Exact Power Sampling for Language Models with Reusable Exploration

Anonymous authors
Paper under double-blind review

Abstract

Power sampling sharpens a base model' s sequence distribution at inference time, eliciting reasoning performance comparable to reinforcement learning with verifiable rewards (RLVR) without additional training or external verifiers. However, existing methods largely rely on asymptotic approximations without finite-budget exactness guarantees, leaving efficient exact power sampling a key challenge. Our key insight is that local prefix-tree exploration yields globally valid, progressively tighter bounds on target mass, enabling exact sampling even when exploration is incomplete. Building on this structure, we introduce RSPS, an adaptive rejection sampler that uses sparse prefix refinement to guarantee exactness from the first output sample. We prove that acceptance and rejection feedback provide a stochastic oracle for prefix exploration, and amortize exploration costs across samples by reusing the resulting refinements. Experiments across multiple tasks and models show that, compared with existing methods, RSPS reduces distributional error to the level of finite-sample statistical fluctuations under matched compute budgets and meets stricter sampling-fidelity requirements at lower computational cost.

1 Introduction

Sampling from power distributions is an important problem in test-time scaling research. It stems from the view that the reasoning gains of large models after reinforcement learning (RL) post-training can also be understood as distribution sharpening: generation favors high-probability reasoning trajectories already supported by the base model, thereby improving performance (He et al., 2025; Karan & Du, 2025). One specific sharpening target is the sequence-level power distribution (Azizi et al., 2026). Specifically, given an input x and the base model's distribution over complete sequences $p_\theta(y_{1:T} \mid x)$, power sampling targets

$$\pi_\alpha(y_{1:T} \mid x) = \frac{p_\theta(y_{1:T} \mid x)^\alpha}{Z_\alpha(x)}, \qquad Z_\alpha(x) = \sum_{y'} p_\theta(y' \mid x)^\alpha,$$

where $\alpha > 1$. Existing methods primarily approximate this target through trajectory transitions in Markov chain Monte Carlo (MCMC), or importance weighting and resampling in sequential Monte Carlo (SMC) (Karan & Du, 2025; Azizi et al., 2026). Both are asymptotic approximation methods and share a formulation centered on the evolution of distributions: target probabilities are realized indirectly through transitions, weighting, and redistribution among candidate trajectories. Under the corresponding convergence conditions, the MCMC state distribution approaches the target distribution as the number of transitions increases (Tierney, 1994), while the weighted particle distribution of SMC approaches the target distribution as the number of particles increases (Chopin, 2004).

This property gives rise to two problems with asymptotic approximation methods. First, accuracy and sample count are coupled: obtaining accurate samples requires generating many sampled trajectories even when only a few outputs are needed. Second, these methods require hyperparameter tuning, yet do not guarantee the fidelity of the resulting distribution after the sampling cost has been incurred (Karan & Du, 2025; Azizi et al., 2026). Consequently, one often has to accept either redundancy or inaccuracy, and sometimes both.

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.

Moving beyond approximation methods, a more desirable solution would therefore produce sufficiently accurate samples from the very first sample, with further computation generating as many exact trajectories as needed, thereby decoupling accuracy from sample count.

Rejection sampling provides a foundation for this goal (Dymetman et al., 2012). Given a valid envelope that bounds the unnormalized target, exact samples can be obtained by drawing candidates from the normalized envelope and accepting them according to the ratio of the target probability to the envelope. The difficulty lies in efficiency: for a sequence-level power target, constructing an envelope distribution that tightly bounds the target is challenging, and the slack between them ultimately manifests as a high rejection rate.

We represent the sequence-level power target on an autoregressive prefix tree and use its hierarchical partition of the space and computable upper bounds on regional mass to construct a valid envelope that can be continually tightened through local refinement. In this representation, each prefix corresponds to all complete sequences that begin with it, and a mutually exclusive and exhaustive collection of prefix regions covers the entire target space. When $\alpha \geq 1$, the unnormalized power mass of the region corresponding to prefix $y_{<t}$ is at most $p_\theta(y_{<t})^\alpha$, yielding a valid upper bound for an unexpanded region without enumerating its suffixes. Combining these upper bounds with the base continuation distributions within the regions gives a pointwise envelope that bounds the target. Further partitioning a prefix region into its child regions preserves complete coverage of the space while monotonically tightening the envelope. Thus, even when exploration is incomplete, the envelope conditions required for exactness remain satisfied, and further exploration can improve the tightness of the envelope while preserving this guarantee.

Based on this structure, we propose the Rejection-Guided Sparse Prefix Sampler (RSPS), which decouples sampling exactness under a given target from the requested sample count. RSPS sparsely expands prefix regions along the paths actually visited and retains model distributions computed during acceptance and rejection, allowing refinements obtained through local exploration to be reused in subsequent sampling. We prove that this update process, which depends on the sampling history, preserves the correctness of the output distribution, guarantees exactness from the first returned sample, and supports further sample generation on demand. To explain efficiency, we further characterize the complementary roles of acceptance and rejection feedback: acceptance feedback identifies regions of the target distribution worth reusing, while rejection feedback points to regions whose upper bounds still have substantial room for tightening. These two types of feedback constitute a dual stochastic oracle for prefix exploration, providing a mechanistic explanation for amortizing exploration costs over subsequent samples through continual envelope tightening and reuse of existing computations. Across the six model–dataset configurations in Table 1, RSPS uses fewer FLOPs than the primary baseline, Power-SMC (Azizi et al., 2026), while achieving equal or higher reported accuracy.

Our main contributions are as follows.

- **Exact sequence-level power sampling.** Prefix probabilities provide computable bounds for unexplored subtrees, enabling a valid envelope that tightens through refinement while preserving exactness.
- **On-demand sampling with reusable exploration.** RSPS combines rejection correction and sparse prefix refinement, guaranteeing exact samples from the first output while reusing previous model computations.
- **Efficient exact sampling.** Acceptance and rejection feedback guide refinement as a stochastic oracle, amortizing exploration costs across samples.

2 Prefix Tree

2.1 Power Sampling and the Prefix-Tree Representation

Given a prompt x, let $y_{1:T}$ denote a complete output terminated by EOS, where T is the actual termination position (Parys et al., 2025). Let $\mathcal{Y}$ be the set of complete outputs with positive probability. With termination enforced at EOS or a preset maximum length, p_θ

is normalized over $\mathcal{Y}$. Below, we omit the conditioning on x in probabilities, abbreviating $\pi_\alpha(y_{1:T} \mid x)$ and $Z_\alpha(x)$ as $\pi_\alpha(y_{1:T})$ and Z_α; we allow $\alpha \geq 1$, with the target reducing to the base distribution when $\alpha = 1$.

Prefix trees are a common structure in sampling, mapping complete outputs to regions that can be progressively subdivided: the empty prefix is the root, token extensions are edges, and complete outputs are leaves. Let $y_{<t} = y_{1:t-1}$ denote the prefix before the t-th token is generated, which maps to a node in the prefix tree; $y_{<t} \preceq y'$ means that the former is a prefix of the latter and thus an ancestor node of the latter. A leaf represents a complete sequence $y_{<T+1} = y_{1:T}$, and the conditional probability of an empty continuation is taken to be 1. A prefix node $y_{<t}$ represents its descendant complete outputs, so $p_\theta(y_{<t}) = \sum_{y' \succeq y_{<t}} p_\theta(y')$.

The unnormalized power mass of a leaf is $p_\theta(y_{1:T})^\alpha$. The *true power mass* of the region represented by prefix $y_{<t}$ is defined as the sum of the masses of its descendant leaves:

$$M(y_{<t}) := \sum_{y' \succeq y_{<t}} p_\theta(y')^\alpha. \tag{1}$$

The probability that the target falls in this region is $M(y_{<t})/Z_\alpha$. Merely normalizing $p_\theta(y_t \mid y_{<t})^\alpha$ locally ignores subsequent paths. Computing M generally requires enumerating complete continuations, so we instead construct regional upper bounds below.

2.2 Partitioning, Envelopes, and Refinement of Prefix Space

As discussed above, naive rejection sampling is inefficient for power sampling because of the substantial slack between the envelope distribution and the target distribution. In this section, we draw on the idea of regional upper bounds from adaptive rejection sampling (Maddison et al., 2014), partitioning the output space with frontier nodes and constructing progressively refinable envelopes for the regions.

We define a set of prefixes as a *frontier* $\mathcal{F}$ if every root-to-leaf path passes through exactly one of them. A frontier node $y_{<t} \in \mathcal{F}$ corresponds to a subtree region containing all complete outputs $y' \succeq y_{<t}$; these regions are mutually exclusive and cover $\mathcal{Y}$.

The true power mass $M(y_{<t})$ of this region is generally difficult to compute. Since the conditional probabilities of complete continuations sum to 1, and their power sum is at most 1 when $\alpha \geq 1$, the prefix probability yields a *computable upper bound* $U(y_{<t})$:

$$\begin{aligned} U(y_{<t}) &:= p_\theta(y_{<t})^\alpha, \\ M(y_{<t}) = U(y_{<t}) &\sum_{y' \succeq y_{<t}} p_\theta(y' \mid y_{<t})^\alpha \leq U(y_{<t}). \end{aligned} \tag{2}$$

Thus, U is a rigorous upper bound on the subtree's power mass and is greater than or equal to the true mass M.

The subsequent rejection sampling procedure also requires a pointwise envelope for each output. For $y_{1:T}$ and its unique frontier prefix $y_{<t}$, weight the region by $U(y_{<t})$ and then generate a continuation according to $q_{\mathcal{F}}(y_{t:T} \mid y_{<t})$. Provided that $q_{\mathcal{F}}(y_{t:T} \mid y_{<t}) \geq p_\theta(y_{t:T} \mid y_{<t})^\alpha$, the following is guaranteed for every complete output $y_{1:T}$:

$$U(y_{<t}) q_{\mathcal{F}}(y_{t:T} \mid y_{<t}) \geq p_\theta(y_{1:T})^\alpha \tag{3}$$

We further define the total envelope mass as $E(\mathcal{F}) := \sum_{y_{<t} \in \mathcal{F}} U(y_{<t}) \geq Z_\alpha$. Selecting a region according to U and generating a continuation yields the rejection sampling proposal.

To reduce envelope slack, replace a nonterminal frontier node $y_{<t}$ with all its reachable children to obtain a new frontier $\mathcal{F}'$. The region is subdivided, while coverage of $\mathcal{Y}$ is preserved. The difference in total envelope mass before and after refinement is

$$E(\mathcal{F}) - E(\mathcal{F}') = U(y_{<t}) \left[1 - \sum_v p_\theta(v \mid y_{<t})^\alpha \right] \geq 0. \tag{4}$$

Here, v ranges over all reachable successor tokens, including EOS. The new pointwise envelope for each output is also no larger than the old one, because $p_\theta(v \mid y_{<t})^\alpha \leq p_\theta(v \mid y_{<t})$. The algorithm below refines regions as they are visited.

Extension to general inference targets Power sampling can be viewed as a special case of the more general approximate inference problem with target $p_\theta(y_{1:T})\Phi(y_{1:T})$, where $\Phi(y_{1:T}) \geq 0$ is a sequence weight; setting $\Phi(y_{1:T}) = p_\theta(y_{1:T})^{\alpha-1}$ recovers power sampling. When the reweighting potential admits verifiable prefix upper bounds, the envelope and refinement mechanism above also applies to such targets; specific conditions and examples are given in Appendix B.

3 Method

Building on the envelope construction in Section 2, the Rejection-Guided Sparse Prefix Sampler (RSPS) repeats three steps: selecting a frontier region according to its envelope mass, generating a continuation token by token within that region using a low-temperature proposal and applying rejection correction, and refining the visited regions after acceptance or rejection. Figure 1 illustrates how RSPS, MCMC, and SMC operate on the prefix tree.

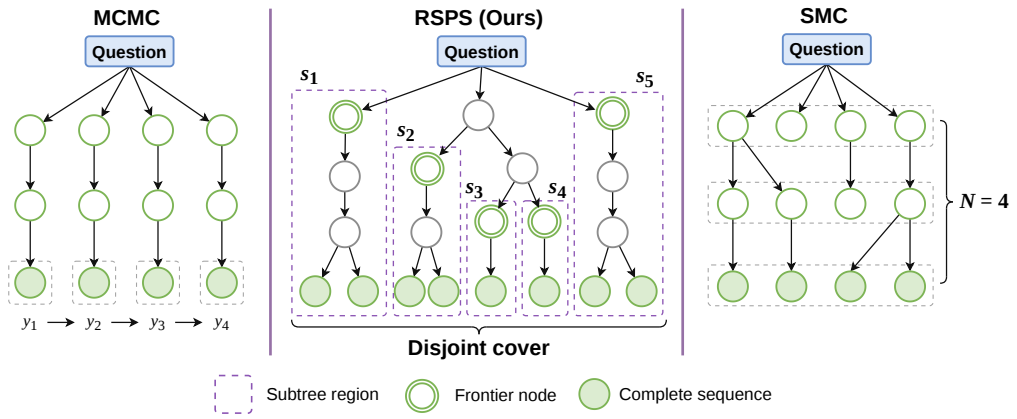


Figure 1: Illustration of different sampling algorithms on a prefix tree. MCMC generates a complete sequence at a time and makes probabilistic transitions between sequences. SMC maintains a fixed number of particles at each level and advances them in parallel through extension and resampling. RSPS maintains a set of frontier nodes, none of which is an ancestor of another, and extends sequences from the frontier. Double circles denote frontier nodes, and green dashed boxes indicate the corresponding subtree regions.

Frontier selection and rejection correction. Fix the current frontier $\mathcal{F}$ and write $E = E(\mathcal{F})$ for brevity. An attempt first selects a starting prefix $y_{<t}$ with probability $U(y_{<t})/E$ and then generates a continuation from that prefix. Let $q_{\mathcal{F}}(v \mid y_{<j})$ denote the next-token proposal distribution during continuation. It must cover the positive-probability support of the base distribution and, as described in Section 2.2, satisfy the single-step correction condition $a_j = p_\theta(y_j \mid y_{<j})^\alpha / q_{\mathcal{F}}(y_j \mid y_{<j}) \leq 1$. The acceptance probability of a complete candidate is therefore

$$A_q(y_{1:T}; y_{<t}) = \prod_{j=t}^{T} \frac{p_\theta(y_j \mid y_{<j})^\alpha}{q_{\mathcal{F}}(y_j \mid y_{<j})}. \tag{5}$$

Here, T is the actual termination position at which EOS is generated or the maximum length is reached, rather than a prespecified continuation length.

Low-temperature proposals and tokenwise decisions. Let $C_\alpha(y_{<t}) = \sum_v p_\theta(v \mid y_{<t})^\alpha \leq 1$ denote the local power sum. Within the selected region, RSPS uses a low-temperature proposal with temperature $1/\alpha$:

$$q_{\mathcal{F}}(v \mid y_{<t}) = \frac{p_\theta(v \mid y_{<t})^\alpha}{C_\alpha(y_{<t})}, \qquad a(v) = C_\alpha(y_{<t}) \leq 1. \tag{6}$$

Lemma 1 (Local optimality of the low-temperature proposal) *For a fixed prefix, the proposal in Eq. 6 is the unique proposal over positive-probability successor tokens that gives the single-step correction factor $a(v)$ zero variance.*

This removes randomness from the correction factor at a fixed prefix, but the complete path still requires rejection correction; the proof of the lemma is given in Appendix A.1. Substituting the low-temperature proposal into Eq. 5 yields

$$A_q(y_{1:T}; y_{<t}) = \prod_{j=t}^{T} C_\alpha(y_{<j}). \tag{7}$$

Since $C_\alpha \leq 1$, each attempt requires only one uniformly sampled threshold; tokenwise early rejection is possible as soon as the cumulative acceptance probability falls below this threshold.

Sparse refinement and parallel sampling. RSPS retains unvisited sibling subtrees as token groups and separates child prefixes only along the paths actually visited, avoiding explicit expansion of the entire vocabulary at each level. Each batch of attempts fixes the current envelope and generates continuations in parallel. At the end of the batch, distributions already computed along accepted and rejected paths are used to refine the frontier, and the cache is retained for reuse by subsequent samples. Token-group weights, splitting invariants, tokenwise decisions, and the complete algorithm are given in Appendix A.

Blockwise sampling for long sequences The cost of exact sampling from power distributions over long sequences may grow exponentially with length (Block & Polyanskiy, 2023; Gheissari et al., 2018), so RSPS serves as a sampling kernel within blocks of a finite horizon. Each block generates a set of exact samples that share refinement results; the next block uses a virtual root to connect these prefixes and their continuation subtrees, inherits the sample weights, and continues sampling.

The total variation error of a sequence can be bounded by the sum of upper bounds on the conditional distribution error at each step (Abate et al., 2014). RSPS samples exactly from the block target determined by the given candidate prefixes, weights, and horizon, eliminating within-block sampling bias and its accumulation; the target approximation introduced by the finite horizon and candidate set remains.

4 Experiments

We compare RSPS with Power-SMC (Azizi et al., 2026), MH power sampling (Karan & Du, 2025), greedy decoding, and low-temperature decoding on Qwen3.5-9B (Qwen Team, 2026) and gemma-4-12B (Team et al., 2026) using three reasoning benchmarks. Acc denotes accuracy and FLOPs are normalized to greedy decoding within each setting.

Accuracy. Across the six model–dataset pairs in Table 1, RSPS outperforms the primary baseline, Power-SMC, by 0.2–1.3 percentage points in five pairs, while both achieve 70.6% on MATH500 with Qwen3.5-9B. RSPS also has higher acc than MH power sampling, greedy decoding, and low-temperature decoding in all six pairs; its margin over MH power sampling is 0.6–3.6 percentage points.

Computational cost. The relative FLOPs of RSPS range from 27.8–42.3×, compared with 40.9–64.8× for Power-SMC. Comparing each pair under the same model and dataset, RSPS uses 5.6%–48.7% fewer FLOPs than Power-SMC and 19.6%–42.2% fewer than MH power sampling (40.2–56.1×). Thus, across these six pairs, RSPS achieves higher or equal reported accuracy at lower computational cost than both power sampling baselines, while using more computation than greedy and low-temperature decoding.

5 Analysis

We analyze exactness, exploration-cost amortization, and the cost–error tradeoff.

Table 1: Answer accuracy (acc, %) and relative FLOPs for two models on engineering subsets of three datasets. The FLOPs of greedy decoding are normalized to 1.00× for each model–dataset pair.

		Greedy decoding	Low-temperature decoding	MH power sampling	Power-SMC	**RSPS** (ours)
Qwen3.5-9B						
GPQA Diamond	acc	58.2	62.8	63.9	63.2	64.5
	FLOPs	1.00×	0.98×	52.6×	64.8×	42.3×
MATH500	acc	64.0	62.6	70.0	70.6	70.6
	FLOPs	1.00×	0.88×	47.7×	60.4×	33.6×
MMLU-Pro	acc	76.2	75.3	76.1	77.6	78.5
	FLOPs	1.00×	0.85×	52.9×	59.6×	30.6×
gemma-4-12B						
GPQA Diamond	acc	63.2	79.2	85.3	88.2	88.9
	FLOPs	1.00×	1.02×	56.1×	40.9×	38.6×
MATH500	acc	77.2	79.6	81.2	81.8	82.0
	FLOPs	1.00×	0.97×	54.5×	62.2×	39.3×
MMLU-Pro	acc	64.7	75.6	79.3	81.2	81.9
	FLOPs	1.00×	0.90×	40.2×	43.5×	27.8×

5.1 Sampling Exactness Guarantee

Lemma 2 (Sampling exactness under adaptive refinement) *Let $\mathcal{H}_{k-1}$ be the history before attempt k. For any valid frontier determined by this history, an accepted sequence Y_k satisfies*

$$\Pr(Y_k = y_{1:T} \mid \text{acc}, \mathcal{H}_{k-1}) = \pi_\alpha(y_{1:T}). \tag{8}$$

Every sample returned in attempt-index order therefore follows the target, including the first.

The proof is in Appendix C.1. Refinement changes efficiency but preserves the valid envelope and exactness guarantee.

Approximate reference distribution For a fixed MATH500 input, Qwen3.5-9B, and power exponent from Section 4, we collect 120,000 responses to depth 64 and their model probabilities. The reference set $\mathcal{Y}_{\text{ref}}$ counts each distinct response once and preserves early termination. It covers the main power mass with small reference error, but is not exhaustive ground truth. Both methods use this same reference for relative comparison.

For $d \in \{32, 64\}$, let $\mathcal{Y}_d$ contain distinct outputs truncated to length $\min(d, T)$, where T is the termination position. Writing these trajectories as $y_{<t}$, define $Z_{\alpha,d} := \sum_{y_{<t} \in \mathcal{Y}_d} p_\theta(y_{<t})^\alpha$. The expected trajectory power mass is

$$\mu_d = \sum_{y_{<t} \in \mathcal{Y}_d} \frac{p_\theta(y_{<t})^\alpha}{Z_{\alpha,d}} \, p_\theta(y_{<t})^\alpha \quad = \frac{\sum_{y_{<t} \in \mathcal{Y}_d} p_\theta(y_{<t})^{2\alpha}}{Z_{\alpha,d}}. \tag{9}$$

Sampling accuracy Figure 2 compares the sample mean $N^{-1} \sum_{i=1}^{N} p_\theta(y^{(i)}_{1:\min(d,T_i)})^\alpha$ with the reference expectation, where T_i is trajectory i's termination position. Under exact sampling, its expectation equals μ_d. RSPS stays close to the reference at both depths, with fluctuations narrowing as trajectory counts grow, consistent with the exactness lemma. SMC approaches the reference at depth 64 with many trajectories, but remains lower at depth 32 and varies more overall. Both methods fluctuate widely with few trajectories, while RSPS medians remain close to the reference.

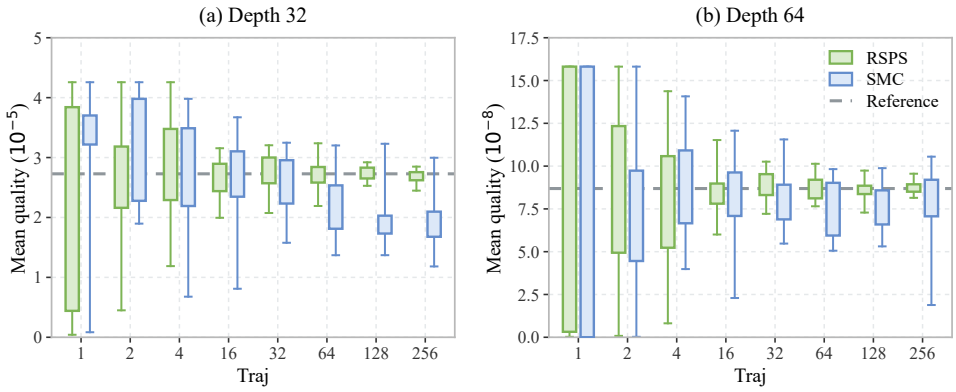


Figure 2: Mean trajectory power mass across repeated runs with different trajectory counts: (a) depth 32; (b) depth 64. The gray dashed line is the reference estimate of μ_d; green and blue box plots show RSPS and SMC, respectively.

5.2 Rejection Guidance and Cost Amortization

The acceptance rate is Z_α/E. Acceptance and rejection feedback identify reusable target regions and envelope slack, respectively.

Lemma 3 (Dual stochastic oracle) *Fix $\mathcal{F}$ and $E = E(\mathcal{F})$, interpreting token groups as their child prefixes. Let S be the selected starting prefix, including the child sampled within a selected group. For any prefix $y_{<j}$,*

$$\Pr(y_{<j} \preceq Y \mid \text{acc}, \mathcal{F}) = \frac{M(y_{<j})}{Z_\alpha}. \tag{10}$$

For any effective starting prefix $y_{<t} \in \mathcal{F}$, if $E > Z_\alpha$,

$$\Pr(S = y_{<t} \mid \text{rej}, \mathcal{F}) = \frac{U(y_{<t}) - M(y_{<t})}{E - Z_\alpha}. \tag{11}$$

The proof is in Appendix C.2. Accepted paths visit prefixes according to target mass, identifying regions likely to serve future samples. Rejected attempts select regions in proportion to their removable envelope mass $U(y_{<t}) - M(y_{<t})$, revealing potential refinement gains without computing $M(y_{<t})$.

RSPS caches next-token distributions along visited paths and reuses their refinements. Accepted paths supply existing model evaluations; rejected paths reveal slack. For a fixed envelope, one accepted sample needs E/Z_α attempts on average. Refinement cannot increase E, so this metric cannot worsen. Token groups add only linearly many structural units in the visited path length. Reuse across samples amortizes earlier exploration costs.

5.3 Computational Cost and Distributional Error

Section 4 shows lower FLOPs for RSPS at equal or higher reported accuracy. Here we examine prefix expansions for a fixed input: RSPS costs more per trajectory but needs fewer trajectories for comparable error, reducing total cost over some budgets. Expansion counts proxy model computation and cannot be directly converted to task FLOPs.

At a block boundary, RSPS and SMC both retain N weighted prefix trajectories and prune other continuation regions. Both therefore incur finite-horizon and candidate-set error relative to the full-sequence target; SMC also incurs within-block sampling error (Section 3). We exclude single-chain MCMC because it retains only one trajectory rather than a comparable weighted candidate set.

Trajectory cost Using the model, input, and exponent from Section 5.1, we generate trajectories of up to 32 tokens, preserving early EOS. Let $C(N)$ count expansions for N trajectories, with $C(0) = 0$. The additional and average costs are $c_N = C(N) - C(N-1)$ and $\bar{c}_N = C(N)/N$. Both methods count one expansion per newly evaluated next-token distribution; rejected RSPS attempts are included.

Figure 3(a,b) plots trajectory count vertically against additional and average costs. In (b), RSPS is on the left and SMC on the right; distance from zero indicates cost. RSPS's additional cost falls toward SMC's, while its average cost gap narrows but remains slightly positive. Initially, a loose envelope and little cached computation make rejection and exploration costly. Oracle-guided refinement and cache reuse then reduce later costs and amortize this initial work. These costs differ from full-task FLOPs in Section 4.

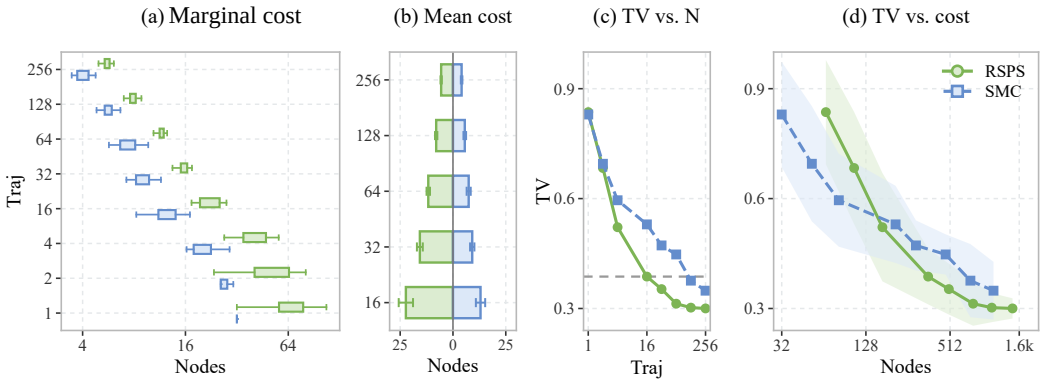


Figure 3: RSPS (green) and SMC (blue): (a) additional trajectory cost; (b) average trajectory cost; (c) trajectory count versus global TV error represented on $\mathcal{Y}_{32}$; (d) cumulative expansion cost versus the same error. The reference aggregates depth-64 response probabilities backward through the tree.

Error relative to the reference target To assess finite candidates against the longer-sequence target, we aggregate the reference from Section 5.1 onto $\mathcal{Y}_{32}$. Raising response probabilities to power α, summing backward, and normalizing gives

$$\widehat{M}_{64}(y_{<t}) = \sum_{\substack{y' \in \mathcal{Y}_{\text{ref}} \\ y' \succeq y_{<t}}} p_\theta(y')^\alpha, \qquad q_{\text{ref}}(y_{<t}) = \frac{\widehat{M}_{64}(y_{<t})}{\sum_{y' \in \mathcal{Y}_{\text{ref}}} p_\theta(y')^\alpha}. \tag{12}$$

Unlike the full subtree mass M (Section 2.1), $\widehat{M}_{64}$ is estimated from the reference set; q_{ref} includes continuations beyond depth 32.

Let $\mathcal{S}_N \subseteq \mathcal{Y}_{32}$ be RSPS's distinct retained trajectories, weighted by their within-block power masses. SMC retains $\{(y^{(i)}_{1:\min(32,T_i)}, \bar{w}_i)\}_{i=1}^N$ with normalized particle weights, summing weights for identical trajectories. Their weighted distributions are

$$\begin{aligned} p_{32}^{\text{RSPS}}(y_{<t}) &= \frac{\mathbf{1}\{y_{<t} \in \mathcal{S}_N\}\, p_\theta(y_{<t})^\alpha}{\sum_{y_{<j} \in \mathcal{S}_N} p_\theta(y_{<j})^\alpha}, \\ p_{32}^{\text{SMC}}(y_{<t}) &= \sum_{i=1}^N \bar{w}_i\, \mathbf{1}\{y^{(i)}_{1:\min(32,T_i)} = y_{<t}\}, \qquad \sum_{i=1}^N \bar{w}_i = 1. \end{aligned} \tag{13}$$

We measure global error by total variation distance (TV):

$$\text{TV}(q_{\text{ref}}, p_{32}) = \frac{1}{2} \sum_{y_{<t} \in \mathcal{Y}_{32}} \left| q_{\text{ref}}(y_{<t}) - p_{32}(y_{<t}) \right|, \tag{14}$$

Here p_{32} denotes either method's distribution. Lower TV means a closer match to the reference, reflecting finite candidates, horizon truncation, and sampling approximation together.

In Figure 3(c), RSPS reaches TV 0.35 with about 32 trajectories, versus 256 for SMC, yielding lower error per trajectory.

Error at equal cost In Figure 3(d), SMC is better at very low budgets, but RSPS has lower TV beyond about 140 expansions. At 560 expansions, TV is about 0.35 for RSPS and 0.45 for SMC. Fewer required trajectories offset RSPS's higher per-trajectory cost. This explains the trend in Section 4, although fixed-input expansion counts cannot determine real-task FLOPs.

6 Related Work

Power sampling. Power sampling sharpens complete-sequence probabilities without training. Prefix-dependent normalizers make tokenwise low-temperature decoding differ from the global power target (Karan & Du, 2025; Azizi et al., 2026). MH-based MCMC (Karan & Du, 2025) repeatedly resamples suffixes, matching or exceeding RL-post-trained models on reasoning tasks but requiring costly serial transitions. Power-SMC (Azizi et al., 2026) advances trajectories in parallel with tokenwise importance correction and adaptive resampling. Scalable Power Sampling (Ji et al., 2026) uses Monte Carlo lookahead to rescale low-temperature conditionals by estimated continuation contributions. MH and SMC converge asymptotically without finite-budget exactness guarantees; token-level rescaling also approximates conditionals. RSPS instead uses valid envelopes and rejection correction for exact block-target samples regardless of requested sample count.

Adaptive rejection sampling. CARS (Parys et al., 2025), the closest work, handles 0–1 constraints by excluding subtrees proven infeasible. Power-sampling rejection does not imply zero subtree mass, so RSPS handles nonbinary weights by tightening envelopes and applying rejection correction. It draws on A* Sampling's spatial partitions and regional bounds (Maddison et al., 2014), and OS*'s rejection-guided refinement (Dymetman et al., 2012).

7 Conclusion

RSPS uses prefix probabilities to bound the full target space and tightens this envelope through sparse refinement. Rejection correction guarantees exactness from the first sample and supports on-demand generation. Acceptance and rejection feedback identify reusable regions and envelope slack, guiding exploration while cached computations amortize its cost.

Across six configurations, RSPS matches or exceeds Power-SMC and MH accuracy with fewer FLOPs (Table 1). Prefix reuse reduces average trajectory cost; in the fixed-input budgets studied, fewer trajectories also reduce the cost of reaching comparable error. For long sequences, exactness applies to the current block target; finite-horizon and candidate-set approximation remains.

References

Alessandro Abate, Frank Redig, and Ilya Tkachev. On the effect of perturbation of conditional probabilities in total variation. *Statistics & Probability Letters*, 88:1–8, 2014.

Seyedarmin Azizi, Erfan Baghaei Potraghloo, Minoo Ahmadi, Souvik Kundu, and Massoud Pedram. Power-SMC: Low-latency sequence-level power sampling for training-free LLM reasoning, 2026. URL `https://arxiv.org/abs/2602.10273`.

Adam Block and Yury Polyanskiy. The sample complexity of approximate rejection sampling with applications to smoothed online learning. In *The Thirty Sixth Annual Conference on Learning Theory*, pp. 228–273. PMLR, 2023.

Nicolas Chopin. Central limit theorem for sequential Monte Carlo methods and its application to bayesian inference. *The Annals of Statistics*, 32(6):2385–2411, 2004. doi: 10.1214/009053604000000698. URL https://doi.org/10.1214/009053604000000698.

Marc Dymetman, Guillaume Bouchard, and Simon Carter. The OS* algorithm: A joint approach to exact optimization and sampling, 2012. URL https://arxiv.org/abs/1207.0742.

Nicolas Emmenegger, Theo X. Olausson, Armando Solar-Lezama, and Chara Podimata. Conformal language modeling via posterior sampling, 2026. URL https://arxiv.org/abs/2606.03731.

Reza Gheissari, Eyal Lubetzky, and Yuval Peres. Exponentially slow mixing in the mean-field swendsen-wang dynamics. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1981–1988. SIAM, 2018.

Andre Wang He, Daniel Fried, and Sean Welleck. Rewarding the unlikely: Lifting GRPO beyond distribution sharpening. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 25548–25560. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.1298. URL https://aclanthology.org/2025.emnlp-main.1298/.

Xiaotong Ji, Rasul Tutunov, Matthieu Zimmer, and Haitham Bou Ammar. Scalable power sampling: Unlocking efficient, training-free reasoning for LLMs via distribution sharpening, 2026. URL https://arxiv.org/abs/2601.21590.

Aayush Karan and Yilun Du. Reasoning with sampling: Your base model is smarter than you think, 2025. URL https://arxiv.org/abs/2510.14901.

Chris J. Maddison, Daniel Tarlow, and Tom Minka. A* sampling. In *Advances in Neural Information Processing Systems*, volume 27, pp. 3086–3094, 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/hash/937debc749f041eb5700df7211ac795c-Abstract.html.

PaweĹ Parys, Sairam Vaidya, Taylor Berg-Kirkpatrick, and Loris D'Antoni. Constrained adaptive rejection sampling. *arXiv preprint arXiv:2510.01902*, 2025.

Qwen Team. Qwen3.5-9B model card, 2026. URL https://huggingface.co/Qwen/Qwen3.5-9B.

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, et al. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026.

Luke Tierney. Markov chains for exploring posterior distributions. *The Annals of Statistics*, 22(4):1701–1728, 1994. doi: 10.1214/aos/1176325750. URL https://doi.org/10.1214/aos/1176325750.

Stephen Zhao, Rob Brekelmans, Alireza Makhzani, and Roger Grosse. Probabilistic inference in language models via twisted sequential monte carlo. *arXiv preprint arXiv:2404.17546*, 2024.

A Implementation Details and Proofs for RSPS

A.1 Local Optimality of the Low-Temperature Proposal

Fix a prefix $y_{<t}$ and, over its positive-probability successor tokens, write $p_v = p_\theta(v \mid y_{<t})$ and $q_v = q_{\mathcal{F}}(v \mid y_{<t})$ for brevity. Regardless of the choice of proposal distribution q, the mean of the single-step correction factor $a(v) = p_v^\alpha / q_v$ is $\mathbb{E}_q[a] = \sum_v p_v^\alpha = C_\alpha(y_{<t})$. Proposals differ

in whether this factor varies substantially with the sampled token. By the Cauchy–Schwarz inequality,

$$\mathrm{Var}_q[a] = \sum_v \frac{p_v^{2\alpha}}{q_v} - (\mathbb{E}_q[a])^2 \geq \frac{\left(\sum_v p_v^\alpha\right)^2}{\sum_v q_v} - (\mathbb{E}_q[a])^2 = 0. \tag{15}$$

Equality holds if and only if $q_v \propto p_v^\alpha$. Normalization then yields Eq. 6, in which every possible token has the same factor $C_\alpha(y_{<t})$. This makes the single-step correction stable and easy to compute. The factors at all visited steps must still be multiplied along the complete path, because C_α generally differs across prefixes.

A.2 Tokenwise Decisions and Adaptive Refinement

In practice, the b attempts in a batch share the active units and envelope weights present at the start of the batch, but independently select starting points and generate continuations. Each attempt draws $u \sim \mathrm{Unif}(0,1)$ only once. By Eq. 7, the cumulative acceptance probability upon reaching prefix $y_{<m+1}$ is $\prod_{j=t}^{m} C_\alpha(y_{<j})$, which can also be written in logarithmic form as

$$\ell_m = \sum_{j=t}^{m} \log C_\alpha(y_{<j}). \tag{16}$$

Since every $C_\alpha \leq 1$, ℓ_m can only decrease. Once $\ell_m < \log u$, no subsequent token can turn the candidate into an accepted sample, so the attempt can stop at the current prefix and record a rejection. If a terminal leaf is reached without crossing the threshold, the complete sequence is accepted. If the starting point is itself a terminal leaf, there are no subsequent steps and the acceptance probability is 1.

After the entire batch finishes, the next-token distributions already computed along accepted and rejected paths are written to the cache, and the frontier is updated jointly. Even if the same prefix is visited by multiple parallel paths, it is processed only once; the new envelope is used starting with the next batch. Accepted sequences are collected in attempt-index order, so differences in the completion times of parallel tasks do not affect the return order. Refinement always preserves regional coverage and the envelope upper bound, so these batch updates still satisfy the exactness conditions in Section 5.1.

A.3 Hierarchical Expansion for Memory Control

A token group $(y_{<t}, A)$ implicitly represents sibling subtrees that have not yet been instantiated, where A is a nonempty set of reachable successor tokens. Its envelope mass is

$$U(y_{<t}, A) := U(y_{<t}) \sum_{v \in A} p_\theta(v \mid y_{<t})^\alpha. \tag{17}$$

A token group can be understood as "a group of sibling branches that have not yet been assigned individual nodes." First select a group with probability $U(y_{<t}, A)/E$, then sample a token v within the group in proportion to $p_\theta(v \mid y_{<t})^\alpha$. Combining these two selections, the probability of selecting child prefix $y_{<t+1} = y_{<t}v$ is

$$\frac{U(y_{<t}, A)}{E} \frac{p_\theta(v \mid y_{<t})^\alpha}{\sum_{w \in A} p_\theta(w \mid y_{<t})^\alpha} = \frac{U(y_{<t+1})}{E}. \tag{18}$$

Thus, first sampling a token within a group is exactly equivalent to directly sampling from all child prefixes according to their upper bounds. This token is part of selecting the starting point, not the subsequent low-temperature proposal, so acceptance probabilities begin accumulating only after the child prefix. Applying rejection correction to the continuation from that child prefix gives a probability of $p_\theta(y_{1:T})^\alpha/E$ for proposing and accepting the complete sequence $y_{1:T}$; ordinary prefix units satisfy the same expression.

After token v is visited, only the selected child prefix is separated out, while the remaining siblings stay in the residual group $A \setminus \{v\}$; the residual group is removed if it is empty. Splitting a group changes only its storage representation, and the weights of the two parts still sum to the original group weight. Only subsequent refinement of evaluated prefixes

reduces the envelope mass according to Eq. 4. This adds at most a constant number of structural units per visited level, so a path of length t adds $O(t)$ units; memory for next-token distributions and model caches is counted separately.

Algorithm 1: Parallel RSPS (frontier updates by batch)

Input: target sample count N, number of parallel attempts per batch b.
Initialize the active units to $\{\epsilon\}$ and set $U(\epsilon) = E = 1$.
Initialize an empty output list and an empty next-token distribution cache.
While the output list contains fewer than N sequences, run the next batch.
  Fix the active units, their weights, and total mass E for this batch.
  For attempts $i = 1, \ldots, b$, **execute in parallel**:
    Sample a starting unit c_i according to each unit's weight divided by E.
    Independently draw $u_i \sim \mathrm{Unif}(0, 1)$ and set $\ell_i = 0$.
    If c_i is a prefix unit s, set the current prefix to $r = s$.
    If $c_i = (s, A)$ is a token group, perform sampling within the group.
      Sample $v \in A$ with probability $p_\theta(v \mid s)^\alpha / \sum_{w \in A} p_\theta(w \mid s)^\alpha$.
      Set $r = sv$; do not include this token in ℓ_i.
    If r is terminal, record acceptance of r and end this attempt.
    While r is nonterminal and this attempt has not ended:
      Retrieve $p_\theta(\cdot \mid r)$ from the cache or evaluate it.
      Record the visited prefix r and its distribution.
      Compute $C_\alpha(r) = \sum_v p_\theta(v \mid r)^\alpha$.
      Update $\ell_i \leftarrow \ell_i + \log C_\alpha(r)$.
      If $\ell_i < \log u_i$, record rejection at r and end the attempt.
      Otherwise, sample the next token according to $p_\theta(v \mid r)^\alpha / C_\alpha(r)$.
      Set $r \leftarrow rv$ and record the traversed branch.
      If r is terminal, record acceptance of r and end the attempt.
  End parallel attempts; order this batch's results by attempt index.
  Append accepted sequences from this batch to the output list in index order.
  Merge duplicate prefix evaluations and updates across paths.
  Separate selected child prefixes from groups along each path, retaining nonempty residual groups.
  Refine evaluated prefixes along visited paths, reusing the cache.
  Update affected unit weights and the total envelope mass E.
End while; return the first N sequences in the output list.

B Extension to General Inference Targets

Above, the target weight of a complete sequence y is set to $p_\theta(y)^\alpha$. To reweight by rewards, values, or other scores, the target can instead be written as

$$\pi_\Phi(y) = \frac{p_\theta(y)\Phi(y)}{Z_\Phi}, \qquad Z_\Phi = \sum_{y \in \mathcal{Y}} p_\theta(y)\Phi(y) > 0, \quad \Phi(y) \geq 0. \tag{19}$$

Here, $\Phi(y)$ assigns a weight to a complete sequence, and power sampling corresponds to $\Phi(y) = p_\theta(y)^{\alpha-1}$. For example, a reward $R(y)$ can induce the weight $\Phi(y) = \exp(\beta R(y))$, where $\beta \geq 0$; existing methods also learn intermediate scores to guide generation (Zhao et al., 2024). To continue using prefix refinement, the key is to provide a reliable upper bound for suffixes that have not yet been generated. We give two constructions below.

Prefix upper bounds that control individual sequence weights. Let $B(s)$ be a nonnegative prefix function that is nonincreasing along prefix extensions, i.e., $B(sv) \leq B(s)$, and satisfies $B(y) \geq \Phi(y)$ at complete sequences. Then $\Phi(y) \leq B(s)$ for every $y \succeq s$. This yields a regional upper bound and a pointwise envelope of the same form as in power

sampling:

$$M_\Phi(s) := \sum_{y \succeq s} p_\theta(y)\Phi(y) \le U_\Phi(s) := p_\theta(s)B(s),$$
$$\overline{f}_{\mathcal{F},\Phi}(y) := p_\theta(y)B(s) \ge p_\theta(y)\Phi(y), \qquad s \in \mathcal{F},\ s \preceq y. \tag{20}$$

After refining prefix s, the sum of the upper bounds of its child regions is

$$\sum_v U_\Phi(sv) = p_\theta(s) \sum_v p_\theta(v \mid s)B(sv) \le p_\theta(s)B(s) = U_\Phi(s). \tag{21}$$

Thus, refinement still tightens the envelope. To sample, first select a region according to $U_\Phi(s)$, generate a continuation to y using the base model within the selected positive-weight region, and then accept with probability $\Phi(y)/B(s)$; the accepted weight is exactly $p_\theta(y)\Phi(y)$. For example, express the cumulative-score threshold of Emmenegger et al. (2026) on prefixes: each completed claim has a score in $[0, 1]$, and $B(s)$ indicates whether the product of these scores still meets a fixed threshold. Scores of completed claims remain fixed, and continuations only add new claims, so the product cannot increase and $B(s)$ is nonincreasing; the same indicator on complete sequences can serve as $\Phi(y)$.

Stepwise rejection when only an upper bound in expectation is available. Sometimes the learned quantity is a prediction $h(s)$ of "how high the final score is likely to be when generation continues from this prefix," which need not upper-bound every individual continuation. In this case, one can instead control the conditional average of the child-node scores. Let $h(s) \ge 0$, set $h(y) = \Phi(y)$ at leaves, and assume that every nonterminal prefix satisfies the same one-sided residual bound

$$\sum_v p_\theta(v \mid s)h(sv) - h(s) \le \varepsilon, \qquad \varepsilon \ge 0. \tag{22}$$

If the maximum generation length is $T_{\max}$, this underestimation can be compensated for using the maximum number of remaining steps:

$$H(s) = h(s) + (T_{\max} - |s|)\varepsilon \quad \text{(nonterminal prefixes)}, \qquad H(y) = \Phi(y) \quad \text{(terminal leaves)}. \tag{23}$$

At each step, the correction term decreases by at least ε, so Eq. 22 implies

$$H(s) \ge \sum_v p_\theta(v \mid s)H(sv) \quad \Longrightarrow \quad H(s) \ge \mathbb{E}_{p_\theta}[\Phi(Y) \mid s]. \tag{24}$$

The conclusion on the right can be obtained level by level from the leaves to the root. Thus, $p_\theta(s)H(s)$ is an upper bound on the subtree's target mass, and the sum of the child-node upper bounds after refinement does not exceed the parent-node upper bound.

Here, $H(s)$ controls the *expected continuation weight*, so sampling instead uses stepwise rejection: first select a starting point according to $p_\theta(s)H(s)$; at a nonterminal prefix r with a positive upper bound, choose the next step or reject with probabilities

$$\Pr(r \to rv) = \frac{p_\theta(v \mid r)H(rv)}{H(r)}, \qquad \Pr(\text{rej} \mid r) = 1 - \sum_v \frac{p_\theta(v \mid r)H(rv)}{H(r)} \tag{25}$$

Equation 24 ensures that the right-hand sides are valid probabilities. If generation proceeds from a frontier prefix $s = s_0$ to a complete sequence $s_L = y$, writing $s_{j+1} = s_j v_j$, the H factors along the path cancel, giving

$$\frac{p_\theta(s)H(s)}{E_\Phi(\mathcal{F})} \prod_{j=0}^{L-1} \frac{p_\theta(v_j \mid s_j)H(s_{j+1})}{H(s_j)} = \frac{p_\theta(y)\Phi(y)}{E_\Phi(\mathcal{F})},$$
$$E_\Phi(\mathcal{F}) = \sum_{s \in \mathcal{F}} p_\theta(s)H(s). \tag{26}$$

Thus, this construction also yields exact samples from the target distribution. Both constructions use the same prefix partitioning and refinement approach: B directly provides an envelope for each sequence, while H handles upper bounds in expectation through stepwise transitions.

C Proofs for the Analysis Section

C.1 Exactness Under Adaptive Refinement

Fix the complete history $\mathcal{H}_{k-1}$ before an attempt begins, which determines the active units, total envelope mass E_k, and continuation proposal. To handle token groups uniformly, temporarily view each active group (s, A) as the child prefixes $\{sv : v \in A\}$. Since tokens within a group are selected according to their power weights, the probability of actually selecting sv is

$$\frac{U(s,A)}{E_k} \cdot \frac{p_\theta(v \mid s)^\alpha}{\sum_{w \in A} p_\theta(w \mid s)^\alpha} = \frac{U(sv)}{E_k}. \tag{27}$$

Together with the existing prefix units, these effective starting prefixes form a mutually exclusive and exhaustive frontier. For a complete sequence $y_{1:T}$, let $y_{<t}$ be its unique effective starting prefix. By Eq. 5, the factors in the continuation proposal and acceptance probability cancel term by term:

$$\begin{aligned} \Pr(Y_k = y_{1:T}, \text{acc} \mid \mathcal{H}_{k-1}) &= \frac{U(y_{<t})}{E_k} \cdot q_{\mathcal{F}}(y_{t:T} \mid y_{<t}) \cdot A_q(y_{1:T}; y_{<t}) \\ &= \frac{p_\theta(y_{1:T})^\alpha}{E_k}, \\ \Pr(\text{acc} \mid \mathcal{H}_{k-1}) &= \frac{Z_\alpha}{E_k}. \end{aligned} \tag{28}$$

Therefore, $\Pr(Y_k = y_{1:T} \mid \text{acc}, \mathcal{H}_{k-1}) = \pi_\alpha(y_{1:T})$. This equality holds for any valid frontier determined by previous outcomes. Conditional on all previous attempts having been rejected, the k-th attempt still follows π_α if accepted; summing over the possible indices of the first acceptance shows that the first returned sample also follows π_α. Repeating the same argument conditional on the returned samples and the complete history of attempts shows that the conditional distribution of each subsequent returned sample remains π_α.

Under parallel execution, the attempts in a batch share a fixed envelope and use independent random numbers; after ordering them by attempt index, the same conditional argument applies. Token-group splitting at the end of the batch preserves the sum of weights, while frontier refinement preserves coverage and pointwise upper bounds, so the next batch continues to satisfy the premises of the proof.

C.2 The Dual Stochastic Oracle

We use the effective starting-prefix representation from the previous subsection: if an active unit is a token group, first view it as the individual child prefixes within the group. Fix the current state $\mathcal{F}$ and total envelope mass E; these effective starting regions remain nonoverlapping and cover all complete sequences. Equation 28 has shown that any complete sequence y is proposed and accepted with probability $p_\theta(y)^\alpha / E$. Summing over all complete sequences that pass through prefix r therefore gives

$$\begin{aligned} \Pr(r \preceq Y, \text{acc} \mid \mathcal{F}) &= \sum_{y \succeq r} \Pr(Y = y, \text{acc} \mid \mathcal{F}) \\ &= \frac{1}{E} \sum_{y \succeq r} p_\theta(y)^\alpha = \frac{M(r)}{E}. \end{aligned} \tag{29}$$

Taking r to be the empty prefix gives the acceptance rate of this attempt, $\Pr(\text{acc} \mid \mathcal{F}) = Z_\alpha / E$. Normalizing the expression above by this rate yields Eq. 10 in the main text: the frequency with which accepted samples pass through a prefix equals the target probability of that prefix region.

For rejection feedback, first consider *which region an attempt starts from.* For any effective starting prefix s, the probability of starting at s is $U(s)/E$, while the joint probability of starting at s and eventually being accepted is $M(s)/E$. Thus, conditional on selecting s,

the acceptance rate is $M(s)/U(s)$ and the rejection rate is $1 - M(s)/U(s)$; multiplying the starting-region probability by this rejection rate gives

$$\begin{aligned}\Pr(S = s, \text{rej} \mid \mathcal{F}) &= \frac{U(s)}{E}\left(1 - \frac{M(s)}{U(s)}\right) \\ &= \frac{U(s) - M(s)}{E}.\end{aligned} \tag{30}$$

Summing the rejection probabilities over all regions and using $\sum_s U(s) = E$ and $\sum_s M(s) = Z_\alpha$ yields

$$\Pr(\text{rej} \mid \mathcal{F}) = \frac{E - Z_\alpha}{E}. \tag{31}$$

Normalizing Eq. 30 by this probability then gives Eq. 11 in the main text. Intuitively, the more a region's upper bound exceeds its true mass, the more likely attempts starting from that region are to appear in the rejection records. The algorithm need not know $M(s)$ in advance; repeated sampling itself produces this preference.

We can further examine *the step along a path at which rejection occurs*. For a nonterminal prefix r, the envelope mass that a single refinement can remove is $\Delta(r) = U(r)[1 - C_\alpha(r)]$. True mass is additive across child nodes, and the difference between the current upper bound and the child-node upper bounds is exactly $\Delta(r)$, so

$$U(r) - M(r) = \Delta(r) + \sum_v \big[U(rv) - M(rv)\big]. \tag{32}$$

At a terminal leaf, $U(y) = M(y)$. Applying the expression above repeatedly downward from s gives

$$U(s) - M(s) = \sum_{r \succeq s,\ r \text{ nonterminal}} \Delta(r). \tag{33}$$

In other words, the envelope gap of a region is exactly the sum of the local gains that can still be removed through level-by-level refinement within that region.

For tokenwise decisions, let R be the first prefix at which the cumulative acceptance probability falls below the threshold, and consider a nonterminal prefix r below an effective starting point. Under the low-temperature proposal, whenever the one-step test at r is passed and the path proceeds to child node rv, the "proposal probability $\times$ pass probability" at that step is

$$q_{\mathcal{F}}(v \mid r) C_\alpha(r) = p_\theta(v \mid r)^\alpha. \tag{34}$$

Therefore, the probability of reaching r from the effective starting point without any earlier rejection is $U(r)/E$. The probability of failure at r is $1 - C_\alpha(r)$, so

$$\Pr(R = r, \text{rej} \mid \mathcal{F}) = \frac{U(r)}{E}\big[1 - C_\alpha(r)\big] = \frac{\Delta(r)}{E}. \tag{35}$$

Summing over all possible R below a starting region recovers Eq. 30. This shows that rejection records can indicate both "where a large gap exists" through the starting region and "which refinement step may directly shrink the envelope" through the specific stopping position; accepted paths indicate which prefixes subsequent samples are more likely to revisit.