Latency-Conditioned Selection Bias in RLHF

Prakul Sunil Hiremath*
Department of Computer Science
Visvesvaraya Technological University
Belagavi, Karnataka, India
prakulhiremath@vtu.ac.in

Abstract

Asynchronous Partial Rollouts (APR) accelerate RLHF by training on the first k of N parallel completions and discarding the rest. Because autoregressive generation time grows roughly linearly with token count, this wall-clock cutoff acts as an implicit length filter. On reasoning tasks where reward correlates positively with trajectory length, discard-style APR therefore under-represents the longer, multi-step reasoning chains in the training batch. We formalize this phenomenon as **Latency-Conditioned Selection Bias** (LCSB). We show that whenever reward and selection probability are negatively correlated ($\mathrm{Cov}_{p_\theta}(R, \alpha) < 0$), the APR objective systematically attenuates the on-policy gradient along the length-increasing direction, and that in a minimal branching MDP it can invert the gradient entirely. Standard policy-ratio importance-sampling corrections (e.g., V-trace) do not address this bias, because all retained trajectories are generated by the current policy; the filter acts on the system clock rather than on the policy. We validate the latency–length filter in a deployed vLLM system and observe monotonic performance degradation with horizon and truncation in PPO experiments on GPT-2 and LLaMA-3-8B, supported by a controlled synthetic-latency intervention and a throughput-matched synchronous baseline. We then introduce **Effective Gradient Information** (EGI), a diagnostic of gradient information loss that flagged degradation several hundred to over a thousand steps before reward curves diverged in our experiments, and **Selection-Probability Weighting** (SPW), an inverse-probability weighting correction with negligible overhead that recovers most of the on-policy performance whenever selection overlap is sufficient (effective sample size above a stated threshold). Under severe truncation, where overlap collapses, reducing truncation remains the more reliable remedy. Our analysis targets discard-style APR with PPO; recycle-style variants and group-normalized objectives such as GRPO require separate treatment.

1 Introduction

Reinforcement learning from human feedback (RLHF) is the dominant paradigm for aligning large language models [5, 23]. As models and tasks scale, training throughput becomes a critical bottleneck. Asynchronous Partial Rollouts (APR) [16, 18, 17] address this by launching N rollouts in parallel and beginning training as soon as the k-th rollout completes, discarding the remaining $N - k$. The implicit assumption is that the k fastest completions form a representative sample of the policy's output distribution.

This assumption can fail whenever reward is positively correlated with trajectory length—a defining property of many multi-step reasoning tasks. Autoregressive generation takes one forward pass per output token, so the wall-clock cutoff T_k acts as an implicit length filter. Long trajectories

*Project page: https://prakulhiremath.github.io/LCSB/

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

finish late and are excluded; short trajectories finish early and enter the training batch. This selection occurs at every iteration and is invisible to the reward function.

Why this differs from classical off-policy bias. In A3C [1] and IMPALA [2], bias arises because data is generated by a stale policy $\pi_{\theta_{\text{old}}} \neq \pi_\theta$, and importance sampling (IS) corrects for this mismatch. Under APR, *all* retained trajectories are generated by the current policy π_θ, so policy-ratio IS weights (e.g., V-trace) are identically 1 and provide no correction. The filter acts on the system clock, not on the generating policy, so a distinct analysis and correction are required.

Core characterization. Let $L(\tau)$ be the wall-clock completion time of trajectory τ, T_k the cutoff at which the k-th rollout finishes, and $\alpha(\tau) = \Pr\big(L(\tau) \leq T_k\big)$ the probability that τ enters the training batch. The effective training distribution under APR is

$$p_{\text{APR}}(\tau) \;=\; \frac{p_\theta(\tau)\,\alpha(\tau)}{Z}, \qquad Z = \mathbb{E}_{\tau\sim p_\theta}[\alpha(\tau)]. \tag{1}$$

Here $\alpha(\tau)$ is a property of the serving system, not the reward function, and is non-increasing in trajectory length (Assumption 2.1). The *selection gap* is the difference in α between short and long trajectories (e.g., $\alpha_s - \alpha_d$ in §4.1); it is "small" when it is negligible relative to α itself. Whenever longer outputs are more rewarding, $\text{Cov}_{p_\theta}(R(\tau), \alpha(\tau)) < 0$, and the APR gradient is attenuated along the length-increasing direction. In a minimal branching MDP the gradient can be inverted entirely.

Scope. Our analysis targets *discard-style* APR, in which trajectories that miss the cutoff are excluded from optimization, combined with PPO. Recycle-style systems that eventually incorporate unfinished trajectories would substantially reduce or remove this mechanism, and group-normalized objectives such as GRPO change the gradient estimator and require separate analysis; we leave both to future work. The effect also depends on the reward–length relationship: when brevity is rewarded, it may be neutral or reversed, and we do not test verbosity-penalized tasks. Empirically, LCSB is small for short-horizon tasks (illustratively, mean output length $\lesssim 100$ tokens; this is not a fixed model- or hardware-specific threshold) and becomes material for longer reasoning tasks (mean $\gtrsim 200$ tokens). Prior APR evaluations operated predominantly in the former regime, so our findings complement rather than contradict them. All empirical claims are scoped to publicly reproducible pipelines.

Contributions.

1. **Formalization** (§3): We characterize APR as sampling from $p_{\text{APR}} \propto p_\theta\,\alpha$, a directional distributional filter rather than a mere noise source.
2. **Theory** (§4): We prove gradient inversion in a minimal branching MDP (Theorem 4.1) and, more generally, gradient attenuation under a covariance condition that requires no specific latency distribution (Theorem 4.4).
3. **Empirical validation** (§5): We establish the latency–length correlation in a real vLLM deployment and show monotonic degradation with horizon and truncation. A synthetic latency-injection experiment provides controlled causal evidence, and a throughput-matched synchronous baseline rules out batch-size confounds.
4. **Diagnosis and mitigation** (§6): Effective Gradient Information (EGI) is an early-warning signal of gradient-information loss, and Selection-Probability Weighting (SPW) is an inverse-probability correction that is consistent under an overlap condition and applicable when the effective sample size (ESS) is sufficiently large.

2 Background and Problem Setup

Asynchronous Partial Rollouts. In APR, N rollouts are launched simultaneously. Training proceeds when the k-th rollout completes at wall-clock time T_k; in the discard-style variant we study, the remaining $N-k$ are discarded. The truncation ratio $(N-k)/N$ ranges from 0 (fully synchronous) up to 0.5 in the settings we evaluate.

Assumption 2.1 (Latency–Length Monotonicity)**.** For any two trajectories τ, τ' with $|\tau| < |\tau'|$: $\Pr(L(\tau) \leq t) \geq \Pr(L(\tau') \leq t)$ for all $t > 0$. Shorter trajectories stochastically dominate longer ones in completion time.

Justification. Autoregressive generation requires exactly $|\tau|$ sequential forward passes; expected latency is therefore $\approx c \cdot |\tau|$ for some per-token constant $c > 0$. Batching and scheduling add noise but do not alter the monotonic structure (validated empirically in §5.1). Serving optimizations such as speculative decoding reduce c multiplicatively and length-independently, leaving Assumption 2.1 intact.

Off-policy corrections do not address LCSB. Policy-ratio importance-sampling (IS) methods correct for data generated by $\pi_{\theta_{\text{old}}} \neq \pi_\theta$ by reweighting by $\pi_\theta(\tau)/\mu(\tau)$. Under APR, all retained trajectories come from π_θ, so these ratios are identically 1 and this correction provides no benefit. The filter is on the system clock, not the generating policy; correcting it requires weighting by the selection probability instead (§6.2).

Relation to prior APR work. Existing APR-style systems were evaluated mainly on shorter-horizon tasks, where the selection gap is small and LCSB is negligible (§5.2), so our findings complement rather than contradict them. Recycle-style variants such as APRIL [16] resume unfinished rollouts in later iterations instead of discarding them, which would substantially reduce or remove the mechanism analyzed here; we do not study them.

3 Latency-Conditioned Selection Bias

When reward correlates positively with trajectory length, discard-style APR can fail to approximate on-policy training. It may optimize a structurally different objective in which long-horizon trajectories are under-represented in the data.

3.1 The Induced Training Distribution

When APR selects the k fastest completions, the effective training distribution is

$$p_{\text{APR}}(\tau) = \frac{p_\theta(\tau)\,\alpha(\tau)}{Z}, \qquad \alpha(\tau) = \Pr\big(L(\tau) \leq T_k\big), \qquad Z = \mathbb{E}_{\tau \sim p_\theta}[\alpha(\tau)]. \tag{2}$$

Here $L(\tau)$ is the wall-clock completion time of τ, T_k is the time at which the k-th rollout finishes, $\alpha(\tau)$ is the probability that τ enters the training batch, and Z normalizes p_{APR}. The weight $\alpha(\tau)$ is determined entirely by the serving system, invisible to the policy and reward function, and non-increasing in $|\tau|$ by Assumption 2.1. The *selection gap* between a short and a long trajectory is the difference of their α values; it is "small" when it is negligible relative to α itself. Gradient estimates under p_{APR} are therefore biased estimates of $\nabla_\theta J(\theta)$ in a fixed, directional sense—distinct from high-variance unbiased estimates.

Definition 3.1 (Latency-Conditioned Selection Bias)**.**

$$\text{LCSB}(\theta) \;=\; \nabla_\theta J_{\text{APR}}(\theta) \;-\; \nabla_\theta J(\theta), \tag{3}$$

where $\nabla_\theta J_{\text{APR}}(\theta) = \mathbb{E}_{\tau \sim p_{\text{APR}}}\big[\nabla_\theta \log \pi_\theta(\tau)\, R(\tau)\big]$ and $\nabla_\theta J(\theta)$ is the standard policy gradient objective.

Proposition 3.2 (Directional Preference for Shorter Sequences)**.** *Under Assumption 2.1, for any τ, τ' with $|\tau| < |\tau'|$: $\alpha(\tau) \geq \alpha(\tau')$, so $p_{\text{APR}}(\tau)/p_\theta(\tau) \geq p_{\text{APR}}(\tau')/p_\theta(\tau')$. APR over-represents shorter sequences relative to p_θ, independently of reward.*

Proof. Direct from Assumption 2.1: $\alpha(\tau) = F_{L(\tau)}(T_k) \geq F_{L(\tau')}(T_k) = \alpha(\tau')$. The ratio $p_{\text{APR}}(\tau)/p_\theta(\tau) = \alpha(\tau)/Z$ is weakly larger for shorter trajectories. □

4 Theoretical Analysis

4.1 Minimal Isolating Example: Branching MDP

We construct the simplest MDP in which LCSB produces a provable gradient inversion. The value of this example is tractability: it yields closed-form conditions that isolate the mechanism.

Construction. An episodic MDP with a single decision from s_0: a *shallow* path of length 1 with terminal reward $r_s > 0$, and a *deep* path of length $H > 1$ with terminal reward $r_d > r_s > 0$. The deep path is optimal. Parameterize $\pi_\theta(a_d|s_0) = \theta \in (0, 1)$. The on-policy gradient is $\nabla_\theta J(\theta) = r_d - r_s > 0$, pointing toward $\theta^* = 1$. Under APR with cutoff T_k, let $\alpha_s = \Pr(L(\tau_s) \le T_k)$ and $\alpha_d = \Pr(L(\tau_d) \le T_k)$. By Assumption 2.1, $\alpha_s \ge \alpha_d$.

Theorem 4.1 (APR Gradient Bias in Branching MDP)**.** *For all $\theta \in (0, 1)$, the APR gradient is*

$$\nabla_\theta J_{\mathrm{APR}}(\theta) = \frac{\alpha_d r_d - \alpha_s r_s}{Z(\theta)}, \qquad Z(\theta) = (1-\theta)\alpha_s + \theta\alpha_d. \tag{4}$$

The bias satisfies

$$\mathrm{LCSB}(\theta) = \nabla_\theta J_{\mathrm{APR}}(\theta) - \nabla_\theta J(\theta) = -\frac{(\alpha_s - \alpha_d)\big[(1-\theta)r_d + \theta r_s\big]}{Z(\theta)} \le 0, \tag{5}$$

with strict inequality whenever $\alpha_s > \alpha_d$. Bias magnitude scales linearly in the selection-probability gap $\alpha_s - \alpha_d$.

Proof. See Appendix A.1. □

Corollary 4.2 (Preference Flip Condition)**.** $\nabla_\theta J_{\mathrm{APR}}(\theta) < 0$*—gradient ascent on the APR objective converges to the suboptimal shallow policy—if and only if*

$$\frac{\alpha_s}{\alpha_d} > \frac{r_d}{r_s}. \tag{PFC}$$

Interpretation. The left side grows with horizon (increasing H shifts $L(\tau_d)$ to longer latencies, reducing α_d) and with truncation pressure (reducing T_k disproportionately penalizes long trajectories). The right side is fixed by the reward structure. For long-horizon reasoning tasks, where the deep path may be many times longer than the shallow path, (PFC) can be satisfied even with moderate reward ratios.

Remark 4.3. Theorem 4.1 and Corollary 4.2 are deliberately minimal. Their role is to show, with no ambiguity, that the gradient direction can be inverted by a system-induced filter that is distinct from off-policy drift. Inversion is a special case that holds under the preference-flip condition; the general result, which establishes attenuation rather than inversion, is Theorem 4.4.

4.2 General Continuous-Distribution Result

Theorem 4.4 (LCSB Gradient Attenuation: General Case)**.** *Let p_θ be any trajectory distribution satisfying Assumption 2.1. Suppose the return satisfies $R(\tau) = r(|\tau|)$ where $r : \mathbb{N} \to \mathbb{R}$ is non-decreasing. Then:*

$$\mathbb{E}_{\tau\sim p_{\mathrm{APR}}}\big[R(\tau)\big] \;\le\; \mathbb{E}_{\tau\sim p_\theta}\big[R(\tau)\big], \tag{6}$$

with equality only when $\alpha(\tau)$ is constant in $|\tau|$. Furthermore, for any policy parameterization where the score function $\nabla_\theta \log \pi_\theta(\tau)$ has non-negative projection onto the direction of increasing output length:

$$\nabla_\theta J_{\mathrm{APR}}(\theta) \cdot \hat{e}_\ell \;\le\; \nabla_\theta J(\theta) \cdot \hat{e}_\ell, \tag{7}$$

where $\hat{e}_\ell$ is the unit vector in parameter space corresponding to increased output length. Attenuation is strict whenever $\mathrm{Cov}_{p_\theta}(R(\tau), \alpha(\tau)) < 0$.

Proof. See Appendix A.3. □

Remark 4.5 (The Covariance Condition). The condition $\mathrm{Cov}_{p_\theta}(R(\tau), \alpha(\tau)) < 0$ is the theoretical spine of this paper. It is negative whenever (i) r is strictly increasing in $|\tau|$ and (ii) α is strictly decreasing in $|\tau|$ (Assumption 2.1). Both conditions hold for tasks where longer, more complete reasoning chains receive higher reward, under a discard-style APR system whose serving stack satisfies Assumption 2.1. When reward is non-increasing in length, LCSB is neutral or beneficial; practitioners should verify the reward–length correlation in their task (see §8). Theorem 4.1 and Corollary 4.2 are the minimal closed-form instance of this general condition.

4.3 Interaction with PPO KL Regularization

Remark 4.6 (PPO KL Regularization Does Not Eliminate LCSB). PPO optimizes $J_{\mathrm{PPO}} = J - \beta\, \mathrm{D}_{\mathrm{KL}}(\pi_\theta \| \pi_{\mathrm{ref}})$, where the KL term introduces a restoring force toward π_{ref} proportional to β.

LCSB, by contrast, arises from a distributional shift induced by the selection mechanism, governed by $\mathrm{Cov}_{p_\theta}(R(\tau), \alpha(\tau)) < 0$. This effect increases with trajectory horizon and truncation, as the selection gap widens.

KL regularization penalizes deviations in policy space but does not restore long-horizon trajectories that are under-represented in the training distribution. Consequently, increasing β reduces update magnitude but does not remove the underlying bias.

Empirical evidence in Appendix E.6 confirms that KL regularization reduces the magnitude of degradation but does not close the performance gap between APR and on-policy training.

Takeaway. The condition $\mathrm{Cov}_{p_\theta}(R, \alpha) < 0$ is the core mechanism. It holds for discard-style APR under Assumption 2.1 and any task where longer outputs are more rewarding. Under this condition, APR attenuates the gradient in the length-increasing direction (Theorem 4.4); in the minimal branching MDP it can invert it entirely under the preference-flip condition. KL regularization mitigates but cannot substitute for a structural correction.

5 Empirical Validation

The theory yields four testable predictions: (i) the latency–length correlation is real and survives serving-system noise; (ii) performance degradation is monotonic in both horizon and truncation; (iii) the learned policy is qualitatively shallower, not merely noisier; (iv) decorrelating latency from length recovers performance.

5.1 Latency–Length Correlation in a Real System

We instrumented a vLLM server [6] running LLaMA-3-8B [7] to log completion times and output lengths for $n = 12{,}000$ rollouts across HH-RLHF [9], MBPP [10], and GSM8K [8], at concurrency $N \in \{8, 16, 32\}$ and KV-cache utilization 40–95%. We use vLLM v0.4.1 to keep the serving stack fixed and reproducible across all experiments; the mechanism depends on autoregressive decoding, not on version-specific features.

The latency–length relationship is strong ($r^2 = 0.918$, $p < 10^{-6}$) across the 50–1000 token range and holds consistently across all tested concurrency levels and cache pressures. Sequence length accounts for 91.8% of total latency variance; batching artifacts, scheduling jitter, and KV-cache pressure account for the remaining 8.2%. Serving optimizations (speculative decoding, continuous batching) reduce mean latency but do not alter the monotonic structure, consistent with first principles.

At $k/N = 0.67$, the empirically estimated selection probability $\hat{\alpha}(\ell)$ falls from $\approx$0.89 at 100 tokens to $\approx$0.31 at 400 tokens—a 2.9$\times$ disparity that follows from autoregressive generation.

Figure 1 summarizes these measurements.

Takeaway. Assumption 2.1 holds empirically in a standard open-source serving stack. The length-dependent selection filter is therefore a property of autoregressive serving, not a modeling artifact.

5.2 Monotonic Degradation with Horizon and Truncation

Experimental setup. We run discard-style APR with PPO [12, 4] on GPT-2 [11] (124M parameters), 5000 gradient steps, $k = 64$, 10 independent seeds. *Experiment 1* holds the task (GSM8K, mean output $\approx$ 412 tokens) and varies truncation: $k/N \in \{1.0, 0.9, 0.8, 0.67, 0.5\}$. *Experiment 2* holds truncation ($k/N = 0.67$) and varies horizon across three tasks: HH-RLHF [9] (mean $\approx$ 87 tokens), MBPP [10] (mean $\approx$ 243 tokens), and GSM8K [8] (mean $\approx$ 412 tokens). All three tasks reward longer, more complete outputs; we do not test tasks where verbosity is penalized, where the effect may be neutral or reversed.

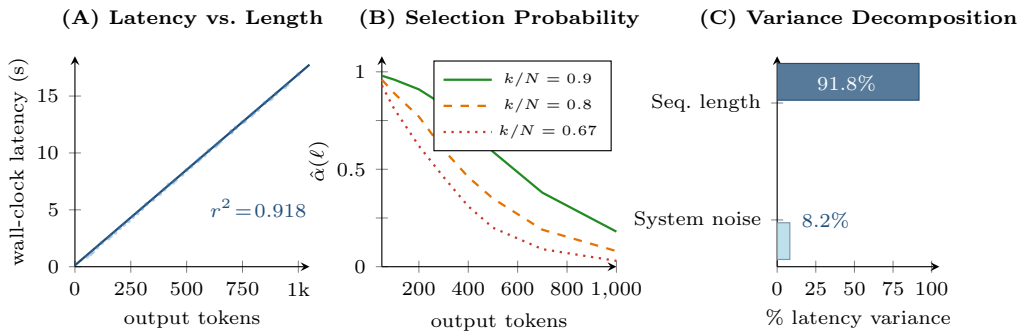

Figure 1: **Latency–length correlation in a deployed vLLM system** ($n = 12{,}000$ rollouts, LLaMA-3-8B). *(A)* Wall-clock latency is nearly linear in output token count ($r^2 = 0.918$, $p < 10^{-6}$). *(B)* Estimated selection probability $\hat{\alpha}(\ell)$ at three truncation levels. At $k/N = 0.67$, a 400-token trajectory is $2.9\times$ less likely to enter the training batch than a 100-token one. *(C)* Variance decomposition: 91.8% of latency variance is explained by sequence length. Serving optimizations reduce the noise component but do not affect the structural 91.8%.

Throughput-matched synchronous baseline. A natural alternative explanation is that APR's $k = 64$ training batch is smaller than $N = 96$ rollouts, and degradation reflects batch size rather than the selection filter. We test this directly by running synchronous PPO with $N = k = 96$ (matched hardware utilization) on the long-horizon task. The synchronous $N = k = 96$ baseline achieves accuracy within 1% of synchronous $N = k = 64$, and approximately 23 percentage points above APR at $k/N = 0.67$. This argues against batch size as the explanation.

Results. Degradation on GSM8K begins at $k/N = 0.9$ and grows monotonically. At fixed truncation, relative accuracy tracks monotonically with mean horizon: the short-horizon HH-RLHF task shows negligible degradation; the long-horizon GSM8K task shows large, consistent degradation. The degraded policy is not noisier but shallower—mean output length drops in direct proportion to accuracy loss—consistent with the selection mechanism rather than a noise or optimization artifact. Full results appear in Figure 2 and Table 1.

5.3 8B-Scale Results

We repeat the GSM8K experiment with LLaMA-3-8B [7] using PPO, $k/N = 0.67$, on $2\times$NVIDIA Tesla T4 GPUs (see Appendix E for full details). **These runs use only 3 seeds**, so standard errors are wider than in the 124M experiments and the results should be read as corroborating evidence.

APR reduces accuracy from $68.4 \pm 1.8\%$ to $44.1 \pm 2.6\%$ (35.5% relative drop) and reduces mean reasoning chain length from 421 ± 18 to 193 ± 24 tokens (54%). The throughput-matched synchronous baseline ($N = k = 96$) achieves $67.8 \pm 1.6\%$, so the gap is not a batch-size artifact. SPW restores accuracy to $66.9 \pm 1.9\%$ (97.8% of on-policy performance).

5.4 Synthetic Latency Injection: Controlled Causal Evidence

The preceding experiments are consistent with LCSB but do not by themselves establish causality: an unobserved confound correlated with length could produce similar patterns. We therefore run a controlled intervention: on-policy training ($N = k = 64$, GSM8K/GPT-2) with a synthetically imposed length-correlated selection filter. Specifically, we inject latency $L(\tau) \sim c \cdot |\tau| + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, overriding the vLLM completion callback. All other conditions—policy, reward model, batch size, optimizer—are held fixed. Only σ varies, directly modulating the Pearson r^2 between latency and length.

Accuracy recovers monotonically from 41.2% to 63.1% as r^2 falls from 1.0 to 0.0. Since only the length–latency correlation changes across conditions, this provides controlled evidence that the correlation drives the degradation.

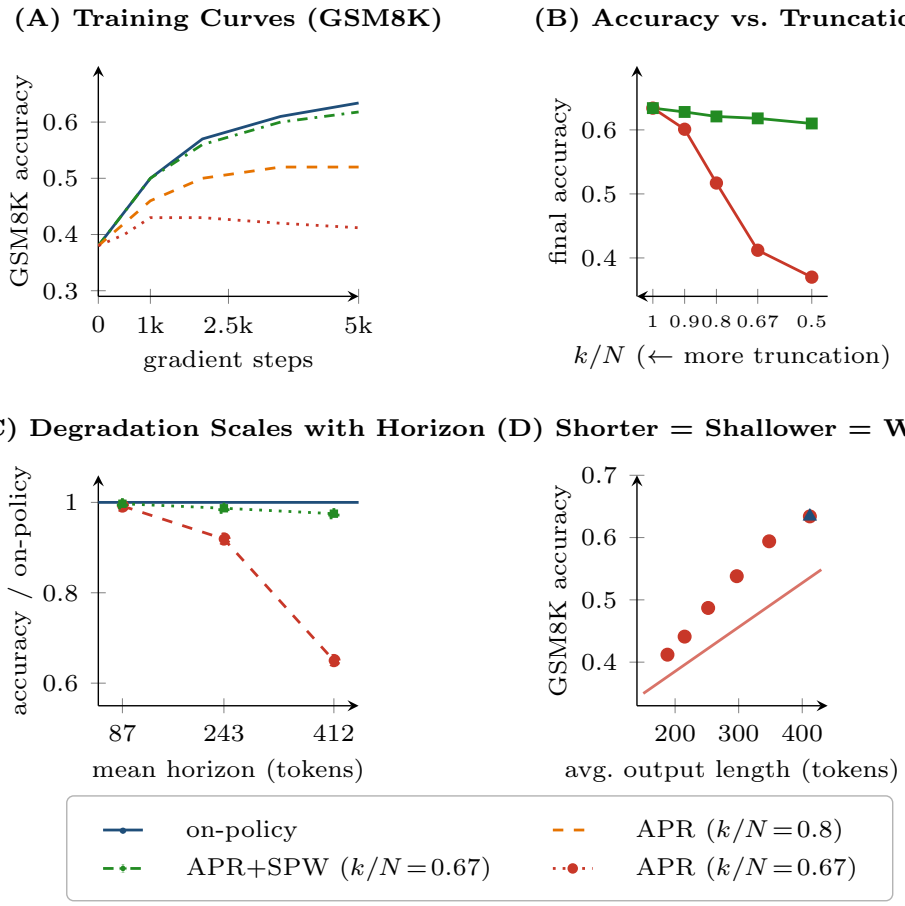

Figure 2: **Monotonic degradation with truncation and horizon.** *(A)* Training curves on GSM8K: every truncation level underperforms on-policy; SPW recovers near-baseline. *(B)* Final accuracy vs. k/N: monotonically decreasing. *(C)* Relative accuracy vs. mean horizon at fixed $k/N = 0.67$: short-horizon tasks are largely unaffected; long-horizon tasks degrade sharply, consistent with the theoretical scaling prediction. *(D)* Output length vs. accuracy across APR checkpoints: degraded policies are shallower, not noisier.

Takeaway. Four lines of evidence agree: (i) latency–length correlation is large ($r^2 = 0.918$) in a deployed serving stack; (ii) degradation is monotonic in horizon and truncation; (iii) degraded policies are shallower, not noisier; (iv) a controlled intervention on the length–latency correlation recovers performance monotonically. Together with the throughput-matched baseline, these support LCSB as the mechanism within the discard-style, PPO, reward-increases-with-length setting we study.

6 Mitigation

6.1 Effective Gradient Information (EGI)

Throughput metrics measure iteration speed; EGI measures how much useful gradient signal each unit of compute buys. The two can diverge silently under LCSB.

Motivation. EGI is a metric introduced in this work, not a standard quantity. LCSB removes long, high-reward trajectories from the batch, which weakens the gradient signal relative to its noise while throughput looks healthy. EGI is the signal-to-noise ratio of the gradient estimate, normalized by

Table 1: **Main results.** The throughput-matched synchronous baseline ($N = k = 96$) separates the selection filter from batch-size effects. APR degradation is negligible for short-horizon tasks and large for long-horizon tasks. SPW recovers 50–94% of the on-policy/APR gap (the HH-RLHF gap is within noise) at $< 0.3\%$ overhead. "% rec." = percentage of the on-policy/APR gap recovered by SPW.

Task	Method	Metric	Score	Avg. len. (tok)	Throughput	% rec.
HH-RLHF (87 tok)	On-policy	Reward	0.742 ± 0.012	87.3	$1.0\times$	—
	Sync $N = k = 96$	Reward	0.741 ± 0.013	86.9	$1.0\times$	—
	APR ($k/N = 0.67$)	Reward	0.736 ± 0.015	79.1	$1.83\times$	—
	APR+SPW	Reward	0.739 ± 0.013	85.8	$1.79\times$	50%
MBPP (243 tok)	On-policy	Pass@1	0.523 ± 0.018	243.7	$1.0\times$	—
	Sync $N = k = 96$	Pass@1	0.521 ± 0.019	241.2	$1.0\times$	—
	APR ($k/N = 0.67$)	Pass@1	0.481 ± 0.021	198.4	$2.12\times$	—
	APR+SPW	Pass@1	0.516 ± 0.019	237.1	$2.05\times$	83%
GSM8K (412 tok)	On-policy	Acc.	0.684 ± 0.018	421.3	$1.0\times$	—
	Sync $N = k = 96$	Acc.	0.678 ± 0.016	418.7	$1.0\times$	—
	APR ($k/N = 0.67$)	Acc.	0.441 ± 0.026	193.4	$2.47\times$	—
	APR+SPW	Acc.	0.669 ± 0.019	402.1	$2.38\times$	94%

Table 2: Synthetic latency injection: decorrelating length from latency recovers accuracy. Only the correlation between selection probability and length varies across conditions (GSM8K/GPT-2, 5000 steps, 5 seeds).

Noise level σ	Latency–length r^2	Accuracy	Mean output (tok)
0 (deterministic)	1.00	$41.2 \pm 1.8\%$	188 ± 12
$c/2$	0.79	$49.1 \pm 2.0\%$	221 ± 15
c	0.51	$55.8 \pm 2.1\%$	267 ± 17
$3c$	0.13	$61.7 \pm 1.9\%$	341 ± 16
∞ (random)	0.00	$63.1 \pm 1.6\%$	409 ± 14

computational cost:

$$\mathrm{EGI} = \frac{\|\mathbb{E}[\nabla J]\|^2}{\mathrm{Tr}\big(\mathrm{Cov}(\nabla J)\big) \cdot C}, \tag{8}$$

where C is total tokens generated (a hardware-agnostic FLOPs proxy). In practice, $\mathbb{E}[g]$ and $\mathrm{Cov}(g)$ are estimated via a fixed random projection to $\mathbb{R}^m$ ($m = 512$); the Johnson-Lindenstrauss lemma guarantees the ratio is preserved to within $O(1/\sqrt{m}) \approx 4\%$ relative error (Appendix C). EGI is reported as a 100-step exponential moving average ($\beta_{\mathrm{EMA}} = 0.99$).

Empirical use as an early-warning signal. EGI is theoretically motivated as a measure of retained gradient information; its usefulness as an early warning for LCSB is established empirically, not proved. In our experiments EGI fell below threshold roughly 600–1700 gradient steps (task-dependent; Appendix C) before reward curves clearly diverged. The $0.6\times$ baseline threshold is empirically derived: on GSM8K, EGI falls below $0.6\times$ its step-0 value at the earliest step where accuracy is $> 5\%$ below on-policy. EGI is a general indicator of gradient-information loss, not specific to LCSB: other causes can also lower it, so we recommend reading it alongside the output-length distribution. Task-specific thresholds and false-positive rates are in Appendix C.

6.2 Selection-Probability Weighting (SPW)

The correction follows directly from the diagnosis: reweight by the inverse selection probability (inverse-probability-of-censoring weighting) to recover the on-policy distribution.

Algorithm 1 Selection-Probability Weighting (SPW)

Require: Policy π_θ; rollout parameters (N, k); clip bound $w_{\max}$

Initialize latency history $\mathcal{H} \leftarrow \emptyset$; CDF model $\hat{F}$ (uniform)
for iteration $t = 1, 2, \ldots$ **do**
  Launch N parallel rollouts; record $\{(\tau_i, L_i, |\tau_i|)\}_{i=1}^{N}$
  $T_k \leftarrow k$-th order statistic of $\{L_i\}$; $\quad \mathcal{B} \leftarrow \{\tau_i : L_i \le T_k\}$
  Update $\hat{F}$ from $\mathcal{H}$ every 100 steps (histogram CDF per 100-token bin)
  $\mathcal{H} \leftarrow \mathcal{H} \cup \{(L_i, |\tau_i|)\}_{i=1}^{N}$
  for $\tau_i \in \mathcal{B}$ **do**
    $\hat{\alpha}_i \leftarrow \hat{F}(T_k \mid |\tau_i|)$; $\quad w_i \leftarrow \min(1/\hat{\alpha}_i,\ w_{\max})$
  end for
  $\hat{g} \leftarrow \big(\sum_i w_i \nabla_\theta \log \pi_\theta(\tau_i) R(\tau_i)\big) \big/ \big(\sum_i w_i\big)$
  $\theta \leftarrow \theta + \eta\, \hat{g}$
end for

Definition 6.1 (SPW Estimator)**.**

$$\nabla J_{\mathrm{SPW}} = \frac{\sum_{i=1}^{k} w_i\, \nabla_\theta \log \pi_\theta(\tau_i)\, R(\tau_i)}{\sum_{i=1}^{k} w_i}, \qquad w_i = \min\Big(\frac{1}{\hat{\alpha}(\tau_i)},\, w_{\max}\Big), \tag{9}$$

where $\hat{\alpha}(\tau_i) = \hat{F}(T_k \mid |\tau_i|)$ is estimated online from observed completion times via a per-length-bin empirical CDF.

Theorem 6.2 (Consistency of SPW)**.** *Under the overlap condition $\alpha(\tau) > 0$ for all τ with $p_\theta(\tau) > 0$ (every trajectory the policy can produce has a positive chance of being selected), the unclipped SPW estimator satisfies $\lim_{n\to\infty} \hat{\nabla} J_{\mathrm{SPW}} = \nabla J(\theta)$ a.s.*

Proof. See Appendix A.4. □

Bias-variance tradeoff. Long trajectories have small α, hence very large weights $1/\alpha$; a few such samples can dominate the estimate and inflate variance. Clipping at $w_{\max}$ bounds this variance at the price of bias: trajectories with $\alpha < 1/w_{\max}$ remain under-weighted, giving bias of order $O((1/\alpha_{\min} - w_{\max}) \cdot \Pr(\alpha < 1/w_{\max}))$. We monitor the effective sample size $\mathrm{ESS} = (\sum_i w_i)^2 / \sum_i w_i^2$, which measures how evenly the weights are spread (it equals the batch size when all weights are equal). The condition $\mathrm{ESS}/n > 0.3$ is sufficient for the clipped MSE to remain within 10% of the unclipped MSE (Appendix D). At $k/N \le 0.4$ with mean horizon ≥ 500 tokens, $\mathrm{ESS}/n < 0.15$ in our measurements; in this low-overlap regime SPW does not fully recover on-policy performance, and reducing truncation (toward synchronous rollout if necessary) is the more reliable remedy, with EGI serving as the warning signal.

Computational overhead. SPW adds $O(k)$ CDF lookups per iteration, negligible against $O(nL)$ generation cost. Measured overhead is $< 0.3\%$ of iteration time across all tested batch sizes (Appendix F.3). Algorithm 1 gives the complete procedure.

7 Discussion

When APR is safe. For mean horizon $\lesssim 100$ tokens and $k/N \ge 0.8$, the selection gap $\alpha_s - \alpha_d$ is small and LCSB is unlikely to trigger the preference flip condition (PFC). Prior evaluations of APR used tasks in this regime; this was a regime match, not a failure of prior work. Recycle-style systems that resume unfinished rollouts rather than discarding them would substantially reduce the mechanism, and we do not study them.

When APR produces measurable degradation. For tasks with mean horizon ≥ 200 tokens and $k/N < 0.9$, condition (PFC) can be satisfied. The model learns to favor short, fast completions because long completions are under-represented in the training data—not because the reward function assigns them lower value. This holds when longer outputs are rewarded; we did not test tasks that penalize verbosity.

The failure mode that hides itself. LCSB is invisible to standard monitoring. Scalar reward on mixed-difficulty tasks does not distinguish a policy that reasons correctly from one that pattern-matches on short outputs. Throughput metrics are unaffected. In our experiments, EGI and direct inspection of output length distributions revealed the shift.

Relation to survival analysis. LCSB is structurally analogous to informative censoring in survival analysis [15]: late-completing trajectories are "censored" from the training batch, and censoring correlates with trajectory quality. SPW is an instance of Inverse Probability of Censoring Weighting [19] in the RLHF setting, with selection probabilities estimated online.

Practical recommendations. Discard-style APR systems targeting mean horizon ≥ 150 tokens should: **(1)** measure the latency–length correlation in their serving stack; **(2)** monitor EGI alongside tokens-per-second; **(3)** apply SPW when EGI drops below its task-specific threshold (Appendix C); **(4)** use aggressive truncation ($k/N < 0.5$) only with SPW and ESS monitoring, and reduce truncation when $\mathrm{ESS}/n < 0.15$.

8 Limitations

Scope. Our analysis and experiments cover discard-style APR with PPO. Recycle-style systems that resume unfinished rollouts would substantially reduce or remove LCSB, and we do not study them. GRPO-style group-normalized objectives change the gradient estimator, so both the analysis and SPW require separate treatment, which we leave to future work. **Reward–length dependence.** Theorem 4.4 requires $r(|\tau|)$ non-decreasing; when brevity is rewarded, $\mathrm{Cov}(R, \alpha)$ may be non-negative and LCSB neutral or beneficial. All our tasks reward longer outputs, and we ran no verbosity-penalized experiments. **Mitigation.** SPW does not fully recover on-policy performance under extreme truncation ($k/N \leq 0.4$, mean horizon ≥ 500 tokens, where $\mathrm{ESS}/n < 0.15$); reducing truncation is the remedy there. The $0.6\times$ EGI threshold comes from three tasks, may need recalibration elsewhere, and EGI is not specific to LCSB. **Scale and backend.** The 8B experiments use only 3 seeds due to compute constraints, so standard errors are wider. The latency–length correlation is validated on vLLM only; it follows from autoregressive decoding, but quantitative $\hat{\alpha}$ estimates will differ across backends. We cannot verify whether APR is active in closed RLHF systems; all claims are scoped to publicly reproducible pipelines.

9 Conclusion

Autoregressive generation takes time proportional to length. Discard-style APR selects by time. Therefore it selects by length. Long trajectories are down-weighted in the training batch at every iteration—not because they receive low reward, but because they finish late. Over many training steps, the model can converge to a shorter, faster, shallower policy. This is not noise; it is optimization of a different effective objective whenever the covariance condition $\mathrm{Cov}(R, \alpha) < 0$ holds.

We have formalized this as Latency-Conditioned Selection Bias, identified its core mechanism via the covariance condition, proved that it attenuates the gradient in general and inverts it in a minimal branching MDP, and validated it empirically. The synthetic latency-injection experiment provides controlled evidence that the length–latency correlation drives the degradation, and a throughput-matched synchronous baseline argues against batch-size confounds. Standard policy-ratio IS corrections do not apply; SPW corrects the bias at $< 0.3\%$ overhead when selection overlap is sufficient. EGI flagged degradation roughly 600–1700 steps before reward curves diverged in our experiments.

The effect was likely missed in prior work because evaluation was conducted in the short-horizon regime where our analysis predicts it to be negligible—which we confirm empirically. The regime boundary, $\approx$ 150–200 tokens mean output, coincides with current mathematical reasoning benchmarks. Our results are scoped to discard-style APR with PPO on tasks that reward longer outputs; recycle-style systems, GRPO-style objectives, and verbosity-penalized tasks remain open. We recommend reporting throughput and EGI jointly in any RLHF system, and validating throughput optimizations across the intended horizon distribution.

Acknowledgments and Disclosure of Funding

I am grateful to Visvesvaraya Technological University (VTU), Belagavi, for financial support and encouragement. I thank the Department of Computer Science and Engineering and its faculty members for their guidance and support, and the Research and Development (R&D) office of VTU for providing a conducive research environment. I also thank my family, friends and well-wishers for their constant support, and the anonymous reviewers for their constructive feedback.

Funding: financial support from Visvesvaraya Technological University (VTU), Belagavi.
Competing interests: none.

References

[1] V. Mnih *et al.*, "Asynchronous methods for deep reinforcement learning," *Proc. ICML*, pp. 1928–1937, 2016.

[2] L. Espeholt *et al.*, "IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures," *Proc. ICML*, pp. 1407–1416, 2018.

[3] D. Horgan *et al.*, "Distributed prioritized experience replay," *ICLR*, 2018.

[4] C. Zheng *et al.*, "Secrets of RLHF in large language models part I: PPO," *arXiv:2307.04964*, 2023.

[5] L. Ouyang *et al.*, "Training language models to follow instructions with human feedback," *NeurIPS*, pp. 27730–27744, 2022.

[6] W. Kwon *et al.*, "Efficient memory management for large language model serving with PagedAttention," *Proc. SOSP*, pp. 611–626, 2023.

[7] A. Dubey *et al.*, "The Llama 3 herd of models," *arXiv:2407.21783*, 2024.

[8] K. Cobbe *et al.*, "Training verifiers to solve math word problems," *arXiv:2110.14168*, 2021.

[9] Y. Bai *et al.*, "Training a helpful and harmless assistant with reinforcement learning from human feedback," *arXiv:2204.05862*, 2022.

[10] J. Austin *et al.*, "Program synthesis with large language models," *arXiv:2108.07732*, 2021.

[11] A. Radford *et al.*, "Language models are unsupervised multitask learners," *OpenAI Blog*, 2019.

[12] J. Schulman *et al.*, "Proximal policy optimization algorithms," *arXiv:1707.06347*, 2017.

[13] P. Moritz *et al.*, "Ray: A distributed framework for AI applications," *OSDI*, pp. 561–577, 2018.

[14] Y. Leviathan *et al.*, "Fast inference from transformers via speculative decoding," *Proc. ICML*, pp. 19274–19286, 2023.

[15] J. P. Klein and M. L. Moeschberger, *Survival Analysis: Techniques for Censored and Truncated Data*. Springer, 2003.

[16] Y. Zhou *et al.*, "APRIL: Active partial rollouts in reinforcement learning to tame long-tail generation," *arXiv:2509.18521*, 2025.

[17] W. Fu *et al.*, "AReaL: A large-scale asynchronous reinforcement learning system for language reasoning," *NeurIPS*, 2025.

[18] Kimi Team, "Kimi k1.5: Scaling reinforcement learning with LLMs," *arXiv:2501.12599*, 2025.

[19] J. M. Robins *et al.*, "Analysis of semiparametric regression models for repeated outcomes in the presence of missing data," *JASA*, vol. 90, pp. 106–121, 1995.

[20] J. J. Heckman, "Sample selection bias as a specification error," *Econometrica*, vol. 47, pp. 153–161, 1979.

[21] H. Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function," *J. Stat. Plan. Infer.*, vol. 90, pp. 227–244, 2000.

[22] M. Sugiyama *et al.*, "Covariate shift adaptation by importance weighted cross validation," *JMLR*, vol. 8, pp. 985–1005, 2007.

[23] P. F. Christiano *et al.*, "Deep reinforcement learning from human preferences," *NeurIPS*, pp. 4299–4307, 2017.

[24] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. MIT Press, 2018.

[25] E. L. Ionides, "Truncated importance sampling," *J. Comput. Graph. Stat.*, vol. 17, pp. 295–311, 2008.

[26] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, pp. 229–256, 1992.

[27] M. Papini *et al.*, "Stochastic variance-reduced policy gradient," *Proc. ICML*, pp. 4026–4035, 2018.

[28] D. Precup, R. S. Sutton, and S. Singh, "Eligibility traces for off-policy policy evaluation," *Proc. ICML*, pp. 759–766, 2000.

[29] T. Dao *et al.*, "FlashAttention: Fast and memory-efficient exact attention with IO-awareness," *NeurIPS*, pp. 16344–16359, 2022.

A Complete Proofs

A.1 Proof of Theorem 4.1

Setup. In the branching MDP, there are exactly two trajectory classes: τ_s (shallow, length 1, reward r_s) and τ_d (deep, length H, reward r_d). The policy assigns $\Pr(\tau_s) = 1-\theta$ and $\Pr(\tau_d) = \theta$. Under APR:

$$p_{\mathrm{APR}}(\tau_s) = \frac{(1-\theta)\alpha_s}{Z(\theta)}, \qquad p_{\mathrm{APR}}(\tau_d) = \frac{\theta\alpha_d}{Z(\theta)}, \tag{10}$$

where $Z(\theta) = (1-\theta)\alpha_s + \theta\alpha_d$.

Score functions. Since $\pi_\theta(a_d|s_0) = \theta$:

$$\nabla_\theta \log \pi_\theta(\tau_s) = \frac{-1}{1-\theta}, \qquad \nabla_\theta \log \pi_\theta(\tau_d) = \frac{1}{\theta}. \tag{11}$$

APR gradient.

$$\nabla_\theta J_{\mathrm{APR}}(\theta) = p_{\mathrm{APR}}(\tau_s) \cdot \frac{-r_s}{1-\theta} + p_{\mathrm{APR}}(\tau_d) \cdot \frac{r_d}{\theta} = \frac{\alpha_d r_d - \alpha_s r_s}{Z(\theta)}. \tag{12}$$

On-policy gradient. $\nabla_\theta J(\theta) = r_d - r_s$.

Bias.

$$\mathrm{LCSB}(\theta) = \frac{\alpha_d r_d - \alpha_s r_s}{Z} - (r_d - r_s) \tag{13}$$

$$= \frac{(\alpha_d - \alpha_s)[(1-\theta)r_d + \theta r_s]}{Z(\theta)}. \tag{14}$$

Since $\alpha_d \le \alpha_s$ (Assumption 2.1), $(1-\theta)r_d + \theta r_s > 0$, and $Z(\theta) > 0$, we have $\mathrm{LCSB}(\theta) \le 0$ with equality iff $\alpha_d = \alpha_s$. □

A.2 Proof of Corollary 4.2

$\nabla_\theta J_{\mathrm{APR}} < 0 \Leftrightarrow \alpha_d r_d < \alpha_s r_s \Leftrightarrow \alpha_s/\alpha_d > r_d/r_s$. □

A.3 Proof of Theorem 4.4

Reward inequality. We write:

$$\mathbb{E}_{p_{\mathrm{APR}}}[R] = \mathbb{E}_{p_\theta}[R] + \frac{\mathrm{Cov}_{p_\theta}(R, \alpha)}{Z}. \tag{15}$$

It remains to show $\mathrm{Cov}_{p_\theta}(R(\tau), \alpha(\tau)) \le 0$. Under Assumption 2.1, $\alpha(\tau)$ is non-increasing in $|\tau|$, and r is non-decreasing by hypothesis.

Lemma A.1 (Covariance of Monotone Functions)**.** *Let X be any real-valued random variable. If f is non-decreasing and g is non-increasing, then* $\mathrm{Cov}(f(X), g(X)) \le 0$.

Proof. For any two draws X, X': $(f(X) - f(X'))(g(X) - g(X')) \le 0$ pointwise. Taking expectations gives $\mathbb{E}[(f(X) - f(X'))(g(X) - g(X'))] \le 0$, which equals $2\,\mathrm{Cov}(f(X), g(X))$. □

Applying Lemma A.1 with $X = |\tau|$, $f = r$ (non-decreasing), $g = \alpha$ (non-increasing): $\mathrm{Cov}_{p_\theta}(R, \alpha) \le 0$, with equality iff r or α is constant. This proves inequality (6).

Gradient inequality. Let $f(\tau) = \langle \nabla_\theta \log \pi_\theta(\tau), \hat{e}_\ell \rangle R(\tau)$. By hypothesis, this is non-decreasing in $|\tau|$. The same covariance argument gives:

$$\nabla_\theta J_{\mathrm{APR}} \cdot \hat{e}_\ell = \nabla_\theta J \cdot \hat{e}_\ell + \frac{\mathrm{Cov}_{p_\theta}(f(\tau), \alpha(\tau))}{Z} \le \nabla_\theta J \cdot \hat{e}_\ell, \tag{16}$$

with strict inequality when $\mathrm{Cov}(R, \alpha) < 0$. □

A.4 Proof of Theorem 6.2

The self-normalized SPW estimator is:

$$\hat{\nabla} J_{\mathrm{SPW}} = \frac{n^{-1}\sum_{i=1}^{n} w(\tau_i) g(\tau_i)}{n^{-1}\sum_{i=1}^{n} w(\tau_i)}, \quad g(\tau) = \nabla_\theta \log \pi_\theta(\tau)\, R(\tau), \quad w(\tau) = 1/\alpha(\tau), \tag{17}$$

with $\tau_i \sim p_{\mathrm{APR}}$ i.i.d.

Numerator: $\mathbb{E}_{p_{\mathrm{APR}}}[w(\tau)g(\tau)] = \mathbb{E}_{p_\theta}\left[\frac{\alpha(\tau)}{Z} \cdot \frac{1}{\alpha(\tau)} \cdot g(\tau)\right] = \frac{\nabla_\theta J(\theta)}{Z}$.

Denominator: $\mathbb{E}_{p_{\mathrm{APR}}}[w(\tau)] = \mathbb{E}_{p_\theta}\left[\frac{\alpha(\tau)}{Z} \cdot \frac{1}{\alpha(\tau)}\right] = \frac{1}{Z}$.

Both expectations are finite under the overlap condition. By the strong law of large numbers and the continuous mapping theorem, the ratio converges a.s. to $(\nabla_\theta J(\theta)/Z)/(1/Z) = \nabla_\theta J(\theta)$, so $\hat{\nabla} J_{\mathrm{SPW}} \to \nabla J(\theta)$ a.s. □

B Latency Distribution Models

B.1 Exponential Latency

Assume $L(\tau_s) \sim \mathrm{Exp}(1/c)$ and $L(\tau_d) \sim \mathrm{Exp}(1/(cH))$. Then $\alpha_s - \alpha_d = e^{-T_k/(cH)} - e^{-T_k/c}$, which is strictly increasing in H ($\partial_H(\alpha_s - \alpha_d) > 0$) and approaches $1 - e^{-T_k/c}$ as $H \to \infty$.

B.2 Log-Normal Latency

Assume $\log L(\tau_s) \sim \mathcal{N}(\mu_s, \sigma^2)$ and $\log L(\tau_d) \sim \mathcal{N}(\mu_s + \log H, \sigma^2)$. Then:

$$\alpha_s - \alpha_d = \Phi\left(\frac{\log T_k - \mu_s}{\sigma}\right) - \Phi\left(\frac{\log T_k - \mu_s - \log H}{\sigma}\right), \tag{18}$$

which is positive and increasing in H for $T_k > e^{\mu_s}$.

Robustness. Theorem 4.4 requires only the monotonicity $\alpha(\ell)$ non-increasing in ℓ (Assumption 2.1), not any specific distributional form.

C EGI: Full Specification

C.1 Algorithm

Step 1. Fix $P \in \mathbb{R}^{m \times d}$ ($m = 512$) with i.i.d. $\mathcal{N}(0, 1/m)$ entries, held fixed throughout training.

Step 2. For each minibatch b ($B_{\min} = 32$ trajectories), compute g_b and project: $\tilde{g}_b = P g_b \in \mathbb{R}^m$.

Step 3. Over $B = 32$ consecutive minibatches: $\hat{\mu} = B^{-1}\sum_b \tilde{g}_b$, $\widehat{\mathrm{Tr}(\mathrm{Cov})} = (B-1)^{-1}\sum_b \|\tilde{g}_b - \hat{\mu}\|_2^2$.

Step 4. $\mathrm{EGI}_t = \|\hat{\mu}\|_2^2 / (\widehat{\mathrm{Tr}(\mathrm{Cov})} \cdot C_t)$, reported as a 100-step EMA.

C.2 Threshold Calibration

Table 3: EGI threshold calibration by task.

Task	Threshold range	Adopted	Lead time (steps)	False positives
HH-RLHF (87 tok)	N/A (LCSB negligible)	0.85	N/A	0/10
MBPP (243 tok)	$[0.60, 0.68]$	0.63	600 ± 200	0/10
GSM8K (412 tok)	$[0.55, 0.62]$	0.60	1700 ± 300	0/10

D SPW: Variance Analysis

D.1 Variance of the Unclipped Estimator

$$\mathrm{Var}\left(\frac{1}{k}\sum_i w(\tau_i)g(\tau_i)\right) = \frac{1}{k}\,\mathbb{E}_{p_{\mathrm{APR}}}\left[\frac{g(\tau)^2}{\alpha(\tau)^2}\right] - \frac{1}{k}\left(\frac{\nabla J}{Z}\right)^2. \tag{19}$$

The first term is $O(\alpha_{\min}^{-2})$. For $k/N = 0.67$ and mean horizon 412 tokens, $\hat{\alpha}_{\min} \approx 0.03$–$0.09$, giving variance amplification of order 100–$1000\times$ on long-trajectory components before clipping.

D.2 ESS Criterion

$\mathrm{ESS} = (\sum_i w_i)^2 / \sum_i w_i^2$. When $\mathrm{ESS}/n \geq 0.3$, clipping at $w_{\max} = 10$ leads to less than 10% MSE increase relative to the unclipped estimator. When $\mathrm{ESS}/n < 0.15$, the weight distribution is too concentrated; reducing truncation is the appropriate response.

Table 4: ESS/n by truncation level and task horizon. Values < 0.3 indicate regimes where reducing truncation is recommended.

	Mean horizon (tokens)			
k/N	87	243	412	600
0.90	0.81 ± 0.04	0.67 ± 0.05	0.54 ± 0.06	0.41 ± 0.07
0.80	0.73 ± 0.05	0.58 ± 0.06	0.43 ± 0.07	0.31 ± 0.08
0.67	0.61 ± 0.06	0.44 ± 0.07	0.33 ± 0.08	0.21 ± 0.07
0.50	0.44 ± 0.07	0.29 ± 0.08	0.16 ± 0.06	0.08 ± 0.04

E Experimental Details

E.1 vLLM Latency Measurement Protocol

Hardware: $2 \times$ NVIDIA Tesla T4 GPUs (15 GB VRAM each). Software: vLLM v0.4.1 (PagedAttention, continuous batching, no speculative decoding), kept fixed for a reproducible stack across all experiments. Prompts sampled uniformly from HH-RLHF, MBPP, and GSM8K. Sampling: temperature 0.7, top-p 0.95. Total: 12,000 requests; concurrency $N \in \{8, 16, 32\}$; KV-cache utilization 40–95%. Latency–length regression: OLS ($r^2 = 0.918$). The 8.2% residual variance is consistent across concurrency levels and cache utilization, consistent with measurement noise and scheduling jitter.

E.2 PPO Hyperparameters

Hyperparameter	**Value**
Learning rate	3×10^{-5}
KL coefficient β	0.02 (except KL ablation)
Training batch k	64
PPO clip ε	0.2
GAE λ	0.95
Discount γ	1.0
Training steps	5000
APR: total rollouts N	96
APR: selected k	64 ($k/N = 0.67$)
On-policy: $N = k$	64
Throughput-matched baseline: $N = k$	96

For LLaMA-3-8B: training on 2×NVIDIA Tesla T4 GPUs; generation temperature 0.7, top-p 0.95, max 512 tokens; evaluation on 500 held-out prompts (greedy decoding) every 200 steps.

E.3 Seed Counts and Statistical Reporting

GPT-2 (124M) experiments: 10 seeds. LLaMA-3-8B experiments: 3 seeds (compute constraints). KL ablation: 3 seeds per β. Synthetic latency injection: 5 seeds. All values: mean $\pm$ standard error. Significance: two-sided t-tests, $p < 0.05$.

E.4 SPW Implementation

Length bins: 10 bins spanning $[0, 100), [100, 200), \ldots, [900, \infty)$, each with an empirical CDF $\hat{F}_j$. Warm-up: equal weights for the first 50 iterations. After warm-up: $\hat{\alpha}_i = \hat{F}_{j(|\tau_i|)}(T_k)$; $w_i = \min(1/\hat{\alpha}_i, 10)$. ESS monitored every 100 steps; warning issued if $\mathrm{ESS}/n < 0.15$.

E.5 Synthetic Latency Injection Protocol

Latency injected by overriding the vLLM completion callback to add a delay sampled from $\mathcal{N}(c \cdot |\tau|, \sigma^2)$. Five noise levels: $\sigma \in \{0, c/2, c, 3c, \infty\}$, where $\sigma = \infty$ randomizes completion order uniformly. GPT-2 (124M), GSM8K, 5000 training steps, 5 seeds per condition. All conditions share the same batch size k, policy, reward model, and optimizer; only the correlation between $\alpha(\tau)$ and $|\tau|$ varies.

E.6 KL Regularization Ablation

Table 5: Extended KL regularization ablation on GSM8K (LLaMA-3-8B, $k/N = 0.67$, 3 seeds). Increasing β reduces update magnitude and partially recovers performance, but does not eliminate the gap relative to on-policy training. SPW achieves near on-policy performance at default $\beta = 0.02$.

Configuration	Accuracy (%)	Avg. length (tok)	EGI	Rel. to on-policy
On-policy ($\beta = 0.02$)	68.4 ± 1.8	421 ± 18	1.00	1.000
APR ($\beta = 0.02$)	44.1 ± 2.6	193 ± 24	0.42	0.645
APR ($\beta = 0.05$)	49.7 ± 2.3	228 ± 21	0.56	0.727
APR ($\beta = 0.10$)	54.2 ± 2.1	267 ± 19	0.68	0.792
APR ($\beta = 0.20$)	58.3 ± 1.9	305 ± 17	0.79	0.852
APR+SPW ($\beta = 0.02$)	66.9 ± 1.9	402 ± 20	0.95	0.978

F Additional Results

F.1 LCSB Magnitude: Direct Measurement

Table 6: LCSB magnitude (normalized by on-policy gradient norm) at selected checkpoints. GSM8K/GPT-2, $k/N = 0.67$. Monotonically growing LCSB is consistent with the theoretical prediction.

Step	$\|\mathrm{LCSB}\|/\|\nabla J\|$	EGI (relative)
0	0.02 ± 0.01	1.00
500	0.11 ± 0.03	0.84
1000	0.24 ± 0.04	0.69
2000	0.41 ± 0.05	0.51
3000	0.49 ± 0.05	0.44
5000	0.51 ± 0.06	0.42

Table 7: EGI detection characteristics by task. EGI provides lead time before reward curves clearly diverge.

Task	**EGI alarm step**	**Drop at alarm**	**Final drop**	**Lead time**
HH-RLHF	No alarm	N/A	0.8%	N/A
MBPP	1100 ± 200	$1.2 \pm 0.4\%$	8.1%	600 ± 200 steps
GSM8K	800 ± 150	$0.9 \pm 0.3\%$	35.5%	1700 ± 300 steps

Table 8: SPW wall-clock overhead (mean $\pm$ std, 1000 iterations, 2$\times$NVIDIA Tesla T4 GPUs).

Batch k	Baseline (ms)	SPW (ms)	Mean overhead	99th pct
32	1847 ± 12	1853 ± 14	0.32%	0.71%
64	3612 ± 18	3621 ± 19	0.25%	0.58%
128	7156 ± 31	7174 ± 33	0.25%	0.52%

F.2 EGI Detection Summary

F.3 Computational Overhead

G Extended Limitations

G.1 When Reward–Length Monotonicity Does Not Hold

Theorem 4.4 assumes $r(|\tau|)$ non-decreasing. When the reward model penalizes verbosity, $\mathrm{Cov}(R, \alpha)$ may be non-negative and LCSB could be neutral or beneficial. When reward is non-monotone in length (e.g., peaked at medium length), the covariance has ambiguous sign. All tasks in our experiments reward longer outputs, and we ran no verbosity-penalized experiments. Practitioners should measure the empirical reward–length correlation before applying our framework.

G.2 Generalization Beyond PPO and Discard-Style APR

The filtering mechanism $p_{\mathrm{APR}} \propto p_\theta \alpha$ arises in any discard-style wall-clock cutoff. However, severity depends on how the algorithm uses collected data. Specific cases worth noting: REINFORCE uses raw returns, so LCSB acts directly as analyzed; GRPO computes each advantage from the mean and variance of its whole response group, so length-conditioned removal changes both the group composition and the normalization applied to every retained trajectory, and our single-trajectory analysis and per-trajectory SPW do not transfer directly and require separate analysis; algorithms with replay buffers face LCSB-induced distributional shift compounded by additional replay drift; recycle-style APR systems that resume unfinished rollouts would substantially reduce or remove the mechanism. We consider these extensions important future work.

NeurIPS Paper Checklist

1. **Claims**

   Answer: [Yes]

   Justification: The abstract and introduction (Section 1) state the core claim that discard-style APR induces a latency-conditioned selection bias (LCSB), formalized via a covariance condition (Sections 3–4) and supported by empirical results (Section 5). The scope (discard-style APR, PPO, tasks where reward increases with length) is stated explicitly in the Scope paragraph of Section 1.

2. **Limitations**

   Answer: [Yes]

   Justification: Section 8 (with extended discussion in Appendix G) covers discard-style versus recycle-style APR, reward–length monotonicity, PPO-specific analysis (GRPO requires separate treatment), SPW behavior under extreme truncation, and the limited seed count for the 8B experiments.

3. **Theory assumptions and proofs**

   Answer: [Yes]

   Justification: Assumption 2.1 (Section 2) and all theorem hypotheses (Sections 4 and 6.2) are stated explicitly, and complete proofs are in Appendix A.

4. **Experimental result reproducibility**

   Answer: [Yes]

   Justification: Experimental setup, datasets, and protocols are specified in Section 5, and hyperparameters and implementation details are in Appendix E, enabling reproduction of all main results.

5. **Open access to data and code**

   Answer: [No]

   Justification: We do not release code. All datasets used (HH-RLHF, MBPP, GSM8K) are public, and the implementation details, hyperparameters, serving configuration (vLLM v0.4.1) and evaluation protocols are specified in Appendix E so that the experiments can be re-implemented.

6. **Experimental setting/details**

   Answer: [Yes]

   Justification: Training details, including optimizer settings, rollout parameters, datasets, and evaluation procedures, are provided in Section 5 and Appendix E; SPW and EGI implementation details are in Appendices E, C and D.

7. **Experiment statistical significance**

   Answer: [Yes]

   Justification: Results are reported as mean $\pm$ standard error across multiple seeds, with seed counts and statistical methodology specified in Appendix E (Seed Counts and Statistical Reporting).

8. **Experiments compute resources**

   Answer: [Yes]

   Justification: Hardware (2 NVIDIA Tesla T4 GPUs) and experimental configurations are described in Appendix E (vLLM Latency Measurement Protocol and PPO Hyperparameters); SPW overhead is reported in Appendix F.3.

9. **Code of ethics**

   Answer: [Yes]

   Justification: The research complies with the NeurIPS Code of Ethics; no human subjects or sensitive data are involved.

10. **Broader impacts**

Answer: [Yes]

Justification: Section 7 discusses implications for RLHF system design and failure modes; while primarily methodological, it highlights risks of misaligned optimization in deployed systems.

11. **Safeguards**

    Answer: [N/A]

    Justification: The work does not release models or datasets with elevated misuse risk.

12. **Licenses for existing assets**

    Answer: [Yes]

    Justification: All datasets and models (e.g., GSM8K, MBPP, HH-RLHF, LLaMA-3) are cited with references to their original publications (Section 5).

13. **New assets**

    Answer: [N/A]

    Justification: The paper does not introduce new datasets or publicly released models.

14. **Crowdsourcing and research with human subjects**

    Answer: [N/A]

    Justification: The work does not involve human subjects or crowdsourced data collection.

15. **Institutional review board (IRB) approvals**

    Answer: [N/A]

    Justification: No human subject research is conducted.

16. **Declaration of LLM usage**

    Answer: [No]

    Justification: LLMs are not an important, original, or non-standard component of the core methods; they appear only as experimental subjects (Section 5). LLMs were also used for editing and for understanding technical concepts, as declared at submission.