

BEACON: Bayesian Optimal Stopping for Efficient LLM Sampling

Guangya Wan^{1,*}, Zixin Stephen Xu^{1,*}, Sasa Zorc¹, Manel Baucells¹

Mengxuan Hu², Hao Wang¹, Sheng Li^{1,†}

¹University of Virginia ²Capital One

*Equal contribution †Corresponding author

Abstract

Best-of- N (BoN) sampling improves LLM output quality by generating multiple candidates and selecting the highest-reward response, but fixing N wastes compute, since the budget is paid regardless of whether a strong response has appeared. We introduce BEACON (Bayesian Efficient Adaptive Criterion for Optimal N-stopping), an adaptive sampling method grounded in Bayesian sequential search. BEACON generates responses sequentially, updates posterior beliefs over the reward distribution via conjugate prior updates, and stops when the expected marginal benefit of further sampling no longer justifies its cost. Under a specified reward-distribution model, the resulting stopping rule admits closed-form posterior updates and a precomputable table-lookup index policy, with low inference overhead. On reasoning and alignment benchmarks, BEACON dynamically allocates samples to uncertain queries, matching fixed BoN accuracy to within 0.7 points while using 50–88% fewer samples on average and outperforming prior adaptive sampling methods at comparable or smaller budgets. We further demonstrate BEACON’s utility for post-training data generation and extensions to batch-parallel inference and noisy reward. These results connect Bayesian optimal stopping theory with LLM usage, providing a model-grounded stopping criterion under explicit distributional assumptions and practical efficiency gains for reward-guided LLM sampling.

1 Introduction

Large Language Models (LLMs) are increasingly deployed in settings where both response quality and inference efficiency matter (Wang and Li, 2026; Sui et al., 2025; Yuan et al., 2025b; Yang et al., 2025). A common approach to improve quality is sampling-based selection: generate multiple candidates, score them with a reward model or consistency metric, and return the best (Wang et al., 2022;

Nakano et al., 2021; Snell et al., 2025). Methods such as Best-of- N sampling are now commonly used for complex reasoning, safety alignment, and preference data generation (Ichihara et al., 2025; Yuan et al., 2024). Yet they waste compute by design: the number of generations is fixed in advance, regardless of whether a strong response appears early or the prompt is inherently easy (Wang et al., 2025a; Fu et al., 2024). As models scale, this motivates a principled **economy of sampling**: when to pay for another generation, and when the current best is good enough to stop (Huang et al., 2025a).

Prior work addresses this question by making sampling budgets adaptive, but mostly through indirect proxies for the value of another sample. Consistency-based methods estimate difficulty or confidence from answer agreement (Aggarwal et al., 2023; Wang et al., 2022; Taubenfeld et al., 2025; Wan et al., 2025a), but agreement can be misleading when multiple incorrect responses converge, and is poorly defined for open-ended tasks with many valid answers (Wang et al., 2025a; Fu et al., 2024). Reward-based methods adapt Best-of- N by learning auxiliary stopping models (Cobbe et al., 2021; OpenAI, 2022; Zhang et al., 2024; Damani et al., 2025; Komiya et al., 2026) or allocating samples across prompts using batch-level reward-distribution estimates (Raman et al., 2026). These approaches either require domain-specific training or rely on static reward estimates, leaving the online stop/continue decision to heuristic rules rather than a formal sequential decision criterion with optimality guarantees.

To bridge this gap, we leverage principles from Bayesian optimal stopping (Baucells and Zorc, 2024) and reformulate reward-guided LLM sampling as a sequential search problem. Rather than estimating reward distributions in a separate offline stage, we treat each query’s reward distribution as a latent object to be inferred online: every generated response both reveals information about the

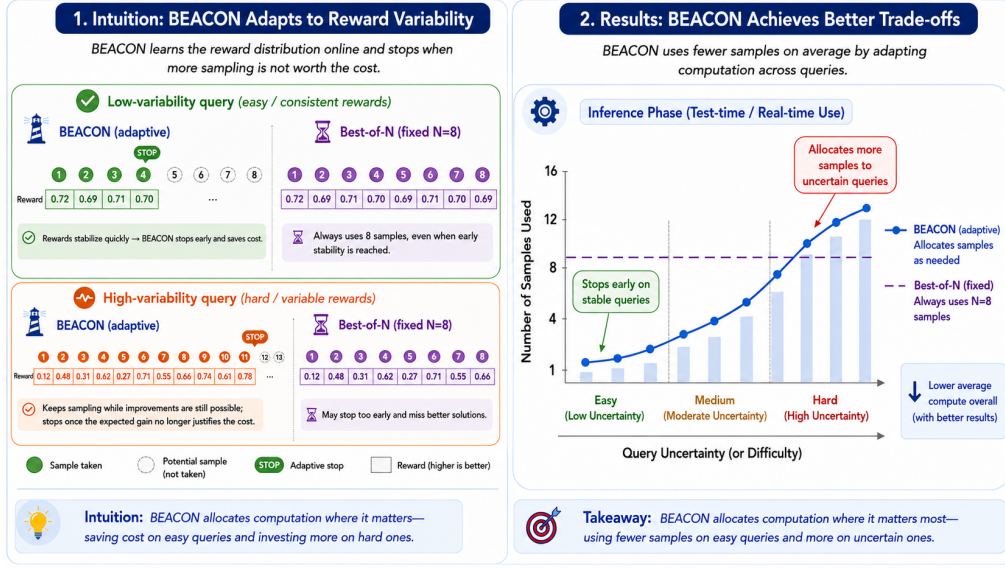


Figure 1: **BEACON adaptive sampling versus fixed Best-of- N sampling.** *Left:* BEACON stops when rewards stabilize and continues when they remain variable, whereas Best-of- N uses a fixed budget for every query. *Right:* schematic of how BEACON’s per-query sample count grows with query uncertainty while Best-of- N stays fixed, yielding lower average inference cost.

distribution of attainable rewards and incurs computational cost (Toth and Oberhauser, 2020). This turns adaptive sampling into a stop/continue decision: continue only when the posterior expected improvement from another sample exceeds its cost, and otherwise return the best response found so far. Because the decision is updated after each observed reward, the policy naturally allocates more samples to queries with uncertain or variable rewards and stops earlier when the reward distribution stabilizes.

Building on this principle, we introduce the **Bayesian Efficient Adaptive Criterion for Optimal N-stopping (BEACON)**, an adaptive sampling method that combines sequential search with online Bayesian learning over reward-model signals. BEACON continues sampling only when the posterior expected marginal benefit of another response exceeds its sampling cost. As a default continuous instantiation, we use a **Normal-Inverse-Gamma (NIG)** model for reward scores, which admits closed-form posterior updates and enables a reusable **Universal Index Policy (UIP)** for table-lookup stopping decisions. After each sample, BEACON updates sufficient statistics, compares an index-based continuation value against a cost threshold, and stops when further sampling is no longer justified. Experiments on reasoning and alignment benchmarks show that BEACON substantially reduces average inference cost while maintaining comparable performance. We further

study extensions to efficient preference-data generation, batch-parallel inference, and robust stopping under noisy or misspecified reward signals. Our main contributions are as follows:

- **Bayesian formulation:** We formulate reward-guided LLM sampling as Bayesian optimal stopping over an unknown, query-specific reward distribution, weighing the posterior expected benefit of an additional sample against its cost.
- **Efficient stopping rule:** From this Bayesian formulation, we derive a UIP-based stopping rule for reward-guided LLM sampling. With a NIG reward model, BEACON supports closed-form posterior updates and a pre-computable index table, making each online stop/continue decision constant time. We clarify the assumptions under which the policy is optimal and provide alternatives when they are violated.
- **Empirical validation:** Across reasoning and alignment benchmarks, BEACON matches fixed BoN accuracy to within 0.7 points while using 50–88% fewer samples on average. We further demonstrate utility for iterative DPO data generation, batch-parallel inference, and noisy rewards.

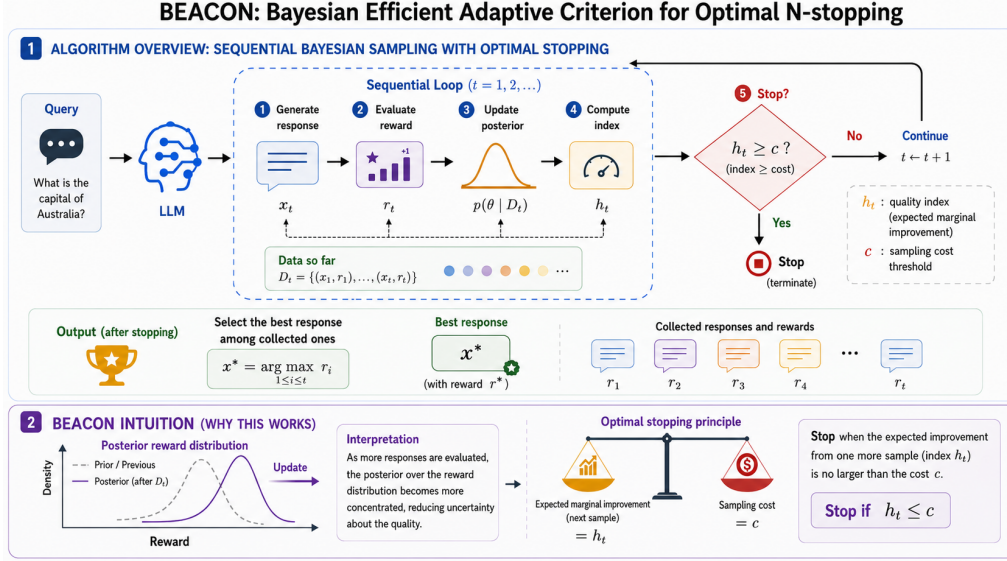


Figure 2: **BEACON framework**. *Top*: An illustration of sequential sampling of LLM responses with reward model evaluation. *Bottom*: Intuition of optimal stopping mechanism, which updates Bayesian posterior beliefs about reward distribution parameters after each sample and stops based on the optimal sampling policy, comparing the index-based threshold to the cost.

2 BEACON: LLM Sampling as Bayesian Sequential Search

We formulate adaptive LLM sampling as a **Bayesian sequential search** problem and derive a **Bayes-optimal stopping rule** under a specified **reward-distribution model**. Specifically, we treat the reward model $R(x, y)$ as providing scalar quality scores: for a fixed prompt x and policy LLM, repeated sampling induces a query-dependent **reward distribution** $f(r | x)$ that marginalizes over both response generation and reward evaluation. Adaptive sampling can therefore be viewed as a finite-horizon **sequential search** problem, where each generated response both reveals information about $f(r | x)$ and incurs a sampling cost c . Classical sequential search formalizes this quality–cost trade-off through threshold-based stopping rules. When $f(r | x)$ is unknown, Bayesian sequential search instead updates posterior beliefs from the observed rewards. The Universal Index Policy (UIP) of Baucells and Zorc (2024) provides an optimal index rule for this posterior-standardized setting, which BEACON instantiates for reward-guided LLM sampling (full background in Appendices E.1–E.4).

2.1 Model Setup

Problem Setting. Given a query x and a policy LLM $\pi_\phi(y | x)$, BEACON sequentially generates responses $y_i \sim \pi_\phi(\cdot | x)$ and evaluates each us-

ing a reward model $R(x, y_i) = r_i$. The induced reward distribution $f(r | x)$ is query-dependent and unknown, since it marginalizes over stochastic response generation and, when applicable, stochastic reward evaluation. We model rewards as i.i.d. draws from $f(r | x)$ —a tractable working assumption that is natural under fixed prompt and decoding parameters, though correlations across samples, for example from shared model biases or reasoning patterns, may arise in practice. This assumption should be viewed as a working model for deriving a tractable stopping rule rather than as a claim that LLM samples are strictly independent; empirically, the resulting policy remains effective under realistic reward-model noise and heterogeneous query difficulty. After k samples, let $\mathbf{r}_k = \{r_1, \dots, r_k\}$ and $z_k = \max_i r_i$ denote the current best reward. Each additional sample incurs cost c (constant marginal cost per generated-and-scored candidate, interpretable as the workload’s average decoding plus reward-evaluation cost), and sampling is capped at horizon n . BEACON seeks a stopping time K that maximizes the expected net utility $\mathbb{E}[z_K - Kc]$.

Bayesian Learning with Conjugate Priors. To make sequential updating tractable, we place $f(r | x)$ in a parametric family $\mathcal{F}$ with a conjugate prior, so that posterior updates compress the observed rewards into finite-dimensional sufficient statistics (Diaconis and Ylvisaker, 1979). As our default continuous instantiation, we adopt a Normal–

Inverse-Gamma (NIG) prior for unknown parameters (μ, σ^2) . After the first k_0 samples, the state needed for the stopping decision can be represented by (z_k, μ_k, σ_k) —the running maximum, posterior mean, and posterior predictive scale—together with the sample index k . These quantities update recursively from the previous state and the new reward r_k , giving $\mathcal{O}(1)$ memory and per-step update cost independent of the total number of samples. Closed-form update equations and the choice of k_0 are deferred to Appendix E.2.

Why Normal–NIG as the Default. The NIG instantiation trades modeling fidelity for tractability: it matches the common Gaussian treatment of continuous reward signals, admits closed-form posterior predictive updates, and supports the computationally efficient UIP index rule of [Baucells and Zorc \(2024\)](#). Reward distributions in our experiments are approximately Gaussian in aggregate (Appendix F.5), though per-query distributions can exhibit skewness or outliers. Section 4 introduces a robust update for such cases within the same stop/continue framework, and Appendix E.6 provides a Bernoulli–Beta instantiation for binary-reward settings.

Bellman Equation. On the sufficient-statistic state (z_k, μ_k, σ_k) , the objective $\mathbb{E}[z_K - K \cdot c]$ admits the standard Bellman recursion

$$V_{n,k}(z_k, \mu_k, \sigma_k; c) = \max\{z_k, \mathbb{E}[V_{n,k+1}(z_{k+1}, \mu_{k+1}, \sigma_{k+1}; c)] - c\}, \quad (1)$$

with corresponding optimal stopping rule:

$$\text{Stop iff } H_{n,k}(z_k, \mu_k, \sigma_k; c) = \mathbb{E}[V_{n,k+1}(z_{k+1}, \mu_{k+1}, \sigma_{k+1}; c)] - z_k \leq c. \quad (2)$$

where $H_{n,k}$ represents the expected marginal gain from continued sampling.

Solving this recursion is intractable for general distributions. The next section derives an efficient solution via the UIP, which reduces each stopping decision to a constant-time index lookup.

2.2 Optimal Sampling Policy and Computation

We now derive BEACON’s Bayes-optimal stopping rule under the NIG reward model by applying the Universal Index Policy of [Baucells and Zorc \(2024\)](#) to the Bayesian sequential search formulation above.

Definition 1. For $k_0 \leq k < n$, the h -index function $h_{n,k} : \mathbb{R} \rightarrow (0, \infty)$ maps each standardized best reward $\hat{z} \in \mathbb{R}$ to the unique value $c > 0$ that solves the condition where the expected marginal gain from continuing equals the sampling cost.

Theorem 1 (Bayes-Optimal Policy under the NIG Model). *Under i.i.d. reward sampling and the Normal–Inverse-Gamma reward model, after k_0 bootstrap samples, the Bayes-optimal policy for the induced sequential search objective continues at step $k \geq k_0$ iff*

$$h_{n,k}\left(\frac{z_k - \mu_k}{\sigma_k}\right) > \frac{c}{\sigma_k}. \quad (3)$$

Thus,

$$K = \min\{k \geq k_0 : h_{n,k}(\hat{z}_k) \leq c/\sigma_k\} \wedge n$$

maximizes $\mathbb{E}[z_K - Kc]$ under the specified Bayesian reward model.

Proof sketch. The argument has three steps (full proof in Appendix E.7). **(i)** Under the NIG prior, (z_k, μ_k, σ_k) together with k form sufficient statistics for the Bellman state (Appendix E.3), so the optimal policy depends on the history only through this triple. **(ii)** Standardizing by $\hat{z}_k = (z_k - \mu_k)/\sigma_k$ reduces the problem to a canonical location–scale-normalized sequential search instance. **(iii)** For this canonical problem, the Universal Index Policy of [Baucells and Zorc \(2024\)](#) establishes a strictly decreasing index function $h_{n,k}(\cdot)$ for which myopic continuation $h_{n,k}(\hat{z}_k) > c/\sigma_k$ is Bellman-optimal; unscaling recovers (3). Operationally, BEACON standardizes the current best reward, looks up $h_{n,k}(\hat{z}_k)$, and stops when it falls below the cost-adjusted threshold c/σ_k .

Sensitivity Analysis. The optimal stopping time K exhibits intuitive dependencies on key problem parameters. When sampling cost c increases, exploration is discouraged and the algorithm stops earlier. If the current best reward z_k substantially exceeds the posterior mean μ_k (large $\hat{z}_k$), the framework recognizes an exceptionally high-quality sample has likely been found and stops earlier. Conversely, greater posterior uncertainty (larger σ_k) encourages continued sampling: under BEACON’s Jeffreys prior, μ_k reduces to the sample mean, so $z_k \geq \mu_k$ holds throughout any run and $\hat{z}_k \geq 0$; in this regime, larger σ_k both shrinks the nonnegative $\hat{z}_k$ and lowers the cost-adjusted threshold c/σ_k , jointly widening the continuation margin. Larger

remaining horizon $n - k$ similarly raises $h_{n,k}$, making the algorithm more patient (see Appendix E.8 for proofs and formal statements).

h-Table Precomputation and Complexity. Although Theorem 1 characterizes the optimal stopping rule, evaluating it directly would require solving a Bellman recursion at every step. BEACON sidesteps this by tabulating the h-index function $h_{n,k}(\hat{z})$ once, offline, so that each in-loop stopping decision reduces to a table lookup. By Definition 1, each table entry is the unique normalized sampling cost at which $H_{n,k}(\hat{z}, 0, 1; c) = c$, and the table is built by backward induction over k via the recursion in Appendix E.5. The appendix also specifies the standardized grid, interpolation rule, and numerical safeguards used to make the h-table reproducible. Two properties make this overhead negligible in deployment: (i) the prior is task-agnostic, so the same table serves *every* query at horizon n and the precompute cost is amortized across the entire workload; (ii) deployments using several horizons $\{n_1, \dots, n_J\}$ pay only for distinct horizons, not for queries.

Algorithm Implementation. Algorithm 1 summarizes the BEACON procedure. After initializing Jeffreys’ non-informative priors and loading the precomputed h -table, we draw $k_0 = 3$ bootstrap samples to establish valid posterior parameters. The adaptive loop then alternates between (i) a constant-time stopping check via standardized score $\hat{z}_k = (z_k - \mu_k)/\sigma_k$ and table lookup of $h_{n,k}(\hat{z}_k)$, and (ii) generating a new sample and updating sufficient statistics, with the robust update (Section 4) applied optionally for skewed reward distributions.

3 BEACON Results on Inference Scaling

We evaluate BEACON as a drop-in replacement for Best-of-N at inference time across reasoning and alignment benchmarks. Our evaluation addresses three questions: (i) does BEACON achieve a superior quality-efficiency trade-off relative to fixed BoN and existing adaptive baselines? (ii) is the framework robust across reward model choices, including under degraded reward signals? and (iii) does its adaptive allocation reflect principled response to query characteristics?

3.1 Experimental Setup

Hyperparameter Configuration. BEACON uses a fixed prior, maximum horizon, and sampling

cost. *Prior parameters:* We use Jeffreys’ non-informative prior $(\alpha_0, \nu_0, \mu_0, \beta_0) = (-0.5, 0, 0, 0)$ for task-agnostic deployment, requiring $k_0 = 3$ initial samples for well-defined posteriors (Appendix E.2). *Maximum horizon:* We set $n = 32$ to match standard Best-of- N configurations (Singhi et al., 2025); BEACON’s average sample count is well below this horizon. *Sampling cost:* The stopping rule operates on a fixed per-query cost c_x . Because reward scales vary across models and tasks, we set $c_x = \lambda \sigma_{k_0}(x)$ after the bootstrap samples, where $\sigma_{k_0}(x)$ is the posterior predictive scale and λ is a dimensionless cost ratio. This parameterization expresses c_x in units of reward scale; all theoretical results and stopping decisions depend only on c_x . Higher λ favors efficiency, while lower λ favors quality. Unless otherwise specified, we use the same default λ across models and tasks, with calibration details in Appendix C.

Baselines, Models, and Evaluations. We focus our comparisons on training-free methods and standard Best-of- N (BoN), as these represent widely used inference-time strategies for improving LLM outputs without additional policy optimization. Specifically, we include one-shot **Chain-of-Thought** (CoT) prompting with $K=1$; **Self-Consistency** (SC) (Wang et al., 2022), which performs majority voting over n sampled reasoning paths; **Adaptive-Consistency** (AS) (Aggarwal et al., 2023), a model-agnostic early-stopping method that fits a Dirichlet–multinomial posterior to answer agreement patterns; **RASC** (Wan et al., 2025a), a heuristic adaptive sampler that uses CoT quality scores to guide sampling; standard Best-of- N (BoN), which selects the highest reward-scored candidate from n attempts; and adaptive Best-of- N (**ABoN**) (Raman et al., 2026), a two-stage adaptive allocation method that first spends an exploration budget to estimate per-prompt reward distributions and then uses non-parametric kernel density estimation to allocate the remaining samples. Policy models include LLaMA-3.2-3B (Grattafiori et al., 2024), Qwen2.5-7B-Instruct (Qwen Team, 2025), and Gemini-2.5-Flash (Gemini Team, Google, 2025); reward models include Llama-3.1-Nemotron-70B-Reward and Skywork-Llama-3.1-8B. We use CoT (Wei et al., 2022) prompting with model-specific templates. Evaluation covers: (1) *Reasoning Tasks*, where we report Pass@1 on three mathematical benchmarks, MATH-500 (Lightman et al., 2023), AIME24, and AIME25 (Mathematical Association of America,

Table 1: Comparison of BEACON with baseline sampling methods across different models and tasks. Reasoning accuracy is the unweighted mean over the three benchmarks of MATH-500, AIME24, and AIME25. BEACON achieves a superior trade-off between Accuracy/Win Rate/Scaled Reward $\bar{z}_K$ and efficiency (Avg Sample $\bar{K}$). Upward arrows ($\uparrow$) indicate percentage improvement in accuracy or win rate over CoT baselines; downward arrows ($\downarrow$) show percentage reduction in samples compared to the maximum BoN sample size (32). SC and AS entries are omitted for AlpacaEval 2.0 because majority voting over answers is not defined for open-ended generation. Additional runs for statistical significance reported in Table 5.

Model	Method	Reasoning Tasks (Avg. MATH 500/AIME)			Alignment Task (AlpacaEval 2.0)		
		Accuracy $\uparrow$ (%)	Samples $\downarrow$ ($\bar{K}$)	Reward $\uparrow$ ($\bar{z}_K$)	Win Rate $\uparrow$ (%)	Samples $\downarrow$ ($\bar{K}$)	Reward $\uparrow$ ($\bar{z}_K$)
LLaMA-3.2-3B	Direct CoT	20.0	1.0	-1.15	17.6	1.0	-1.19
	SC	28.5 $\uparrow 42.5\%$	16.0 $\downarrow 50.0\%$	-0.35	-	-	-
	RASC	27.8 $\uparrow 39.0\%$	5.2 $\downarrow 83.8\%$	-0.30	22.0 $\uparrow 25.0\%$	7.0 $\downarrow 78.1\%$	-1.23
	AS	27.8 $\uparrow 39.0\%$	5.6 $\downarrow 82.5\%$	-0.31	-	-	-
	BoN	33.4 $\uparrow 67.0\%$	32.0	1.75	27.5 $\uparrow 56.3\%$	32.0	1.94
	ABoN	33.0 $\uparrow 65.0\%$	25.5 $\downarrow 20.3\%$	1.72	26.5 $\uparrow 50.6\%$	26.2 $\downarrow 18.1\%$	1.89
	BEACON	33.0 $\uparrow 65.0\%$	15.8 $\downarrow 50.6\%$	1.68	27.9 $\uparrow 58.5\%$	14.5 $\downarrow 54.7\%$	1.85
Qwen2.5-7B	Direct CoT	56.9	1.0	-1.66	24.8	1.0	-0.94
	SC	66.2 $\uparrow 16.3\%$	16.0 $\downarrow 50.0\%$	-0.48	-	-	-
	RASC	65.5 $\uparrow 15.1\%$	4.3 $\downarrow 86.6\%$	-0.38	27.5 $\uparrow 10.9\%$	4.0 $\downarrow 87.5\%$	-1.27
	AS	65.3 $\uparrow 14.8\%$	4.6 $\downarrow 85.6\%$	-0.39	-	-	-
	BoN	73.0 $\uparrow 28.3\%$	32.0	2.45	39.6 $\uparrow 59.7\%$	32.0	2.22
	ABoN	72.0 $\uparrow 26.5\%$	25.0 $\downarrow 21.9\%$	2.40	38.0 $\uparrow 53.2\%$	25.8 $\downarrow 19.4\%$	2.20
	BEACON	73.0 $\uparrow 28.3\%$	6.5 $\downarrow 79.7\%$	2.36	38.9 $\uparrow 56.9\%$	7.8 $\downarrow 75.6\%$	2.15
Gemini-2.5-Flash	Direct CoT	80.1	1.0	-0.99	46.7	1.0	-0.62
	SC	82.8 $\uparrow 3.4\%$	16.0 $\downarrow 50.0\%$	-0.34	-	-	-
	RASC	84.4 $\uparrow 5.4\%$	3.5 $\downarrow 89.1\%$	-0.20	48.7 $\uparrow 4.3\%$	3.5 $\downarrow 89.1\%$	-0.64
	AS	84.4 $\uparrow 5.4\%$	3.8 $\downarrow 88.1\%$	-0.21	-	-	-
	BoN	86.0 $\uparrow 7.4\%$	32.0	1.46	53.6 $\uparrow 14.8\%$	32.0	1.83
	ABoN	85.8 $\uparrow 7.1\%$	16.0 $\downarrow 50.0\%$	1.44	53.4 $\uparrow 14.3\%$	16.0 $\downarrow 50.0\%$	1.82
	BEACON	86.4 $\uparrow 7.9\%$	5.0 $\downarrow 84.4\%$	1.41	53.4 $\uparrow 14.3\%$	4.0 $\downarrow 87.5\%$	1.79

Note: SC = Self-Consistency; RASC = Reasoning-Aware SC; AS = Adaptive Sampling; BoN = Best-of- N ; ABoN = Adaptive BoN.

2024); (2) an *Alignment Task* using AlpacaEval 2.0 (Li et al., 2023), where model responses to user instructions are compared against GPT-4o (OpenAI, 2024). For each method, we report task performance together with the average number of generated candidates $\bar{K}$ and the dataset-average normalized selected reward $\bar{z}_K$, and rewards are standardized within each context.

3.2 Main Results

Optimal Performance-Efficiency Tradeoff. Table 1 reports BEACON’s headline performance across models, reward models, and tasks: BEACON matches fixed BoN₃₂ accuracy to within 0.7 points on every model–task pair while using far fewer samples, showing that adaptive stopping captures most of the quality benefit at a fraction of the cost. Figure 3 compares both methods at matched compute budgets: we vary BEACON’s cost parameter c to control its average sample count and compare against BoN at the same N . The result shows a consistent Pareto improvement across much of the compute range: (i) *at matched performance, BEACON requires substantially fewer samples than BoN*; (ii) *at matched sample budget, BEACON*

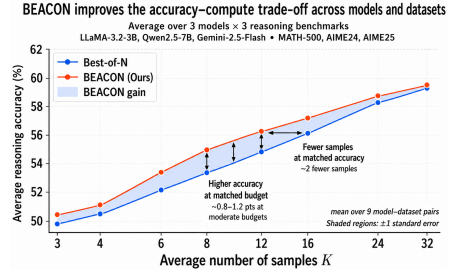


Figure 3: **Aggregated quality–efficiency.** BEACON Pareto-wins BoN across 3 models $\times$ 3 reasoning benchmarks with fewer samples at matched accuracy with higher accuracy/matched budget.

achieves substantially higher accuracy. The advantage is largest at moderate budgets ($K = 6\text{--}14$) and converges at high K where both methods saturate. This dominance reflects BEACON’s dynamic per-query allocation, investing additional samples in queries with high reward variance while terminating early when reward distributions stabilize, unlike BoN’s uniform sampling that allocates identical resources regardless of query characteristics.

Reward Model Sensitivity. BEACON’s stopping behavior is governed by reward-signal uncertainty: when posterior variance is low, additional samples

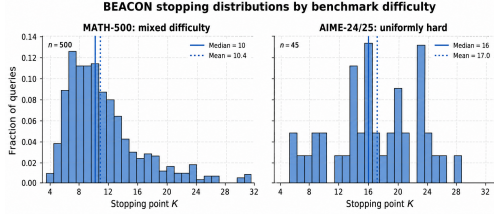


Figure 4: **Stopping-point distributions of BEACON on MATH and AIME.** BEACON terminates early on easier queries (left-skewed distribution) and allocates more samples to harder queries (right-shifted distribution), indicating adaptive compute allocation based on query difficulty.

Table 2: BEACON across reward models on math reasoning (Gemini 2.5 policy). Accuracy here is reported on MATH-500 only, so values are not directly comparable to the three-benchmark average in Table 1. At matched compute, BEACON outperforms BoN under both RMs, with $\bar{K}$ adapting to reward signal characteristics.

RM	Method	$\bar{K}$	Acc.
Nemotron-70B	BoN-N=32	32.0	86.0
	BoN-matched	5.0	83.5
	BEACON	5.0	85.3
Skywork-8B	BoN-N=32	32.0	83.3
	BoN-matched	18.0	80.1
	BEACON	17.8	81.8

are unlikely to improve selection and the algorithm stops; when it remains high, sampling continues. This produces systematic differences across reward models (Table 2): $\bar{K}$ is substantially smaller under the more decisive Nemotron-70B than under the noisier Skywork-8B, with BEACON automatically allocating more compute to the noisier signal. At matched compute, BEACON consistently outperforms BoN under both reward models, indicating that the efficiency gain comes from adaptive allocation rather than any property of a particular reward model. A controlled noise stress test (Appendix F.4) confirms this: as injected Gaussian noise increases, BEACON’s accuracy degrades at a rate comparable to fixed BoN while $\bar{K}$ grows to compensate. BEACON thus inherits the reward model’s quality ceiling without amplifying its errors, preserving its efficiency advantage across reward-signal conditions.

Adaptive Allocation Tracks Query Difficulty. BEACON’s efficiency advantage stems from non-uniform per-query allocation. Figure 4 shows that on MATH-500 (mixed difficulty) the stopping dis-

tribution is concentrated at small K , while on AIME-24/25 (uniformly hard) it shifts substantially to the right, with most queries sampling well beyond the MATH-500 median. This dataset-level shift indicates that BEACON’s allocation calibrates to query difficulty without manual tuning. The same pattern holds across model capability (Table 1): $\bar{K}$ decreases monotonically from LLaMA-3.2-3B to Qwen2.5-7B to Gemini-2.5-Flash, reflecting that more capable models produce reward distributions that stabilize earlier and trigger BEACON’s stopping criterion sooner.

4 Extending BEACON Beyond Inference

Having established BEACON as an efficient drop-in replacement for Best-of-N at inference time, we now demonstrate that the framework extends to three common deployment settings: (i) post-training data generation for iterative DPO, where BEACON reduces sampling cost as the policy improves; (ii) batch-parallel sampling for latency-sensitive inference, where BEACON inherits the parallelism benefits of BoN at matched memory budgets; and (iii) noisy reward distributions encountered in practice, where the framework admits alternative instantiations and a robust update.

Efficiency in Post-Training Data Generation.

We also show BEACON’s utility in efficient data generation for Iterative Direct Preference Optimization (DPO) (Rafailov et al., 2023), a widely-used post-training paradigm that enhances LLM consistency on challenging questions (Xiong et al., 2025; Pang et al., 2024). We evaluate BEACON against the standard Best-of-N (BoN) approach—the conventional method for generating preference pairs in iterative training pipelines (Li et al., 2024; Yuan et al., 2025a); comprehensive experimental details are provided in Appendix D.2. Figure 5 illustrates performance across seven DPO iterations: BEACON achieves *comparable accuracy* to BoN while progressively reducing the average required samples ($\bar{K}$, shown on the secondary y-axis). Notably, $\bar{K}$ decreases over iterations as the policy becomes more consistent and aligned, since stable reward distributions trigger BEACON’s stopping criterion earlier. This demonstrates a compounding benefit: BEACON not only reduces per-iteration sampling cost but does so increasingly as training progresses, making it especially well-suited for iterative post-training pipelines where data generation dominates compute cost.

Iterative DPO: BEACON matches BoN accuracy with fewer samples

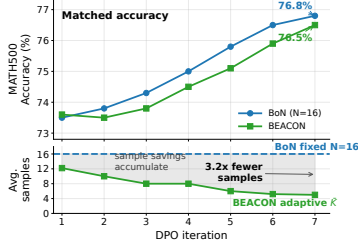


Figure 5: **Iterative DPO on MATH500 with Qwen2.5-7B.** BEACON matches BoN ($N=16$) accuracy across all seven iterations (76.5% vs. 76.8% at iteration 7) while $\bar{K}$ decreases from ~ 12 to ~ 5 , yielding a growing efficiency gap.

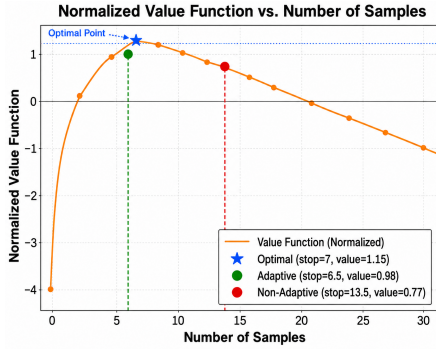


Figure 6: **Robust update keeps stopping near the optimum under reward skew.** Normalized value of the optimization objective $\mathbb{E}[z_K - Kc]$ vs. stopping time K for an illustrative skewed query.

Handling Diverse Reward Distributions. While we instantiate BEACON with a Normal-NIG conjugate model for tractability and empirical fit (Fig. 11), the framework does not require strict normality. The optimality of Theorem 1 is conditional on model well-specification, and per-query reward distributions occasionally exhibit negative skewness (Fig. 11, middle panel) under which the NIG posterior is itself misspecified. To address tail misspecification within the same framework, we introduce an optional *robust updating mechanism* that preserves the informative right tail by replacing values below the 1% posterior-predictive quantile with the current posterior mean before the update. It is a one-sided winsorization that leaves the maximum statistic z_k unchanged, modifies only the posterior parameters (μ_k, σ_k) used for the stopping decision, and affects at most 1% of samples in expectation under the model. Under correct specification, the induced bias is negligible by construction; under skewed misspecification, it substantially reduces stopping-decision error which is a standard trade-

Table 3: Batch-parallel BEACON performance (LLaMA 3.2). Memory overhead is relative to $b = 1$.

Method	Spd. (vs Seq.)	Mem. Ovhd.	Acc. Δ	Avg. Samp.
$b = 1$ (Seq.)	1.00×	Base	0.0%	15.8
$b = 2$	1.65×	+22%	+0.2%	16.1
$b = 4$	2.31×	+48%	+0.8%	17.2
$b = 8$	2.85×	+85%	+1.4%	19.4

off in robust statistics. Figure 6 shows that this mechanism consistently drives BEACON’s stopping points closer to the optimum on skewed distributions while leaving behavior on well-behaved distributions essentially unchanged. For settings where rewards are inherently non-Gaussian (e.g., binary correctness signals), the framework admits a Bernoulli–Beta instantiation with its own optimal-stopping policy; See Appendix E.6 and E.9 for details.

Batch-Parallel Sampling for Reduced Latency.

BEACON naturally supports batch-parallel sampling: it generates b candidates in parallel, scores them, updates the posterior with the batch rewards, and applies the same stopping rule after each batch. Larger b reduces wall-clock latency but makes stopping decisions coarser, since BEACON can terminate only after observing a full batch. Table 3 shows this trade-off: throughput improves with b , while average sample count and memory overhead increase moderately. The same precomputed UIP table is reused for all batch sizes with the same horizon, so batching requires no additional precomputation per batch.

5 Related Work

Parallel Reasoning and Adaptive Inference Al-

location. Parallel inference-time scaling improves LLM outputs by generating multiple candidates and selecting the best among them. Consensus-based methods combine candidates through majority voting, weighted voting, or confidence-based aggregation (Wang et al., 2022; Fu et al., 2025; Wan et al., 2025a; Wang et al., 2025a; Wan et al., 2026), while Best-of- N (BoN) sampling uses an external verifier or reward model (RM) to select the highest-scoring response (Cobbe et al., 2021; Ichihara et al., 2025; Zhang et al., 2024; Ankner et al., 2024). With RM, BoN tends to be more reliable on alignment tasks (Zhou et al., 2025) and on tasks where aggregation is not well-defined (Chen et al., 2024). To improve efficiency, recent adaptive

methods address this by allocating computation based on answer agreement, verifier confidence, or reward-model feedback (Sui et al., 2025; Manvi et al., 2024; Huang et al., 2025b; Wan et al., 2025a; Wang et al., 2025b; Wan et al., 2025b; Aggarwal et al., 2023). The most closely related work, adaptive Best-of- N (ABoN), uses a two-stage heuristic that first estimates per-prompt reward distributions with an exploration budget and then allocates samples via non-parametric density estimation (Raman et al., 2026). BEACON instead formulates adaptivity as a Bayesian sequential decision problem, treating each observed reward as evidence that updates a posterior over the reward distribution and applying optimal stopping theory to decide when to stop sampling.

Optimal Stopping and Bayesian Inference. BEACON builds on decision-theoretic and Bayesian approaches to sequential sampling (Shvartsman et al., 2015; Zheng et al., 2020). Multi-armed bandits study exploration–exploitation trade-offs (Lattimore and Szepesvári, 2020; Slivkins, 2019), including best-arm identification (Audibert and Bubeck, 2010; Gabillon et al., 2012) and extreme bandits for discovering high-value outcomes (Carpentier and Valko, 2014; Lopez et al., 2021). Optimal stopping theory formalizes when to terminate sequential observation (Ferguson, 2012), with foundations in sequential search (Weitzman, 1979) and recent extensions to unknown distributions (Bau-cells and Zorc, 2024). Bayesian methods, including budgeted bandits and Bayesian optimization, use posterior beliefs for sample-efficient exploration of costly objectives (Xia et al., 2016; Snoek et al., 2012; Shahriari et al., 2015). BEACON follows this line but uses conjugate-prior Bayesian updating to obtain closed-form posterior updates and a tractable adaptive stopping rule for improving LLM sample efficiency under moderate sample.

6 Conclusion

We introduced BEACON, a Bayesian optimal-stopping framework for efficient reward-guided LLM sampling. Rather than fixing a Best-of- N budget in advance, BEACON updates posterior beliefs about a query-specific reward distribution and continues sampling only when the expected improvement justifies its cost. With a Normal–Inverse-Gamma instantiation, this yields closed-form updates and constant-time online decisions via a precomputed Universal Index Policy. Empiri-

cally, BEACON preserves much of the quality gain of fixed BoN while substantially reducing average sample count, with extensions to preference-data generation, batch-parallel inference, and noisy reward signals. These results support adaptive, query-level allocation as a more efficient alternative to uniform inference-time scaling.

Limitations

We highlight four limitations of BEACON. First, the optimality of the stopping rule is conditional on the modeling assumptions: rewards are treated as i.i.d. draws from a Normal–Inverse-Gamma family, which holds approximately in aggregate but can be violated at the per-query level through skewness, outliers, or correlations between samples sharing reasoning templates or latent decoding modes. The robust update (Section 4) addresses skewness, but a fuller treatment of sample correlations remains open.

Second, BEACON’s stopping behavior inherits the quality of the reward model. Our experiments demonstrate robustness across reward models of varying decisiveness and under controlled noise injection (Appendix F.4), but systematic biases in the reward signal could propagate into stopping decisions in ways that injected noise does not capture. Integrating reward-model uncertainty quantification and multi-dimensional reward signals is a natural next step.

Third, our evaluation beyond inference-time selection remains preliminary. The iterative DPO experiment in Section 4 uses a single model and benchmark, and should therefore be viewed as an initial demonstration rather than a comprehensive post-training study. In the current implementation, the query-level cost is fixed after bootstrap as $c_x = \lambda \sigma_{k_0}(x)$, where λ is a deployment-level cost ratio. This provides a simple scale-adaptive quality–efficiency knob, but future systems could adjust λ based on latency targets, server load, query importance, or training stage.

Fourth, the iterative DPO experiment uses a correctness-aware reward R_{adapt} (Appendix D.2.5) that incorporates ground-truth final-answer correctness. Both BoN and BEACON receive this identical signal, so the comparison isolates the sampling policy, but such a signal is unavailable at inference time and the absolute accuracies in Figure 5 should not be read as inference-time results.

Acknowledgments

This work is supported in part by the U.S. Office of Naval Research (ONR) Award under Grant Number N00014-24-1-2668, the National Science Foundation (NSF) under Grants IIS-2316306 and CNS-2330215, and the National Institutes of Health (NIH) under Grant R01EB293388. The content is solely the responsibility of the authors and does not necessarily represent the official views of the ONR, NSF, and NIH.

References

- Pranjal Aggarwal, Aman Madaan, Yiming Yang, and Mausam. 2023. [Let’s sample step by step: Adaptive-consistency for efficient reasoning and coding with LLMs](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12375–12396, Singapore. Association for Computational Linguistics.
- Zachary Ankner, Mansheej Paul, Brandon Cui, Jonathan Daniel Chang, and Prithviraj Ammanabrolu. 2024. [Critique-out-loud reward models](#). In *Pluralistic Alignment Workshop at NeurIPS 2024*.
- Jean-Yves Audibert and Sébastien Bubeck. 2010. Best arm identification in multi-armed bandits. In *Proceedings of the 23rd Conference on Learning Theory (COLT)*, pages 41–53.
- Manel Baucells and Saša Zorc. 2024. Search in the dark: The case with recall and gaussian learning. *Operations Research*.
- Alexandra Carpentier and Michal Valko. 2014. Extreme bandits. In *Advances in Neural Information Processing Systems*, volume 27.
- Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, and Denny Zhou. 2024. [Universal self-consistency for large language models](#). In *ICML 2024 Workshop on In-Context Learning*.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. [Training verifiers to solve math word problems](#). *Preprint*, arXiv:2110.14168.
- Mehul Damani, Idan Shenfeld, Andi Peng, Andreea Bobu, and Jacob Andreas. 2025. [Learning how hard to think: Input-adaptive allocation of LM computation](#). In *The Thirteenth International Conference on Learning Representations*.
- Persi Diaconis and Donald Ylvisaker. 1979. Conjugate priors for exponential families. *The Annals of statistics*, pages 269–281.
- Thomas S. Ferguson. 2012. [Optimal stopping and applications](#). Lecture notes, UCLA Department of Statistics Papers.
- Yichao Fu, Junda Chen, Siqi Zhu, Zheyu Fu, Zhongdongming Dai, Aurick Qiao, and Hao Zhang. 2024. [Efficiently serving llm reasoning programs with certindex](#). *arXiv preprint arXiv:2412.20993*.
- Yichao Fu, Xuewei Wang, Yuandong Tian, and Jiawei Zhao. 2025. Deep think with confidence. *arXiv preprint arXiv:2508.15260*.
- Victor Gabillon, Mohammad Ghavamzadeh, and Alessandro Lazaric. 2012. Best arm identification: A unified approach to fixed budget and fixed confidence.

- In *Advances in Neural Information Processing Systems*, volume 25.
- Gemini Team, Google. 2025. [Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities](#). *arXiv preprint arXiv:2507.06261*.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. [The llama 3 herd of models](#). *Preprint*, arXiv:2407.21783.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.
- Audrey Huang, Adam Block, Qinghua Liu, Nan Jiang, Akshay Krishnamurthy, and Dylan J Foster. 2025a. [Is best-of-n the best of them? coverage, scaling, and optimality in inference-time alignment](#). In *Forty-second International Conference on Machine Learning*.
- Chengsong Huang, Langlin Huang, Jixuan Leng, Jiacheng Liu, and Jiaxin Huang. 2025b. [Efficient test-time scaling via self-calibration](#). *Preprint*, arXiv:2503.00031.
- Yuki Ichihara, Yuu Jinnai, Tetsuro Morimura, Kenshi Abe, Kaito Ariu, Mitsuki Sakamoto, and Eiji Uchibe. 2025. [Evaluation of best-of-n sampling strategies for language model alignment](#). *Transactions on Machine Learning Research*.
- Junpei Komiyama, Daisuke Oba, and Masafumi Oyama. 2026. [Best-of-infinity: Asymptotic performance of test-time LLM ensembling](#). In *The Fourteenth International Conference on Learning Representations*.
- Tor Lattimore and Csaba Szepesvári. 2020. *Bandit Algorithms*. Cambridge University Press.
- Jiaxiang Li, Siliang Zeng, Hoi To Wai, Chenliang Li, Alfredo Garcia, and Mingyi Hong. 2024. [Getting more juice out of the SFT data: Reward learning from human demonstration improves SFT for LLM alignment](#). In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.
- Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. AlpacaEval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval.
- Hunter Lightman, Vinit Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. [Let's verify step by step](#). *Preprint*, arXiv:2305.20050.
- Romain Lopez, Inderjit S Dhillon, and Michael I Jordan. 2021. Learning from extreme bandit feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8732–8740.
- Rohin Manvi, Anikait Singh, and Stefano Ermon. 2024. [Adaptive inference-time compute: LLMs can predict if they can do better, even mid-generation](#). *Preprint*, arXiv:2410.02725.
- Mathematical Association of America. 2024. [American invitational mathematics examination - AIME](#). Mathematics Competition. Accessed: May 2025.
- Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2021. [Webgpt: Browser-assisted question-answering with human feedback](#). *Preprint*, arXiv:2112.09332.
- OpenAI. 2022. [Measuring Goodhart's Law](#).
- OpenAI. 2024. GPT-4o system card. *arXiv preprint arXiv:2410.21276*.
- Richard Yuanzhe Pang, Weizhe Yuan, He He, Kyunghyun Cho, Sainbayar Sukhbaatar, and Jason E Weston. 2024. [Iterative reasoning preference optimization](#). In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.
- Qwen Team. 2025. [Qwen2.5 technical report](#). *Preprint*, arXiv:2412.15115.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. [Direct preference optimization: Your language model is secretly a reward model](#). *ArXiv*, abs/2305.18290.
- Vinod Raman, Hilal Asi, and Satyen Kale. 2026. [Adabon: Adaptive best-of-n alignment](#). *Preprint*, arXiv:2505.12050.
- Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. 2015. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175.
- Michael Shvartsman, Vaibhav Srivastava, and Jonathan D. Cohen. 2015. [A theory of decision making under dynamic context](#). In *Advances in Neural Information Processing Systems*, volume 28.
- Nishad Singhi, Hritik Bansal, Arian Hosseini, Aditya Grover, Kai-Wei Chang, Marcus Rohrbach, and Anna Rohrbach. 2025. [When to solve, when to verify: Compute-optimal problem solving and generative verification for llm reasoning](#). *Preprint*, arXiv:2504.01005.
- Aleksandrs Slivkins. 2019. Introduction to multi-armed bandits. *Foundations and Trends in Machine Learning*, 12(1-2):1–286.

- Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2025. [Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning](#). In *The Thirteenth International Conference on Learning Representations*.
- Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, volume 25.
- Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, Hanjie Chen, and Xia Hu. 2025. [Stop overthinking: A survey on efficient reasoning for large language models](#). *Transactions on Machine Learning Research*.
- Amir Taubenfeld, Tom Sheffer, Eran Ofek, Amir Feder, Ariel Goldstein, Zorik Gekhman, and Gal Yona. 2025. [Confidence improves self-consistency in llms](#). *Preprint*, arXiv:2502.06233.
- Csaba Toth and Harald Oberhauser. 2020. [Bayesian learning from sequential data using Gaussian processes with signature covariances](#). In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9548–9560. PMLR.
- Guangya Wan, Mingyang Ling, Xiaoqi Ren, Rujun Han, Sheng Li, and Zizhao Zhang. 2026. [COMPASS: Enhancing agent long-horizon reasoning with evolving context](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3360–3380, San Diego, California, United States. Association for Computational Linguistics.
- Guangya Wan, Yuqi Wu, Jie Chen, and Sheng Li. 2025a. [Reasoning aware self-consistency: Leveraging reasoning paths for efficient LLM sampling](#). In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3613–3635, Albuquerque, New Mexico. Association for Computational Linguistics.
- Guangya Wan, Yuqi Wu, Hao Wang, Shengming Zhao, Jie Chen, and Sheng Li. 2025b. [Derailer-rerailer: Adaptive verification for efficient and reliable language model reasoning](#). *Preprint*, arXiv:2408.13940.
- Pinqiao Wang and Sheng Li. 2026. [Multi-agent debate: A unified agentic framework for tabular anomaly detection](#). *Preprint*, arXiv:2602.14251.
- Xinglin Wang, Shaoxiong Feng, Yiwei Li, Peiwen Yuan, Yueqi Zhang, Chuyi Tan, Boyuan Pan, Yao Hu, and Kan Li. 2025a. [Make every penny count: Difficulty-adaptive self-consistency for cost-efficient reasoning](#). In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 6904–6917, Albuquerque, New Mexico. Association for Computational Linguistics.
- Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. [Self-consistency improves chain of thought reasoning in language models](#). *Preprint*, arXiv:2203.11171.
- Yiming Wang, Pei Zhang, Siyuan Huang, Baosong Yang, Zhuosheng Zhang, Fei Huang, and Rui Wang. 2025b. [Sampling-efficient test-time scaling: Self-estimating the best-of-n sampling in early decoding](#). *Preprint*, arXiv:2503.01422.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022. [Chain of thought prompting elicits reasoning in large language models](#). In *Advances in Neural Information Processing Systems*.
- Martin L. Weitzman. 1979. [Optimal search for the best alternative](#). *Econometrica*, 47(3):641–654.
- Yingce Xia, Tao Qin, Weidong Ma, Nenghai Yu, and Tie-Yan Liu. 2016. Budgeted bandit problems with continuous random costs. In *Asian Conference on Machine Learning*, pages 317–332. PMLR.
- Wei Xiong, Chengshuai Shi, Jiaming Shen, Aviv Rosenberg, Zhen Qin, Daniele Calandriello, Misha Khalman, Rishabh Joshi, Bilal Piot, Mohammad Saleh, Chi Jin, Tong Zhang, and Tianqi Liu. 2025. [Building math agents with multi-turn iterative preference learning](#). In *The Thirteenth International Conference on Learning Representations*.
- Huibo Yang, Mengxuan Hu, Amoreena Most, W. Anthony Hawkins, Brian Murray, Susan E. Smith, Sheng Li, and Andrea Sikora. 2025. [Evaluating accuracy and reproducibility of large language model performance on critical care assessments in pharmacy education](#). *Frontiers in Artificial Intelligence*, 7:1514896.
- Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2025a. [Self-rewarding language models](#). *Preprint*, arXiv:2401.10020.
- Xiu Yuan, Tongzhou Mu, Stone Tao, Yunhao Fang, Mengke Zhang, and Hao Su. 2025b. [Policy decorator: Model-agnostic online refinement for large policy model](#). In *The Thirteenth International Conference on Learning Representations*.
- Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2024. [Scaling relationship on learning mathematical reasoning with large language models](#).
- Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. 2024. [Generative verifiers: Reward modeling as next-token prediction](#). In *The 4th Workshop on Mathematical Reasoning and AI at NeurIPS’24*.

Sue Zheng, David Hayden, Jason Pacheco, and John W. Fisher III. 2020. [Sequential bayesian experimental design with variable cost structure](#). In *Advances in Neural Information Processing Systems*, volume 33.

Enyu Zhou, Guodong Zheng, Binghai Wang, Zhiheng Xi, Shihan Dou, Rong Bao, Wei Shen, Limao Xiong, Jessica Fan, Yurong Mou, Rui Zheng, Tao Gui, Qi Zhang, and Xuanjing Huang. 2025. [RMB: Comprehensively benchmarking reward models in LLM alignment](#). In *The Thirteenth International Conference on Learning Representations*.

A Use of AI Assistants

AI assistants were used for LaTeX formatting, copy-editing, and proofreading of author-written text. The research questions, theoretical development, experimental design, implementation, analysis, and all claims in this paper are the authors’ own. The authors have verified the full content of the paper and take responsibility for it.

B Experimental Setup and Details

B.1 Main Experiment Setup

Code is available at https://github.com/wan19990901/Bayesian_Adaptive_Sampling/.

B.1.1 Policy Models and Reward Models

For our policy (generator) models, we evaluated a range of architectures and sizes to ensure comprehensive assessment of BEACON’s performance:

- **LLaMA-3.2-3B-Instruct**: Accessed and run locally on a GPU node equipped with 2 NVIDIA A100 GPUs.
- **Qwen2.5-7B-Instruct**: Inference conducted via DeepInfra’s API¹.
- **Gemini-2.5-Flash**: Inference conducted via Google’s API².

For our reward models (RMs), we utilized:

- **NVIDIA Llama-3.1-Nemotron-70B-Reward**: Accessed via NVIDIA’s API³ services for response verification.
- **Skywork-Llama-3.1-8B-Reward**: Accessed and run locally on a GPU node equipped with 2 NVIDIA A100 GPUs.

¹<https://deepinfra.com/>

²<https://aistudio.google.com/api-keys>

³https://build.nvidia.com/nvidia/llama-3_1-nemotron-70b-reward

B.1.2 Datasets and Evaluation Benchmarks

We evaluated BEACON on diverse reasoning and alignment tasks:

Reasoning Tasks: The reasoning evaluation focused on mathematical problem-solving, reporting average accuracy (Pass@1) across the following benchmarks. For answer extraction and checking mathematical equivalence, we utilized the expression matching tool from the Math-Verify repository⁴ to ensure consistent and robust evaluation.

- **MATH500**: We used all 500 problems from the MATH500 subset (Lightman et al., 2023) of the MATH dataset (Hendrycks et al., 2021).
- **AIME 2024**: All 30 problems from the American Invitational Mathematics Examination 2024 were used (Mathematical Association of America, 2024).
- **AIME 2025**: All 30 problems from the American Invitational Mathematics Examination 2025 were used.

Alignment Task:

- **AlpacaEval 2.0**: We used the full set of 805 prompts from AlpacaEval 2.0 (Li et al., 2023). The evaluation of response quality, comparing model outputs against a reference (e.g., GPT-4), was conducted using OpenAI’s API for the automated evaluation protocol provided by AlpacaEval. The primary metric reported is the win rate.

B.1.3 Prompting Strategy

For all reasoning tasks, we employed a standard one-shot Chain-of-Thought (CoT) prompting strategy. The models were instructed to act as math assistants and provide step-by-step solutions. Model-specific instruction templates were used where appropriate, but the core reasoning prompt structure was as follows:

⁴<https://github.com/huggingface/Math-Verify>

Standard CoT Reasoning Prompt

You are a math assistant. Solve problems step by step with clear reasoning.

Format:

1. Start with “Let me solve this step by step:”
2. Numbered steps with explanations
3. End with answer in box: $\boxed{42}$

Rules:

- Answer must be integer or simplified fraction
- Use exact box format: $\boxed{42}$
- No text after box
- For fractions: $\boxed{\frac{3}{4}}$
- For negatives: $\boxed{-5}$

Example:

Let me solve this step by step:

1. First step
2. Second step
- $\vdots$
- N) Final step

$\boxed{42}$

[Problem Statement Here]

This standardized prompt ensures consistency in how tasks are presented to the policy models. For AlpacaEval, prompts are used as provided by the benchmark.

C Practical Guidelines for Setting the Sampling Cost

The sampling cost c controls the quality–efficiency trade-off in BEACON. It represents the marginal cost of generating and scoring one additional candidate in the objective

$$\mathbb{E}[z_K - Kc].$$

Larger values of c encourage earlier stopping, while smaller values allow more exploration. The stopping rule uses c only through the normalized ratio c/σ_k , where σ_k is the posterior predictive scale. Thus, the absolute numerical value of c should be interpreted relative to the reward scale rather than as a task-specific constant.

In our experiments, we use a simple scale-adaptive parameterization. After the bootstrap samples, we fix a query-level cost

$$c_x = \lambda \sigma_{k_0}(x),$$

where $\sigma_{k_0}(x)$ is the posterior predictive scale estimated from the initial k_0 samples and λ is a dimensionless cost ratio. Once c_x is set, it is held fixed for the remainder of that query, and BEACON applies the same stopping rule as in Section 2. This parameterization simply expresses sampling cost in units of the query’s initial reward uncertainty, making the rule insensitive to arbitrary reward-scale differences across models and reward functions.

The role of λ is to expose a single deployment-level quality–efficiency knob: smaller values prioritize quality, while larger values prioritize efficiency. Importantly, λ does not change the form of the stopping rule and is not a learned stopping model. In all main experiments, we use a fixed default value of λ across models and tasks unless otherwise specified, and vary it only when tracing quality–efficiency Pareto curves.

For practical use, λ can be selected from a small default range, such as $\{0.01, 0.02, 0.05\}$, depending on the desired compute budget. A lightweight validation sweep may be used to choose a preferred operating point, but it does not require running full BoN-32 on the evaluation set. In deployment, one can instead select λ by targeting an average sample budget or latency constraint on a small held-out calibration set. This is analogous to choosing the operating point of any anytime inference method, rather than training or fitting an additional model.

Across our experiments, values in the range $\lambda = 0.01$ – 0.05 provided a stable trade-off, substantially reducing average sample count while preserving most of the quality gains of fixed BoN. Smaller or harder models may benefit from lower λ , while stronger models or easier workloads can often use larger λ because their reward distributions stabilize earlier.

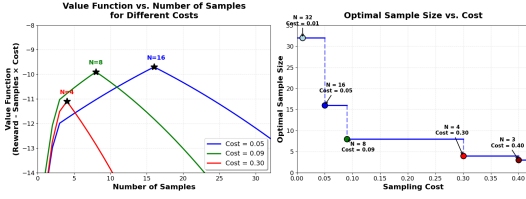


Figure 7: Impact of the sampling cost (c) on the value optimization and the optimal sample size. Higher c results in earlier stopping to reach Bayesian optimality.

D Practical Implementation Details

D.1 Decoding Hyperparameters and BEACON Configuration

Besides cost, the BEACON framework operated with the following core settings for the main experiments:

Decoding Parameters: For all policy model inferences, unless specified otherwise by the API provider’s defaults for instructed models, we used consistent decoding parameters:

- Temperature: 0.7
- Maximum new tokens: 2048

BEACON Framework Configuration:

- Minimum initial samples (k_0): 3. This is required to properly establish the posterior Normal-Inverse-Gamma distribution using Jeffreys’ non-informative prior ($\mu_0 = 0, \nu_0 = 0, \alpha_0 = -0.5, \beta_0 = 0$).
- Maximum sampling horizon ($T_{\max}$ or n): 32.
- H-index function $h_{n,k}(\hat{z}_k)$: Pre-computed based on the methodology in Baucells and Zorc (Baucells and Zorc, 2024) using the specified priors.

D.2 Iterative DPO with BEACON: Experimental Setup

D.2.1 Data Foundation for DPO Preference Generation

The prompts used for generating responses, which subsequently form preference pairs for Direct Preference Optimization (DPO), are drawn from the extensive training set of the MATH dataset (7,500 problems) (Hendrycks et al., 2021). The MATH dataset, comprising problems from American mathematics competitions like AMC 10, AMC 12, and AIME, is highly suitable due to its provision of

full step-by-step solutions, which is beneficial for training models on complex derivations. Its effectiveness in enhancing mathematical reasoning is well-established. While preference data is generated from these MATH prompts, the DPO model’s performance is evaluated on the MATH500 benchmark, as presented in Figure 5 of the main text.

D.2.2 Base Reward Model

To score generated responses, we use Skywork-Llama-3.1-8B as the base reward model, following the main experiments. This model assigns a scalar score R_{learned} to each generated response. For DPO preference-pair construction, R_{learned} is combined with ground-truth final-answer correctness to form the shared correctness-aware reward R_{adapt} defined in Appendix D.2.5.

D.2.3 Preference Pair Generation Strategies for DPO

Preference pairs for each DPO iteration are generated using either fixed-budget BoN or BEACON. To ensure a fair comparison, both methods use the same correctness-aware reward signal R_{adapt} , defined in Appendix D.2.5, for candidate scoring, ranking, and filtering. Thus, the methods differ only in their sampling policy: BoN always generates a fixed number of candidates, while BEACON adaptively determines the number of candidates to generate.

Best-of-N (BoN) Strategy. For each prompt, BoN samples a fixed $N = 16$ candidate responses from the current policy model, Qwen2.5-7B-Instruct. Each response is scored using R_{adapt} . The response with the highest R_{adapt} score is selected as the preferred response y_w , and the response with the lowest R_{adapt} score is selected as the dispreferred response y_l .

BEACON Strategy. For each prompt, BEACON adaptively determines the number of candidate responses K to generate. It begins with $k_0 = 3$ bootstrap samples and can sample up to a maximum horizon of $N_{\max} = 16$. BEACON uses the same R_{adapt} signal for posterior updates, stopping decisions, and final ranking among the collected candidates. From the K collected responses, the response with the highest R_{adapt} score is selected as y_w , and the response with the lowest R_{adapt} score is selected as y_l .

Filtering Rule. For both methods, a prompt is discarded only if all collected candidates have iden-

tical R_{adapt} scores, in which case no meaningful preference pair can be formed. We apply the same filtering rule to both methods.

D.2.4 Iterative Direct Preference Optimization Training

The policy model, Qwen2.5-7B-instruct, is fine-tuned using Direct Preference Optimization (DPO) (Rafailov et al., 2023) on the preference pairs (y_w, y_l) generated by either the BoN or BEACON strategy. Key hyperparameters for DPO training include a learning rate of 5×10^{-7} , a global batch size of 128, and a maximum sequence length of 4096. In each DPO iteration, the policy model is trained for 2 epochs on the newly generated preference dataset. The entire cycle of preference pair generation (using either BoN or BEACON) followed by DPO fine-tuning of the policy model is repeated for a total of 7 iterations.

D.2.5 Correctness-Aware Reward Signal

Because the DPO experiment is conducted on mathematical reasoning tasks with known ground-truth answers, we use a correctness-aware reward signal for preference-pair construction. To avoid giving BEACON an oracle advantage, this same reward signal is used by both BoN and BEACON.

Let R_{learned} be the continuous score produced by the fixed reward model for a generated response. We define the correctness-aware reward R_{adapt} as follows:

- If the response contains the correct final answer in a `\boxed{ }` environment:
 - If $R_{\text{learned}} > 0$, then $R_{\text{adapt}} = 2R_{\text{learned}}$.
 - If $R_{\text{learned}} \leq 0$, then $R_{\text{adapt}} = R_{\text{learned}}/2$.
- If the response contains an incorrect final answer in a `\boxed{ }` environment:
 - If $R_{\text{learned}} > 0$, then $R_{\text{adapt}} = R_{\text{learned}}/2$.
 - If $R_{\text{learned}} \leq 0$, then $R_{\text{adapt}} = 2R_{\text{learned}}$.
- If the response does not contain a final answer in a `\boxed{ }` environment, the reward remains unchanged: $R_{\text{adapt}} = R_{\text{learned}}$.

Both BoN and BEACON use R_{adapt} for preference-pair ranking and filtering. BEACON additionally uses the same R_{adapt} values for its Bayesian posterior updates and stopping decisions. Therefore, the comparison isolates the effect of adaptive stopping rather than differences in reward access.

E Additional Theoretical Analysis and Proofs

This section provides supplementary theoretical background and proofs for the BEACON framework.

E.1 Classical Sequential Search Problem

In the classical sequential search problem (Weitzman, 1979), a decision maker (DM) faces an infinite sequence of independent offers $\{x_1, x_2, \dots\}$ drawn from a known distribution function F with density f . Sampling incurs a constant cost $c > 0$, and the DM may stop at any time, accepting the highest offer observed so far (see Figure 8 for illustration). Since the distribution F is fully known, the predictive distribution remains unchanged throughout the process.

The dynamic programming recursion for the value function after k samples is:

$$V(z) = \max \left\{ z, \int_{-\infty}^{\infty} V(\max\{z, y\}) dF(y) - c \right\}$$

where z is the current best observed offer. The associated gain from one more search is $H(z) = \int_z^{\infty} (y - z) dF(y)$, representing the expected improvement conditional on continuing.

A fundamental property is the **reservation price property**: there exists a threshold r^* such that the optimal policy is to stop if and only if $z \geq r^*$. This reservation price solves:

$$c = H(r^*) = \int_{r^*}^{\infty} (y - r^*) dF(y).$$

The DM thus compares the sampling cost with the expected marginal benefit; once the best offer reaches r^* , the expected gain no longer justifies continued search.

This threshold-based policy has important implications: (1) r^* depends only on F and c , not on the number of samples or their sequence; (2) as c increases, r^* decreases, leading to earlier stopping; and (3) distributions with heavier right tails yield higher reservation prices, reflecting greater potential benefits from continued search.

E.2 Priors and Minimal Sample Size for Bayesian Updating

We employ a conjugate Normal-Inverse-Gamma prior for the unknown mean μ and variance σ^2 of the reward distribution, following (Baucells and

Algorithm 1 BEACON: Bayesian Efficient Adaptive Criterion for Optimal N-stopping

Input: Query x , policy LLM $\pi_\phi(y|x)$, reward model $R(x, y)$, cost c , max samples n , grid size G

Output: Best response y^* and its reward r^*

// Step 1: Initialize prior and h-index table

1 $(\alpha_0, \nu_0, \beta_0, \mu_0) \leftarrow (-0.5, 0, 0, 0)$

// Non-informative prior

2 $h\text{-table} \leftarrow \text{PrecomputeTable}(n, G, \alpha_0, \nu_0)$

// Pre-compute as in §E.5

// Step 2: Generate initial samples and compute baseline parameters

3 Generate $\{y_1, y_2, y_3\} \sim \pi_\phi(y|x)$ and compute $r_i \leftarrow R(x, y_i)$ for $i \in \{1, 2, 3\}$

4 $(\alpha, \nu, \mu, \beta) \leftarrow \text{UpdateHyperParams}(r_1, r_2, r_3, \alpha_0, \nu_0, \beta_0, \mu_0)$

5 $z \leftarrow \max\{r_1, r_2, r_3\}, \sigma \leftarrow \sqrt{\frac{(\nu+1)\beta}{\nu\alpha}}, k \leftarrow 3$

// Step 3: Adaptive sampling loop

6 **while** $k < n$ **do**

7 $\hat{z} \leftarrow (z - \mu)/\sigma, h \leftarrow \text{LookupHIndex}(h\text{-table}, k, \hat{z})$

8 **if** $h \leq c/\sigma$ **then**

9 **break**

// Apply UIP as in (3)

10 **end**

11 $k \leftarrow k + 1$, generate $y_k \sim \pi_\phi(y|x)$ and compute $r_k \leftarrow R(x, y_k)$

// Optional robust low-tail filtering for skewed rewards; omit for the exact NIG-UIP policy

12 $q_{0.01} \leftarrow F_{2\alpha_{k-1}}^{-1}(0.01 \mid \mu, \sigma)$

13 $\tilde{r}_k \leftarrow \begin{cases} \mu & \text{if } r_k < q_{0.01} \\ r_k & \text{otherwise} \end{cases}$

14 $(z, \mu, \sigma) \leftarrow \text{UpdateStats}(z, \mu, \sigma, \tilde{r}_k, k)$

// Updated by (5)

15 **end**

16 $i^* \leftarrow \arg \max_{i \in \{1, \dots, k\}} r_i$ **return** y_{i^*}, r_{i^*}

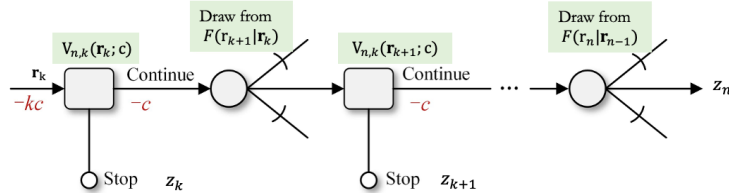


Figure 8: Sequential search problem illustration: At each step, the decision-maker observes a reward r_k and decides whether to stop with the current best reward z_k or continue sampling, incurring cost c .

Zorc, 2024). In this framework, the precision $1/\sigma^2$ follows a Gamma distribution with parameters (α_0, β_0) , and conditional on precision, μ follows a Normal distribution with mean μ_0 and variance σ^2/ν_0 . This structure enables analytical posterior updates with hyperparameters $(\alpha_k, \nu_k, \beta_k, \mu_k)$. Starting with prior $\text{NIG}(\mu_0, \nu_0, \alpha_0, \beta_0)$ and after observing k samples, the posterior parameters be-

come:

$$\begin{aligned} \alpha_k &= \alpha_0 + \frac{k}{2}, \nu_k = \nu_0 + k, \mu_k = \frac{\nu_0 \mu_0 + k \bar{r}_k}{\nu_0 + k}, \\ \beta_k &= \beta_0 + \frac{\sum_{i=1}^k (r_i - \bar{r}_k)^2}{2} + \frac{k \nu_0 (\bar{r}_k - \mu_0)^2}{2(\nu_0 + k)}, \end{aligned} \quad (4)$$

where $\bar{r}_k$ is the sample mean. The posterior predictive distribution follows a Student-t distribution with $2\alpha_k$ degrees of freedom, mean μ_k , and scale parameter $\sigma_k = \sqrt{(\nu_k + 1)\beta_k/(\nu_k \alpha_k)}$.

The minimal sample size k_0 depends on the chosen prior hyperparameters. For BEACON, we adopt Jeffreys' non-informative prior

Table 4: Prior configurations and their associated minimum sample size k_0 required for a well-defined posterior predictive distribution.

Prior Configuration	k_0
$2\alpha_0 > 1, \nu_0, \beta_0 > 0$	0
$2\alpha_0 \in (0, 1], \nu_0 > 0, \beta_0 \geq 0$	1
$2\alpha_0 > 1, \nu_0 > 0, \beta_0 = 0$	1
$\alpha_0, \beta_0 > 0, \nu_0 = 0$	1
$\alpha_0 > 0, \nu_0 = \beta_0 = 0$	2
$2\alpha_0 \in (-1, 0], \nu_0, \beta_0 \geq 0$	2
$2\alpha_0 = -1, \nu_0, \beta_0 \geq 0$	3

$(\alpha_0, \nu_0, \beta_0) = (-1/2, 0, 0)$, which requires $k_0 = 3$ initial observations before a valid posterior predictive distribution emerges. More informative priors require fewer initial samples, as summarized in Table 4.

E.3 Sufficient Statistics for the Search Problem using Bayesian Updating

The Bayesian sequential search model can be significantly simplified through sufficient statistics that encapsulate all relevant information for optimal stopping decisions (Baucells and Zorc, 2024).

Lemma 1. *For any sampling stage k , where $k_0 \leq k \leq n$, the triple (z_k, μ_k, σ_k) together with k constitute sufficient statistics for the value-to-go function $V_{n,k}(\mathbf{r}_k; c)$.*

Proof: Under the NIG conjugate model, the posterior over (μ, σ^2) given $\mathbf{r}_k$ depends on the data only through the four hyperparameters $(\alpha_k, \nu_k, \mu_k, \beta_k)$ in eq. (4). Two of these— $\alpha_k = \alpha_0 + k/2$ and $\nu_k = \nu_0 + k$ —are deterministic functions of k alone, while μ_k and β_k depend on the data. The posterior predictive scale $\sigma_k = \sqrt{(\nu_k + 1)\beta_k/(\nu_k\alpha_k)}$ is a one-to-one function of β_k given k , so (μ_k, σ_k) together with k encode the same information as $(\alpha_k, \nu_k, \mu_k, \beta_k)$. The Bellman recursion $V_{n,k} = \max\{z_k, \mathbb{E}[V_{n,k+1} \mid \text{state}] - c\}$ depends on $\mathbf{r}_k$ only through (i) the running maximum z_k , used in the stopping branch, and (ii) the predictive distribution over r_{k+1} , which is fully determined by (μ_k, σ_k, k) . Hence $(z_k, \mu_k, \sigma_k, k)$ are sufficient. $\square$

This lemma establishes that rather than tracking the complete observation history $\mathbf{r}_k = \{r_1, \dots, r_k\}$, we only need to maintain three key statistics:

- $z_k = \max\{r_1, \dots, r_k\}$: The highest observed reward so far
- μ_k : The posterior mean of the reward distribution

- σ_k : The scale parameter derived from the posterior distribution

These statistics update efficiently according to the following formula when observing a new reward r_{k+1} :

$$z_{k+1} = \max\{z_k, r_{k+1}\}, \quad (5)$$

$$\mu_{k+1} = \mu_k + \frac{r_{k+1} - \mu_k}{\nu_0 + k + 1}, \quad (6)$$

$$\sigma_{k+1} = \sqrt{\frac{1 - \frac{1}{(\nu_0 + k + 1)^2}}{2\alpha_0 + k + 1}} \times \sqrt{(2\alpha_0 + k)\sigma_k^2 + (r_{k+1} - \mu_k)^2}. \quad (7)$$

The posterior predictive distribution follows a Student-t distribution with $2\alpha_k$ degrees of freedom. This statistical sufficiency substantially simplifies both theoretical analysis and practical implementation of BEACON by reducing the problem’s dimensionality from tracking k individual rewards to monitoring just three informative statistics, making the algorithm computationally efficient and mathematically tractable.

E.4 Details of Universal Index Policy

Here we elaborate on the Universal Index Policy.

Definition 2. *For $k_0 \leq k < n$, the index function $h_{n,k} : \mathbb{R} \rightarrow (0, \infty)$ maps each standardized best reward $\hat{z} \in \mathbb{R}$ to the unique value $c > 0$ that solves $H_{n,k}(\hat{z}, 0, 1; c) = c$.*

This definition captures how the h-index represents the exact cost threshold where the expected value of continuing equals that of stopping. The strength of this approach is its reusability—once computed, these indices apply across different queries with similar statistical properties.

Theorem 2. (Baucells and Zorc, 2024) *After $k \geq k_0$ observations, with standardized best reward $\hat{z}_k = (z_k - \mu_k)/\sigma_k$, it is optimal to stop and accept z_k if $c \geq \sigma_k h_{n,k}(\hat{z}_k)$, and continue searching otherwise. Equivalently, stop if and only if $\hat{z}_k \geq h_{n,k}^{-1}(c/\sigma_k)$.*

Theorem 3. (Baucells and Zorc, 2024) *The h-index function $h_{n,k}(\hat{z})$ for $k_0 \leq k < n$ is strictly decreasing, convex, and satisfies $\lim_{\hat{z} \rightarrow \infty} h_{n,k}(\hat{z}) = 0$.*

This theorem establishes BEACON’s core stopping criterion—after normalizing the current best reward, we stop sampling when the cost-adjusted index threshold is reached. This occurs when the current best reward is sufficiently high relative to our uncertainty about the reward distribution, considering the sampling cost.

E.5 Computation of the h-index

Computing the h-index function $h_{n,k}(\hat{z})$ requires solving recursive equations based on the Bellman equation and expected marginal gain function $H_{n,k}$ from §2.1. As derived in (Baucells and Zorc, 2024), we need to solve:

$$H_{n,k}(\hat{z}, 0, 1; c) = H_{k+1,k}(\hat{z}, 0, 1; c) + \int \sigma_u \cdot g(u) dF_{2\alpha_k}(u), \quad (8)$$

where

$$g(u) = \max \left\{ 0, H_{n,k+1} \left(\frac{z_u - \mu_u}{\sigma_u}, 0, 1; \frac{c}{\sigma_u} \right) - \frac{c}{\sigma_u} \right\}$$

with boundary condition $H_{n,n} = 0$, where $z_u = \max\{\hat{z}, u\}$, $\mu_u = u/(\nu_0 + k + 1)$, $\sigma_u = \Lambda_{k+1} \sqrt{2\alpha_0 + k + u^2}$, $\Lambda_{k+1} = \sqrt{\frac{1 - 1/(\nu_0 + k + 1)^2}{2\alpha_0 + k + 1}}$, and $F_{2\alpha_k}$ is the CDF of the Student-t distribution with $2\alpha_k = 2\alpha_0 + k$ degrees of freedom—the posterior predictive distribution at stage k , whose df reflects sequential Bayesian updating. For Jeffreys’ prior ($\alpha_0 = -1/2$) this gives $\text{df} = k - 1$, which is well-defined for $k \geq k_0 = 3$ ($\text{df} \geq 2$). The form of Λ_{k+1} follows from substituting the canonical state ($\sigma_k = 1, \mu_k = 0$) into the posterior update (5); an equivalent form is $\Lambda_{k+1}^2 = (\nu_0 + k)(\nu_0 + k + 2)/[(\nu_0 + k + 1)^2(2\alpha_0 + k + 1)]$. The $-c/\sigma_u$ term inside the max is the normalized continuation cost, required by the gross-gain convention adopted in eq. (2) where H denotes expected marginal gain *before* netting the cost; omitting it would inflate continuation values and shift the precomputed h-index thresholds.

In our implementation, we pre-compute the h-index function $h_{n,k}(\hat{z})$ for each time horizon n using Jeffreys’ non-informative prior ($\alpha_0 = -0.5, \nu_0 = 0$). Algorithm 2 summarizes the construction. Our implementation follows the public h-index construction code⁵, which provides grid construction, full h-table solution, and interpolated lookup routines accelerated with Numba.

⁵https://github.com/wan19990901/Bayesian_Adaptive_Sampling/blob/main/src/adapt_sample/h_index_construction.py

Algorithm 2 Offline construction of the BEACON h-table

Input: Horizon n ; bootstrap start $k_0 = 3$; grid resolution G ; bound Z ; grid-shape parameters (ρ, η) ; prior parameters (α_0, ν_0) .

Output: Lookup table $\mathcal{H} = \{h_{n,k}(\hat{z}_g) : k = k_0, \dots, n-1, \hat{z}_g \in \mathcal{Z}\}$.

Construct normalized cost and state grids $(\mathcal{C}, \mathcal{Z}) \leftarrow \text{BUILDGRIDS}(G, \rho, Z, \eta)$, where the implementation uses defaults $\rho = 0.85$, $Z = 30$, and $\eta = 0.75$. Set the auxiliary value table $\mathcal{M}[n, \hat{z}_g, c_i] \leftarrow 0$ for all $\hat{z}_g \in \mathcal{Z}$ and $c_i \in \mathcal{C}$.

for $k \leftarrow n-1, n-2, \dots, k_0$ **do**

foreach $(\hat{z}_g, c_i) \in \mathcal{Z} \times \mathcal{C}$ **do**

 Initialize with the one-step myopic gain $H_{k+1,k}(\hat{z}_g, 0, 1; c_i)$ and, if $k < n-1$, add the grid-quadrature approximation to the continuation integral in (8). When the next-stage arguments $((z_u - \mu_u)/\sigma_u, c_i/\sigma_u)$ fall between grid points, interpolate $\mathcal{M}[k+1, \cdot, \cdot]$ over both $\hat{z}$ and c . Store the result as $\mathcal{M}[k, \hat{z}_g, c_i] \leftarrow H_{n,k}(\hat{z}_g, 0, 1; c_i)$.

foreach $\hat{z}_g \in \mathcal{Z}$ **do**

 Interpolate over $\mathcal{C}$ to find the unique crossing c^* satisfying $\mathcal{M}[k, \hat{z}_g, c^*] = c^*$. Store $\mathcal{H}[k, \hat{z}_g] \leftarrow \max\{c^*, 0\}$.

return $\mathcal{H}, \mathcal{Z}$

The state grid is built in standardized reward units rather than raw reward units, so the same h-table can be reused across queries once the state is normalized by $\hat{z}_k = (z_k - \mu_k)/\sigma_k$. The reference implementation uses $c_j = G\rho^j$ for the cost grid, with the final boundary point set to $c_{G+1} = 0$, and $\hat{z}_j = Z\{(1-\eta)q_j + \eta q_j^3\}$ for the state grid, where $q_j = (2j - G - 1)/(G - 1)$. The resolution G is an offline accuracy–cost parameter for the precomputed table; the reference implementation treats it as a user-specified input rather than a query-level hyperparameter, and we use $G = 100$ in all reported experiments. At runtime, BEACON clips $\hat{z}_k$ to the tabulated interval and applies linear interpolation between adjacent $\hat{z}$ grid points for the fixed pair (n, k) . Values outside the grid are assigned the nearest boundary value; this is numerically stable because $h_{n,k}(\hat{z})$ is decreasing and approaches zero in the high- $\hat{z}$ tail by Theorem 3.

For numerical stability, the offline solver uses double precision, bounded Student- t grid quadrature, interpolation only within the tabulated cost

and state ranges, and boundary clipping when evaluating out-of-range values. After table construction, we check that entries are nonnegative up to numerical tolerance and weakly decreasing in $\hat{z}$ for each k , matching Theorem 3. These diagnostics catch grid or quadrature errors before the table is used for inference. During inference, BEACON therefore evaluates the stopping condition through constant-time interpolation on this pre-computed h-table, avoiding dynamic-programming calculations at runtime.

E.6 Binary-Reward Variant: Bernoulli–Beta Learning

Model. Consider the sequential generation process in Section 2.1 with binary rewards $r_i \in \{0, 1\}$ and unknown success probability $\theta \in (0, 1)$. Conditional on θ , the draws are i.i.d. Bernoulli(θ). Sampling incurs a per-draw cost $c > 0$ and the horizon is at most n draws. Let $\mathbf{r}_k = (r_1, \dots, r_k)$ and $z_k = \max\{r_1, \dots, r_k\} \in \{0, 1\}$ be the best observed reward by stage k . Adopt the conjugate prior $\theta \sim \text{Beta}(a_0, b_0)$ with $a_0, b_0 > 0$. After k draws with $s_k = \sum_{i=1}^k r_i$ successes and $f_k = k - s_k$ failures, the posterior is

$$\theta \mid \mathbf{r}_k \sim \text{Beta}(a_k, b_k), \quad a_k = a_0 + s_k, \quad b_k = b_0 + f_k,$$

and the posterior-predictive probability of success on the next draw is

$$q_k = \Pr(r_{k+1} = 1 \mid \mathbf{r}_k) = a_k / (a_k + b_k).$$

The sufficient statistics are (z_k, a_k, b_k) . Upon observing r_{k+1} ,

$$z_{k+1} = \max\{z_k, r_{k+1}\}, \quad (9)$$

$$a_{k+1} = a_k + r_{k+1}, \quad b_{k+1} = b_k + 1 - r_{k+1}. \quad (10)$$

Dynamic program. If $z_k = 1$ the process is absorbing and the decision maker stops with value 1. When $z_k = 0$, let $V_{n,k}(0, a_k, b_k; c)$ denote the optimal value (with at most n draws total, at stage k). Then

$$V_{n,k}(0, a_k, b_k; c) = \max\{0, v_k(c)\}, \quad (11)$$

$$V_{n,n}(0, a_n, b_n; c) = 0, \quad V_{n,k}(1, a_k, b_k; c) = 1, \quad (12)$$

where $v_k(c) = q_k + (1 - q_k) V_{n,k+1}(0, a_k, b_k + 1; c) - c$, and $q_k = a_k / (a_k + b_k)$.

It is convenient to rewrite the recursion in remaining-draws form. Let $R_t(a, b; c)$ be the optimal value with $t \in \{0, 1, \dots\}$ draws remaining

and no success yet. With $R_0(\cdot) \equiv 0$,

$$R_t(a, b; c) = \max\{0, q(a, b) + (1 - q(a, b)) R_{t-1}(a, b + 1; c) - c\}, \quad (13)$$

where $q(a, b) = a / (a + b)$. The connection to (11) is $V_{n,k}(0, a_k, b_k; c) = R_{n-k}(a_k, b_k; c)$.

Reservation cost and optimal policy. For each stage, there is a unique cost threshold at which the decision maker is indifferent between stopping and taking one more draw.

Lemma 2 (Basic properties of R_t). *For fixed (t, a, b) with $t \geq 1$ and $a, b > 0$, the map $c \mapsto R_t(a, b; c)$ is continuous and nonincreasing on $[0, \infty)$. Moreover $R_t(a, b; 0) > 0$ and $R_t(a, b; 1) = 0$; hence $\{c > 0 : R_t(a, b; c) > 0\} \subset (0, 1)$.*

Proof: Induct on t . The base $t = 0$ is trivial since $R_0 \equiv 0$. Suppose the claim holds for $t - 1$. The inner term

$$\psi_t(c) := q(a, b) + (1 - q(a, b)) R_{t-1}(a, b + 1; c) - c$$

is continuous as a sum of continuous functions, and is nonincreasing in c : the first two terms are nonincreasing (a constant and a nonnegative multiple of the nonincreasing R_{t-1}), and $-c$ is strictly decreasing. The outer map $x \mapsto \max\{0, x\}$ preserves both continuity and monotonicity, so $R_t = \max\{0, \psi_t\}$ is continuous and nonincreasing in c . For $c = 0$, $R_t(a, b; 0) \geq R_1(a, b; 0) = q(a, b) > 0$. For $c = 1$, since $R_{t-1}(a, b + 1; 1) \leq R_{t-1}(a, b + 1; 0)$ and $q(a, b) \leq 1$, one has $\psi_t(1) \leq q(a, b) - 1 \leq 0$, hence $R_t(a, b; 1) = 0$. $\square$

Lemma 3 (Strictly decreasing “continue” margin). *Define $\phi_t(c) := q(a, b) + (1 - q(a, b)) R_{t-1}(a, b + 1; c) - c$. Then ϕ_t is continuous and strictly decreasing on $[0, \infty)$, with $\phi_t(0) > 0$ and $\phi_t(1) \leq 0$.*

Proof: Continuity follows from Lemma 2. If $0 \leq c_0 < c_1$, then using that R_{t-1} is nonincreasing,

$$\begin{aligned} \phi_t(c_1) - \phi_t(c_0) &= (1 - q)(R_{t-1}(c_1) - R_{t-1}(c_0)) \\ &\quad - (c_1 - c_0) \leq 0 - (c_1 - c_0) < 0, \end{aligned}$$

so ϕ_t is strictly decreasing. The sign claims are from Lemma 2. $\square$

Theorem 4 (Reservation–cost policy). *Fix n and a stage $k < n$ with posterior parameters (a_k, b_k) . There exists a unique reservation cost $h_{n,k}^B(a_k, b_k) \in [0, 1]$ such that*

$$V_{n,k}(0, a_k, b_k; c) > 0 \iff c < h_{n,k}^B(a_k, b_k).$$

Equivalently, $h_{n,k}^B(a_k, b_k)$ is the unique solution c to

$$c = \frac{a_k}{a_k + b_k} + \frac{b_k}{a_k + b_k} V_{n,k+1}(0, a_k, b_k + 1; c). \quad (14)$$

The Bayes-optimal policy is: if $z_k = 1$, stop and obtain 1; if $z_k = 0$, continue iff $c < h_{n,k}^B(a_k, b_k)$.

Proof: Work with R_t at $t = n - k$. By Lemma 3, ϕ_t is strictly decreasing and continuous with $\phi_t(0) > 0$ and $\phi_t(1) \leq 0$, hence it has a unique root $h_t(a, b) \in [0, 1]$. From (13), $R_t = \max\{0, \phi_t\}$, so $R_t > 0$ iff $c < h_t(a, b)$. Continuity gives $\phi_t(h_t) = 0$, which is (14) after mapping (t, a, b) back to (n, k, a_k, b_k) . The stated action rule is exactly the comparison between the two branches in (11), while $z_k = 1$ is absorbing with value 1. $\square$

Adaptation to BEACON. For binary rewards, BEACON replaces the continuous Normal framework with Bernoulli-Beta conjugate learning. The stopping criterion from Theorem 1 becomes the binary reservation cost policy in Theorem 4, while the continuous sufficient statistics (z_k, μ_k, σ_k) of Lemma 1 are replaced with the discrete sufficient statistics (z_k, a_k, b_k) updated via (9), in place of the continuous recursion (5). This maintains BEACON's sequential structure while using binary-specific optimal stopping decisions.

E.7 Proof of Theorem 1

Proof. Under the Normal reward model with Normal-Inverse-Gamma prior, (z_k, μ_k, σ_k) (together with k) are sufficient statistics for the state (Lemma in Appendix E.3). Standardizing gives $\hat{z}_k = (z_k - \mu_k)/\sigma_k$ and reduces the problem to the canonical (location-scale normalized) sequential search instance.

By Theorem 2 (Universal Index Policy), for the canonical problem there exists a strictly decreasing index function $h_{n,k}(\cdot)$ such that the (myopic) continuation rule

$$\text{Continue at step } k \iff h_{n,k}(\hat{z}_k) > \frac{c}{\sigma_k}$$

maximizes the Bellman value function. Translating back to original (unscaled) variables yields exactly condition (3). Therefore the stopping time

$$K = \min\{k \geq k_0 : h_{n,k}(\hat{z}_k) \leq c/\sigma_k\} \wedge n$$

achieves the optimal value $\mathbb{E}[z_K - Kc]$.

Optimality of K follows because: (i) any earlier stop with $h_{n,k}(\hat{z}_k) > c/\sigma_k$ forgoes strictly positive

expected marginal gain; (ii) any continuation with $h_{n,k}(\hat{z}_k) \leq c/\sigma_k$ incurs cost exceeding expected benefit; (iii) strict monotonicity of $h_{n,k}$ implies no alternative rule can dominate. Hence K is the unique optimal stopping time under the stated assumptions. $\square$

E.8 Sensitivity Analysis

Proposition 1. The optimal stopping time K under the policy is influenced by several factors: it decreases with higher sampling cost c and larger current best reward z_k , while increasing with higher posterior mean μ_k and greater posterior scale parameter σ_k . Additionally, the algorithm becomes more patient (tend to continue) when more remaining samples $(n - k)$ are available.

Proof of Proposition 1 Proof: From Theorem 3, we know that the h-index function $h_{n,k}(\hat{z})$ for $k_0 \leq k < n$ is strictly decreasing, convex, and satisfies $\lim_{\hat{z} \rightarrow \infty} h_{n,k}(\hat{z}) = 0$.

Under the UIP stopping criterion in Equation (3), stopping occurs when $h_{n,k}(\hat{z}_k) \leq c/\sigma_k$, equivalently when $\hat{z}_k \geq h_{n,k}^{-1}(c/\sigma_k)$. Since $h_{n,k}$ is strictly decreasing, so is its inverse $h_{n,k}^{-1}$; hence a higher c raises c/σ_k but lowers the $\hat{z}$ -space threshold $h_{n,k}^{-1}(c/\sigma_k)$, making the stopping condition easier to satisfy and leading to earlier stopping (smaller K).

For a fixed standardized best reward $\hat{z}$, the h-index function $h_{n,k}(\hat{z})$ is increasing in $n - k$ (the number of remaining samples) as shown in Bau-cells and Zorc (2024). This means that when more samples remain available (larger $n - k$), the threshold for stopping becomes higher, making the algorithm more "patient" and likely to continue sampling.

With $\hat{z}_k = (z_k - \mu_k)/\sigma_k$, higher z_k increases $\hat{z}_k$, which decreases $h_{n,k}(\hat{z}_k)$ by the monotonicity of h (Theorem 3) and yields earlier stopping. Conversely, higher μ_k decreases $\hat{z}_k$, increasing $h_{n,k}(\hat{z}_k)$ and extending sampling. For the scale parameter σ_k , observe first that under BEACON's Jeffreys non-informative prior ($\nu_0 = 0, \alpha_0 = -1/2$) the posterior mean reduces to the sample mean, $\mu_k = \bar{r}_k$, so $z_k = \max_{i \leq k} r_i \geq \bar{r}_k = \mu_k$ and hence $\hat{z}_k \geq 0$ throughout any run. Under this regime, σ_k enters the stopping inequality $h_{n,k}(\hat{z}_k) \leq c/\sigma_k$ through two reinforcing channels: a larger σ_k shrinks the nonnegative $\hat{z}_k$ (raising $h_{n,k}(\hat{z}_k)$ by monotonicity) and simultaneously lowers the threshold c/σ_k . Both effects widen the

margin $h_{n,k}(\hat{z}_k) - c/\sigma_k$, encouraging continued sampling (larger K).

E.9 Robust Updating Formula and Details

Robust Update Rule. To mitigate negative skewness and extreme left-tail outliers, we modify the standard posterior update by filtering rewards below the 1% posterior-predictive quantile. Specifically,

$$z_{k+1} = \max\{z_k, r_{k+1}\}, \quad (15)$$

$$\tilde{r}_{k+1} = \begin{cases} \mu_k, & \text{if } r_{k+1} < q_{0.01}, \\ r_{k+1}, & \text{otherwise,} \end{cases} \quad (16)$$

$$\mu_{k+1} = \mu_k + \frac{\tilde{r}_{k+1} - \mu_k}{\nu_0 + k + 1}, \quad (17)$$

$$\sigma_{k+1} = \sqrt{\frac{1 - \frac{1}{(\nu_0 + k + 1)^2}}{2\alpha_0 + k + 1}} \times \sqrt{(2\alpha_0 + k)\sigma_k^2 + (\tilde{r}_{k+1} - \mu_k)^2}. \quad (18)$$

where $q_{0.01} = F_{2\alpha_k}^{-1}(0.01 \mid \mu_k, \sigma_k)$ is the 1% quantile of the posterior predictive distribution. Here the subscript convention is that $(\mu_k, \sigma_k, \alpha_k)$ are computed from the first k rewards, so the quantile applied to the incoming reward r_{k+1} uses $2\alpha_k$ degrees of freedom. In Algorithm 1 the loop counter has already been incremented before this test, which is why the corresponding line reads $F_{2\alpha_{k-1}}^{-1}$; the two expressions refer to the same posterior.

Interpretation. This one-sided winsorization caps the influence of extreme left-tail samples at the posterior mean, reducing variance inflation while leaving the maximum statistic z_{k+1} intact unless a new best reward is observed. The adjustment preserves the right-tail fidelity of the reward distribution, which is essential for correctly identifying high-quality responses.

Practical Notes. (i) The choice of threshold p is robust across $[0.5\%, 2\%]$, with $p = 1\%$ as default. (ii) The update is $O(1)$ per step; quantiles can be pre-tabulated for efficiency. (iii) For numerical stability, enforce $\sigma_k \geq 10^{-6}$. (iv) Optionally, robust updates can be activated only when recent empirical skewness is strongly negative (e.g., $\gamma_1 < -0.5$).

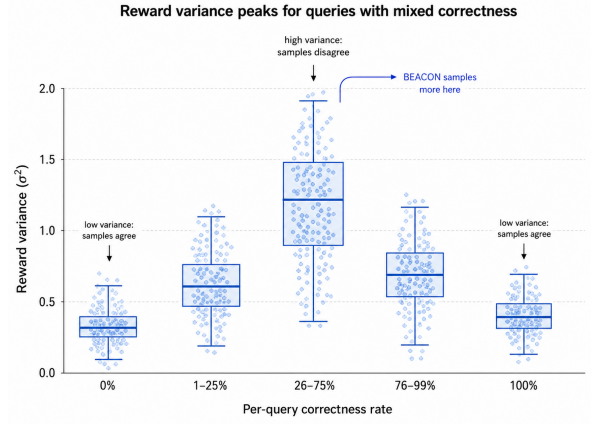


Figure 9: **Reward variance peaks for queries with mixed correctness** (Gemini 2.5 Flash on AIME 2024/25, Nemotron-70B reward model). Queries are stratified by per-query correctness rate (the fraction of $N = 32$ sampled responses that produce the correct final answer), and reward variance is computed across the sampled rewards for each query. Variance is lowest at the extremes—when all samples are correct or all are incorrect, reward signals consistently indicate quality and additional sampling adds little—and peaks for queries with mixed correctness, where the reward distribution carries the most information about which response to select. BEACON’s posterior-variance-based stopping criterion exploits this pattern directly: high posterior variance triggers continued sampling, while low variance triggers early stopping.

F Additional Experimental Results

F.1 Reward Variance Correlates with Response Correctness

A core assumption of BEACON’s adaptive sampling is that reward variance carries information about response quality dispersion: low variance indicates consistent generations, while high variance indicates mixed or unreliable responses. We empirically validate this assumption by stratifying queries by their per-query correctness rate (proportion of generated responses that produce the correct final answer) and computing the corresponding reward variance.

Figure 9 shows the relationship across AIME24/25 queries. Queries where all generations are correct exhibit low reward variance; queries where all generations are incorrect also exhibit low variance, since rewards consistently indicate poor quality; queries with mixed correctness (some correct, some incorrect) exhibit substantially higher variance.

This pattern confirms that reward variance is

informative about whether additional sampling is likely to help: when all samples agree (high or low quality), variance is low and further sampling adds little; when samples disagree, variance is high and further sampling may identify a better response. BEACON’s posterior-variance-based stopping criterion exploits this correlation directly.

F.2 Statistical Significance Analysis of BEACON

To assess the reliability of our results, we present a focused analysis using LLaMA-3.2-3B as our base model. Each experiment was conducted with 5 different random seeds, and we report the error bars as the standard error of the mean (SEM). As shown in Table 5, BEACON achieves comparable performance to the BoN baseline in terms of accuracy ($33.0 \pm 1.6\%$ vs. $33.4 \pm 1.3\%$ for reasoning tasks) and win rate ($27.9 \pm 1.8\%$ vs. $27.5 \pm 2.5\%$ for alignment tasks), with overlapping error margins in both cases indicating no substantial performance difference, while using significantly fewer samples (15.8 ± 1.2 vs. 32.0 for reasoning and 14.5 ± 2.3 vs. 32.0 for alignment).

F.3 Impacts of Cost on the Value Optimizations

The sampling cost, denoted by c , plays a critical role in the optimization of our value function, which serves as the core objective for determining the optimal stopping criterion. As illustrated in Figure 7, variations in the cost parameter directly influence the shape and peak of the value function and the resulting optimal sample size. The left subplot of Figure 7 shows that for any given sampling cost, the value function initially increases with the number of samples as the expected reward grows when we have more sample option to select, but eventually decreases as the cumulative cost outweighs the marginal gain from additional samples. Notably, increasing the sampling cost leads to a lower maximum achievable value and shifts the point of optimal stopping (where the value function peaks) towards a smaller number of samples. The right subplot of Figure 7 further emphasizes this relationship by directly plotting the optimal sample size against the sampling cost. This plot clearly demonstrates a strong inverse correlation: as the cost of obtaining each sample increases, the BEACON framework optimally decides to stop sampling earlier, resulting in a significantly reduced optimal sample size.

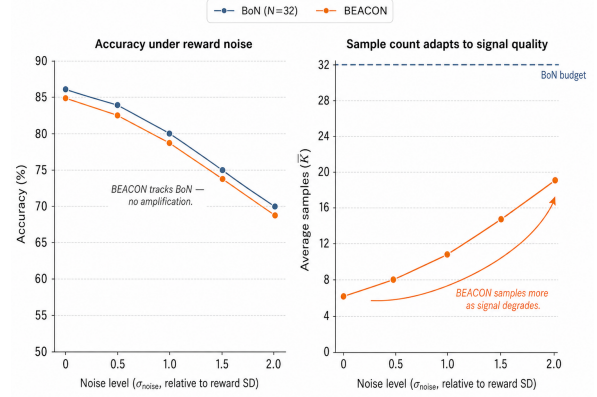


Figure 10: **BEACON’s behavior under reward signal noise** (Gemini 2.5 Flash on AIME 2024/25, Nemotron-70B reward model). Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma_{\text{noise}}^2)$ is injected into reward model outputs, with σ_{noise} scaled relative to the empirical reward standard deviation. **Left:** Accuracy under noise. BEACON tracks BoN ($N = 32$) closely across all noise levels, indicating that adaptive stopping inherits but does not amplify reward model errors. **Right:** Average sample count $\bar{K}$. While BoN’s budget is fixed at $N = 32$, BEACON’s sample count grows with noise as posterior variance remains elevated for longer, yet stays well below the BoN budget even under heavy noise. Together, these results show that BEACON adapts compute to signal quality without catastrophic failure.

F.4 Reward Noise Stress Test

To validate BEACON’s behavior under degraded reward signals, we inject Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma_{\text{noise}}^2)$ into reward model outputs at varying levels and rerun BEACON’s stopping logic and BoN- $N=32$ selection on the perturbed scores. We sweep $\sigma_{\text{noise}} \in \{0, 0.5, 1.0, 1.5, 2.0\}$ relative to the empirical reward standard deviation.

Figure 10 shows accuracy and $\bar{K}$ as functions of noise level. As noise increases, both BEACON and BoN’s accuracy decline at comparable rates, indicating that BEACON inherits but does not amplify reward model errors. The figure shows the framework’s principled response to noisier signals: posterior variance remains elevated for longer, triggering additional sampling. Importantly, BEACON’s efficiency advantage over BoN is preserved across all noise levels— $\bar{K}$ stays well below $N = 32$ even under heavy noise.

These results confirm that BEACON’s behavior under degraded rewards is predictable and consistent with its design: the framework adapts compute to signal quality without catastrophic failure, making it suitable for deployment where reward model

Table 5: Comparison of BEACON with baseline methods using LLaMA-3.2-3B. Results shown as mean $\pm$ SEM across 5 random seeds.

Method	Reasoning (Avg. MATH/AIME)			Alignment (AlpacaEval 2.0)		
	Acc. $\uparrow$ (%)	Samp. $\downarrow$ ($\bar{K}$)	Reward $\uparrow$ ($\bar{z}_K$)	Win $\uparrow$ (%)	Samp. $\downarrow$ ($\bar{K}$)	Reward $\uparrow$ ($\bar{z}_K$)
Direct CoT	20.0 $\pm$ 2.8	1.0	-1.15 $\pm$ 0.10	17.6 $\pm$ 3.5	1.0	-1.19 $\pm$ 0.05
BoN (N-RM)	33.4 $\pm$ 1.3	32.0	1.75 $\pm$ 0.12	27.5 $\pm$ 2.5	32.0	1.94 $\pm$ 0.15
BEACON (N-RM)	33.0$\pm$1.6	15.8$\pm$1.2	1.68$\pm$0.10	27.9$\pm$1.8	14.5$\pm$2.3	1.85$\pm$0.13

quality cannot be guaranteed in advance.

F.5 Normality Analysis of Reward Distributions

While BEACON uses Bayesian updating to estimate the query-specific reward distribution, real-world reward distributions from LLMs may not strictly satisfy the Normal-NIG modeling assumptions. This appendix section visually explores the characteristics of reward distributions observed in our experiments using the Nemotron reward model, providing context for our robust updating mechanism.

Figure 11 presents an aggregated view of reward distributions, conditioned on whether the generated responses were ultimately deemed correct or incorrect. We observe that for both categories, the empirical distributions of rewards are reasonably approximated by a normal distribution, albeit with different means and variances. Specifically, correct answers tend to receive higher mean rewards, but both distributions exhibit a unimodal, bell-like shape characteristic of normality. This overall trend provides a foundational justification for employing a Gaussian-based learning model.

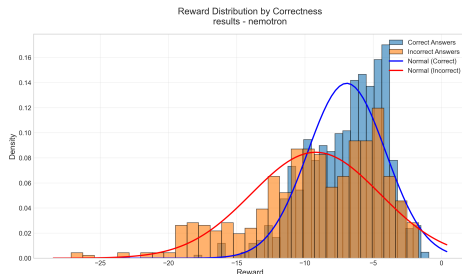


Figure 11: Aggregated reward distributions from the Nemotron RM, separated for responses classified as correct (blue histogram, blue normal fit) and incorrect (orange histogram, red normal fit). Both distributions show approximate normality.

However, analyzing distributions at an aggre-

gate level can mask variations in individual query-specific reward patterns. Therefore, Figure 12 dives into specific cases to illustrate the types of reward distributions encountered for individual prompts. The leftmost panel ("Normal Question 8") depicts a common scenario where the rewards for multiple samples from a single prompt follow an approximately normal distribution, though the specific mean and variance naturally differ from prompt to prompt. In contrast, the middle panel ("Non-Normal Question 3") illustrates an occasional but important pattern: the distribution consists primarily of high-reward samples with a few significantly lower, noisy rewards in the left tail. This type of skewed distribution, or one with outliers, can badly influence standard posterior parameter updates. But it is precisely these instances that motivate BEACON's robust updating formula, which is designed to mitigate the impact of such extreme low-value outliers, thereby maintaining a more stable and reliable estimation of the reward potential focused on the right tail. The rightmost panel ("All Questions") shows the overall distribution of all rewards for context.

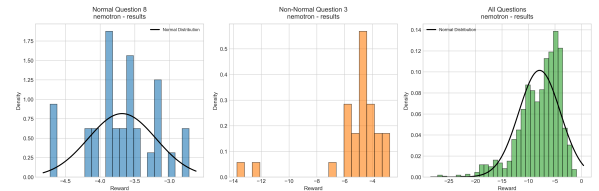


Figure 12: Examples of query-specific reward distributions using the Nemotron RM. Left: A typical case exhibiting approximate normality ("Normal Question 8"). Middle: An occasional case with predominantly high rewards and some low-value outliers ("Non-Normal Question 3"), motivating robust updates. Right: Aggregate distribution of all rewards.

These observations support our approach: while normality is a useful working assumption for the

bulk of reward behaviors, the adaptive robust update mechanism provides resilience against deviations, particularly those caused by uninformative low scores, ensuring BEACON remains effective across diverse and sometimes non-ideal reward landscapes.

F.6 Case Analysis: Solution Diversity and Failure Analysis

The BEACON framework employs an adaptive stopping mechanism that dynamically adjusts the number of samples (K) based on the expected marginal gain from additional sampling, relative to the sampling cost c and posterior uncertainty about the reward distribution (σ_k). This mechanism inherently influences the diversity of generated solution candidates, balancing exploration breadth with computational efficiency. The stopping decision is driven by the consistency of reward model (RM) scores (reflected in σ_k), the quality of the current best response (z_k relative to μ_k), and the remaining sample budget ($n - k$), as governed by the Universal Index Policy (UIP).

Table 6 gives constructed illustrations of BEACON’s behavior across query types, showing qualitatively how these factors interact to determine stopping time and sample diversity. The prompts, responses, and reward values in the table are hypothetical and are included for exposition only; quantitative results are reported in Section 3 and Appendix F.1.

- **Easy Queries with Consistent High Rewards:** For simple queries (e.g., Scenario 1), initial samples $\{y_k\}_{k=1}^{k_0}$ yield uniformly high RM scores with low σ_k . Low posterior variance indicates that further sampling is unlikely to improve the best response, so BEACON stops at or near the bootstrap horizon. This yields low sample diversity, as responses are similar and high-quality.
- **Hard Queries with Consistent Low Rewards:** For queries the policy cannot solve (e.g., Scenario 2), samples consistently receive low RM scores with low σ_k . BEACON also terminates early here, since the expected marginal gain from another draw is small, again yielding low diversity. The corresponding failure mode is premature stopping when the RM systematically underscores a correct response.

- **Moderately Hard Queries with Improving Rewards:** For queries of intermediate difficulty (e.g., Scenario 3), early samples may be incorrect while later ones improve. BEACON continues sampling while σ_k remains elevated and stops once the reward distribution stabilizes, producing moderate K and medium diversity. The corresponding failure mode is delayed stopping when early incorrect samples inflate σ_k .
- **Very Hard or Ambiguous Queries with Inconsistent Rewards:** For open-ended or ambiguous queries (e.g., Scenario 4), RM scores vary widely and σ_k stays high. High variance encourages extended sampling toward the horizon n , maximizing the chance of finding a high-quality response and producing high diversity. The corresponding failure mode is excessive sampling when σ_k remains high because of RM noise rather than genuine quality dispersion.
- **High-Patience Configurations:** Under a small λ (equivalently, small c_x) or a large horizon n , BEACON continues sampling even after a good response has been found, because the cost-adjusted threshold c/σ_k is low. This produces late stopping and high diversity, at the cost of computation that may be unnecessary when the current best response is already adequate.

These regimes follow directly from the stopping rule: BEACON adjusts exploration according to reward consistency (σ_k), the quality of the current best response (z_k relative to μ_k), and the remaining budget ($n - k$). The dataset-level consequences of this behavior are quantified in Figure 4, which reports the empirical distribution of stopping points on MATH-500 and AIME-24/25, and in Figure 9, which shows that reward variance is highest precisely for the mixed-correctness queries where continued sampling is most likely to help.

Table 6: **Constructed illustrations** of BEACON’s stopping behavior across query types. Prompts, responses, and z_k values are hypothetical and chosen for exposition; they are not model outputs or measured reward-model scores.

Scenario Type	Example Prompt	S1 (Out & z_k)	S2 (Out & z_k)	S3 (Out & z_k)	S4 (Out & z_k)	S5 (Out & z_k)	S6 (Out & z_k)	Stops (K)	Diversity
Easy / High Reward	What is $2 + 2$?	Equals 4. ($z_k \approx 0.95$)	The sum is 4. ($z_k \approx 0.98$)	$2+2$ is 4. ($z_k \approx 0.96$)				Early ($K = 3$)	Low
Hard / Low Reward	Simple proof of Fermat’s Last Theorem...	[Failed Proof 1] ($z_k \approx -2.0$)	[Failed Proof 2] ($z_k \approx -2.2$)	[Failed Proof 3] ($z_k \approx -1.8$)				Early ($K = 3$)	Low
Moderately Hard / Improving	Simplify $\sqrt{242}$	Incorrect: $2\sqrt{60.5}$ ($z_k \approx -1.5$)	Error: $\sqrt{200} + \sqrt{42}$ ($z_k \approx -1.2$)	Correct: $11\sqrt{2}$ ($z_k \approx 0.95$)	Correct: $11 \cdot \sqrt{2}$ ($z_k \approx 0.93$)	Correct: $\sqrt{121 \cdot 2} = 11\sqrt{2}$ ($z_k \approx 0.96$)		Moderate ($K = 5$)	Medium
Very Hard / Inconsistent	How did US states get their names?	Brief: “Native words, kings.” ($z_k \approx 0.1$)	Partial: “VA from Virgin Queen...” ($z_k \approx 0.3$)	Flawed: “All named after presidents.” ($z_k \approx -0.5$)	Better: “Native Am. languages...” ($z_k \approx 0.8$)	Comprehensive ($z_k \approx 0.95$)		Late ($K \geq 5$)	High
High Patience / Extended	Outline three approaches to climate change.	Approach A (brief) ($z_k \approx 0.6$)	Approach B (flawed) ($z_k \approx 0.5$)	Approach A (detailed) ($z_k \approx 0.85$)	Approach B (detailed) ($z_k \approx 0.88$)	Approach C (detailed) ($z_k \approx 0.90$)	Approach D ($z_k \approx 0.92$)	Late ($K \geq 6$)	High

Note: S_i denotes Sample i . All entries are hypothetical illustrations; z_k values are chosen to be on the scale of Nemotron-70B reward scores but are not measured values.