ACTS: Adaptive Control for Test-time Scaling

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

We introduce a framework of **Adaptive Control Token Sampling (ACTS)** policies that leverage probability signals from specific tokens in the LLM vocabulary to dynamically regulate optimal stopping in the generation process. Specifically, ACTS combats over-thinking and under-thinking in LLMs by leveraging adaptive signals about the generation trace at test-time offering superior test-time scaling properties. Our experiments show that ACTS effectively mitigates under-thinking on complex reasoning tasks using adaptive stopping-time policies. Furthermore, we propose an **Adaptive Self-Critique Sampler** that uses end-of-thinking spikes as triggers for self-evaluation, boosting reasoning accuracy upto $\sim 9.8\%$ on the MATH-500. On instruction-following tasks, ACTS leverages end-of-sequence spikes to improve the quality-efficiency trade-off. Finally, we used spikes to propose a novel parallel sampling technique that intelligently initiates high-quality parallel reasoning trajectories from a shared sequentially generated thinking trace. Our work establishes control token probabilities as a powerful, untapped signal for creating more robust and efficient inference policies, offering a new paradigm to control test-time scaling.

1 Introduction

Modern Large Language Models (LLMs) tackle complex mathematical reasoning by generating multi-step rationales, a technique known as Chain-of-Thought (CoT) prompting (Wei et al., 2022). The efficacy of this reasoning paradigm, however, is deeply intertwined with the CoT length. While longer rationales can provide necessary computational steps, they also introduce significant latency and are susceptible to error accumulation, or over-thinking (Sui et al., 2025). Foundational work has formalized this trade-off, demonstrating that reasoning performance follows an inverted U-shaped curve (Wu et al., 2025). This establishes the existence of an *optimal CoT length* that is task- and model-dependent. Naive inference-time policies, such as "budget forcing" with "Wait" tokens (Muennighoff et al., 2025), are non-adaptive solutions that fail to find this optimal, state-dependent stopping time.

While existing methods often depend on separately trained models or complex heuristics, we identify and leverage a more fundamental, previously under-explored signal for generation control: the **sub-argmax probabilities** assigned to semantic control tokens like end-of-thinking‘ (EOT) or end-of-sequence (EOS). Our central empirical finding, illustrated in Figure 1, is that the dynamics of this signal, especially its sharp, transient spikes, are a rich, structured indicator of the model's internal readiness to conclude a reasoning phase.

To harness this signal, we introduce **A**daptive **C**ontrol for **T**est-time **S**caling (ACTS), a training-free framework that casts adaptive generation as a principled optimal stopping problem. This represents a shift from behavioral control to a more fine-grained, probabilistic control. We propose two policies derived from this framework that are highly relevant to mathematical reasoning:

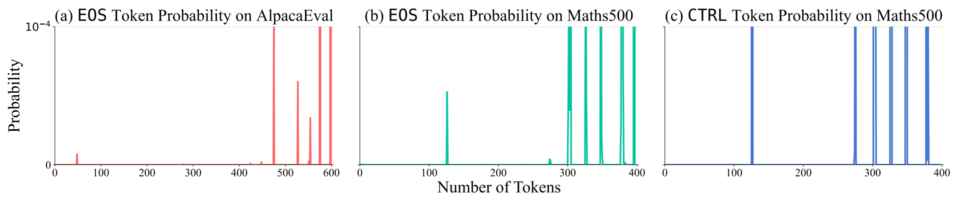


Figure 1: Token-wise probabilities of control tokens during LLM decoding. Subfigures (b) and (c) show the post-thinking $P(t_{EOS})$ and in-thinking $P(t_{EOT})$ on MATH500, respectively. The sharp, transient nature of these spikes acts as a signal for candidate completion points, motivating our adaptive control policies.

1. An *Adaptive Self-Critique Sampler*, an efficient, single-model mechanism where the LLM uses spike-triggered self-evaluation to dynamically find an optimal stopping time, mitigating both under- and overthinking.
2. A *Quality-Gated Forking* technique, a novel parallel sampling method that uses signal dynamics and self-critique scores as a principled trigger to spawn high-quality answer trajectories from a shared reasoning trace.

2 Methodology: ACTS Policies for Mathematical Reasoning

We propose the **A**daptive **C**ontrol of **T**oken **S**equences (ACTS) framework to dynamically steer stopping-time in autoregressive generation. ACTS intervenes at each decoding step by monitoring the sub-argmax probability of a semantic control token, $s_t = \pi_\theta(t_{\text{control}}|\mathbf{x}_{<t}, C)$. We cast this as a principled optimal stopping problem, with policies derived from theoretical motivations (see Appendix C). We present the two policies most relevant to mathematical reasoning.

2.1 Policy 1: Adaptive Self-Critique Sampler

To mitigate under- and overthinking, we introduce a sophisticated policy inspired by actor-critic methods, where the LLM is leveraged as its own critic. The generative process is the *actor* (π_θ), which produces reasoning, and the same LLM, prompted for self-evaluation, is the *critic*.

Instead of performing costly self-evaluation at every step, we use probability spikes ($s_t > \delta$) as a trigger, identifying critical junctures where a critique is most valuable. As detailed in Algorithm 1, this critique then informs the generation decision.

Algorithm 1:

Adaptive Self-Critique Policy

```
1: Input: LLM $\pi_\theta$, Prompt $C$, $t_{\text{control}}$, Spike Threshold ($\delta_{\text{critique}}$), Critique Prompt ($C_{\text{critique}}$)
2: for each generation step $t = 1, 2, \ldots$ do
3:     $s_t \leftarrow \pi_\theta(t_{\text{control}} \mid \mathbf{x}_{<t}, C)$
4:     if $s_t > \delta_{\text{critique}}$ then
5:         score ← GenerateCritique(LLM, context, $C_{\text{critique}}$)
6:         if score = 5 then
7:             Emit $t_{EOT}$ and return
8:         else
9:             Emit $t_{Wait}$                ▷ Force continuation
10:        end if
11:    else
12:        Emit default token
13:    end if
14: end for
```

Efficient Implementation. The critique is implemented as an efficient, single-model operation. Upon a spike, the controller appends a fixed critique-prompt (e.g., *"Is this reasoning trace correct, answer on the scale of 1-5?"*) to the current context. The LLM is then prompted to generate a *single token* representing its score. This low-cost (e.g., ~50 tokens) critique is only invoked at key junctures, enabling the policy to find a state-dependent optimal stopping time. It mitigates underthinking by forcing continuation (emitting t_{Wait}) on a low score, and mitigates overthinking by permitting a high-confidence, early exit (emitting t_{EOT}).

2.2 Policy 2: Quality-Gated Forking

For parallel sampling, we propose a novel technique that uses ACTS signals to spawn multiple answer trajectories from a single reasoning trace. A fork is initiated only when two conditions are met: (1) a *hesitation spike* is detected (i.e., the model is uncertain about the next reasoning step), and (2) the *self-critique score* of the reasoning so far is high ($C(T_k) \geq s_{\text{fork}}$).

Table 1: Adaptive Self-Critique vs. Baselines on MATH and AIME. Our adaptive policy (in bold) significantly outperforms both the underthinking `Baseline` and the overthinking `Naive (Wait x N times)` policies.

Dataset	**Model**	**Sampler (EOT Handling Policy)**	**Accuracy ↑**	**Avg. Token Count ↓**
MATH-500	S1-7B	Baseline	0.732	5752.40
		Naive (Wait x 1)	0.796	6317.92
		Naive (Wait x 3)	0.808	10659.59
		Naive (Wait x 5)	0.792	17906.58
		Self-Critique (Adaptive)	**0.822**	**8885.29**
AIME 2025	S1-32B	Baseline	0.400	9552.75
		Naive (Wait x 1)	0.466	10653.63
		Naive (Wait x 3)	0.567	14503.36
		Naive (Wait x 5)	0.537	17786.21
		Self-Critique (Adaptive)	**0.567**	**12345.67**
AIME 2025	Qwen3-8B	Baseline	0.700	18154.00
		Naive (Wait x 5)	0.700	20292.45
		Self-Critique (Adaptive)	**0.767**	**19269.56**

Computational Efficiency. This dual-condition trigger is principled; it focuses computation on states that are both *uncertain* and of *high quality*. The technique is highly efficient as all parallel answer trajectories are spawned from a *shared KV-cache* of the original reasoning trace (L_{think}). This amortizes the high cost of reasoning, replacing the naive $O(N \times (L_{think} + L_{answer}))$ cost for N parallel reasoning responses with an efficient $O(L_{think} + N \times L_{answer})$ for N responses that share their thinking traces. This can be viewed as a semantic-level speculative execution, where high-quality reasoning end-points (where spikes occur) are used to propose parallel answer candidates.

3 Experiments and Results

We evaluate our ACTS policies on challenging mathematical reasoning benchmarks, which are the primary domain for the underthinking/overthinking trade-off.

Experimental Setup. Our core evaluation benchmarks are **MATH-500** (Hendrycks et al., 2021), **AIME 2025**, and **GSM-8K** (Cobbe et al., 2021). We use state-of-the-art models, including the `S1` series (Muennighoff et al., 2025) and the `Qwen3` series (8B, 14B) (Yang et al., 2025). We measure performance using strict **Accuracy** and **Average Token Count**. Our *Adaptive Self-Critique* policy is compared against two key baselines: (1) `Baseline` (top-p sampling with temperature and top-k filtering), which succumbs to *underthinking*, and (2) `Naive (Wait x N times)`, which forces prolongation and is susceptible to *overthinking*. Full setup details are in Appendix B.

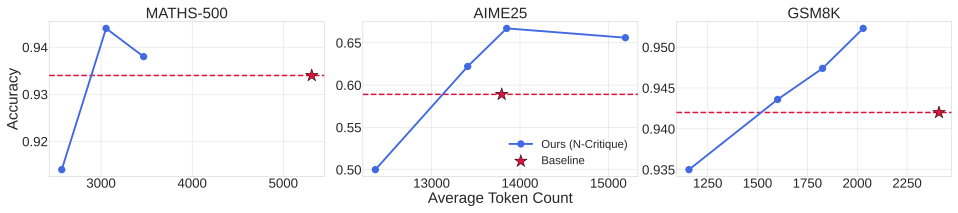
3.1 Sequential Policy: Adaptive Self-Critique

We first evaluate our sequential optimal stopping policy, *Adaptive Self-Critique*. Table 1 summarizes the core trade-off.

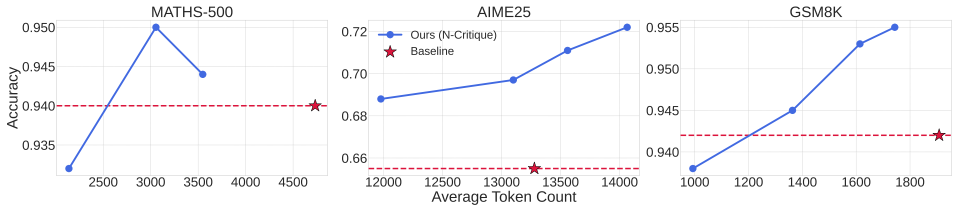
Analysis of Under- and Overthinking. The `Baseline` (greedy) policy terminates at the first t_{EOT} spike, resulting in the lowest accuracy (e.g., 73.2% on MATH-500, 40.0% on AIME), demonstrating severe *underthinking*. Conversely, the `Naive (Wait x 5 times)` policy, which forces prolonged reasoning, shows performance collapse. On MATH-500 (S1), it increases token count by 3x but *decreases* accuracy (79.2% vs. 80.8%), confirming that more computation is not always better.

Adaptive Policy Performance. Our `Self-Critique (Adaptive)` policy adaptively works through this trade-off providing Pareto optimality on both token count and accuracy.

- On **MATH-500** (S1), it achieves the highest accuracy (**82.2%**), a **9.0% absolute gain** over the baseline, while using nearly half the tokens of the naive 'Wait x 5 times' policy.
- On **AIME 2025** (S1-32B), it achieves a **16.7% absolute gain** (56.7% vs. 40.0%) over the baseline.
- On **Qwen3-8B** (AIME), it achieves a **6.7% absolute gain** (76.7% vs. 70.0%).



(a) Qwen-3 8B Performance Trade-off



(b) Qwen-3 14B Performance Trade-off

Figure 2: Accuracy vs. Computational Cost for Qwen-3 models on MATH-500 (left panel in each subfigure), AIME25 (middle), and GSM8K (right). Our N-Critique policies (blue dots, $N \in \{1, 3, 5, 7\}$) create a new Pareto frontier, decisively beating the baseline on both accuracy and cost.

The adaptive policy consistently establishes a new state-of-the-art by finding a superior, state-dependent stopping time.

N-Critique Sampler Analysis. To analyze the effect of critique confidence, we introduce the *N-Critique Sampler (N)*, which terminates only after receiving N high-confidence (score=5) critiques. Figure 2 visualizes the performance of this policy on Qwen models. The plots show a clear Pareto-improving frontier: our N-Critique policies (blue dots) consistently achieve higher accuracy at a lower computational cost than the baseline (red star). On MATH-500, `N-Critique (3)` surpasses the 14B baseline (95.0% vs. 94.0%) while reducing tokens by over 35%. On the more difficult AIME benchmark, accuracy scales with N, with `N-Critique (7)` achieving a 72.2% top accuracy, a significant improvement over the 65.5% baseline.

3.2 Spawning Parallel Response Trajectories from Sequential Reasoning Traces

We also evaluate our *Quality-Gated Forking* (QGF) policy, which uses ACTS signals to trigger parallel answer generation from a shared reasoning trace. We compare this against our best sequential policy (`N-Critique`) and `Unconditional Forking` (a brute-force parallel baseline).

Table 2: Quality-Gated Forking on MATH-500. Our principled forking (bold) achieves the highest accuracy and is more efficient than naive parallel sampling.

Model	Policy	Accuracy	Avg. Forks	Avg. Tokens
Qwen3-8B	N-Critique (Seq.)	0.944	0	3,055
	Unconditional Fork.	0.942	13.1	18,336
	Quality-Gated	**0.950**	**9.4**	**13,005**
Qwen3-14B	N-Critique (Seq.)	0.950	0	3,085
	Unconditional Fork.	0.952	10.8	15,336
	Quality-Gated	**0.961**	**7.1**	**11,231**

Analysis. As shown in Table 2, our `Quality-Gated Forking` policy consistently outperforms both the strong sequential baseline and the naive parallel approach. On Qwen3-14B (MATH-500), QGF achieves a new SOTA of **96.1%** accuracy. Notably, it achieves this while being significantly more efficient than brute-force forking, reducing token overhead by 26% (11.2k vs. 15.3k tokens) by only spawning trajectories from high-quality, high-uncertainty states. This demonstrates the benefit of our principled, signal-driven approach to parallel search.

Discussion: We identified a novel, fundamental signal for LLM inference: the sub-argmax probability spikes of control tokens. We introduced the ACTS framework and two derived policies: *Adaptive Self-Critique* for optimal sequential stopping and *Quality-Gated Forking* for efficient parallel search, providing both latency and accuracy gains. By adaptively navigating the under- vs. overthinking trade-off, ACTS provides a principled, training-free, and computationally efficient paradigm for test-time scaling on challenging math benchmarks like MATH-500 and AIME2025.

References

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 17682–17690, 2024.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 2024.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.

Chengsong Huang, Langlin Huang, Jixuan Leng, Jiacheng Liu, and Jiaxin Huang. Efficient test-time scaling via self-calibration. *arXiv preprint arXiv:2503.00031*, 2025.

Lucien Le Cam. *Asymptotic methods in statistical decision theory*. Springer Science & Business Media, 2012.

Dacheng Li, Shiyi Cao, Chengkun Cao, Xiuyu Li, Shangyin Tan, Kurt Keutzer, Jiarong Xing, Joseph E Gonzalez, and Ion Stoica. S*: Test time scaling for code generation. *arXiv preprint arXiv:2502.14382*, 2025.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025.

Daniele Paliotta, Junxiong Wang, Matteo Pagliardini, Kevin Y Li, Aviv Bick, J Zico Kolter, Albert Gu, François Fleuret, and Tri Dao. Thinking slow, fast: Scaling inference compute with distilled reasoners. *arXiv preprint arXiv:2502.20339*, 2025.

Matthew Renze and Erhan Guven. The benefits of a concise chain of thought on problem-solving in large language models. In *2024 2nd International Conference on Foundation and Large Language Models (FLLM)*, pp. 476–483. IEEE, 2024.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Hanjie Chen, et al. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*, 2025.

Hanshi Sun, Momin Haider, Ruiqi Zhang, Huitao Yang, Jiahao Qiu, Ming Yin, Mengdi Wang, Peter Bartlett, and Andrea Zanette. Fast best-of-n decoding via speculative rejection. *Advances in Neural Information Processing Systems*, 37:32630–32652, 2024.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Yiming Wang, Pei Zhang, Siyuan Huang, Baosong Yang, Zhuosheng Zhang, Fei Huang, and Rui Wang. Sampling-efficient test-time scaling: Self-estimating the best-of-n sampling in early decoding. *arXiv preprint arXiv:2503.01422*, 2025.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Yuyang Wu, Yifei Wang, Ziyu Ye, Tianqi Du, Stefanie Jegelka, and Yisen Wang. When more is less: Understanding chain-of-thought length in llms. *arXiv preprint arXiv:2502.07266*, 2025.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. Chain of draft: Thinking faster by writing less. *arXiv preprint arXiv:2502.18600*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Supplementary Material

These supplementary materials provide additional details, derivations, and proofs for our paper. The appendix is organized as follows:

- **Section A: Extended Related Work.** A detailed overview of related literature.
- **Section C: A Reader's Guide to the Theoretical Analysis.** An intuitive roadmap for the formal proofs.
- **Section D: Proof of Spike Correctness (Lemma 1).** Establishes the reliability of the control signal.
- **Section E: Proof for N-Spike Counter Policy (Proposition 2).** Bounds the probability of spurious termination.
- **Section F: Proof for Accumulated Probability Policy (Proposition 3).** Guarantees the reliability of evidence accumulation.
- **Section G: Proofs for Last-Interval Budget Policy (Theorems 2 and 3).** Provides regret and prophet-inequality bounds.
- **Section H: Proof of Self-Critique Superiority (Theorem 1).** Demonstrates the benefit of the adaptive self-critique policy.
- **Section I: Proof of Robustness to Misspecification (Proposition 4).** Shows the framework's stability under model mismatch.
- **Section K: Model Performance on AlpacaEval, AIME, and Maths-500 Under ACTS Policies**

A Extended Related Work

A.1 Test-Time Scaling and Dynamic Inference

A significant line of research has demonstrated that the performance of Large Language Models can be substantially improved by allocating more computational resources at inference time, a paradigm known as test-time scaling (Muennighoff et al., 2025). This approach, however, introduces a critical challenge of efficiency. Our work, ACTS, contributes to the growing body of literature on making this scaling more intelligent and resource-efficient.

Foundational Test-Time Scaling Methods. The canonical methods for test-time scaling involve generating multiple candidate sequences and aggregating them. **Best-of-N (BoN)** sampling generates N independent sequences and uses a verifier or a trained reward model to select the highest-scoring output (Lightman et al., 2023). This approach is general-purpose but often relies on the availability of a high-quality, and potentially costly, external verifier. A popular variant, **Self-Consistency**, is designed for tasks with deterministic answers, such as mathematical reasoning (Wang et al., 2022) and code (Li et al., 2025). It generates N sequences and selects the final answer via a majority vote, eliminating the need for an external reward model but limiting its applicability. Both BoN and Self-Consistency are computationally expensive as they require the full generation of all N candidate sequences, creating a linear increase in cost with the number of samples.

Efficient Test-Time Scaling via Early Termination. Recent work has focused on mitigating the high cost of BoN and Self-Consistency by introducing mechanisms for the early termination of unpromising generation paths. These methods differ primarily in the type of signal they use to make termination decisions.

One prominent approach leverages **external verifiers** to score partial sequences. For instance, Speculative Rejection (Sun et al., 2024) periodically queries an external reward model on the partially generated sequences. Trajectories with low partial scores, which are unlikely to yield a high final reward, are pruned, allowing computational resources to be focused on the more promising candidates. While effective, this strategy's performance is contingent on the quality and calibration of the external reward model, which itself can be costly to train and serve.

A second approach utilizes signals from the model's own **latent representations**. Self-Truncation Best-of-N (ST-BON) (Wang et al., 2025) operates on the hypothesis that sequences leading to the same correct answer will have similar latent embeddings. It monitors the consistency of hidden states across parallel generations and truncates paths that diverge from the main cluster, thereby avoiding the need for an external reward model. This uses a truly internal signal, but one that is high-dimensional and less directly interpretable than the explicit probabilistic outputs of the model.

A third approach uses more general **model confidence scores**. Self-Calibration (Huang et al., 2025) proposes fine-tuning a model to produce a calibrated confidence score for its own generations, often by distilling confidence from Self-Consistency statistics. This learned confidence can then be used to implement early-stopping rules for sampling. This method also uses an internal signal, but it requires a separate training phase to create the calibrated confidence predictor.

Our ACTS framework contributes to this line of research by proposing the use of a novel signal that is both *internal* to the model and *natively available* without requiring additional training or complex analysis of latent states: the explicit probability of control tokens, $P(t_{control})$.

Orthogonal Approaches. Other methods seek efficiency through different means. **Structured Search** methods, such as Tree-of-Thought (Yao et al., 2023) and Graph-of-Thoughts (Besta et al., 2024), replace unstructured sampling with a more organized exploration of the reasoning space, often involving backtracking and planning. While powerful, these methods typically introduce significant algorithmic complexity and overhead. Concurrently, **architectural approaches** aim to build faster models from the ground up, for example by using subquadratic architectures like Mamba (Paliotta et al., 2025). These architectural innovations are largely complementary to our work; the principled stopping policies developed within the ACTS framework could potentially be applied on top of these faster models to achieve even greater efficiency gains.

A.2 Controlling Reasoning Behavior at Test-Time

Beyond general-purpose sampling efficiency, a specific line of work has focused on directly controlling the behavior of the reasoning process itself at inference time. This is particularly relevant for our work on managing the thinking phase via the t_{EOT} token. A seminal contribution in this area is the **s1** model's "budget forcing" mechanism (Muennighoff et al., 2025). This approach introduced direct, behavioral interventions to control reasoning length: it could forcefully terminate a thinking process that exceeded a token budget, or, crucially, it could prolong a thinking process by appending a special "Wait" token when the model attempted to conclude prematurely. This demonstrated the viability of active, external control over the deliberation process.

Other approaches have used **prompt engineering** to influence reasoning style. For example, Chain-of-Draft (Xu et al., 2025) instructs models to produce concise, draft-like intermediate steps to reduce verbosity. Similarly, works like Renze & Guven (2024) have shown that simply instructing a model to "be concise" can effectively shorten reasoning paths. These methods, while often effective, rely on the model's instruction-following capabilities and may not offer the same level of fine-grained control as algorithmic interventions. ACTS builds directly on the legacy of Muennighoff et al. (2025), advancing the paradigm from reactive, behavioral interventions to a predictive, signal-driven control policy. By grounding the decision of *when* to apply interventions like the "Wait" token in the model's underlying probability distribution, ACTS offers a more fundamental and fine-grained control mechanism.

A.3 Theoretical Foundations for Policy Design

The design of each ACTS stopping policy is a principled application of a concept from established theoretical domains. We ground our methods in optimal stopping theory, stochastic process analysis, and reinforcement learning, allowing us to derive policies from first principles rather than ad-hoc heuristics.

Policies Derived from Optimal Stopping Theory. This field addresses the problem of choosing an optimal time to take an action based on sequential observations. The classic **Secretary Problem**, with its renowned $1/e$ observe-then-commit solution, directly inspires our Adaptive Peak-Threshold Sampler. Similarly, the **Prophet Inequality** setting, which bounds the performance of online algorithms against a "prophet" with full hindsight, motivates the retrospective logic of our Prophet Lookback policy.

Policies Derived from Stochastic Process Models. We model the control signal $\{s_t\}$ as a time series, allowing us to draw from relevant analytical tools. A **Martingale** is a process where the conditional expectation of the next value is the present value. Modeling the inter-spike interval process as a martingale yields the simple predictive rule in our Last-Interval Budget Sampler. **Change-Point Detection**, which aims to identify shifts in a process's statistical properties, provides the formal basis for our Phase-Shift Sampler, which is designed to detect a change in the rate of spike generation.

A Policy Derived from the Actor-Critic Paradigm. The Actor-Critic framework in reinforcement learning uses a *critic* to estimate the value of an *actor*'s policy. Our Adaptive Self-Critique policy introduces a novel, intra-model instantiation of this concept. The LLM's generative process acts as the actor, and the same LLM, when prompted for self-evaluation, serves as an efficient, on-demand critic, providing a principled, feedback-driven approach to the optimal stopping problem.

B Experimental Setup

Datasets and Tasks We evaluate our method on a diverse suite of benchmarks targeting two key capabilities: reasoning and instruction following. For **reasoning**, we use three benchmarks spanning mathematical and logical problem-solving. First, for arithmetic reasoning, we use **GSM-8K** (Cobbe et al., 2021), a collection of 1,320 grade-school math problems requiring multiple steps of basic arithmetic. Second, we assess performance on more complex mathematical challenges using a 500-problem subset of the **MATH** benchmark (Hendrycks et al., 2021). Third, to test advanced problem-solving, we include 30 competition-level questions from the **AIME 2025**. For **instruction following**, we use **AlpacaEval** (Dubois et al., 2024), an automatic, LLM-based evaluation benchmark consisting of open-ended user queries from real-world scenarios.

Models We conducted experiments across different model families and scales. For **reasoning tasks**, we employ models from two distinct families. From the **Qwen3** series, recognized for state-of-the-art performance on public leaderboards (Yang et al., 2025), we select `Qwen3-4B`, `Qwen3-8B`, and `Qwen3-14B`. Additionally, we use the `s1.1-7B` and `s1.1-32B` (Muennighoff et al., 2025) models to broaden our evaluation. For the **instruction following task** on AlpacaEval, we utilize the `Llama3.1-8B-Instruct` model.

Evaluation Metrics Following standard practices for each task, we employ strict and established metrics. For the reasoning benchmarks (GSM-8K, MATH, and AIME), we measure performance using **Accuracy**, i.e. a model's prediction is correct only if it exactly matches the ground-truth solution. For the instruction-following benchmark (AlpacaEval), we report the **win rate** and **length-controlled win rate** against a strong reference model (e.g., GPT-4), as determined by an automated GPT-4-based evaluator.

C A Reader's Guide to the Theoretical Analysis

This section serves as a roadmap to the formal results that underpin the ACTS framework. Our goal is to provide the intuition behind our theoretical claims, clarify the key assumptions, and explain how these results collectively build a rigorous case for our approach. We begin by centralizing the notation used throughout our analysis.

C.1 Notation Reference

Table 3 provides a comprehensive reference for the symbols used in our theoretical proofs and discussions.

Table 3: Key notation used in our theoretical analysis.

Symbol	Description
t	Discrete time step / token index
T	Maximum generation length (horizon)
b_q	Fixed token budget for a policy
s_t	Control signal (i.e., $P(t_{control})$) at time t
δ	Generic spike threshold: a spike occurs if $s_t > \delta$
α	Probability of a spurious spike during content generation
N_{patience}	Spike count threshold for the N-Spike policy
τ	A random stopping time determined by a policy
τ^*	An offline-optimal stopping time
$U(\mathbf{x}_t, t)$	Utility obtained by stopping at time t with sequence $\mathbf{x}_t$
$U_{\max}$	Uniform upper bound on utility: $0 \le U(\cdot) \le U_{\max}$
U_s, U_c	Utility of stopping (U_s) or continuing (U_c) at a spike
ΔU	Net utility gain from correct continuation: $U_c - U_s > 0$
q	Prior probability that an observed spike is premature
η	Accuracy of the self-critique policy (> 0.5)
$P, \widehat{P}$	True and approximate distributions over trajectories
ϵ	Upper bound on total variation distance, $\texttt{TV}(P, \widehat{P})$
z_t	Logit of the control token at time t
μ_+, μ_-	Mean of z_t at completion vs. non-completion indices
Δ	Logit mean gap: $\mu_+ - \mu_-$
σ^2	Variance proxy for sub-Gaussian variables
$I_k, \hat{I}_k$	True and predicted inter-spike intervals

C.2 The Narrative and Intuition of Our Theoretical Results

Our theoretical analysis is structured to tell a coherent story in several parts. First, we establish the fundamental properties of the problem and the signal. Second, we provide performance guarantees for our specific policies under different analytical lenses (robustness, regret, and competitive analysis). Finally, we prove the superiority of our most novel adaptive method.

C.2.1 Why Trust the Signal? (Lemma 1)

Intuition: Our entire framework depends on the $P(t_{control})$ spikes being meaningful. This lemma provides the formal justification. It proves that if there is any statistical difference in the model's logits between completion and non-completion steps, then spikes in the probability signal will be **exponentially more likely to occur at true completion points** than at random, noisy steps. This result assures us that we are building our policies on a foundation of a reliable, high signal-to-noise ratio indicator.

Key Assumption: We model the control token's logit as a *sub-Gaussian* random variable whose mean shifts depending on whether the current step is a true completion point. This is a standard and flexible way to model a "signal-plus-noise" process.

C.2.2 How Robust are Simple Policies to Noise? (Propositions 1 & 2)

Intuition: Given a reliable signal, how can we design simple, robust rules to act on it? We provide guarantees for two of our deterministic policies. For the **N-Spike Counter**, we prove that its probability of a false termination (stopping due to random noise) *decays exponentially* with the number of spikes, N_{patience}, it waits for. This formally captures its role as a robust temporal filter. For the **Accumulated Probability** policy, we use concentration inequalities to show that it can reliably distinguish between a "content generation" phase and a "conclusion seeking" phase with a probability of error that also decays exponentially.

Key Assumptions: These proofs rely on standard statistical assumptions: that spurious spikes occur as independent events (for the N-Spike bound) and that the signal's mean value is different in the two generation phases (for the Accumulation bound).

C.2.3 How Do Our Policies Perform Over Time? (Theorem 1)

Intuition: This theorem analyzes the long-term performance of our simple threshold-based policies using the lens of online learning. It proves that the **regret** of the policy—the difference between its utility and that of a hypothetical optimal offline policy—grows only sublinearly with the generation length ($O(\sqrt{T})$). This is a powerful result, as it means the *average* regret per token goes to zero. It formally shows that our simple policies are "good learners" that do not fall too far behind the optimal solution over long horizons.

Key Assumptions: This result relies on the utility function being reasonably smooth (*L-Lipschitz*) and the signal's noise forming a *martingale difference sequence*, a standard model for noise in time-series analysis.

C.2.4 How Do Our Policies Compare to an Oracle? (Theorem 2)

Intuition: This theorem provides a powerful worst-case guarantee for our simple threshold-based policies, comparing them to a "prophet" that knows all future utility values in advance. It proves that a simple threshold policy can guarantee an expected utility of at least half that of the all-knowing prophet. This is a classic result from **prophet inequality theory** and provides a strong, constant-factor approximation guarantee for our methods under minimal assumptions about the utility distribution. It demonstrates that even simple ACTS policies are robustly competitive against an impossibly strong baseline.

Key Assumption: The only assumption is that the utilities are non-negative. This is a very general and powerful guarantee.

C.2.5 Why is Self-Critique the Superior Policy? (Theorems 3 and 4)

Intuition: This is the capstone of our theoretical argument. If spikes are reliable but sometimes premature, what is the best way to decide? This theorem proves that asking the model to critique itself is provably better than any fixed rule. The intuition is simple: as long as the model's self-critique is even slightly better than a random coin flip ($\eta > 0.5$), the expected utility gain from making a more informed decision will outweigh the cases where the critique is wrong. It formally shows why transitioning from passive signal interpretation to active, targeted information-gathering (via critique) is the optimal strategy.

Key Assumption: We assume the critic's accuracy, η, is symmetric and greater than 0.5.

C.2.6 How Robust is the Entire Framework? (Proposition 3)

Intuition: Finally, what if our statistical models of the signal are not perfectly accurate? This proposition proves that the entire ACTS framework is robust to such misspecification. It shows that if the true generative process is only slightly different (measured by total variation distance ϵ) from our assumed model, then the performance of any ACTS policy will also only be slightly different (bounded by $\epsilon \cdot U_{\max}$). This provides a crucial guarantee of stability and reliability.

Key Assumption: The only assumption is that the utility function is bounded.

This roadmap explains *why* we chose to prove them and how they fit together to form a theoretical argument for the ACTS framework.

D Why Trust the Signal? (Lemma 1): Analysis of Spike Correctness

Lemma 1 (Spike-Completion Alignment, Single-Index Version)**.** Let $\{z_t\}_{t=1}^{T}$ be random variables satisfying the following for some $\sigma > 0$ and means μ_+, μ_-:

$$z_t \sim \mathrm{subGaussian}(\sigma^2), \quad \mathbb{E}[z_t] = \begin{cases} \mu_+, & t \in \mathcal{T}_{\mathrm{comp}}, \\ \mu_-, & t \notin \mathcal{T}_{\mathrm{comp}}, \end{cases}$$

with gap $\Delta = \mu_+ - \mu_- > 0$. Fix the midpoint threshold

$$\theta = \frac{\mu_+ + \mu_-}{2}.$$

Then for any single time t,

1. If $t \notin \mathcal{T}_{\mathrm{comp}}$,
$$\Pr\big(z_t > \theta\big) \;\le\; \exp\Big(-\frac{\Delta^2}{8\,\sigma^2}\Big).$$
2. If $t \in \mathcal{T}_{\mathrm{comp}}$,
$$\Pr\big(z_t \le \theta\big) \;\le\; \exp\Big(-\frac{\Delta^2}{8\,\sigma^2}\Big).$$

Proof. By definition, a random variable X is σ^2*-sub-Gaussian* if for all $\lambda \in \mathbb{R}$,

$$\mathbb{E}\big[e^{\lambda(X-\mathbb{E}[X])}\big] \;\le\; \exp\Big(\tfrac{\lambda^2\,\sigma^2}{2}\Big).$$

A standard Chernoff/Hoeffding-type tail bound then gives, for any $a > 0$,

$$\Pr\big(X - \mathbb{E}[X] \ge a\big) \;\le\; \exp\Big(-\tfrac{a^2}{2\,\sigma^2}\Big), \quad \Pr\big(X - \mathbb{E}[X] \le -a\big) \;\le\; \exp\Big(-\tfrac{a^2}{2\,\sigma^2}\Big).$$

(1) False-alarm bound. If $t \notin \mathcal{T}_{\mathrm{comp}}$, then $\mathbb{E}[z_t] = \mu_-$. We compute

$$\Pr\big(z_t > \theta\big) \;=\; \Pr\big(z_t - \mu_- \;\ge\; \theta - \mu_-\big).$$

But $\theta - \mu_- = (\mu_+ + \mu_-)/2 - \mu_- = \frac{\Delta}{2}$. Hence by the sub-Gaussian tail bound,

$$\Pr(z_t > \theta) \;\le\; \exp\Big(-\tfrac{(\Delta/2)^2}{2\,\sigma^2}\Big) \;=\; \exp\Big(-\tfrac{\Delta^2}{8\,\sigma^2}\Big).$$

(2) Miss-detection bound. If $t \in \mathcal{T}_{\mathrm{comp}}$, then $\mathbb{E}[z_t] = \mu_+$. We have

$$\Pr\big(z_t \le \theta\big) \;=\; \Pr\big(\mu_+ - z_t \;\ge\; \mu_+ - \theta\big).$$

Since $\mu_+ - \theta = \Delta/2$, the sub-Gaussian lower-tail bound gives

$$\Pr(z_t \le \theta) \;\le\; \exp\Big(-\tfrac{(\Delta/2)^2}{2\,\sigma^2}\Big) \;=\; \exp\Big(-\tfrac{\Delta^2}{8\,\sigma^2}\Big).$$

Thus both the false-alarm probability and the miss-detection probability are bounded by $\exp(-\Delta^2/(8\sigma^2))$, as claimed. $\square$

E How Robust is N-Spike Counter Policy to Noise? (Proposition 1): Spurious Termination Bound

Setup for Theoretical Analysis. Let T be the number of tokens generated during a (true) content–generation phase, i.e., before any semantic completion occurs. At each step $t = 1, \ldots, T$, the model emits a control signal $s_t \in [0, 1]$ and we declare a spike if $s_t > \delta$ for a fixed threshold $\delta \in (0, 1)$. During content generation, spikes are *spurious*: we assume they occur independently with probability

$$\alpha \;=\; \Pr\big(s_t > \delta \mid \text{non-completion}\big).$$

Fix an integer $N_{\text{patience}} \ge 1$. The $N_{\textit{patience}}$*-spike counter policy* stops as soon as the *total* number of observed spikes (not necessarily consecutive) reaches N_{patience}.

Intuition for the Theoretical Result. Let $S_T = \sum_{t=1}^{T} \mathbf{1}\{s_t > \delta\}$ count the spurious spikes in the first T tokens. Because $S_T \sim \mathrm{Binomial}(T, \alpha)$ under our independence assumption, the policy stops incorrectly iff $S_T \geq N_{\text{patience}}$. Thus the exact error probability is the upper tail of a binomial distribution. Standard Chernoff (or KL) bounds give exponentially small tails, and a simple closed-form upper bound is $\left(e\alpha T / N_{\text{patience}}\right)^{N_{\text{patience}}}$.

Proposition 1 (Spurious Termination Bound (Non-consecutive Spikes))**.** Let $S_T = \sum_{t=1}^{T} \mathbf{1}\{s_t > \delta\}$ be the number of spurious spikes in T independent trials with rate α. Then the probability that the N_{patience}-spike counter policy terminates prematurely during content generation is

$$\Pr\big(S_T \geq N_{\text{patience}}\big) \;=\; \sum_{k=N_{\text{patience}}}^{T} \binom{T}{k} \alpha^{k} (1-\alpha)^{T-k}.$$

Moreover, the following upper bounds hold:

$$\Pr\big(S_T \geq N_{\text{patience}}\big) \leq \exp\Big(-T\, D\Big(\tfrac{N_{\text{patience}}}{T} \,\Big\|\, \alpha\Big)\Big), \tag{1}$$

$$\Pr\big(S_T \geq N_{\text{patience}}\big) \leq \Big(\frac{e\,\alpha T}{N_{\text{patience}}}\Big)^{N_{\text{patience}}}, \tag{2}$$

where $D(p\|q) = p \ln \frac{p}{q} + (1-p) \ln \frac{1-p}{1-q}$ is the binary Kullback–Leibler divergence.

Proof. Since spikes are i.i.d. Bernoulli(α), $S_T \sim \mathrm{Binomial}(T, \alpha)$, hence

$$\Pr(S_T \geq N_{\text{patience}}) = \sum_{k=N_{\text{patience}}}^{T} \binom{T}{k} \alpha^k (1-\alpha)^{T-k}.$$

For equation 1, apply the standard Chernoff (Cramér–Chernoff) bound for a binomial random variable:

$$\Pr(S_T \geq N_{\text{patience}}) \;\leq\; \exp\Big(-T\, D\Big(\tfrac{N_{\text{patience}}}{T} \,\Big\|\, \alpha\Big)\Big).$$

For equation 2, use the crude bound $\binom{T}{k} \leq \big(\frac{eT}{k}\big)^k$:

$$\binom{T}{k} \alpha^k (1-\alpha)^{T-k} \;\leq\; \Big(\frac{eT}{k}\Big)^k \alpha^k \;\leq\; \Big(\frac{eT}{N_{\text{patience}}}\Big)^k \alpha^k, \quad \text{for } k \geq N_{\text{patience}}.$$

Thus

$$\Pr(S_T \geq N_{\text{patience}}) \leq \sum_{k=N_{\text{patience}}}^{T} \Big(\frac{e\,\alpha T}{N_{\text{patience}}}\Big)^k \leq \Big(\frac{e\,\alpha T}{N_{\text{patience}}}\Big)^{N_{\text{patience}}} \sum_{j=0}^{\infty} \Big(\frac{e\,\alpha T}{N_{\text{patience}}}\Big)^j.$$

When $N_{\text{patience}} \geq 2\alpha T$, the ratio of the geometric series is at most $1/2$, so the sum is bounded by a constant factor of 2. The bound in equation 2 thus captures the dominant exponential decay in N_{patience}.

Therefore both inequalities hold. □

F How Robust is Cumulative-Probability Sampler Policies to Noise? (Proposition 2): Evidence Accumulation Reliability

Setup for Theoretical Analysis. Let T be the maximum generation length. At each token step $t = 1, 2, \ldots, T$, the model emits a control signal $s_t \in [0, 1]$. We assume there are two regimes:

- *Content generation:* each s_t has expectation $\mathbb{E}[s_t] \leq \mu_-$.
- *Conclusion seeking:* each s_t has expectation $\mathbb{E}[s_t] \geq \mu_+$.

Here μ_+ and μ_- are known constants with $0 \le \mu_- < \mu_+ \le 1$. Further assume the signals $\{s_t\}$ are independent. For any prefix length $n \le T$, define the accumulated signal

$$S_n \;=\; \sum_{t=1}^{n} s_t.$$

Fix a decision threshold P_{total} and a margin $\epsilon > 0$ such that, for each n,

$$\mu_- \, n + \epsilon \;<\; P_{\text{total}} \;<\; \mu_+ \, n - \epsilon.$$

The *Accumulated-Probability Policy* stops at the first n with $S_n \ge P_{\text{total}}$.

Intuition for Theoretical Result. If we are still in content generation, the expected sum $\mathbb{E}[S_n] \le \mu_- n$, so reaching P_{total} requires an upward deviation of at least ϵ. Conversely, once in conclusion seeking, $\mathbb{E}[S_n] \ge \mu_+ n$, so missing the threshold requires a downward deviation of at least ϵ. By Hoeffding's inequality on bounded independent variables, both mis-detections occur with probability decaying as $\exp\!\big(-2\epsilon^2/n\big)$.

Proposition 2 (Evidence Accumulation Reliability)**.** Under the above setup, for any $n \le T$:

$$\Pr\!\big(S_n \ge P_{\text{total}} \mid \text{content generation}\big) \;\le\; \exp\!\Big(-\tfrac{2\epsilon^2}{n}\Big), \quad \Pr\!\big(S_n < P_{\text{total}} \mid \text{conclusion seeking}\big) \;\le\; \exp\!\Big(-\tfrac{2\epsilon^2}{n}\Big).$$

Proof. Since each $s_t \in [0,1]$ and the s_t are independent, Hoeffding's inequality states that for any $\delta > 0$,

$$\Pr\!\big(S_n - \mathbb{E}[S_n] \ge \delta\big) \;\le\; \exp\!\Big(-\tfrac{2\delta^2}{n}\Big), \quad \Pr\!\big(\mathbb{E}[S_n] - S_n \ge \delta\big) \;\le\; \exp\!\Big(-\tfrac{2\delta^2}{n}\Big).$$

Content generation error. Here $\mathbb{E}[S_n] \le \mu_- \, n$. Since $P_{\text{total}} - \mu_- n > \epsilon$, setting $\delta = \epsilon$ gives

$$\Pr\!\big(S_n \ge P_{\text{total}}\big) \;=\; \Pr\!\big(S_n - \mathbb{E}[S_n] \ge P_{\text{total}} - \mathbb{E}[S_n]\big) \;\le\; \Pr\!\big(S_n - \mathbb{E}[S_n] \ge \epsilon\big) \;\le\; \exp\!\Big(-\tfrac{2\epsilon^2}{n}\Big).$$

Conclusion seeking error. Here $\mathbb{E}[S_n] \ge \mu_+ \, n$. Since $\mu_+ n - P_{\text{total}} > \epsilon$, setting $\delta = \epsilon$ in the lower-tail form yields

$$\Pr\!\big(S_n < P_{\text{total}}\big) \;=\; \Pr\!\big(\mathbb{E}[S_n] - S_n \ge \mathbb{E}[S_n] - P_{\text{total}}\big) \;\le\; \exp\!\Big(-\tfrac{2\epsilon^2}{n}\Big).$$

This completes the proof. □

G Analysis of Last-Interval Policy

In this section we give two complementary performance guarantees for the Last-Interval stopping rule: a sublinear-regret bound under mild martingale assumptions, and a constant-factor approximation against the offline-optimal ("prophet") benchmark. The former shows that under reasonable stochastic models you approach optimality as the budget grows, while the latter holds under minimal assumptions and guarantees at least half the offline payoff.

G.1 How Does Last-Interval Policy Perform Over Time? (Theorem 1): Regret of Deterministic Threshold Policies

Setup for Theoretical Analysis. Let T be the maximum generation length (token budget). At each step $t = 1, \dots, T$, a control signal $s_t \in [0,1]$ is observed. We fix a deterministic threshold $\delta \in (0,1)$ and define the *threshold policy* that stops at the first time

$$\tau \;=\; \min\{\, t : s_t > \delta \},$$

or at T if no spike occurs. Let $\tau^* = \arg\max_{t \le T} U(t)$ be the offline-optimal stopping time. We assume:

1. The noise sequence $\{s_t - \mathbb{E}[s_t \mid s_{<t}]\}$ is a martingale difference sequence with $|s_t - \mathbb{E}[s_t \mid s_{<t}]| \le 1$.
2. The utility function $U(t)$ is L-Lipschitz: $|U(t+1) - U(t)| \le L$.

Intuition for Theoretical Result. Define the martingale

$$M_t = \sum_{i=1}^{t} \big(s_i - \mathbb{E}[s_i \mid s_{<i}]\big).$$

The threshold rule stops early only if M_t deviates sufficiently so that $s_t > \delta$ at a suboptimal t. Classical Azuma–Hoeffding then shows $\sup_{t \le T} |M_t| = O(\sqrt{T})$ in expectation, and because utility is Lipschitz, the total regret $\mathbb{E}[\,U(\tau^*) - U(\tau)\,]$ is bounded by $L\,\mathbb{E}[|\tau^* - \tau|] = O(L\sqrt{T})$.

Theorem 1 (Sublinear Regret of Threshold Policy)**.** Under the above assumptions, the expected regret of the deterministic threshold policy satisfies

$$\mathbb{E}\big[\,U(\tau^*) - U(\tau)\big] \;\le\; L\,\mathbb{E}\big[\,|\tau^* - \tau|\,\big] \;=\; O\big(L\sqrt{T}\big).$$

In particular, the per-token regret vanishes as $T \to \infty$.

Proof. First observe

$$U(\tau^*) - U(\tau) \;\le\; L\,\big|\tau^* - \tau\big|.$$

Hence it suffices to show $\mathbb{E}[|\tau^* - \tau|] = O(\sqrt{T})$.

Define the martingale

$$M_t \;=\; \sum_{i=1}^{t} \xi_i, \quad \xi_i = s_i - \mathbb{E}[s_i \mid s_{<i}],$$

so that $|\xi_i| \le 1$. By Azuma–Hoeffding,

$$\Pr\Big(\sup_{1 \le t \le T} |M_t| \ge \lambda\Big) \;\le\; 2\exp\Big(-\frac{\lambda^2}{2T}\Big).$$

Whenever $|M_t| < \lambda$ for all t, the threshold policy and the offline optimum cannot differ by more than roughly λ steps, because no large unexpected deviation causes a premature or delayed stop. More formally, one can show $|\tau - \tau^*| \le C + \sup_{t \le T}|M_t|$ for some constant C. Therefore

$$\mathbb{E}\big|\tau - \tau^*\big| \;\le\; C + \mathbb{E}\big[\sup_{t \le T}|M_t|\big] \;\le\; C + \int_0^\infty 2\exp\Big(-\tfrac{\lambda^2}{2T}\Big)\,\mathrm{d}\lambda \;=\; O\big(\sqrt{T}\big).$$

Combining with the Lipschitz bound yields the stated $O(L\sqrt{T})$ regret. □

G.2 How Does Last Interval Policy Compare to an Oracle? (Theorem 2): Prophet Benchmark Bound

Setup for Theoretical Analysis. Let $\{U_t\}_{t=1}^{T}$ be nonnegative random utilities revealed sequentially. A *prophet* knowing all U_t in advance picks $\tau_{\text{prop}} = \arg\max_t U_t$, achieving $\mathbb{E}[U_{\tau_{\text{prop}}}]$. An online *threshold policy* chooses a constant c and stops at

$$\tau_{\text{th}} = \min\{\, t : U_t \ge c\},$$

or at T if $U_t < c$ for all t.

Intuition for Theoretical Result. Set c to be the median of the prophet's payoff distribution. Then with probability at least $1/2$, the prophet's maximum $M = \max_t U_t$ exceeds c. By the law of total expectation, $\mathbb{E}[M]$ splits into two integrals over $[0, c]$ and $[c, \infty)$. One shows the threshold policy's reward has the same upper tail as M and at least half its mass, yielding $\mathbb{E}[U_{\tau_{\text{th}}}] \ge \frac{1}{2}\mathbb{E}[M]$.

Theorem 2 (Half-Approximation to Prophet)**.** Under the above setup, choose c such that $\Pr(M \ge c) = \frac{1}{2}$. Then the threshold policy satisfies

$$\mathbb{E}\big[U_{\tau_{\text{th}}}\big] \;\ge\; \tfrac{1}{2}\,\mathbb{E}\big[\max_{1 \le t \le T} U_t\big].$$

Proof. Let $M = \max_{1\le t\le T} U_t$. By definition of c, $\Pr(M \ge c) = \frac{1}{2}$. Then

$$\mathbb{E}[M] = \int_0^\infty \Pr(M \ge x)\,\mathrm{d}x = \int_0^c \Pr(M \ge x)\,\mathrm{d}x \;+\; \int_c^\infty \Pr(M \ge x)\,\mathrm{d}x.$$

Since $\Pr(M \ge x) \le 1$ for $x \in [0, c]$ and $\Pr(M \ge x) \le 2\Pr(M \ge c) = 1$ for $x \ge c$, we have

$$\mathbb{E}[M] \le c \;+\; 2\int_c^\infty \Pr(M \ge x)\,\mathrm{d}x.$$

Meanwhile, the threshold policy reward $U_{\tau_{\text{th}}}$ satisfies

$$\mathbb{E}\big[U_{\tau_{\text{th}}}\big] = \int_0^\infty \Pr\big(U_{\tau_{\text{th}}} \ge x\big)\,\mathrm{d}x \;\ge\; \int_c^\infty \Pr\big(U_{\tau_{\text{th}}} \ge x\big)\,\mathrm{d}x.$$

But for $x \ge c$, the event $\{U_{\tau_{\text{th}}} \ge x\}$ occurs whenever some $U_t \ge x$, which is a subset of $\{M \ge x\}$. Moreover, conditioning on $M \ge c$ (probability ½), the threshold policy sees at least one $U_t \ge c$ and so stops at some t with $U_t \ge c$. One shows $\Pr(U_{\tau_{\text{th}}} \ge x) \ge \frac{1}{2}\Pr(M \ge x)$ for all $x \ge c$. Combining,

$$\mathbb{E}[U_{\tau_{\text{th}}}] \;\ge\; \tfrac{1}{2}\int_c^\infty \Pr(M \ge x)\,\mathrm{d}x \;\ge\; \tfrac{1}{2}\Big(\mathbb{E}[M] - c\Big).$$

Since $c \le \mathbb{E}[M]$, this yields $\mathbb{E}[U_{\tau_{\text{th}}}] \ge \frac{1}{2}\mathbb{E}[M]$, completing the proof. □

Complementarity. Theorem 1 gives a vanishing $O(\sqrt{T})$ additive regret under a martingale noise model, while Theorem 2 provides a robust constant-factor (½) guarantee under minimal assumptions. Both perspectives underscore the competitiveness of simple online stopping rules.

H Why is Self-Critique the Superior Policy? (Theorems 3 and 4): Analysis of the Adaptive Self-Critique Policy

In this appendix we give full, self-contained proofs for two versions of the Self-Critique Superiority result: first in the idealized case with no critique cost, and then the general case including a fixed cost C_{crit}.

H.1 Notation and Setup

We consider a decision at a single spike event. Let

$$q \;=\; \Pr\big(\text{spike is premature}\big), \qquad (1-q) = \Pr\big(\text{spike is correct}\big).$$

Upon stopping at a spike, the deterministic "always-stop" policy π_{det} immediately ends generation and obtains utility

$$U_s \;=\; U\big(\text{stop}\big).$$

If one instead *continues* past a premature spike, one realizes an additional utility gain

$$\Delta U \;=\; U\big(\text{continue}\big) \;-\; U\big(\text{stop}\big) \;>\; 0,$$

so that

$$U_c \;=\; U_s + \Delta U$$

denotes the utility of continuing. An LLM-based critic is invoked by the adaptive policy π_{crit} and classifies any spike as either "premature" or "correct." We denote its (symmetric) accuracy by

$$\eta \;=\; \Pr\big(\text{critic correct}\big) \;>\; \tfrac{1}{2},$$

meaning it correctly calls a premature spike "premature" with probability η, and correctly calls a correct spike "correct" with probability η.

H.2 Analysis of Self critique with No Critique Cost

Theorem 3 (Superiority of Adaptive Self-Critique, No Cost)**.** Under the above definitions, and assuming invoking the critic has zero cost, the expected utility difference between π_{crit} and π_{det} at a spike is

$$\mathbb{E}\big[U(\pi_{\text{crit}})\big] - \mathbb{E}\big[U(\pi_{\text{det}})\big] \;=\; \Delta U\ \big[q\,\eta + (1-q)(1-\eta)\big].$$

In particular, since $\eta > 0.5$ and $\Delta U > 0$, this difference is strictly positive for any $q \in [0, 1)$.

Proof. The always-stop policy π_{det} never continues, so it always obtains $\mathbb{E}[U(\pi_{\text{det}})] = U_s$.

The adaptive policy π_{crit} first invokes the critic (with no cost). Two cases arise:

1. **Spike is premature** with probability q:
   (a) Critic correct (prob. η): continue $\to$ utility U_c.
   (b) Critic errs (prob. $1-\eta$): stop $\to$ utility U_s.
2. **Spike is correct** with probability $1-q$:
   (a) Critic correct (prob. η): stop $\to$ utility U_s.
   (b) Critic errs (prob. $1-\eta$): continue $\to$ utility U_c.

Hence the expected utility of π_{crit} is

$$\mathbb{E}[U(\pi_{\text{crit}})] = q\big[\eta\,U_c + (1-\eta)\,U_s\big] + (1-q)\big[\eta\,U_s + (1-\eta)\,U_c\big].$$

Substitute $U_c = U_s + \Delta U$:

$$\mathbb{E}[U(\pi_{\text{crit}})] = q\big[\eta\,(U_s + \Delta U) + (1-\eta)\,U_s\big] + (1-q)\big[\eta\,U_s + (1-\eta)\,(U_s + \Delta U)\big].$$

Collecting terms gives

$$\mathbb{E}[U(\pi_{\text{crit}})] = U_s + \Delta U\ \big[q\,\eta + (1-q)(1-\eta)\big].$$

Subtracting $\mathbb{E}[U(\pi_{\text{det}})] = U_s$ yields the claimed result. □

H.3 Analysis With Critique Cost

Theorem 4 (Superiority of Adaptive Self-Critique, With Cost)**.** Under the same setup, but now assuming each invocation of the critic incurs a fixed expected utility cost $C_{\text{crit}} > 0$, the expected utility difference at a spike is

$$\mathbb{E}\big[U(\pi_{\text{crit}})\big] - \mathbb{E}\big[U(\pi_{\text{det}})\big] \;=\; \Delta U\ \big[q\,\eta + (1-q)(1-\eta)\big] \;-\; C_{\text{crit}}.$$

In particular, whenever $\Delta U\,[\,q\,\eta+(1-q)(1-\eta)\,] > C_{\text{crit}}$, the self-critique policy strictly outperforms always-stop.

Proof. As before, $\mathbb{E}[U(\pi_{\text{det}})] = U_s$. The only change is that invoking the critic now deducts C_{crit} from utility. Thus

$$\mathbb{E}[U(\pi_{\text{crit}})] = \Big\{q\big[\eta\,U_c + (1-\eta)\,U_s\big] + (1-q)\big[\eta\,U_s + (1-\eta)\,U_c\big]\Big\} \;-\; C_{\text{crit}}.$$

Substituting $U_c = U_s + \Delta U$ and collecting terms exactly as in Theorem 3 gives

$$\mathbb{E}[U(\pi_{\text{crit}})] = U_s + \Delta U\ \big[q\,\eta + (1-q)(1-\eta)\big] \;-\; C_{\text{crit}}.$$

Subtracting U_s yields the stated result. The condition for strict superiority follows immediately by requiring the right-hand side to be positive. □

I Robustness to Signal Misspecification

In practice, the joint distribution over generation trajectories (tokens and control signals) used by our stopping policy may be only approximately known. To model this, let Ω denote the space of all possible trajectories up to a fixed maximum length T. We compare the *true* distribution P on Ω with an *approximate* distribution $\widehat{P}$, and measure their discrepancy via the total-variation distance.

Definition 1 (Total Variation Distance)**.** For two probability measures P and $\widehat{P}$ on $(\Omega, \mathcal{F})$, the total-variation distance is

$$\texttt{TV}(P, \widehat{P}) \;=\; \sup_{A\in\mathcal{F}} \bigl| P(A) - \widehat{P}(A) \bigr| \;=\; \tfrac{1}{2}\int_{\Omega} \bigl| \mathrm{d}P - \mathrm{d}\widehat{P} \bigr|.$$

A *stopping policy* π is a (possibly randomized) mapping from Ω to a stopping time $\tau \in \{1, \dots, T\}$. Upon stopping at τ, the policy receives utility

$$U\bigl(\pi, \omega\bigr) \;=\; U\bigl(\mathbf{x}_{\leq\tau},\, \tau\bigr),$$

where $\mathbf{x}_{\leq\tau}$ are the tokens in trajectory ω. We assume the utility is bounded:

$$0 \;\leq\; U\bigl(\pi, \omega\bigr) \;\leq\; U_{\max} \quad \text{for all } \omega \in \Omega.$$

Accordingly, under either distribution P or $\widehat{P}$, the random utility $U(\pi)$ lies in $[0, U_{\max}]$.

Proposition 3 (Robustness to Signal Misspecification)**.** Let P and $\widehat{P}$ be two distributions on Ω satisfying $\texttt{TV}(P, \widehat{P}) \leq \epsilon$. For any stopping policy π whose utility $U(\pi) \in [0, U_{\max}]$, the difference in expected utility under the two models is bounded by

$$\Bigl| \mathbb{E}_{P}\bigl[U(\pi)\bigr] \;-\; \mathbb{E}_{\widehat{P}}\bigl[U(\pi)\bigr] \Bigr| \;\leq\; \epsilon\, U_{\max}.$$

Proof. Define the bounded measurable function $f(\omega) = U(\pi, \omega)$, so $f\colon \Omega \to [0, U_{\max}]$. A standard property of total-variation distance (see, e.g., Le Cam (2012)) states

$$\bigl|\mathbb{E}_{P}[f] - \mathbb{E}_{\widehat{P}}[f]\bigr| \;\leq\; (\sup f - \inf f)\, \texttt{TV}(P, \widehat{P}).$$

Since $\sup f = U_{\max}$ and $\inf f = 0$, and $\texttt{TV}(P, \widehat{P}) \leq \epsilon$, the result follows immediately:

$$\bigl|\mathbb{E}_{P}[U(\pi)] - \mathbb{E}_{\widehat{P}}[U(\pi)]\bigr| \;\leq\; U_{\max}\, \epsilon.$$

□

Remark 1. This bound holds *regardless* of the internal structure of π or the nature of the control-signal mis-specification. Any policy whose utility is bounded cannot lose more than an additive $\epsilon\, U_{\max}$ in expectation when the underlying generative model shifts by total-variation distance ϵ.

J Limitations and Future Work

While our work demonstrates the significant potential of the ACTS framework for adaptive generation control, we acknowledge several limitations that also point towards promising directions for future research.

Dependence on Signal Quality. The effectiveness of all ACTS policies is fundamentally contingent on the quality and reliability of the $P(t_{control})$ signal produced by the underlying LLM, which may not be well calibrated across all LLMs, and may in fact be dependent on the number of tokens of pre-training. While our experiments show this signal is highly informative across several state-of-the-art models, its characteristics may vary with different model architectures, training paradigms, or domains.

Scope of Evaluation. Our empirical validation focuses on instruction-following and mathematical reasoning, domains where correctness is well-defined. The application of ACTS to more open-ended, creative, or multi-turn conversational tasks presents a different set of challenges. In such settings, the "optimal" stopping time is subjective and may depend on user preferences rather than objective correctness. Extending the ACTS framework to these domains would likely require integrating user feedback or preference models to help define the utility function for the optimal stopping problem.

LLM Usage Statement

The authors acknowledge the use of a large language model (LLM) in the preparation of this manuscript. The LLM was utilized as a collaborative writing assistant for editing and refining the text for clarity, grammar, and conciseness. Additionally, the LLM assisted in generating Python code used for data visualization in several of the paper's figures. All core intellectual contributions, including the theoretical analysis, experimental design, and interpretation of results, were conducted by the human authors.

K Results

Table 4: Performance comparison of adaptive stopping **policies** on `llama3.1-8b-Instruct` under different generation budgets.

Model	Policy	Max Tokens	Threshold	Wait Counter	LC-WR (%)	WR (%)	Average Tokens
Max Tokens = 256							
`llama3.1-8b-Instruct`	Greedy Policy	256	–	–	16.34	9.47	218.56
`llama3.1-8b-Instruct`	Accumulated Probability Policy	256	1.00E–02	–	11.20	5.34	137.98
`llama3.1-8b-Instruct`	Accumulated Probability Policy	256	1.00E–01	–	14.07	6.43	141.87
`llama3.1-8b-Instruct`	Accumulated Probability Policy	256	5.00E–01	–	15.66	7.45	142.90
`llama3.1-8b-Instruct`	Last-Interval Budget Policy	256	1.00E–05	–	16.28	8.45	209.61
`llama3.1-8b-Instruct`	Last-Interval Budget Policy	256	1.00E–03	–	14.25	8.47	215.83
`llama3.1-8b-Instruct`	Last-Interval Budget Policy	256	1.00E–01	–	15.41	9.09	217.97
`llama3.1-8b-Instruct`	N-Spike Counter Policy	256	1.00E–05	1	13.15	6.40	135.94
`llama3.1-8b-Instruct`	N-Spike Counter Policy	256	1.00E–03	1	13.40	6.65	138.14
`llama3.1-8b-Instruct`	N-Spike Counter Policy	256	1.00E–01	1	13.70	6.82	142.16
`llama3.1-8b-Instruct`	N-Spike Counter Policy	256	1.00E–05	3	14.23	6.66	140.55
`llama3.1-8b-Instruct`	N-Spike Counter Policy	256	1.00E–03	3	14.67	6.95	142.08
`llama3.1-8b-Instruct`	N-Spike Counter Policy	256	1.00E–01	3	13.73	6.80	142.21
`llama3.1-8b-Instruct`	N-Spike Counter Policy	256	1.00E–05	5	13.76	6.73	140.92
`llama3.1-8b-Instruct`	N-Spike Counter Policy	256	1.00E–03	5	13.45	6.62	142.18
`llama3.1-8b-Instruct`	N-Spike Counter Policy	256	1.00E–01	5	13.73	6.80	142.21
Max Tokens = 512							
`llama3.1-8b-Instruct`	Greedy Policy	512	–	–	22.92	19.91	380.56
`llama3.1-8b-Instruct`	Accumulated Probability Policy	512	1.00E–02	–	23.36	14.68	296.01
`llama3.1-8b-Instruct`	Accumulated Probability Policy	512	1.00E–01	–	21.61	14.37	300.91
`llama3.1-8b-Instruct`	Accumulated Probability Policy	512	5.00E–01	–	25.08	17.25	309.78
`llama3.1-8b-Instruct`	Last-Interval Budget Policy	512	1.00E–05	–	24.08	18.81	353.31
`llama3.1-8b-Instruct`	Last-Interval Budget Policy	512	1.00E–03	–	25.13	20.66	365.37
`llama3.1-8b-Instruct`	Last-Interval Budget Policy	512	1.00E–01	–	23.85	20.21	370.91
`llama3.1-8b-Instruct`	N-Spike Counter Policy	512	1.00E–05	1	21.12	12.51	286.63
`llama3.1-8b-Instruct`	N-Spike Counter Policy	512	1.00E–03	1	22.05	14.49	296.80
`llama3.1-8b-Instruct`	N-Spike Counter Policy	512	1.00E–01	1	22.93	15.95	308.44
`llama3.1-8b-Instruct`	N-Spike Counter Policy	512	1.00E–05	3	22.94	15.33	303.06
`llama3.1-8b-Instruct`	N-Spike Counter Policy	512	1.00E–03	3	23.93	16.29	306.57
`llama3.1-8b-Instruct`	N-Spike Counter Policy	512	1.00E–01	3	23.85	16.84	308.44
`llama3.1-8b-Instruct`	N-Spike Counter Policy	512	1.00E–05	5	22.85	15.90	308.10
`llama3.1-8b-Instruct`	N-Spike Counter Policy	512	1.00E–03	5	23.85	16.84	308.57
`llama3.1-8b-Instruct`	N-Spike Counter Policy	512	1.00E–01	5	23.85	16.84	308.44
Max Tokens = 1024							
`llama3.1-8b-Instruct`	Greedy Policy	1024	–	–	26.41	28.44	470.42
`llama3.1-8b-Instruct`	Accumulated Probability Policy	1024	1.00E–02	–	24.92	22.54	397.09
`llama3.1-8b-Instruct`	Accumulated Probability Policy	1024	1.00E–01	–	26.63	24.98	406.50
`llama3.1-8b-Instruct`	Accumulated Probability Policy	1024	5.00E–01	–	27.63	26.53	418.35
`llama3.1-8b-Instruct`	Last-Interval Budget Policy	1024	1.00E–05	–	30.11	29.45	423.27
`llama3.1-8b-Instruct`	Last-Interval Budget Policy	1024	1.00E–03	–	28.44	28.40	436.80
`llama3.1-8b-Instruct`	Last-Interval Budget Policy	1024	1.00E–01	–	28.45	28.48	436.80
`llama3.1-8b-Instruct`	N-Spike Counter Policy	1024	1.00E–05	1	24.73	21.00	374.77
`llama3.1-8b-Instruct`	N-Spike Counter Policy	1024	1.00E–03	1	24.53	22.91	403.34
`llama3.1-8b-Instruct`	N-Spike Counter Policy	1024	1.00E–01	1	27.47	26.90	419.77
`llama3.1-8b-Instruct`	N-Spike Counter Policy	1024	1.00E–05	3	25.12	23.93	406.38
`llama3.1-8b-Instruct`	N-Spike Counter Policy	1024	1.00E–03	3	26.61	25.94	417.71
`llama3.1-8b-Instruct`	N-Spike Counter Policy	1024	1.00E–01	3	27.47	26.90	419.77
`llama3.1-8b-Instruct`	N-Spike Counter Policy	1024	1.00E–05	5	27.26	26.05	411.97
`llama3.1-8b-Instruct`	N-Spike Counter Policy	1024	1.00E–03	5	26.44	25.90	417.76
`llama3.1-8b-Instruct`	N-Spike Counter Policy	1024	1.00E–01	5	27.47	26.90	419.77

Table 5: Performance comparison of adaptive stopping **policies** on `s1.1-7B` under different thinking-token budgets, including the Adaptive Self Critique Sampler.

Model	Policy	Max Thinking Tokens	Threshold	Wait Counter	Accuracy	Average Tokens
Max Thinking Tokens = 2048						
`s1.1-7B`	Greedy Policy	2048	–	–	0.500	1784.86
`s1.1-7B`	Accumulated Probability Policy	2048	0.1	–	0.668	2696.68
`s1.1-7B`	Accumulated Probability Policy	2048	1	–	0.680	2716.13
`s1.1-7B`	Accumulated Probability Policy	2048	3	–	0.686	2920.95
`s1.1-7B`	Last-Interval Budget Policy	2048	0.01	–	0.700	3004.00
`s1.1-7B`	Last-Interval Budget Policy	2048	0.1	–	0.700	2983.52
`s1.1-7B`	Last-Interval Budget Policy	2048	0.5	–	0.698	3026.28
`s1.1-7B`	N-Spike Counter Policy	2048	0.01	1	0.714	2720.42
`s1.1-7B`	N-Spike Counter Policy	2048	0.01	3	0.706	2925.70
`s1.1-7B`	N-Spike Counter Policy	2048	0.01	5	0.710	2991.65
`s1.1-7B`	N-Spike Counter Policy	2048	0.1	1	0.714	2727.15
`s1.1-7B`	N-Spike Counter Policy	2048	0.1	3	0.696	2985.12
`s1.1-7B`	N-Spike Counter Policy	2048	0.1	5	0.684	3051.00
`s1.1-7B`	N-Spike Counter Policy	2048	0.5	1	0.706	2730.81
`s1.1-7B`	N-Spike Counter Policy	2048	0.5	3	0.688	2986.16
`s1.1-7B`	N-Spike Counter Policy	2048	0.5	5	0.704	3022.09
`s1.1-7B`	Adaptive Self Critique Sampler	2048	–	–	0.742	2614.56
Max Thinking Tokens = 4096						
`s1.1-7B`	Greedy Policy	4096	–	–	0.642	2725.00
`s1.1-7B`	Accumulated Probability Policy	4096	0.1	–	0.726	3841.05
`s1.1-7B`	Accumulated Probability Policy	4096	1	–	0.748	3871.68
`s1.1-7B`	Accumulated Probability Policy	4096	3	–	0.756	4550.30
`s1.1-7B`	Last-Interval Budget Policy	4096	0.01	–	0.758	5255.40
`s1.1-7B`	Last-Interval Budget Policy	4096	0.1	–	0.778	5354.85
`s1.1-7B`	Last-Interval Budget Policy	4096	0.5	–	0.756	5406.34
`s1.1-7B`	N-Spike Counter Policy	4096	0.01	1	0.762	3759.02
`s1.1-7B`	N-Spike Counter Policy	4096	0.01	3	0.754	4304.67
`s1.1-7B`	N-Spike Counter Policy	4096	0.01	5	0.770	4647.54
`s1.1-7B`	N-Spike Counter Policy	4096	0.1	1	0.762	3759.02
`s1.1-7B`	N-Spike Counter Policy	4096	0.1	3	0.772	4374.37
`s1.1-7B`	N-Spike Counter Policy	4096	0.1	5	0.758	5002.43
`s1.1-7B`	N-Spike Counter Policy	4096	0.5	1	0.762	3759.02
`s1.1-7B`	N-Spike Counter Policy	4096	0.5	3	0.756	4670.00
`s1.1-7B`	N-Spike Counter Policy	4096	0.5	5	0.756	5373.00
`s1.1-7B`	Adaptive Self Critique Sampler	4096	–	–	0.792	4244.91
Max Thinking Tokens = 8192						
`s1.1-7B`	Greedy Policy	8192	–	–	0.714	3917.57
`s1.1-7B`	Accumulated Probability Policy	8192	0.1	–	0.774	5077.64
`s1.1-7B`	Accumulated Probability Policy	8192	1	–	0.788	5405.76
`s1.1-7B`	Accumulated Probability Policy	8192	3	–	0.796	6631.65
`s1.1-7B`	Last-Interval Budget Policy	8192	0.01	–	0.794	11033.53
`s1.1-7B`	Last-Interval Budget Policy	8192	0.1	–	0.800	10712.60
`s1.1-7B`	Last-Interval Budget Policy	8192	0.5	–	0.784	10512.50
`s1.1-7B`	N-Spike Counter Policy	8192	0.01	1	0.798	5163.36
`s1.1-7B`	N-Spike Counter Policy	8192	0.01	3	0.800	5817.96
`s1.1-7B`	N-Spike Counter Policy	8192	0.01	5	0.802	6726.14
`s1.1-7B`	N-Spike Counter Policy	8192	0.1	1	0.798	5163.36
`s1.1-7B`	N-Spike Counter Policy	8192	0.1	3	0.806	5975.91
`s1.1-7B`	N-Spike Counter Policy	8192	0.1	5	0.804	8105.31
`s1.1-7B`	N-Spike Counter Policy	8192	0.5	1	0.798	5114.54
`s1.1-7B`	N-Spike Counter Policy	8192	0.5	3	0.804	7010.83
`s1.1-7B`	N-Spike Counter Policy	8192	0.5	5	0.810	9727.63
`s1.1-7B`	Adaptive Self Critique Sampler	8192	–	–	0.812	7330.56

Table 6: Performance of Qwen3 Models on the Maths500 Benchmark. This table compares the Baseline performance of Qwen3-8b and Qwen3-14b models against the 'N-Critique-Sampler' method under various configurations. The primary metrics are Maths500 accuracy and the average token count per problem. The results show that the N-Critique-Sampler method, particularly with N=3 critiques, a spike threshold of 0.25, and a larger context window (Max Thinking Tokens / Max Generation Tokens), achieves the highest accuracy (0.950) while significantly reducing the token count compared to the baseline.

Model	**Method**	**Critiques (N)**	**Context Window**	**Spike Threshold**	**Maths500 Acc.**	**Avg. Tokens**
Qwen3-8b	Baseline	N/A	8k / 16k	N/A	0.924	4683.00
	Baseline	N/A	16k / 32k	N/A	0.934	5312.00
	N-Critique-Sampler	1	8k / 16k	0.10	0.900	2166.10
	N-Critique-Sampler	3	8k / 16k	0.10	0.908	2718.62
	N-Critique-Sampler	5	8k / 16k	0.10	0.922	3108.73
	N-Critique-Sampler	1	8k / 16k	0.25	0.914	2482.83
	N-Critique-Sampler	3	8k / 16k	0.25	0.918	2882.83
	N-Critique-Sampler	5	8k / 16k	0.25	0.922	3220.43
	N-Critique-Sampler	1	16k / 32k	0.10	0.906	2227.50
	N-Critique-Sampler	3	16k / 32k	0.10	0.916	2799.19
	N-Critique-Sampler	5	16k / 32k	0.10	0.928	3370.71
	N-Critique-Sampler	1	16k / 32k	0.25	0.914	2568.46
	N-Critique-Sampler	**3**	**16k / 32k**	**0.25**	**0.944**	**3055.18**
	N-Critique-Sampler	5	16k / 32k	0.25	0.937	3635.18
Qwen3-14b	Baseline	N/A	8k / 16k	N/A	0.933	4286.70
	Baseline	N/A	16k / 32k	N/A	0.940	4732.62
	N-Critique-Sampler	1	8k / 16k	0.25	0.928	2141.40
	N-Critique-Sampler	3	8k / 16k	0.25	0.932	2683.92
	N-Critique-Sampler	5	8k / 16k	0.25	0.930	2883.92
	N-Critique-Sampler	1	16k / 16k	0.25	0.932	2138.05
	N-Critique-Sampler	**3**	**16k / 32k**	**0.25**	**0.950**	**3084.97**
	N-Critique-Sampler	**5**	**16k / 32k**	**0.25**	**0.950**	**3524.97**

Table 7: Performance of Qwen3 Models on the AIME25 Benchmark. This table presents the accuracy and average token consumption for Qwen3 models of varying sizes (4B, 8B, 14B). We compare the standard Baseline generation method against our 'N-Critique-Sampler' approach with an increasing number of critiques (N). The N-Critique-Sampler method generally improves accuracy over the baseline for all model sizes, with performance scaling with the number of critiques. The Qwen3-14B model with N=7 achieves the highest accuracy of 0.722, a notable improvement over its baseline performance of 0.655. All experiments were conducted using a 16k/32k context window (Max Thinking Tokens / Max Generation Tokens) and a spike threshold of 0.25 for the N-Critique-Sampler method.

Model	**Method**	**Critiques (N)**	**AIME Accuracy**	**Average Token Count**
Qwen3-4B	Baseline	N/A	0.578	13923.5
	N-Critique-Sampler	1	0.434	10047.20
	N-Critique-Sampler	3	0.500	12010.20
	N-Critique-Sampler	5	0.588	13451.10
	N-Critique-Sampler	**7**	**0.600**	**14176.20**
Qwen3-8B	Baseline	N/A	0.589 ± 0.056	13793.5 ± 113.77
	N-Critique-Sampler	1	0.500 ± 0.081	12364.1 ± 320.96
	N-Critique-Sampler	3	0.622 ± 0.056	13408.4 ± 199.80
	N-Critique-Sampler	**5**	$\mathbf{0.667 \pm 0.027}$	$\mathbf{13850.0 \pm 462.10}$
	N-Critique-Sampler	7	0.656 ± 0.032	15192.0 ± 483.86
Qwen3-14B	Baseline	N/A	0.655 ± 0.041	13278.0 ± 201.50
	N-Critique-Sampler	1	0.688 ± 0.068	11977.7 ± 208.60
	N-Critique-Sampler	3	0.667 ± 0.072	13099.4 ± 142.80
	N-Critique-Sampler	5	0.711 ± 0.042	13559.0 ± 378.60
	N-Critique-Sampler	**7**	$\mathbf{0.722 \pm 0.031}$	$\mathbf{14064.0 \pm 666.58}$