsing, and mean logit aggregation all require a single linear pass over the token sequence. Consequently, the total cost of evaluating $f(x)$ for a single candidate is negligible relative to a single diffusion model forward pass, which requires an $O(L^2)$ attention computation over the full sequence length L. This asymptotic gap ensures that the Nb verifier evaluations required per denoising step under S^3 do not meaningfully increase the total inference budget, making repeated evaluation practical even at large particle counts.

H.6 Verifier–Correctness Rank Correlation

This analysis evaluates the default composite verifier in Equation (11), corresponding to the PRM-free S^3 configuration. The S^3+PRM control in Appendix H.7 replaces the composite verifier f entirely; therefore, the component-wise analysis below does not apply to that setting. We assess whether each verifier component provides a useful ranking signal for final-answer correctness on MATH-500. For each example $i \in \{1, \ldots, n\}$, let $z_{k,i}$ denote the score assigned by verifier component s_k, and let $y_i \in \{0, 1\}$ indicate whether the final answer is correct. We measure the association between each component score and correctness using Spearman's rank correlation,

$$\rho_k = \mathrm{Corr}\left(\mathrm{rank}(z_k), \mathrm{rank}(y)\right),$$

with tied observations assigned their average rank. Since correctness is binary, the incorrect and correct examples form two tied-rank groups. Statistical significance is evaluated using the conventional two-sided large-sample approximation

$$t_k = \rho_k \sqrt{\frac{n-2}{1-\rho_k^2}}, \qquad t_k \sim t_{n-2},$$

with $n = 500$ and 498 degrees of freedom. Table 15 shows that the composite verifier has the strongest positive association with final correctness. This supports the use of complementary structural, arithmetic, and answer-reachability signals together, as their combination provides a more informative ranking criterion than any individual component alone. In contrast, the non-degeneracy component has essentially zero correlation because nearly all evaluated generations satisfy this criterion, leaving little variation to distinguish correct from incorrect outputs.

The model-confidence component s_4 is reported as N/A rather than omitted. In the recorded evaluation, s_4 was constant at 0.5 for every sample and therefore had zero variance. Because Spearman

Table 16: Average effective output tokens (↓), total latency (↓, seconds), and wall time (↓, seconds) for each ablation variant and dataset ($N = 4$, $b = 2$, $K = 8$, LLaDA-8B-Instruct, `block_length`=64).

Dataset	Method	Avg Eff. Tokens (↓)	Total Latency (s) (↓)	Wall Time (s) (↓)
ARC-C	Baseline	5.67	3648.21	1829.87
	BoK	5.43	28803.25	14431.86
	Look-ahead only	6.08	29450.49	14756.30
	Tilting only	5.43	28804.53	14431.20
	S^3	4.34	29100.42	14580.21
GSM8K	Baseline	3.03	3919.06	1964.63
	BoK	3.06	30938.32	15497.02
	Look-ahead only	3.08	31655.13	15855.10
	Tilting only	3.06	30942.79	15500.48
	S^3	3.07	31660.45	15858.32
MATH	Baseline	3.85	1530.82	770.71
	BoK	3.40	12131.98	6091.68
	Look-ahead only	4.40	12413.41	6231.07
	Tilting only	3.40	12128.29	6088.90
	S^3	3.43	12412.98	6230.43
TruthfulQA	Baseline	5.87	2120.41	1064.81
	BoK	3.06	16743.28	8394.48
	Look-ahead only	5.62	17119.66	8583.10
	Tilting only	3.06	16743.15	8394.40
	S^3	4.05	17110.60	8578.05

correlation is the Pearson correlation between ranked variables, a constant variable also has zero rank variance, making the correlation denominator undefined. Consequently, neither ρ nor a corresponding p-value can be computed. Importantly, this should not be interpreted as zero correlation; rather, it indicates that s_4 provided no sample-level ranking information in this evaluation and therefore could not help distinguish correct from incorrect outputs.

H.7 Optional Verifier Substitution: Process Reward Model

The composite verifier is a scoring choice, not an intrinsic part of the search. To isolate these two components and test whether the conclusions depend on a particular scorer, we replace f with Qwen2.5-Math-PRM-7B. Only the verifier call in Algorithm 1 changes: each one-step clean prediction $\hat{x}0^{(i,j,t)}$, representing the model's clean reconstruction from the current partial state, is passed to the PRM instead of Equation (11), and the resulting scalar is used directly as si, j, t. The PRM outputs a correctness probability for each reasoning step; we average these probabilities across the candidate's segmented steps, yielding a score in $[0, 1]$ without further rescaling. For S^3+PRM, expansion, weighting, SSP resampling, majority voting, and NLL tie-breaking remain unchanged; only the intermediate scorer is replaced. For BoK+PRM, the PRM scores the K completed outputs and the highest-scoring output is returned. The search hyperparameters $(N, b, \tau, \mathcal{K})$ remain fixed at their default values. FK-Steering+PRM uses the same PRM substitution during its resampling rounds.

Domain alignment determines whether the substitution helps. At the default budget, the PRM improves S^3 by $+8.20$ pp on MATH-500 and $+4.09$ pp on GSM8K, but reduces accuracy by 4.01 pp on ARC-Challenge and 5.12 pp on TruthfulQA; BoK exhibits the same direction of change. This behavior is consistent with the PRM's training domain: Qwen2.5-Math-PRM-7B is designed for mathematical reasoning rather than general-domain answer ranking. A second mismatch is specific to diffusion decoding. Standard PRMs are trained to evaluate clean, sequentially completed reasoning steps, whereas S^3 must score intermediate states that are still being resolved. Using the one-step clean prediction instead of the raw masked state reduces this mismatch, but does not eliminate it. The PRM control therefore demonstrates that the S^3 mechanism can operate with an external learned scorer, not that such a scorer is required. The default composite verifier remains preferable on the

Table 17: Compute cost per inference strategy at the default operating point of Table 11. PRM: Qwen2.5-Math-PRM-7B.

Method	Verifier	NFE	PRM Calls	Model (PFLOP)	PRM (PFLOP)	Total (PFLOP)	PRM Overhead
Baseline	–	64	0	0.131	0.000	0.131	–
BoK	–	512	0	1.049	0.000	1.049	–
S^3	Intrinsic	512	0	1.049	0.000	1.049	–
FK-Steering	Intrinsic	512	0	1.049	0.000	1.049	–
BoK + PRM	Qwen-PRM-7B	512	8	1.049	0.014	1.063	+1.4%
S^3 + PRM	Qwen-PRM-7B	512	512	1.049	0.918	1.966	+87.4%
FK-Steering + PRM	Qwen-PRM-7B	512	64	1.049	0.115	1.163	+10.9%

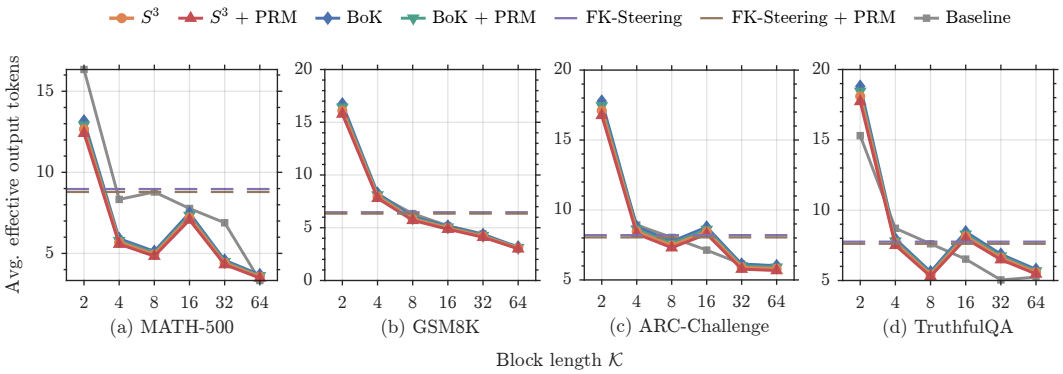


Figure 16: Average effective output tokens across block lengths. Dashed lines denote the block-independent FK-Steering variants.

two general-domain benchmarks because it is explicitly defined on partial states and requires neither training nor labels.

The substitution also changes the computational regime. It requires a second 7B model and evaluates every frontier candidate at every denoising step, with only the parent score cached. The resulting call counts and FLOP overhead are quantified in Appendix I.2, making S^3+PRM substantially more expensive than the training-free default.

Matched-compute results clarify the operating regime. After accounting for PRM cost in Appendix F.4, BoK+PRM outperforms S^3+PRM at every matched-compute point on MATH-500 and ARC-Challenge. This result is informative rather than contradictory: trajectory-level search is advantageous only when intermediate scores are sufficiently informative relative to their cost. S^3 queries the verifier TNb times on one-step reconstructions from partially resolved states, whereas BoK+PRM scores only K completed outputs. When the verifier is expensive and most reliable on complete text, output-level selection is a better use of compute; when intermediate signals are cheap and informative, reallocating compute during denoising is more effective. The default composite verifier lies in the latter regime by construction, which is why reward-model-free S^3, rather than the PRM substitution, is the paper's primary configuration and remains the strongest of the evaluated configurations on both general-domain benchmarks.

I Ablation Study: Computational Cost and Output Conciseness

We report three supplementary metrics alongside accuracy to characterize both computational cost and output behavior. **Average effective output tokens** measures the mean length of the generated responses after removing special end-of-text (<|endoftext|>) tokens and serves as a proxy for response conciseness, with shorter outputs generally reflecting more direct generations. **Total latency**

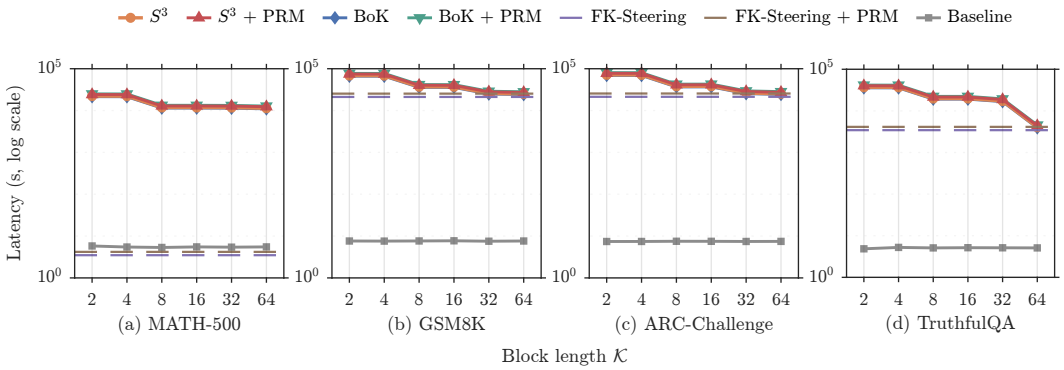


Figure 17: End-to-end latency across block lengths (logarithmic y-axis). Dashed lines denote the block-independent FK-Steering variants.

Table 18: Wall-clock time (seconds) vs. block length. FK-Steering variants are block-length-independent (rightmost columns).

Dataset	Method	Block Length						Block-Independent	
		2	4	8	16	32	64	FK-Steer.	FK-Steer. + PRM
MATH-500	Baseline	4.75	4.48	4.37	4.51	4.44	4.51	2.86	3.43
	BoK	10 750.02	10 717.07	5 748.84	5 746.09	5 711.94	5 507.98		
	BoK + PRM	12 523.22	12 484.84	6 697.10	6 693.89	6 654.12	6 416.51		
	S^3	11 082.50	11 048.53	5 926.64	5 923.80	5 888.60	5 678.33		
	S^3 + PRM	12 079.93	12 042.90	6 460.03	6 456.94	6 418.57	6 189.37		
GSM8K	Baseline	6.24	6.21	6.26	6.31	6.16	6.24	10 534.60	12 641.52
	BoK	33 191.94	33 092.33	17 924.24	17 865.85	12 443.35	12 021.84		
	BoK + PRM	38 666.90	38 550.85	20 880.82	20 812.79	14 495.87	14 004.82		
	S^3	34 218.50	34 115.80	18 478.60	18 418.40	12 828.20	12 393.65		
	S^3 + PRM	37 298.17	37 186.22	20 141.67	20 076.06	13 982.74	13 509.08		
ARC-Chal.	Baseline	6.10	6.12	6.18	6.15	6.11	6.13	10 687.86	12 825.43
	BoK	34 610.41	34 484.36	18 383.83	20 189.97	12 699.53	12 196.73		
	BoK + PRM	40 319.34	40 172.51	21 416.21	23 520.27	14 794.30	14 208.56		
	S^3	35 680.83	35 550.89	18 952.40	20 814.40	13 092.30	12 573.95		
	S^3 + PRM	38 892.10	38 750.47	20 658.12	22 687.70	14 270.61	13 705.61		
TruthfulQA	Baseline	3.97	4.29	4.17	4.22	4.20	4.17	1 755.86	2 107.03
	BoK	17 989.25	17 934.11	9 631.69	9 627.15	8 364.80	2 003.75		
	BoK + PRM	20 956.56	20 892.33	11 220.43	11 215.15	9 744.55	2 334.26		
	S^3	18 545.62	18 488.78	9 929.58	9 924.90	8 623.50	2 065.72		
	S^3 + PRM	20 214.72	20 152.76	10 823.24	10 818.14	9 399.61	2 251.63		

measures the end-to-end wall-clock time of a complete evaluation run, including model initialization, dataset preparation, and all forward passes. **Wall time** reports the average cumulative completion time across batches, measured from the start of the run; it therefore reflects when batches are completed rather than their individual processing duration. For approximately uniform batch sizes, this quantity is expected to be roughly half of the total latency. All configurations follow Table 11.

I.1 Component Ablation at the Default Budget

Table 16 reports metrics for the five ablation conditions at the matched default budget of Table 11. Baseline diffusion incurs the lowest latency by design, performing a single trajectory with no search overhead. BoK and Tilting only have nearly identical latency because both draw K independent sequences without inter-step communication. Look-ahead only and S^3 incur slightly higher wall time than BoK due to per-step verifier scoring and SSP resampling, yet S^3 achieves the best accuracy on GSM8K, ARC-C, and TruthfulQA within a wall-time budget comparable to Look-ahead only, confirming that tilting adds negligible overhead.

Table 19: **Both traced runs.** K=8 terminal samples for BoK, Nb=8 surviving particles for S^3, 512 NFE for both.

		q. 1 $(p-q)$	**q. 19** (3)
Baseline	returns	p^2 ✗	0 ✗
BoK	distinct answers / plurality	8/8 / 1/8	4/8 / 3/8
	correct answer in pool	no	yes (2/8)
	returns	p^2 ✗	0 ✗
S^3	plurality / lineages at t=0	6/8 / 1	7/8 / 2
	$\min_t \mathrm{ESS}_t/Nb$	0.933	0.971
	returns	$p-q$ ✓	3 ✓
Cost	wall clock, BoK → S^3	34.8 → 35.9 s	34.2 → 35.1 s

I.2 Compute Budget and FLOPs Analysis

Table 17 reports the compute budget and cost of each inference strategy at the default operating point. NFE counts diffusion-model evaluations only, while intrinsic verifier operations are excluded and optional PRM evaluations are reported separately. The number of PRM calls depends on how frequently each method invokes the verifier. BoK+PRM scores its $K = 8$ completed candidates once, requiring 8 PRM calls, whereas FK-Steering+PRM scores $K_{\mathrm{FK}} = 8$ particles across eight resampling rounds, for a total of 64 calls. In contrast, S^3+PRM evaluates all $Nb = 8$ frontier candidates at each of the $T = 64$ denoising steps. Because parent scores are cached, this requires $TNb = 512$ PRM calls rather than $2Nb$ calls per step. S^3+PRM retains majority voting with NLL tie-breaking, while BoK+PRM returns the completed candidate with the highest of the K PRM scores; neither rule requires an additional PRM call. At the default operating point, the resulting PRM overheads are approximately 1.4% for BoK+PRM, 10.9% for FK-Steering+PRM, and 87.4% for S^3+PRM.

These differences show that matching diffusion-model NFE does not imply matching total FLOPs once an external verifier is introduced. Accordingly, the accuracy-versus-NFE comparison in Figure 3 isolates performance at matched DLM evaluations, while Table 17 explicitly accounts for the auxiliary PRM cost. For the default intrinsic-verifier methods, total model FLOPs continue to scale directly with NFE. Figure 4 complements this comparison by plotting the same seven configurations on the total-FLOP axis, and Table 7 extends the accounting across the full budget grid. The three training-free configurations are identical on the NFE and total-FLOP axes, whereas the PRM variants shift according to verifier cost, with S^3+PRM incurring the largest shift of $1.88\times$.

I.3 Effect of Block Size

Figures 16 and 17 report average effective output tokens and end-to-end latency across all block sizes for LLaDA-8B-Instruct, comparing the single-trajectory Baseline against S^3 multi-block decoding; the corresponding wall-clock times are listed in Table 18. Larger block sizes substantially reduce wall time because fewer autoregressive blocks are decoded sequentially, but may sacrifice fine-grained denoising structure. S^3 consistently matches or improves over the Baseline in accuracy across most configurations, with modest additional latency from particle expansion and per-step verifier scoring.

J Qualitative Trajectory-Level Comparison

To illustrate how different compute allocations affect the denoising process, we compare the three methods on the same prompt and model, using the same decoding schedule and matched diffusion-model NFE. We record the one-step clean prediction $\hat{x}_0$ for each active trajectory at every denoising step. Figure 18 shows how the decoded text evolves, while Figure 19 summarizes the answers carried by the active trajectories. Table 19 shows a complementary case where all eight BoK samples give different answers, leaving terminal selection limited to the sampled outputs. These examples illustrate the structural limitation underlying Corollary 1: BoK samples from the unchanged p_0 and performs selection only after generation, whereas S^3 can reallocate compute among trajectories during denoising.

Denoising progress, MATH-500 q. 19 Answer: **3**

Baseline one stream, 64 NFE

0.16 To the smallest possible value of a such that all the roots of the $x^3 ax^2 +$

0.33 To determine the smallest possible value of a such that all the roots of the polynomial $x^3 + ax^2 + ax + 1 = 0$ are real, we

1.00 ...the discriminant *are not straightforward to compute directly*. Instead, we can use the fact that the discriminant *of the coefficients* must be non-negative. ⓪ ✗

BoK eight independent streams, 512 NFE

0.33 stream 1 is the baseline trajectory; the other seven denoise in parallel with no communication

1.00 ...since there is no quadratic term, the coefficient a must be zero. ⓪

1.00 ...becomes $a(x+1)^3 = 0$, which means $a = -1$. Since a must be positive, the only value is $a = 1$. ①

1.00 ...is given by $\Delta = a^2 - 4$. Setting the discriminant to zero gives $a = 2$. ②

1.00 ...is symmetric and has a specific form that allows us to determine the condition directly. ③

S^3 four lineages, eight scored children per step, 512 NFE

0.16 For the polynomial $x^3 + ax^2 + ax + 1 = 0$ to have real

0.33 For the polynomial $x^3 + ax^2 + ax + 1 = 0$ to have all real roots, the discriminant of the cubic polynomial must be non-negativ

1.00 ...The discriminant $a - 3 \geq 0$, which implies $a \geq 3$. Therefore, the smallest possible value of a is 3. ③ ✓

Figure 18: **Sample denoising progress.** This shows how the baseline, BoK, and S^3 evolve during denoising. While the baseline and BoK can drift toward incorrect solutions, S^3 progressively favors trajectories that remain consistent with the correct solution.

Telescoping check. For each surviving lineage, we verify that $\sum_t \log G_t$ matches the terminal reward tilt $\tau f(x_0)$, as required by Proposition 2. Across both runs, the equality holds up to numerical precision, confirming that the implemented incremental potentials telescope to the intended terminal reward tilt. The check is performed separately for each lineage because different survivors may have different ancestry paths. This validates the pathwise weighting identity, not exact finite-particle sampling from the Gibbs target.

Scope. These two examples are qualitative illustrations of the search mechanism and are not used to establish performance. Population-level analyses of look-ahead predictiveness, resampling behavior, and telescoping are provided in Appendices E.1, E.2, and E.4, respectively. The main accuracy conclusions are based on the benchmark-level results in Section 4.

MATH-500 q. 19 Let $a > 0$ with all roots of $x^3 + ax^2 + ax + 1 = 0$ real; find the smallest a. Answer: **3**

$\hat{x}_0$	$t/T = 0$	0.33	0.66	1.00
Baseline 1 stream	3 ×1	3 ×1	0 ×1	0 ×1
BoK 8 streams	3 ×8	3 ×7 −3 ×1	3 ×4 0 ×3 1 ×1	3 ×2 0 ×2 1 ×2 2 ×1 ⎵ ×1
S^3 8 children	3 ×8	3 ×8	3 ×8	3 ×6 0 ×2

Baseline

Nothing scores the trajectory, so the drift at t/T=0.66 is never caught.

Returns 0 ✗

BoK

The 8 streams never interact, so each drifts on its own and the pool disperses. The correct answer survives in 2/8 and is outvoted 3 : 2.

Returns 0 ✗

S^3

Every child is scored and resampled each step, so drifting children are replaced before they commit. 2 of 4 founder lineages remain at t=0.

Returns 3 ✓

Figure 19: **The same budget, three allocations.** Chips show the answers carried by the one-step clean predictions $\hat{x}_0$ at different denoising stages. Green denotes the correct answer. BoK and S^3 use the same 512 NFE.