Cost-aware Stopping for Bayesian Optimization

Qian Xie[*]
Cornell University
QX66@CORNELL.EDU

Linda Cai[*]
University of California, Berkeley
TCAI@BERKELEY.EDU

Alexander Terenin
Cornell University
AVT28@CORNELL.EDU

Peter I. Frazier
Cornell University
PF98@CORNELL.EDU

Ziv Scully
Cornell University
ZIVSCULLY@CORNELL.EDU

Abstract

In automated machine learning and other applications of Bayesian optimization, deciding when to stop evaluating expensive black-box functions is an important practical consideration. While several adaptive stopping rules have been proposed, in the cost-aware setting they lack guarantees ensuring they stop before incurring excessive function evaluation costs. We propose a *cost-aware stopping rule* for Bayesian optimization that adapts to varying evaluation costs and is free of heuristic tuning. Our rule is grounded in a theoretical connection to state-of-the-art cost-aware acquisition functions, namely the Pandora's Box Gittins Index (PBGI) and log expected improvement per cost. We prove a theoretical guarantee bounding the expected cumulative evaluation cost incurred by our stopping rule when paired with these two acquisition functions. In experiments on synthetic and empirical tasks, including hyperparameter optimization and neural architecture size search, we show that combining our stopping rule with the PBGI acquisition function consistently matches or outperforms other acquisition-function–stopping-rule pairs in terms of *cost-adjusted simple regret*, a metric capturing trade-offs between solution quality and cumulative evaluation cost.

1 Introduction

Bayesian optimization is a framework designed to efficiently find approximate solutions to optimization problems involving expensive-to-evaluate black-box functions, where derivatives are unavailable. Such problems arise in applications like hyperparameter tuning [22], robot control optimization [18], and material design [27]. It works by iteratively, (a) forming a probabilistic model of the black-box objective function based on data collected thus far, then (b) optimizing a function called the acquisition function, which balances exploration-exploitation tradeoffs, to carefully choose a new point at which to observe the unknown function in the next iteration.

In this work, we consider the *cost-aware* setting, where one must pay a cost to collect each data point, and study *adaptive stopping rules* that choose when to stop the optimization process. After stopping at some terminal time, we measure performance in terms of simple regret, which is the difference in value between the best solution found so far and the global optimum. Collecting a data point can reduce simple regret, but incurs cost in order to do so.

As an example, consider using a cloud computing environment to tune the hyperparameters of a classifier in order to optimize a performance metric on a given test set. Training and evaluating

[*]Equal contribution. Correspondence to: Qian Xie <QX66@CORNELL.EDU>.
Code available at: `https://github.com/QianJaneXie/CostAwareStoppingBayesOpt`.

Preprint. Under review.

test error takes some number of CPU or GPU hours, that may depend on the hyperparamaters used. These come with a financial cost, billed by the cloud computing provider, which define our cost function. The objective value is the business value of deploying the trained model under the given hyperparameters—a given function of the model's accuracy. From this perspective, simple regret can be understood as the opportunity cost for deploying a suboptimal model instead of the optimal one. Motivated by the need to balance cost-performance tradeoff in examples such as above, we aim to design stopping rules that optimizes *expected cost-adjusted simple regret*, defined as the sum of simple regret and the cumulative cost of data collection.

Several stopping rules have been proposed for Bayesian optimization. Simple heuristics—such as fixing a maximum number of iterations or stopping when improvement falls below a threshold—are widely used in practice [16, 17], but can either stop too early or lead to unnecessary evaluations. Other approaches include acquisition-function-based rules that stop when, for instance, the probability of improvement, expected improvement, or knowledge gradient drop below a preset threshold [16, 19, 8]; confidence-bound-based methods like UCB-LCB [17]; the expected minimum simple regret test [11]; as well as methods based on probabilistic regret bounds (PRB) [25]. Most of these target the classical setting which does not explicitly incorporate function evaluation costs.

In this work, we study how to design cost-aware stopping rules, motivated by two primary factors. First, state-of-the-art cost-aware acquisition functions such as the Pandora's Box Gittins Index (PBGI)[26] and log expected improvement per cost (LogEIPC) [1] have not yet been studied in the adaptive stopping setting. This is important because—as our experiments in Section 4 will show—for best performance, one should pair different acquisition functions with different stopping rules. Second, while certain stopping rules, such as UCB-LCB, are guaranteed to achieve a low simple regret, they are not necessarily guaranteed to do so with low evaluation costs. This is important because—as our experiments will show—UCB-LCB will often incur high evaluation costs, resulting in a high cost-adjusted simple regret.

Our main methodological contribution is a stopping rule that is provably Bayesian-optimal in the independent evaluation case. This stopping rule is constructed based on ideas from Pandora's Box theory, which forms the theoretical foundation for the recently-proposed PBGI acquisition function. It also aligns closely with an existing stopping rule previously proposed for expected improvement in the classical non-cost-aware setting: when paired with LogEIPC, our proposal extends this stopping rule to the cost-aware setting, thereby establishing a unified approach for cost-aware Bayesian optimization under both kinds of acquisition functions. Our specific contributions are as follows:

1. *A Novel Cost-Aware Stopping Criterion.* We propose an adaptive stopping rule derived naturally from Pandora's box theory, establishing a unified and principled framework applicable to cost-aware Bayesian optimization.
2. *Theoretical Guarantees.* We prove in Theorem 1 that our stopping rule, when paired with the PBGI or LogEIPC acquisition functions, satisfies a theoretical upper bound on expected cumulative cost up to stopping, which constitutes the first theoretical guarantee of this type for any adaptive stopping rule for Bayesian optimization.
3. *Empirical Validation.* Our experiments show that combining the PBGI acquisition function with our proposed stopping rule consistently matches or outperforms other combinations of acquisition functions and stopping rules.

2 Bayesian Optimization and Adaptive Stopping

In black-box optimization, the goal is to find an approximate optimum of an unknown objective function $f : X \to \mathbb{R}$ using a limited number of function evaluations at points $x_1, \ldots, x_T \in X$ where X is the search space and T is a given search budget, potentially chosen adaptively. The convention measures performance in terms of *expected simple regret* [9, Sec. 10.1], which is given by

$$\mathcal{R} = \mathbb{E}\left[\min_{1 \le t \le T} f(x_t) - \inf_{x \in X} f(x)\right] \tag{1}$$

where the expectation is taken over all sources of randomness, including the sequence of points $x_1, \ldots, x_T$ selected by the algorithm. Bayesian optimization approaches this problem by building a probabilistic model of f—typically a Gaussian process [21], conditioned on the observed data points

$(x_t, y_t)_{t=1}^T$, where $y_t = f(x_t)$. An acquisition function $\alpha_t : X \to \mathbb{R}$ then guides the selection of new samples by carefully balancing the exploration-exploitation tradeoff arising from uncertainty about f.

2.1 Cost-aware Bayesian Optimization

Cost-aware Bayesian optimization [13, 2] extends the above setup to account for the fact that evaluation costs can vary across the search space. For instance, in the hyperparameter tuning example of Section 1, costs can vary according to the time needed to train a machine learning model under a given set of hyperparameters x. Cost-aware Bayesian optimization handles this by introducing a cost function $c : X \to \mathbb{R}^+$, which may be known or unknown depending on the setup. The cost function, or observed costs, are then used to construct the acquisition function α_t.

In this work, we focus primarily on the *cost-per-sample* formulation [6, 5, 26] of cost-aware Bayesian optimization, which seeks methods achieving a low *expected cost-adjusted simple regret*

$$\mathcal{R}_c = \mathbb{E}\left[\underbrace{\min_{1\le t\le T} f(x_t) - \inf_{x\in X} f(x)}_{\text{simple regret}} + \underbrace{\sum_{t=1}^{T} c(x_t)}_{\text{cumulative cost}} \right]. \tag{2}$$

One can also work with the *expected budget-constrained* formulation [26], which incorporates budget constraints explicitly, and seeks algorithms which achieve a low simple regret under an expected evaluation budget. Here, performance is evaluated in terms of

$$\mathcal{R} = \mathbb{E}\left[\min_{1\le t\le T} f(x_t) - \inf_{x\in X} f(x)\right] \qquad \text{where } x_1, .., x_T \text{ satisfy} \qquad \mathbb{E}\sum_{t=1}^{T} c(x_t) \le B. \tag{3}$$

The stopping rules we study can be applied in this setting as well: we discuss this in Section 3.2.

For both settings, we work with a number of acquisition functions—chiefly, *log expected improvement per cost (LogEIPC)* [1] and the *Pandora's Box Gittins Index (PBGI)* [26]. These, respectively, are

$$\alpha_t^{\text{LogEIPC}}(x) = \log \frac{\text{EI}_{f|x_{1:t},y_{1:t}}(x; y^*_{1:t})}{c(x)} \tag{4}$$

and

$$\alpha_t^{\text{PBGI}}(x) = \begin{cases} g \quad \text{where } g \text{ solves} \quad \text{EI}_{f|x_{1:t},y_{1:t}}(x; g) = c(x), & x \notin \{x_1, \dots, x_t\} \\ f(x), & x \in \{x_1, \dots, x_t\} \end{cases} \tag{5}$$

where $\text{EI}_\psi(x; y) = \mathbb{E}\left[\max(y - \psi(x), 0)\right]$ is the expected improvement at point x with respect to some random function $\psi : X \to \mathbb{R}$ and a comparator point y, $y^*_{1:t} = \min_{1\le s\le t} y_s$ is the best observed value up to time t, and $c(x)$ is the evaluation cost at point x. In the classical uniform-cost setting, we also consider the classical *lower confidence bound* and *Thompson sampling* acquisition functions: see Garnett [9] for additional details.

2.2 Adaptive Stopping Rules

At present, to the best of our knowledge, stopping rules for Bayesian optimization have only been studied in the classical setting without explicit evaluation costs. Current methods can be broadly categorized as follows:

1. *Criteria based on convergence or significance of improvement.* This includes empirical convergence, namely stopping when the best value remains unchanged for a fixed number of iterations, or the *global stopping strategy (GSS)* [3], which stops when the improvement is no longer statistically significant relative to the inter-quartile range (i.e., the range between the 25th percentile and the 75th percentile) of prior observations.
2. *Acquisition-based criteria.* This includes stopping rules built from acquisition functions such as the *probability of improvement (PI)* [16], *expected improvement (EI)* [15, 19, 11], and *knowledge gradient (KG)* [8]. These approaches typically stop when the acquisition value falls below a fixed threshold—a predetermined constant, median of the initial acquisition values, or the cost of sampling—but typically assume uniform costs and do not adapt to evaluation costs which can vary across the search space.

3. *Regret-based criteria.* This includes stopping rules based on confidence bounds such as *UCB-LCB* [17] and the *gap of expected minimum simple regrets* [11]. These stop when certain regret bounds fall below a preset or data-driven threshold. The related *probabilistic regret bound (PRB)* stopping rule [25] stops when a candidate point is estimated to be ϵ-optimal with probability at least $1 - \delta$.

In settings beyond Bayesian optimization, Chick and Frazier [6], have proposed a cost-aware stopping rules for finite-domain sequential sampling problems with independent values. Although this formulation allows for varying costs, it does not extend to general Bayesian optimization settings which use correlated Gaussian process models.

3 A Stopping Rule Based on the Gittins Index

In this work, we propose a new stopping rule tailored for two state-of-the-art acquisition functions used in cost-aware Bayesian optimization: LogEIPC and PBGI, introduced in Section 2. As discussed by Xie et al. [26], these acquisition functions are closely connected, arising from two different approximations of the intractable dynamic program which defines the Bayesian-optimal policy for cost-aware Bayesian optimization. We now show that this connection can be used to obtain a principled stopping criterion to be paired with both acquisition functions.

A stopping rule for PBGI. To derive a stopping rule for PBGI, consider first the Pandora's Box problem, from which it is derived. Pandora's Box can be seen as a special case of cost-per-sample Bayesian optimization, as introduced in Section 2, where $X = \{1, \dots, N\}$ is a discrete space and f is assumed uncorrelated. In this setting, the observed values do not affect the posterior distribution (i.e., $f \mid x_{1:t}, y_{1:t} = f$), and thus the acquisition function value at point x is time-invariant and can be written simply as $\alpha^{\mathrm{PBGI}}(x)$, where $\mathrm{EI}_f(x; \alpha^{\mathrm{PBGI}}(x)) = c(x)$. Following Weitzman [24], one can show using a Gittins index argument that selecting x_t to minimize α^{PBGI} is Bayesian-optimal under minimization—the algorithm which does so achieves the smallest expected cost-adjusted simple regret, among all adaptive algorithms.

One critical detail applied in the optimality argument of Weitzman [24] is that the policy defined by α^{PBGI} is *not* Bayesian-optimal for any fixed deterministic T. Instead, it is optimal only when the stopping time T is chosen according to the condition[2]

$$\min_{x \in X \setminus \{x_1, \dots, x_t\}} \alpha^{\mathrm{PBGI}}(x) \geq y^*_{1:t}, \tag{6}$$

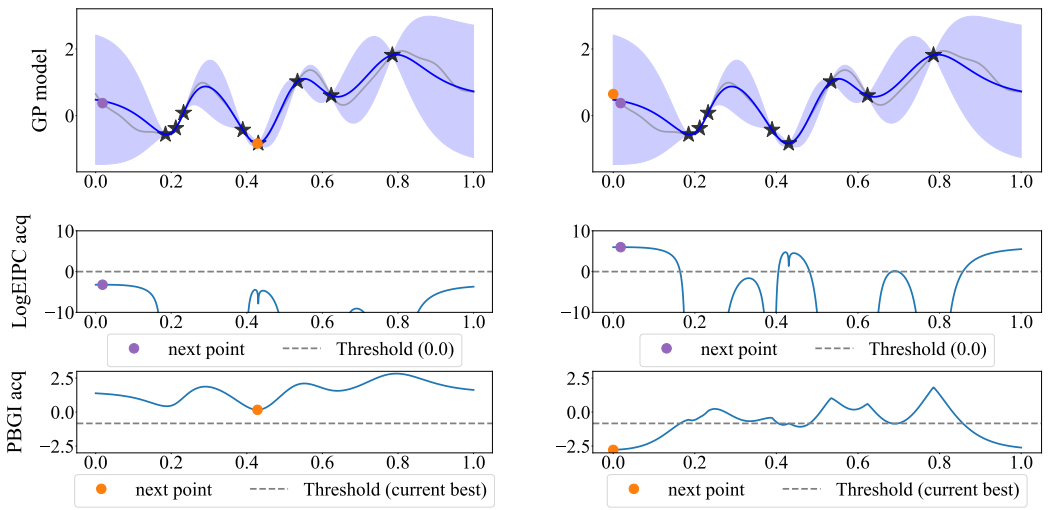
where, as before, $y^*_{1:t}$ is the best value observed so far.

In the correlated setting we study, Xie et al. [26] extend α^{PBGI} to an acquisition function by recomputing it at each time step t based on the posterior mean and variance, which defines α^{PBGI}_t as in Eq. (5). For extending the stopping rule, a subtle design choice arises: should we use $\alpha^{\mathrm{PBGI}}_{t-1}$ (before update) or α^{PBGI}_t (after update) in Eq. (6)? While prior work [10] adopts the former, we choose the latter for two main reasons.

First, it more faithfully reflects the intuition behind Weitzman's original stopping rule. In the independent setting, $\alpha^{\mathrm{PBGI}}(x)$ represents a kind of *fair value* for point x [12]. For evaluated points $x \in \{x_1, \dots, x_t\}$, this is simply the observed function value. For unevaluated points $x \notin \{x_1, \dots, x_t\}$, the fair value reflects uncertainty in $f(x)$ and the cost $c(x)$. The stopping rule (6) then says to *stop when the best fair value is among the already-evaluated points*, namely, when no point provides positive expected gain relative to its cost if evaluated in the next round, conditioned on current observations. In the correlated setting, the fair value naturally extends to $\alpha^{\mathrm{PBGI}}_t(x)$. Second, using α^{PBGI}_t yields tangible empirical gains in cost-adjusted regret, as shown in Appendix C.2.

Connection with LogEIPC. The above reasoning may at first seem to be fundamentally tied to the Pandora's Box problem and its Gittins-index-theoretic properties. We now show that it admits a second interpretation in terms of log expected improvement per cost, which arises from a completely

[2]As explained in Xie et al. [26], alternative tie-breaking rules also preserve optimality in cost-per-sample optimization, and there exists a tie-breaking rule that yields optimality under expected-budget constraints. For simplicity, we use the stopping-as-early-as-possible tie-breaking rule throughout this paper.



(a) $c(x) \equiv 0.1$ (b) $c(x) \equiv 0.0001$

Figure 1: Illustration of the PBGI/LogEIPC stopping rule under the uniform-cost setting. In this example with 8 initial points being evaluated: when the cost-per-sample is large ($c(x) \equiv 0.1$), the maximum LogEIPC acquisition value falls below the threshold 0.0, indicating stopping; the minimum PBGI acquisition value exceeds the current best observed value, also indicating stopping. When the cost-per-sample is small ($c(x) \equiv 0.0001$), the maximum LogEIPC acquisition value remains above the threshold 0.0, indicating no stopping; the minimum PBGI acquisition value is smaller than the current best observed value and again indicating no stopping.

different one-time-step approximation to the Bayesian-optimal dynamic program for cost-aware Bayesian optimization in the general correlated setting.

To show this, we start with the definition above, and apply a sequence of transformations. Consider an unevaluated point $x \in X \backslash \{x_1, \ldots, x_t\}$. Because $\alpha_t^{\mathrm{PBGI}}(x)$ is defined in Eq. (5) to be the unique solution to

$$\mathrm{EI}_{f|x_{1:t},y_{1:t}}(x; \alpha_t^{\mathrm{PBGI}}(x)) = c(x) \tag{7}$$

and since $\mathrm{EI}(x; y)$ is strictly monotone (increasing under minimization) in y, any inequality involving $\alpha_t^{\mathrm{PBGI}}(x; c)$ can be lifted through the EI function without changing its direction. This means that

$$\alpha_t^{\mathrm{PBGI}}(x) \geq y^*_{1:t} \qquad \text{holds if and only if} \qquad \mathrm{EI}\big(x; y^*_{1:t}\big) \leq c(x). \tag{8}$$

This means the PBGI stopping rule from Eq. (6) is equivalent to stopping when

$$\mathrm{EI}\big(x; y^*_{1:t}\big) \leq c(x) \quad \text{for all } x \in X \backslash \{x_1, \ldots, x_t\} \tag{9}$$

meaning *stop when no point's expected improvement is worth its evaluation cost*. Rearranging the inequality and taking logs, we can rewrite this condition using the LogEIPC acquisition function:

$$\max_{x \in X \backslash x_{1:t}} \alpha_t^{\mathrm{LogEIPC}}(x; y^*_{1:t}) \leq 0. \tag{10}$$

We call the stopping rule given by the equivalent conditions (6), (9), and (10) the *PBGI/LogEIPC stopping rule*. Figure 1 gives an illustration of how the rule behaves in a simple setting, demonstrating that it is more conservative—in the sense of preferring to stop earlier—when the cost is high.

3.1 A Theoretical Guarantee on Cost up to Stopping

In the discrete Pandora's Box setting, maximizing $\alpha_t^{\mathrm{PBGI}}(x)$ is Bayesian-optimal in more than one way: doing so not only achieves the best possible expected cost-adjusted simple regret, under appropriately-scaled costs, it also achieves the best *expected simple regret, among all algorithms satisfying an expected budget constraint* of the form

$$\mathbb{E}\left[\sum_{t=1}^{T} c(x_t)\right] \leq B \tag{11}$$

where $B > 0$ is the total budget, and the cost function used in defining $\alpha_t^{\mathrm{PBGI}}(x)$ is $\lambda c(x)$ for some cost-scaling factor λ which depends on B. A proof is given by Xie et al. [26, Theorem 2].

This result, at first, appear to be completely Pandora's-Box-theoretic: it requires X to be discrete and f to be uncorrelated. In the more general correlated setting of cost-aware Bayesian optimization, it is not a priori clear how much costs our stopping rule will incur in expectation. To understand this, we prove the following upper bound on the expected cumulative cost up to the stopping time.

Theorem 1. *Consider minimizing objective function f, drawn from a stochastic process with constant mean μ, over a compact, non-empty domain X, and let c be a known cost function with finite mean. Assume the algorithm begins with one initial evaluation at a pre-selected point x_1 in X with cost $C := c(x_1)$, and subsequently runs either the PBGI or LogEIPC acquisition function with the PBGI/LogEIPC stopping rule. Let the stopping time be defined as $\tau := \min_{t\geq 1}\left\{\max_{x\in X} \alpha_t^{\mathrm{LogEIPC}}(x) \leq 0\right\}$. Let $U := \mu - \mathbb{E}[\min_{x\in X} f(x)] < \infty$ denote the upper bound on expected cumulative improvement $\mathbb{E}\left[\sum_{t=2}^{\tau}(y^*_{1:t-1} - y^*_{1:t})\right]$. Then the expected cumulative cost up to stopping is bounded by*

$$\mathbb{E}\left[\sum_{t=1}^{\tau} c(x_t)\right] \leq C + U. \tag{12}$$

When $f \sim \mathcal{GP}(\mu, K)$, the term U on the right-hand side can be further upper bounded using classic results on the expected supremum of a Gaussian process; see, for example, Lifshits [14, Theorem 10.1]. A proof of Theorem 1 is given in Appendix A. The proof also indicates that the theorem extends to acquisition functions beyond PBGI and LogEIPC that ensure sufficient expected improvement relative to cost before the stopping time τ; see condition (17). This result demonstrates that the PBGI stopping rule avoids excessive spending not just in Pandora's Box [26], but also Bayesian optimization with general correlations. As discussed below, Theorem 1 has another important implication for settings where there is no direct conversion between objective function values and evaluation costs.

3.2 Practical Implementation Considerations for the PBGI/LogEIPC Stopping Rule

Choosing a cost scaling factor λ. In practice, evaluation costs often come in units that differ from how benefit is measured—for example, run time versus monetary costs. To reconcile this, we multiply costs by a scaling factor $\lambda > 0$, i.e., we use cost function $\lambda c(x)$ instead of $c(x)$, in order to ensure that costs $\lambda c(x)$ and objective values $f(x)$ have the same units. For example, if one is willing to spend 1000 seconds of time to improve the objective by 0.01, then one should set $\lambda = 10^{-5}$.

If one knows the unit conversion explicitly—for instance, the hourly rate of a compute cluster and the business value of improved model performance—then the scaling factor λ can be set directly. Otherwise, Theorem 1 gives a principled heuristic for choosing λ. When applied the scaled-cost setting, Theorem 1 bounds the expected total unscaled cost by $C + U/\lambda$. This means that if one has an expected cost budget B, one can safely use scaling factor $\lambda = (B - C)/U$. This solves a significant obstacle raised by Xie et al. [26].

Unknown costs. In practice, evaluation costs are often not known in advance. They can be modeled in one of two ways: (i) as deterministic functions based on domain knowledge, for instance in hyperparameter tuning by setting them proportional to the number of model parameters, or (ii) as stochastic quantities modeled using a Gaussian process over the log costs. For the latter, Astudillo et al. [2] proposed a general framework for handling unknown costs, which can also be applied to our stopping rules. For the LogEIPC stopping rule, following Astudillo et al. [2, Proposition 2], the $c(x)$ in Eq. (10) can be replaced with

$$\frac{1}{\mathbb{E}[1/c(x)]} = \exp\left\{\mu_{\ln c}(x) - (\sigma_{\ln c}(x))^2/2\right\}. \tag{13}$$

Alternatively, one could also replace it with

$$\mathbb{E}[c(x)] = \exp\left\{\mu_{\ln c}(x) + (\sigma_{\ln c}(x))^2/2\right\}. \tag{14}$$

The difference in sign before the variance term reflects how each formulation responds to predictive uncertainty: (13) encourages exploration more than (14). For PBGI, under the unknown-cost setting, $c(x)$ in (5) should be replaced by $\mathbb{E}[c(x)]$ using (14), reflecting the manner in which costs enter the respective root-finding problem. This means that in unknown-cost settings, the equivalence between the PBGI and LogEIPC stopping rules holds when replacing $c(x)$ by (14), but not by (13).

Preventing spurious stops. Although our stopping rule has theoretical guarantees, in practice, it—like other adaptive stopping rules—can still suffer from spurious stops caused by two main sources which we now discuss. First, early in the optimization process, the Gaussian process' model parameters often fluctuate significantly as new data points are collected, causing unstable stopping signals. To mitigate this, we enforce a stabilization period consisting of the first several evaluations, during which we do not allow any stopping rule to trigger. Second, imperfect optimization of the acquisition function—which is especially common in higher-dimensional search spaces—can lead to misleading stopping signals. To handle this, we use a *debounce* strategy, requiring the stopping rule to consistently indicate stopping over several consecutive iterations before stopping optimization.

4 Experiments

To evaluate our proposed PBGI/LogEIPC stopping rule, we design three complementary sets of experiments that progressively assess its behavior. First, we consider an idealized, low-dimensional setting where the Gaussian process model exactly matches the true objective: this lets us isolate the rule's behavior in the absence of any modeling or optimization error. Then, we move to an 8-dimensional synthetic problem and evaluate PBGI/LogEIPC stopping rule when each acquisition-function minimization must be approximated via a numerical optimizer. Finally, we test in practical scenarios with potential model mismatch, using the LCBench hyperparameter tuning benchmark and the NATS neural architecture size search benchmark. We compare the performance of different acquisition function and stopping rule pairs, which we describe next.

Acquisition functions. We consider four common acquisition functions that were discussed in Section 2: *log expected improvement per cost (LogEIPC)*, *Pandora's Box Gittins Index (PBGI)*, *lower confidence bound (LCB)*, and *Thompson sampling (TS)*—chosen for their competitive performance, computational efficiency, and close connections to existing stopping rules.

Baselines. We compare the proposed PBGI/LogEIPC stopping rule against several stopping rules from prior work: *UCB-LCB* [17], *LogEIPC-med* [11], *SRGap-med* [11], and *PRB* [25]. Additionally, we include two simple heuristics used in practice: *convergence* and *GSS*. UCB-LCB stops once the gap between upper and lower confidence bounds falls below a configurable threshold θ, LogEIPC-med stops when the log expected improvement per cost drops beneath $\log(\eta)$ plus the median of its initial T values, SRGap-med stops when the simple-regret gap declines below χ times the median of its initial T values, PRB triggers once a probabilistic regret bound satisfies the regret tolerance ε and confidence δ parameters, convergence stops as soon as the best observed value remains unchanged for w iterations, and GSS stops if the recent improvement is less than $\phi \times \mathrm{IQR}$ over the past w trials where IQR denotes the inter-quartile range of current observations. In addition, we consider *hindsight-optimal*, which picks the true best stopping point in hindsight, as this gives a lower bound on cost-adjusted simple regret.

In our experiments, unless specified otherwise, we follow parameter values recommended in the literature, and set $\theta = 0.01$, $\eta = 0.01$, $\chi = 0.01$, $T = 20$, $\epsilon = 0.05$, $\delta = 0.05$, $w = 5$, and $\phi = 0.01$. Each experiment is repeated with 50 random seeds to assess variability, and we report the mean with error bars (2 times the standard error) for each stopping rule. Each trial (i.e., a run with a distinct random seed) is capped at a fixed number of iterations; if a stopping rule is not triggered within this limit, the stopping time is set to the cap. Details on the iteration cap are provided in Appendix B.

4.1 Bayesian Regret

We first evaluate our cost-aware stopping rule in a controlled synthetic setting, where the ground-truth optima are known. Specifically, we sample objective functions from Gaussian processes with Matérn 5/2 kernels with length scale 0.1, defined over spaces of dimension $d = 1$ and $d = 8$. We consider a variety of evaluation cost structures, including uniform costs, linearly increasing costs in terms of parameter magnitude, and periodic costs that vary non-monotonically over the domain.

In the one-dimensional experiments, we perform exhaustive grid search over $[0, 1]$ to isolate the effect of stopping behavior from numerical optimization. In the 8-dimensional setting, due to instability from higher dimensionality, we apply Polyak averaging [20] with a window size of 20 when computing the stopping condition for the LogEIPC/PBGI rule. See Figure 6 in Appendix B for

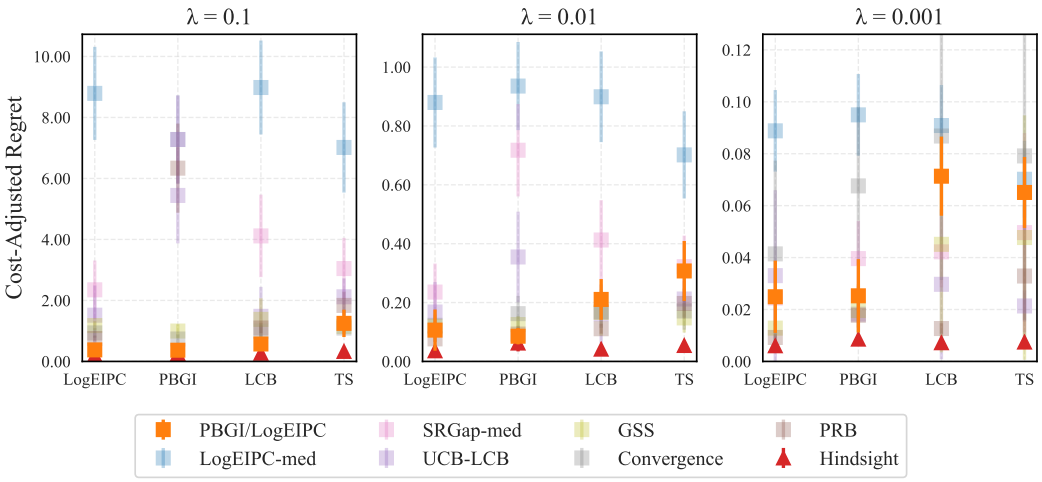


Figure 2: Comparison of cost-adjusted regret of different combinations of acquisition functions and stopping rules in 1-dimensional Bayesian regret experiments. Objective functions are sampled from a Gaussian process with a Matérn-5/2 kernel, using a linear cost function and cost-scaling factors $\lambda = 0.1, 0.01, 0.001$. We see that the matched PBGI/LogEIPC combinations perform best for $\lambda = 0.1$ and 0.01, while slightly suboptimal than various acquisition functions paired with PRB at $\lambda = 0.001$. We also see that matched PBGI consistently achieves a cost-adjusted regret which is close to the hindsight optimal.

an illustration. We omit PRB in the 8D experiment due to its high computational overhead. More details on our experiment setup, and computational considerations are provided in Appendix C.

Figure 2 shows a comparison of cost-adjusted regret for acquisition function + stopping rule pairing with different cost-scaling factors λ in the 1-dimensional setting. As can be observed from the plot, LogEIPC/PBGI stopping rule pairing with LogEIPC or PBGI acquisition function delivers competitive performance for each cost-scaling factor λ, and is particularly strong in handling high cost scenarios—here, those with large λ.

In the 1-dimensional setting, the Bayesian optimization problem is relatively straightforward and all acquisition functions attain nearly identical hindsight optima, independent of cost scale or cost type. In the 8-dimensional experiments, however, clear gaps emerge: Figure 3 compares cost-adjusted regret under uniform, linear, and periodic cost regimes. Under uniform and linear costs, LogEIPC, PBGI, and LCB perform similarly and substantially outperform Thompson Sampling (TS), while in the periodic case, LogEIPC and PBGI outperform LCB and TS. In every regime, combining our stopping rule with LogEIPC, PBGI, or LCB yields regret levels nearly matching the hindsight optimum. This shows the PBGI/LogEIPC stopping rule to be relatively robust against challenges in acquisition function optimization. Further discussion of our experiments across all cost types and cost-scaling factor can be found in Appendix C.

4.2 Empirical Benchmarks: Hyperparameter Optimization and Neural Architecture Search

Benchmarks. We consider two AutoML benchmarks: LCBench [28] and NATS-Bench [7].

LCBench provides training data of 2,000 hyperparameter configurations evaluated on multiple OpenML datasets. In the *unknown-cost* experiments, we consider the full training (at 200 epochs) time as the cost. For the known-cost setting, we observe that training time to scale approximately linearly with the number of model parameters. Based on a linear regression fit (see Appendix B), we adopt $10^{-3} \times p$, where p is the number of model parameters, as a proxy of training time.

NATS-Bench defines a search space of 32,768 neural architectures varying in channel sizes across layers, evaluated on *cifar10-valid*, *cifar100*, and *ImageNet16-120*. As in LCBench, for the known-cost experiments, we approximate the full training + validation time at 90 epochs using $\alpha F + \beta$, where F is the number of FLOPs (floating point operations); see Appendix B for details.

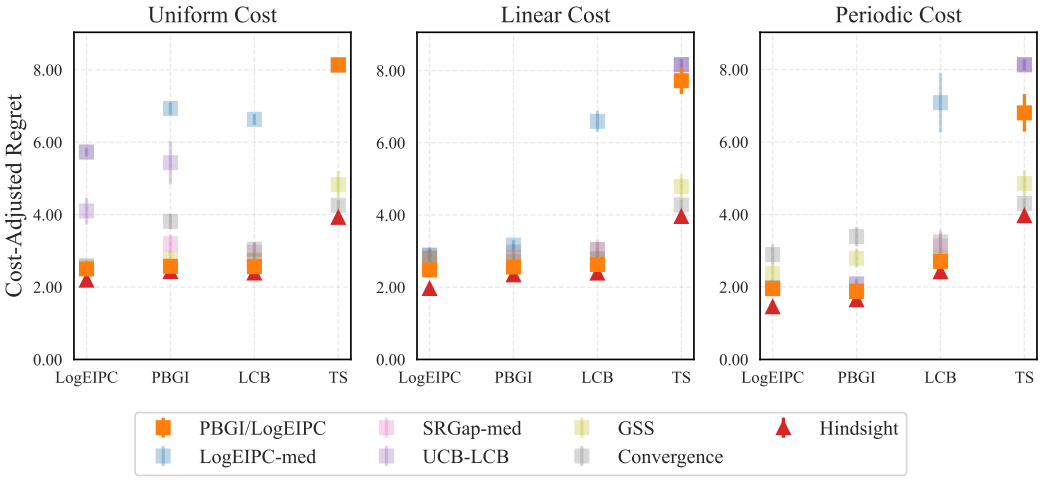


Figure 3: Comparison of cost-adjusted simple regret of different combinations of acquisition functions and stopping rules in 8-dimensional Bayesian regret experiments. Objective functions are sampled from a Gaussian process with a Matérn-5/2 kernel, and three different cost functions scaled by $\lambda = 0.01$. Here, the PBGI/LogEIPC stopping rule consistently yields the lowest cost-adjusted regret that is close to hindsight optimal, when paired with LogEIPC, PBGI, and LCB.

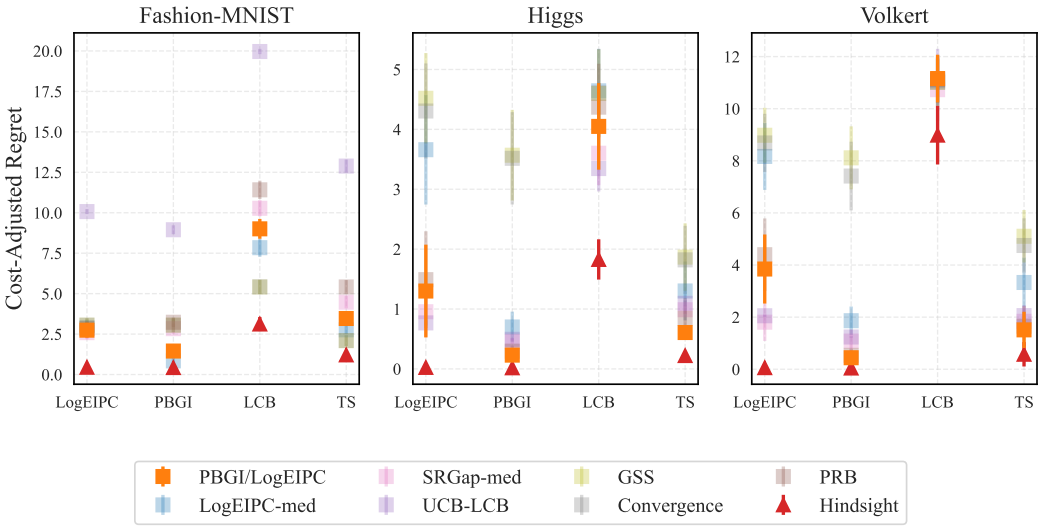


Figure 4: Comparison of cost-adjusted simple regret across different combinations of acquisition functions and stopping rules on LCBench. Objective functions are validation errors on three classification tasks, with proxy runtime as the cost function and a cost-scaling factor of $\lambda = 10^{-4}$. The matched PBGI combination consistently performs best on *higgs* and *volkert*, and second-best on *Fashion-MNIST*, closely approaching the hindsight optimal in all cases. Notably, in addition to PBGI and LogEIPC, Thompson Sampling also performs competitively, and the PBGI/LogEIPC stopping rule remains effective when paired with it.

Metrics. We report the mean and error bars (2 times standard error) of the *cost-adjusted simple regret*, where we consider the evaluation costs to be some cost-scaling factor λ (see Section 3.2) multiplied with cumulative runtime. Simple regret is computed as the difference between (a) test error of the configuration with best validation error at the stopping time and (b) the best test error over all configurations. We present the results using proxy runtime here, deferring the results using actual runtime—an unknown-cost scenario—to Appendix C. Additionally, we report the means and error bars of cumulative runtime and test error pairs at stopping in Appendix C.

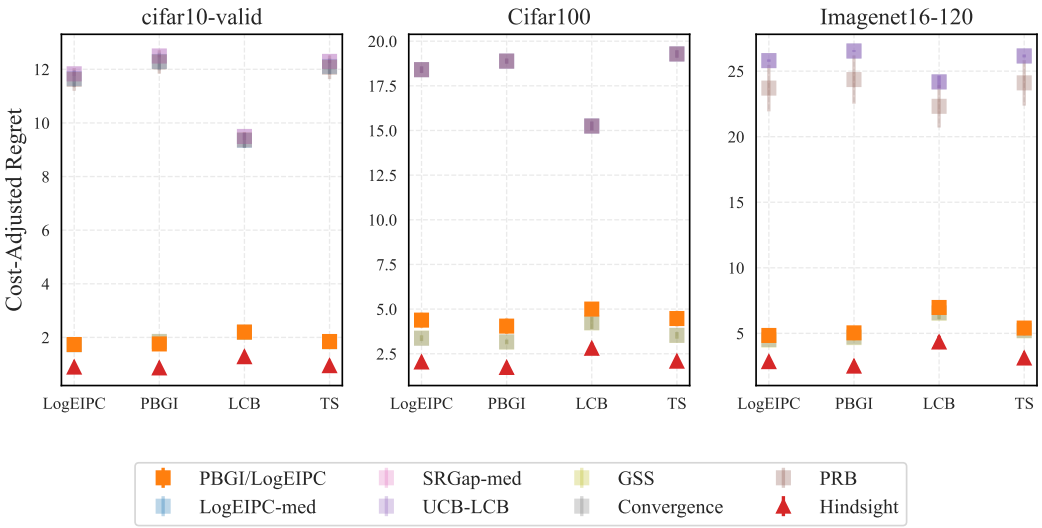


Figure 5: Comparison of cost-adjusted simple regret across different combinations of acquisition functions and stopping rules on NATS-Bench. Objective functions are validation errors on three image classification tasks, with proxy runtime as the cost and a cost-scaling factor of $\lambda = 10^{-4}$. Our stopping rule consistently approaches the hindsight optimal, though it is not always the top performer—GSS and Convergence can achieve lower regret. In contrast, the remaining stopping rules are rarely triggered within the 200-iteration budget. See Table 1 in Appendix C.3 for details.

Experiment results. Figure 4 and Figure 5 present cost-adjusted regret for combinations of acquisition function and stopping rule on classification tasks from LCBench and NATS, with cost-scaling factor $\lambda = 10^{-4}$. Additional results under different cost-scaling factors and the unknown-cost setting are provided in Appendix C, along with visual comparisons between adaptive and fixed-iteration stopping. Overall, our PBGI/LogEIPC stopping rule consistently performs strongly in terms of cost-adjusted simple regret, particularly when paired with the PBGI acquisition function. We also report, in Table 1 of Appendix C, how often each stopping rule fails to trigger within the 200-iteration cap: baselines such as SRGap-med and UCB-LCB often fail to stop early—particularly on NATS—whereas our PBGI/LogEIPC stopping rule reliably stops before the cap.

As we transition from model match to model mismatch, where the true objective function does not align perfectly with the Gaussian process prior, we find that the PBGI/LogEIPC stopping rule, when paired with the PBGI acquisition function, continues to deliver performance closest to the hindsight optimum. However, pairing the same stopping rule with the LogEIPC acquisition function becomes noticeably less competitive on LCBench datasets. This degradation appears to stem from the relative advantage of PBGI over LogEIPC in higher dimensions or misspecified settings: we illustrate this in more detail in Appendix C, which shows that the PBGI acquisition function maintains stronger performance under mis-specification, thus contributing to the improved overall cost-adjusted regret. These results highlight that, while our stopping rule is broadly robust, the choice of acquisition function remains an important consideration when facing model mismatch.

5 Conclusion

We develop the *PBGI/LogEIPC stopping rule* for Bayesian optimization with varying evaluation costs and adaptive stopping. Paired with either the PBGI or LogEIPC acquisition functions, it (a) comes with a theoretical guarantee bounding the expected cumulative cost (Theorem 1), and (b) shows strong empirical performance in terms of cost-adjusted simple regret (Section 4). We believe our framework can be extended to settings involving noisy or multi-fidelity evaluations, batched queries, and alternative objective formulations—for instance, applying a sigmoid transformation to test error rather than a linear one, to reflect real-world considerations such as a user's willingness to adopt models whose performance may change sharply once test error falls below a certain threshold.

Acknowledgments and Disclosure of Funding

QX thanks Nairen Cao for his helpful suggestions, and AT thanks James T. Wilson for helpful comments on stopping rules for Bayesian optimization. LC is funded by the European Union (ERC-2022-SYG-OCEAN-101071601). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them. AT was supported by Cornell University, jointly via the Center for Data Science for Enterprise and Society, the College of Engineering, and the Ann S. Bowers College of Computing and Information Science. PF was supported by AFOSR FA9550-20-1-0351. ZS was supported by the NSF under grant no. CMMI-2307008.

References

[1] Sebastian Ament, Samuel Daulton, David Eriksson, Maximilian Balandat, and Eytan Bakshy. Unexpected improvements to expected improvement for bayesian optimization. *Advances in Neural Information Processing Systems*, 36:20577–20612, 2023.

[2] Raul Astudillo, Daniel Jiang, Maximilian Balandat, Eytan Bakshy, and Peter I Frazier. Multi-step budgeted bayesian optimization with unknown evaluation costs. In *Advances in Neural Information Processing Systems*, 2021.

[3] Eytan Bakshy, Lili Dworkin, Brian Karrer, Konstantin Kashin, Benjamin Letham, Ashwin Murthy, and Shaun Singh. Ae: A domain-agnostic platform for adaptive experimentation. In *Conference on neural information processing systems*, pages 1–8, 2018.

[4] Maximilian Balandat, Brian Karrer, Daniel Jiang, Samuel Daulton, Ben Letham, Andrew G Wilson, and Eytan Bakshy. Botorch: A framework for efficient monte-carlo bayesian optimization. *Advances in neural information processing systems*, 33:21524–21538, 2020.

[5] J Massey Cashore, Lemuel Kumarga, and Peter I Frazier. Multi-step bayesian optimization for one-dimensional feasibility determination. *arXiv preprint arXiv:1607.03195*, 2016.

[6] Stephen E Chick and Peter Frazier. Sequential sampling with economics of selection procedures. *Management Science*, 58(3):550–569, 2012.

[7] Xuanyi Dong, Lu Liu, Katarzyna Musial, and Bogdan Gabrys. Nats-bench: Benchmarking nas algorithms for architecture topology and size. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3634–3646, 2021.

[8] Peter Frazier and Warren B Powell. The knowledge-gradient stopping rule for ranking and selection. In *2008 Winter Simulation Conference*, pages 305–312. IEEE, 2008.

[9] Roman Garnett. *Bayesian optimization*. Cambridge University Press, 2023.

[10] Evangelia Gergatsouli and Christos Tzamos. Weitzman's rule for pandora's box with correlations. *Advances in Neural Information Processing Systems*, 36:12644–12664, 2023.

[11] Hideaki Ishibashi, Masayuki Karasuyama, Ichiro Takeuchi, and Hideitsu Hino. A stopping criterion for bayesian optimization by the gap of expected minimum simple regrets. In *International Conference on Artificial Intelligence and Statistics*, pages 6463–6497. PMLR, 2023.

[12] Robert Kleinberg, Bo Waggoner, and E Glen Weyl. Descending price optimally coordinates search. *arXiv preprint arXiv:1603.07682*, 2016.

[13] Eric Hans Lee, Valerio Perrone, Cedric Archambeau, and Matthias Seeger. Cost-aware bayesian optimization. In *ICML Workshop on Automated Machine Learning*, 2020.

[14] Mikhail Lifshits. Lectures on gaussian processes. In *Lectures on Gaussian Processes*, pages 1–117. Springer, 2012.

[15] Marco Locatelli. Bayesian algorithms for one-dimensional global optimization. *Journal of Global Optimization*, 10:57–76, 1997.

[16] Romy Lorenz, Ricardo P Monti, Ines R Violante, Aldo A Faisal, Christoforos Anagnostopoulos, Robert Leech, and Giovanni Montana. Stopping criteria for boosting automatic experimental design using real-time fmri with bayesian optimization. *arXiv preprint arXiv:1511.07827*, 2015.

[17] Anastasia Makarova, Huibin Shen, Valerio Perrone, Aaron Klein, Jean Baptiste Faddoul, Andreas Krause, Matthias Seeger, and Cedric Archambeau. Automatic termination for hyperparameter optimization. In *International Conference on Automated Machine Learning*, pages 7–1. PMLR, 2022.

[18] Ruben Martinez-Cantin. Bayesian optimization with adaptive kernels for robot control. In *International Conference on Robotics and Automation*, 2017.

[19] Vu Nguyen, Sunil Gupta, Santu Rana, Cheng Li, and Svetha Venkatesh. Regret for expected improvement over the best-observed value and stopping condition. In *Asian conference on machine learning*, pages 279–294. PMLR, 2017.

[20] Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.

[21] Carl Edward Rasmussen and Christopher KI Williams. *Gaussian Processes for Machine Learning*. Cambridge University Press, 2006.

[22] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in Neural Information Processing Systems*, 2012.

[23] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

[24] Martin L Weitzman. Optimal search for the best alternative. *Econometrica*, 1979.

[25] James Wilson. Stopping bayesian optimization with probabilistic regret bounds. *Advances in Neural Information Processing Systems*, 37:98264–98296, 2024.

[26] Qian Xie, Raul Astudillo, Peter Frazier, Ziv Scully, and Alexander Terenin. Cost-aware bayesian optimization via the pandora's box gittins index. *Advances in Neural Information Processing Systems*, 37:115523–115562, 2024.

[27] Yichi Zhang, Daniel W Apley, and Wei Chen. Bayesian optimization for materials design with mixed quantitative and qualitative variables. *Scientific Reports*, 2020.

[28] Lucas Zimmer, Marius Lindauer, and Frank Hutter. Auto-pytorch tabular: Multi-fidelity metalearning for efficient and robust autodl. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(9):3079 – 3090, 2021.

A Theoretical Analysis and Calculations

In this section we provide a restatement involving the cost-scaling factor λ and the proof of Theorem 1, as well as additional discussions on its implication in budget constraint setting.

Theorem 1. *Consider minimizing objective function f, drawn from a stochastic process with constant mean μ, over a compact, non-empty domain X. Let c be a known cost function with finite mean and λ be the cost-scaling factor. Assume the algorithm begins with one initial evaluation at a pre-selected point x_1 in X with cost $C := c(x_1)$, and subsequently runs either the PBGI or LogEIPC acquisition function with the PBGI/LogEIPC stopping rule. Let the stopping time be defined as $\tau := \min_{t\geq 1}\left\{\max_{x\in X} \alpha_t^{\mathrm{LogEIPC}}(x) \leq 0\right\}$. Let $U := \mu - \mathbb{E}[\min_{x\in X} f(x)] < \infty$ denote the upper bound on expected cumulative improvement $\mathbb{E}\left[\sum_{t=2}^{\tau}(y^*_{1:t-1} - y^*_{1:t})\right]$. Then the expected cumulative cost up to stopping is bounded by*

$$\mathbb{E}\left[\sum_{t=1}^{\tau} c(x_t)\right] \leq C + \frac{U}{\lambda}. \tag{15}$$

Proof. We treat the two policies (i.e., the two acquisition function + stopping rule pairs) together and write $\mathcal{F}_t = \sigma(\{x_i, y_i\}_{i=1}^t)$ for the filtration generated by the observations. Here σ refers to σ-algebra. Since we are minimizing, the one–step improvement after iteration t is $(y^*_{1:t-1} - y^*_{1:t})$. Since we initialize at the prior mean, we have $\mathbb{E}[y_1^*] := \mu$ and the quantity $y_1^* - \min_{x\in X} f(x)$ has finite expectation $U < \infty$.

Step 1: A pointwise lower bound on the expected improvement before stopping. Denote the posterior expected improvement function as $\alpha_t^{\mathrm{EI}}(x) := \mathrm{EI}_{f|x_{1:t},y_{1:t}}(x; y^*_{1:t})$. While stopping hasn't occurred ($t < \tau$), by the stopping criteria definition, $\max_{x\in X} \alpha_t^{\mathrm{EI}}(x)/c(x) \geq \lambda$. Hence there exists at least one point with $\alpha_t^{\mathrm{EI}}(x)/c(x) \geq \lambda$.

Policy (A) PBGI. For each x we defined a threshold $\alpha_t^{\mathrm{PBGI}}(x)$ by $\mathrm{EI}_{f|x_{1:t},y_{1:t}}\left(x; \alpha_t^{\mathrm{PBGI}}(x)\right) = \lambda c(x)$. Since $\mathrm{EI}_{f|x_{1:t},y_{1:t}}$ is increasing in its first argument,

$$y^*_{1:t} \;\geq\; \alpha_t^{\mathrm{PBGI}}(x) \iff \alpha_t^{\mathrm{EI}}(x)/c(x) \;\geq\; \lambda.$$

The existence of a point with ratio at least λ therefore implies that the set $\mathcal{S}_t := \{x : y^*_{1:t-1} \geq \alpha_t^{\mathrm{PBGI}}(x)\}$ is non-empty. PBGI chooses x_t with the *smallest* threshold, and thus

$$\alpha_t^{\mathrm{EI}}(x_t) = \mathrm{EI}_{f|x_{1:t},y_{1:t}}(x; y^*_{1:t}) \geq \mathrm{EI}_{f|x_{1:t},y_{1:t}}\left(x; \alpha_t^{\mathrm{PBGI}}(x)\right) = \lambda\, c(x_t). \tag{16}$$

Policy (B) (Log)EIPC. By definition x_t maximizes $\log\left(\alpha_t^{\mathrm{EI}}(x)/c(x)\right)$, and thus $\alpha_t^{\mathrm{EI}}(x_t) \geq \lambda\, c(x_t)$.

Thus for *both* policies,

$$\alpha_t^{\mathrm{EI}}(x_t) \;\geq\; \lambda\, c(x_t), \qquad \text{for all } t \leq \tau. \tag{17}$$

Step 2: The total expected improvement is at most U. Set $\Delta_t := y^*_{1:t-1} - y^*_{1:t} \geq 0$ for $t \leq \tau$. Conditional on $\mathcal{F}_{t-1}$ and the choice of x_t,

$$\mathbb{E}[\Delta_t \mid \mathcal{F}_{t-1}, x_t] = \alpha_t^{\mathrm{EI}}(x_t).$$

Taking expectations and summing up to τ,

$$\mathbb{E}\left[\sum_{t=2}^{\tau} \alpha_t^{\mathrm{EI}}(x_t)\right] = \mathbb{E}\left[\sum_{t=2}^{\tau} \Delta_t\right] = \mathbb{E}\left[y_1^* - y^*_{1:\tau}\right].$$

Since we always have $y^*_{1:\tau} \geq \min_{x\in X} f(x)$, then

$$\mathbb{E}\left[y_1^* - y^*_{1:\tau}\right] \leq \mu - \mathbb{E}\left[\min_{x\in X} f(x)\right] = U.$$

Step 3: Expected cost control. Summing (17) over $t \leq \tau$, taking expectation, and applying the result of Step 2, we have

$$\lambda \mathbb{E}\left[\sum_{t=2}^{\tau} c(x_t)\right] \leq \mathbb{E}\left[\sum_{t=2}^{\tau} \alpha_t^{\mathrm{EI}}(x_t)\right] \leq U.$$

We conclude that

$$\mathbb{E}\left[\sum_{t=1}^{\tau} c(x_t)\right] = \mathbb{E}[c(x_1)] + \mathbb{E}\left[\sum_{t=2}^{\tau} c(x_t)\right] \leq C + \frac{U}{\lambda},$$

which completes the proof. □

Extension to other acquisition functions. From the proof, we see that this bound extends to any acquisition function that satisfies the condition (17). That is, the result holds more generally to acquisition functions beyond PBGI and LogEIPC, as long as each selected point ensures sufficient expected improvement relative to its cost before stopping at time τ.

Implications for budget-constrained settings and PBGI-D. Suppose the user specifies a fixed budget B. Theorem 1 suggests that setting cost-scaling factor $\lambda = U/B$ when using either PBGI or LogEIPC policy ensures that the expected cumulative cost is always within the budget. If this choice of λ proves conservative in practice—resulting in lower-than-anticipated spending—it can be paired with the PBGI-D rule introduced by Xie et al. [26], which starts with $\lambda = \lambda_0$ and reduces λ by half each time stopping is triggered. Using $\lambda_0 = U/B$ as the initial λ value ensures that the initial fixed-λ phase of PBGI-D aligns with the budget in expectation, and provides a principled initial value λ_0 before adaptive decay starts.

B Experimental Setup

All experiments are implemented based on BoTorch [4]. Each Bayesian optimization procedure is initialized with $2(d+1)$ random samples, where d is the dimension of the search domain. For Bayesian regret experiments, we follow the standard practice to generate the initial random samples using a quasirandom Sobol sequence. For empirical experiments, we randomly sample configuration IDs from a fixed pool of candidates—2,000 for LCBench and 32,768 for NATS-Bench. All computations are performed on CPU.

Each experiment is repeated with 50 random seeds, and we report the mean with error bars (2 times the standard error) for each stopping rule. We also impose a cap on the number of iterations: 100 for 1D Bayesian regret, 500 for 8D Bayesian regret, and 200 for empirical experiments. If a stopping rule is not triggered before reaching this cap, we treat the stopping time as equal to the cap.

Gaussian process models. For all experiments, we follow the standard practice to apply Matérn kernels with smoothness $5/2$ and length scales learned from data via maximum marginal likelihood optimization, and standardize input variables to be in $[0, 1]^d$. For empirical experiments, we standardize output variables to be zero-mean and unit-variance, but not for Bayesian regret experiments. In this work, we consider the noiseless setting and set the fixed noise level to be 10^{-6}. In the unknown-cost experiments, we follow Astudillo et al. [2] to model the objective and the logarithm of the cost function using independent Gaussian processes.

Acquisition function optimization. For the 1D Bayesian regret experiments, we optimize over 10,001 grid points. For the 8D Bayesian regret experiment, we use BoTorch's 'gen_candidates_torch', a gradient-based optimizer for continuous acquisition function maximization. For the empirical experiments, since LCBench and NATS-Bench provide only 2,000 and 32,768 configurations respectively, we optimize the acquisition function by simply applying an argmin/argmax over the acquisition values of the unevaluated configurations, without using any gradient-based methods.

Acquisition function and stopping rule parameters. For PBGI, we follow Xie et al. [26] to compute the Gittins indices using 100 iterations of bisection search without any early stopping or other performance and reliability optimizations.

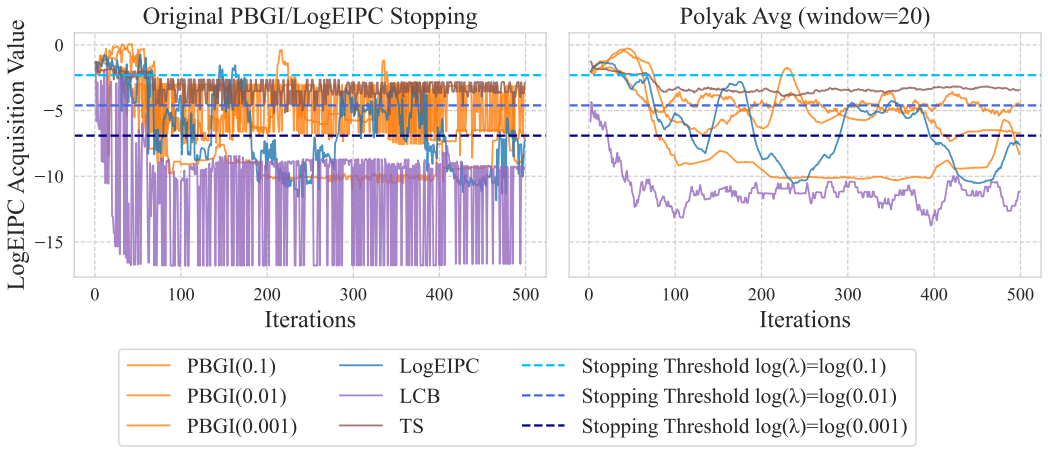


Figure 6: Comparison of the raw and Polyak-averaged PBGI/LogEIPC stopping rule statistic (equal to the LogEIPC acquisition value) in synthetic 8D experiments for multiple acquisition functions, with linear cost and stopping thresholds at $\log(0.1)$, $\log(0.01)$ and $\log(0.001)$. *Left:* The unaveraged statistic exhibits large, high-frequency fluctuations due to the difficulty of acquisition optimization in high dimensions. *Right:* Applying Polyak averaging (window=20) smooths these wiggles, yielding a more stable stopping signal.

For UCB/LCB-based acquisition functions and stopping rules, we follow the original GP-UCB paper Srinivas et al. [23] and the choice in UCB/LCB based stopping rules [17, 11] to use the schedule $\beta_t = 2\log(dt^2\pi^2/6\delta)$, where d is the dimension. We also adopt their choice of $\delta = 10^{-1}$ and a scale-down factor of 5.

For SRGap-med, which stops when the simple regret gap falls below χ times the median of its initial $T = 20$ values, we set $\chi = 0.1$ in the empirical experiments, instead of the default value $\chi = 0.01$ recommended in the literature. This adjustment is made because SRGap-med tends to stop too late on the LCBench datasets, likely due to the relatively small initial regret values and insignificant drop over time.

For PRB, we follow Wilson [25] to use the schedule $N_t = \max(\lceil 64 * 1.5^{t-1} \rceil, 1000)$ for number of posterior samples, risk tolerance $\delta = 0.05$. The error bound ϵ is set to be 0.1 for Bayesian regret experiments and 0.5% of the best test error (i.e., mis-classification rate) among all configurations for the empirical experiments.

For the 8D Bayesian regret experiments, we apply Polyak averaging [20] over 20 iterations to mitigate oscillations in the optimizer. Figure 6 illustrates the challenges these oscillations pose when computing stopping rule statistics and shows the improvement with Polyak averaging. For consistency, we also apply 20-iteration averaging to the GSS and Convergence baselines.

Omitted baselines. We omit the PRB baseline for the 8D Bayesian regret experiment due to its high computational cost — requiring up to global optimum computations of 1000 sample paths per iteration. We also omit the KG acquisition function and stopping rule for the same reason, as they are shown to be computationally intensive in the runtime experiments of Xie et al. [26].

Objective functions: Bayesian regret. In all Bayesian regret experiments, each objective function f is sampled from a Gaussian process prior with a Matérn 5/2 kernel and a lengthscale of 0.1, using a different seed.

Objective functions: empirical. In the empirical experiments, the validation error (scaled out of 100) is used as the objective function during the Bayesian optimization procedure, while the cost-adjusted simple regret is reported based on the corresponding test error.

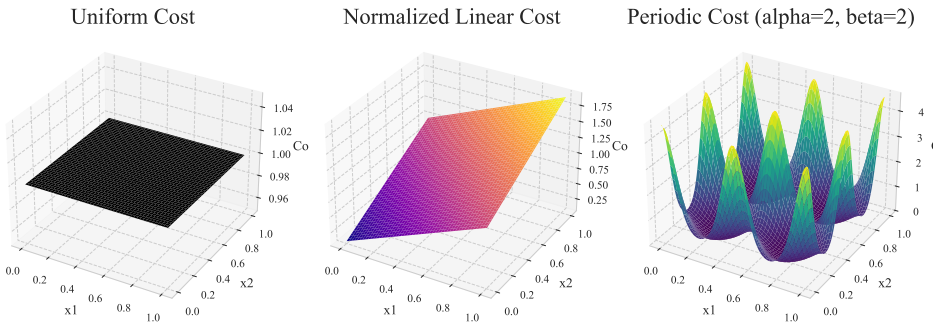


Figure 7: Surface plots of cost functions over $[0, 1]^2$: (Left) Uniform cost of 1 across domain. (Middle) Normalized linear cost increasing with the mean of x_1 and x_2. (Right) Periodic cost with $\alpha = 2$, $\beta = 2$, normalized by Bessel-based factor.

Cost functions: Bayesian regret. In Bayesian regret experiments, we consider three types of evaluation costs: uniform, linear, and periodic. These costs are normalized such that $\mathbb{E}_{x\in[0,1]^d}[c(x)]$ is approximately 1 and their expressions (prior to cost scaling) are given below.

In the *uniform* cost setting, each evaluation incurs a constant cost of 1.

In the *linear* cost setting, the cost increases proportionally with the average coordinate value of the input:

$$\text{linear\_cost}(x) = \frac{1 + 20 \cdot \left(\frac{1}{d}\sum_{i=1}^{d} x_i\right)}{11}.$$

In the *periodic* cost setting, the evaluation cost fluctuates across the domain. Following [2], we define the periodic cost as

$$\text{periodic\_cost}(x) = \frac{\exp\left(\frac{\alpha}{d}\sum_{i=1}^{d}\cos\left(2\pi\beta(x_i - x_i^*)\right)\right)}{\left[I_0\left(\frac{\alpha}{d}\right)\right]^d},$$

where x_i^* denotes the coordinate of the global optimum of f, and I_0 is the modified Bessel function of the first kind, which acts as a normalization constant. We set $\alpha = 2$ and $\beta = 2$ to induce noticeable variation in cost across the domain, while ensuring that costly evaluations can still be worthwhile.

A visualization of the three cost functions is provided in Figure 7.

Cost functions: empirical. In the *unknown-cost* experiments, we treat runtime, i.e., the provided full model training time (200 epochs for LCBench and 90 epochs for NATS) as evaluation costs (prior to cost-scaling).

In the *known-cost* experiments, for LCBench, we use 0.001 times the number of model parameters as a proxy for runtime. This proxy is motivated by our observation of an approximately linear relationship between the number of model parameters and the actual runtime, with slope close to 0.001 (see Figure 8). Importantly, the number of model parameters can be computed in advance, before the Bayesian optimization procedure, based on the network structure and classification task, as we explain in detail below.

In a feedforward neural network like shapedmlpnet with shape 'funnel', the model parameters are determined by input size (number of features), output size (e.g., number of classes), number of layers, size of each layer. The input size and output size are given by:

- Fashion-MNIST:
  - Input dimension: 784 (each image has 28×28 pixels, flattened into a vector of length 784)
  - Number of Output Features (output_feat): 10 (corresponding to 10 clothing categories)
- Adult:

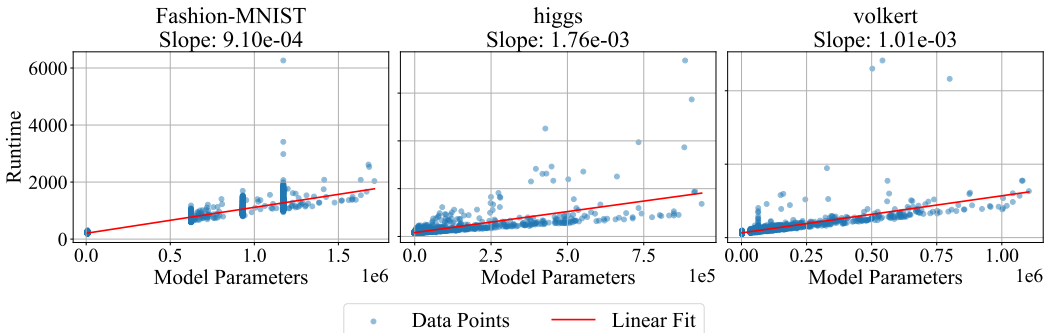


Figure 8: Empirical relationship between the number of model parameters and runtime for three LCBench datasets. Each subplot shows a scatter plot of actual runtime (y-axis) against number of model parameters (x-axis), along with a fitted linear regression line. The observed linear trend supports using 0.001 times the number of model parameters as a proxy for runtime. For Fashion-MNIST and volkert, the fitted slopes are close to 0.001. The slope for higgs is slightly higher, possibly due to a few outliers.

  - Input Dimension: 14 (the dataset comprises 14 features, including both numerical and categorical attributes)
  - Number of Output Features: 1 (binary classification: income $> 50K$ or $\leq 50K$)
- Higgs:
  - Input Dimension: 28 (each instance has 28 numerical features)
  - Number of Output Features: 1 (binary classification: signal or background process)
- Volkert:
  - Input Dimension: 147 (the dataset consists of 147 features)
  - Number of Output Features: 10 (corresponding to 10 classes)

The number of layers (num_layers) and size of each layer (max_unit) can be obtained from the configuration. With these information, we can compute the total number of model parameters (weights and biases) based on the layer-wise structure as follows:

$$\text{layer\_params}_{i \to i+1} = \text{layer}_i \cdot \text{layer}_{i+1} + \text{layer}_{i+1} \quad \text{(weights + biases)} \tag{18}$$

$$\text{model\_params} = \sum_{i=0}^{L-1} \text{layer\_params}_{i \to i+1} \tag{19}$$

$$\text{where} \quad \text{layer}_0 = \text{input\_dim}, \quad \text{layer}_L = \text{output\_feat} \tag{20}$$

Similarly for NATS-Bench, we use $\alpha \times \#\text{FLOPs} + \beta$ as a proxy for runtime (see Figure 9 for a visualization of the linear relationship), where the number of FLOPs (floating point operations) can also be computed in advance, as it is determined solely by the architecture's structure and the fixed input shape. Specifically, for *cifar10-valid*, we set $\alpha = 1$, $\beta = 400$; for *cifar100*, we set $\alpha = 2$, $\beta = 550$; and for *ImageNet16-120*, we set $\alpha = 1$, $\beta = 1000$.

Specifically, FLOPs are precomputed and stored for each architecture in NATS-Bench. Since each architecture corresponds to a deterministic computational graph and all inputs (e.g., CIFAR-10 images) have a fixed shape, the FLOPs required for a forward pass can be calculated analytically—without executing the model on data.

C Additional Experiment Results

In this section, we present additional experimental results to further evaluate the performance of different acquisition-function–stopping-rule combinations across various settings. We also include alternative visualizations to aid interpretation of the results.

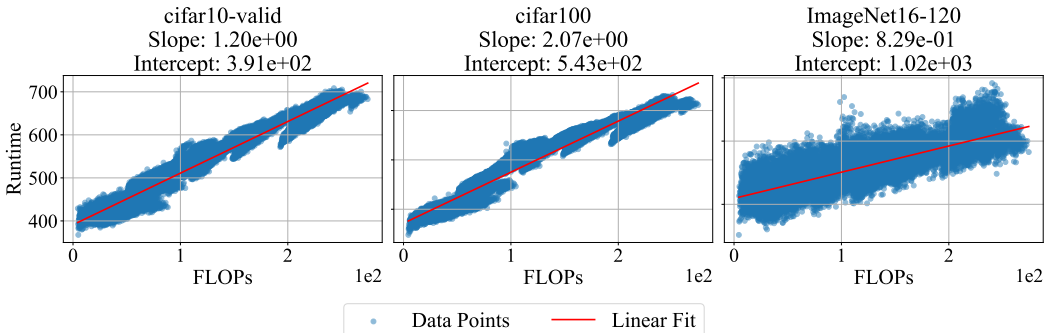


Figure 9: Empirical relationship between the number of FLOPs and runtime for three NATS-Bench datasets. Each subplot shows a scatter plot of actual runtime (y-axis) against number of FLOPs (x-axis), along with a fitted linear regression line. The observed linear trend supports using $\alpha \times \#\text{FLOPs} + \beta$ as a proxy for runtime.

C.1 Runtime Comparison

First, we compare the runtime between our PBGI/LogEIPC stopping rule with several baselines. We measure the CPU time (in seconds) of the computation of the stopping rule, excluding the acquisition function computation and optimization.

From results in Figure 10 we can see that our PBGI/LogEIPC stopping rule is as efficient as SRGap-med and UCB-LCB. In contrast, PRB is significantly more time-consuming, as it involves optimizing over up to 1000 samples.

C.2 Order of Stopping and Posterior Updates

Following the discussions in Section 3, we always compute our proposed stopping rule with respect to the optimal acquisition function value of the *next round*—namely, the one which is obtained after posterior updates have been performed. One could alternatively consider checking the stopping criteria *before* posterior updates, which is backward-looking rather than forward-looking. Figure 11 provides an empirical comparison between the two choices, showing that stopping *after* the posterior update leads to stronger empirical performance.

This suggests that the theoretical guarantee for the Gittins index policy in the correlated Pandora's Box setting by Gergatsouli and Tzamos [10], which is based on the *before-posterior-update* stopping, could potentially be improved by adopting the *after-posterior-update* stopping.

C.3 Additional Experiment Results: Empirical

This subsection presents additional experiment results on hyperparameter optimization over the three LCBench datasets and neural architecture size search over the three NATS datasets.

Number of trials where stopping fails. We count the number of trials in which a stopping rule fails to trigger within our iteration cap of 200 and present the results in Table 1. From the table, we observe that on datasets from the NATS benchmark, regret-based and acquisition-based stopping rules—except for ours—often fail to stop early. On LCBench datasets, some regret-based stopping rules such as SRGap-med and UCB-LCB also frequently exceed the cap. In contrast, our stopping rule consistently stops early, which aligns with our theoretical result in Theorem 1.

Alternative visualization: cost-adjusted regret vs iteration. We provide an alternative visualization of cost-adjusted simple regret by plotting its mean and error bars at fixed iterations, along with the mean and error bars of the stopping iterations for each rule. This allows us to compare the performance of adaptive stopping rules not only against the hindsight-optimal adaptive stopping but also against the hindsight-optimal fixed-iteration stopping.

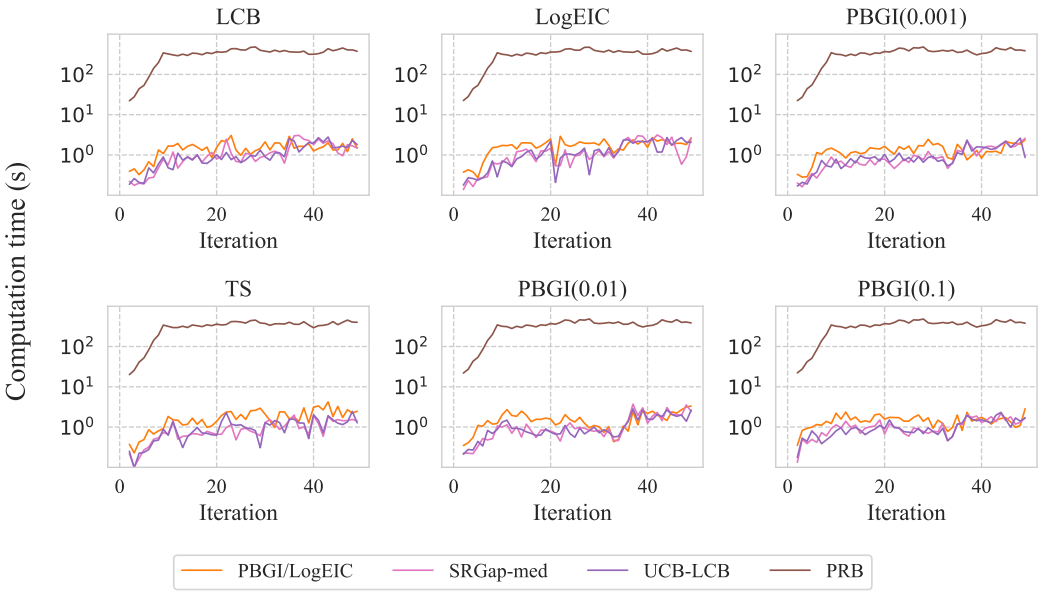


Figure 10: Evolution of per-iteration computation time (in log scale) for different stopping stopping rules when paired with six acquisition policies on the 8-dimensional Bayesian regret benchmark. Each subplot shows the average runtime (in seconds) over 50 iterations under one acquisition function—LCB, Thompson Sampling, LogEIPC, PBGI($\lambda = 10^{-1}$), PBGI($\lambda = 10^{-2}$), and PBGI($\lambda = 10^{-3}$). Curves correspond to four stopping criteria: our PBGI/LogEIPC stopping rule, SRGap-med, UCB–LCB, and the probabilistic regret bound (PRB). Convergence and GSS can be applied using only the best observed value and thus require no additional computation time, thus they are omitted here. LogEIPC-med relies on the same underlying statistical computations as the LogEIPC rule, and therefore its runtime is not measured separately. The results should that PRB incurs significant computational overhead compared to other stopping rules.

Table 1: Number of trials (out of 50) where each stopping rule failed to trigger within 200 iterations, for each dataset in the LCBench (first three) and NATS (last three) benchmarks and each acquisition function. Results are identical across acquisition functions.

Dataset	**PBGI**	**LogEIPC-med**	**SRGap-med**	**UCB-LCB**	**GSS**	**Convergence**	**PRB**
Fashion-MNIST	0	0	47	50	0	0	0
higgs	0	0	50	50	0	0	0
volkert	0	0	50	50	0	0	0
Cifar10	0	48	50	50	0	0	49
Cifar100	0	50	50	50	0	0	50
ImageNet	0	50	50	50	0	0	45

As shown in the empirical setting in Figure 12-14, cost-adjusted regret generally decreases in the early iterations and then increases. The turning point is exactly the hindsight-optimal fixed-iteration stopping point, and our PBGI/LogEIPC stopping rule consistently performs close to this optimum, particularly when paired with the PBGI acquisition function.

Cost model mismatch: proxy runtime vs actual runtime. In the known-cost setting of hyperparameter tuning, a practical approach is to use a proxy for runtime as the evaluation cost during the Bayesian optimization procedure. In our case, we use the number of model parameters scaled by a constant factor, which can be known in advance and has been shown to correlate well with the actual runtime. However, for reporting performance, one may prefer to use the actual runtime to better reflect real-world cost. To assess the impact of this cost model mismatch, we compare the cost-adjusted simple regret obtained when evaluation costs are computed using either the proxy runtime or the actual runtime. As shown in Figure 15 and Figure 16, our PBGI/LogEIPC stopping

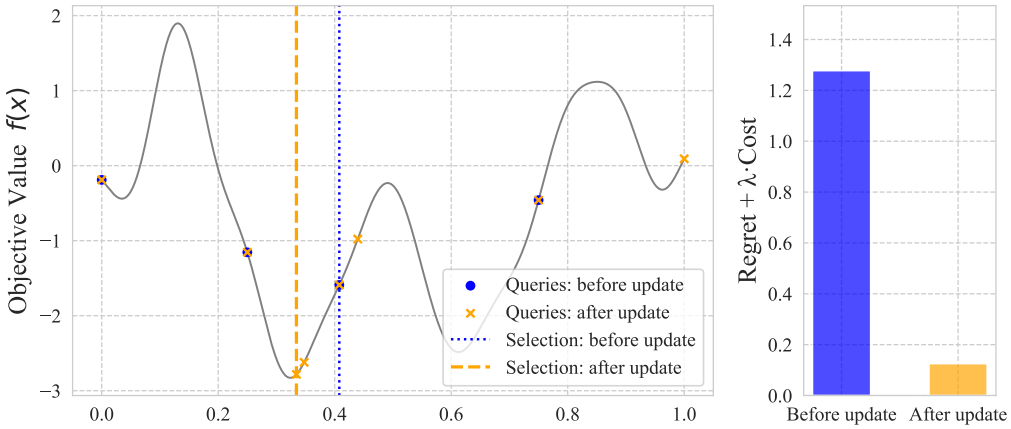


Figure 11: Illustration of a single draw from a Matérn-$5/2$ Gaussian process on $[0, 1]$ with lengthscale 0.1, optimized using PBGI acquisition function under uniform cost and cost-scaling factor $\lambda = 0.01$. We compare two variants of PBGI stopping rules: the *before-posterior-update* (this-round) stopping rule and the *after-posterior-update* (next-round) stopping rule. **Left:** The latent objective function (solid gray) and evaluation sequences for *before-posterior-update* stopping (blue circles) and *after-posterior-update* stopping (orange crosses). The dotted blue line and the dashed orange line mark the best observed value under each respective rule. **Right:** Cost-adjusted regret for each stopping rule. In this example, *after-posterior-update* stopping achieves strictly lower cost-adjusted regret despite performing more evaluations.

rule remains close to the hindsight optimal even when there is a mismatch, although its ranking may shift slightly (e.g., from best to second-best on the *higgs* dataset).

Unknown-cost. As discussed in Section 3.2, there are two variants of the (Log)EIPC acquisition functions and corresponding stopping rules under the unknown-cost setting. We refer to the variant using (13) as (Log)EIPC-inv, and the one using (14) as (Log)EIPC-exp. Recall that while (Log)EIPC-exp is equivalent to PBGI as a stopping rule (but not as an acquisition function), (Log)EIPC-inv differs from both in terms of both the acquisition function and the stopping rule. As shown in Figure 17, the results are similar to those in the known-cost setting.

C.4 Additional Experiment Results: Bayesian Regret

In this section, we present the complete synthetic results. Figures 18 to 20 show the 1D experiments, and Figures 21 to 23 show the 8D experiments. Each figure corresponds to one cost setting (uniform, linear or periodic) and three values of the cost-scaling factor, $\lambda \in 10^{-1}, 10^{-2}, 10^{-3}$. In all of the experimental results, we observe that PBGI/LogEIPC acquisition function + PBGI/LogEIPC stopping achieves cost-adjusted regret that is not only competitive with the baselines, but is also competitive regarding the *best in hindsight* fixed iteration stopping and often competitive even comparing to *hindsight optimal* stopping. These results indicate that our automatic stopping rule can replace manual selection of stopping times without loss in performance. Figures 18 to 23 also show that our PBGI/LogEIPC stopping rule outperforms other baselines when the cost-scaling factor λ is large, indicating that it's an especially suitable stopping criteria when evaluation is expensive.

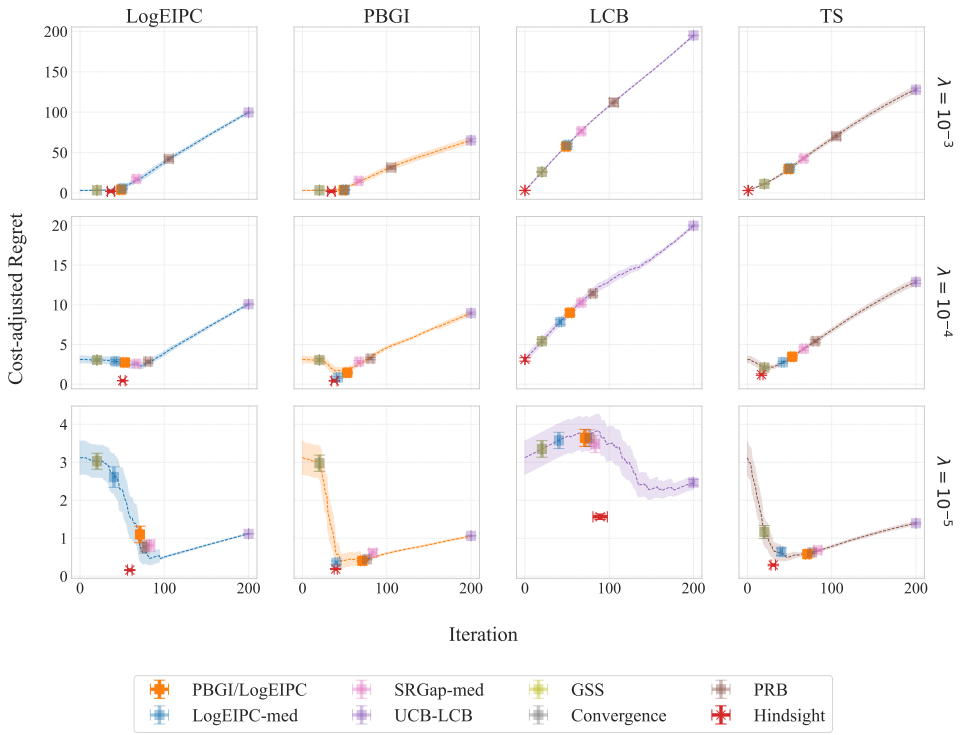


Figure 12: Comparison of cost-adjusted simple regret of different combinations of acquisition functions and stopping rules on the *Fashion-MNIST* dataset. The objective function is the validation error, and the evaluation cost is the proxy runtime, scaled by three different cost-scaling factors $\lambda = 10^{-3}, 10^{-4}, 10^{-5}$. We can see that the PBGI/LogEIPC stopping rule consistently achieves cost-adjusted regret close to the hindsight optimal adaptive stopping as well as the hindsight optimal fixed-iteration stopping when paired with the PBGI acquisition function, though not always the best.

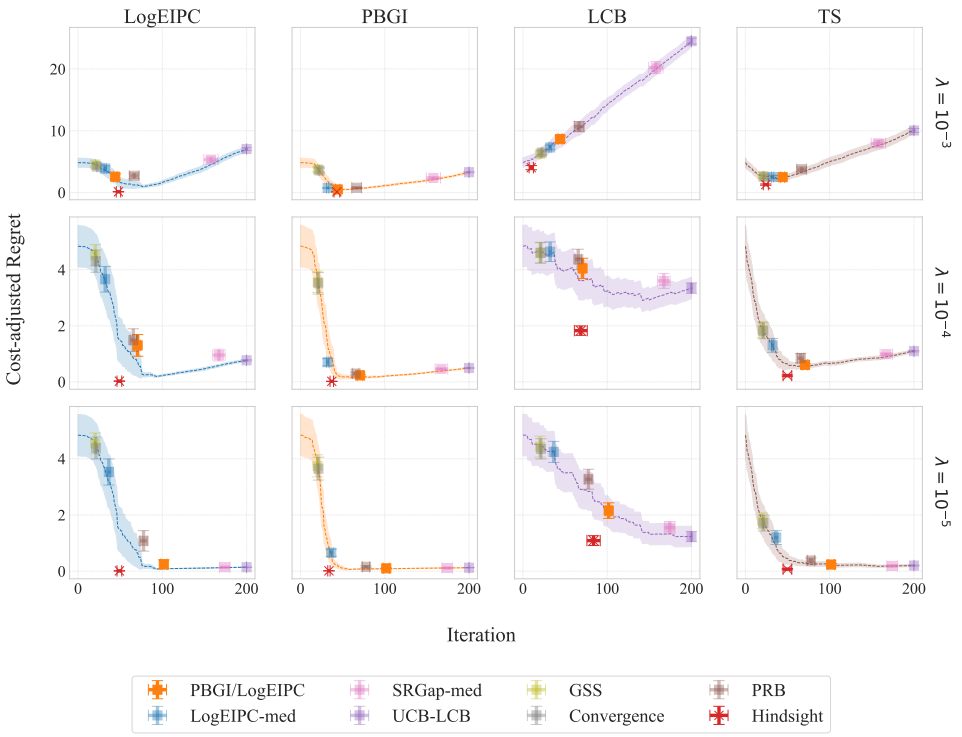


Figure 13: Comparison of cost-adjusted simple regret of different combinations of acquisition functions and stopping rules on the *higgs* dataset. The objective function is the validation error, and the evaluation cost is the proxy runtime, scaled by three different cost-scaling factors $\lambda = 10^{-3}, 10^{-4}, 10^{-5}$. We can see that the PBGI/LogEIPC stopping rule consistently achieves the best cost-adjusted regret when paired with the LogEIPC, PBGI, or TS acquisition function, particularly with PBGI. These combinations not only approach the hindsight optimal adaptive stopping but also perform comparably to the hindsight optimal fixed-iteration stopping.

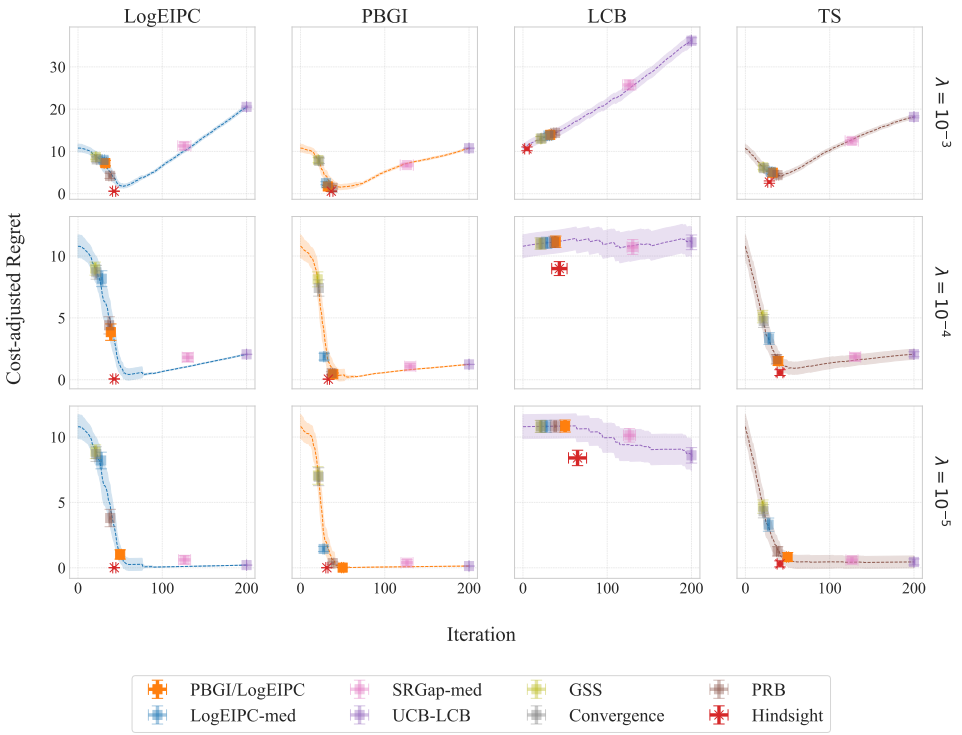


Figure 14: Comparison of cost-adjusted simple regret of different combinations of acquisition functions and stopping rules on the *volkert* dataset. The objective function is the validation error, and the evaluation cost is the proxy runtime, scaled by three different cost-scaling factors $\lambda = 10^{-3}, 10^{-4}, 10^{-5}$. We can see that the PBGI/LogEIPC stopping rule consistently achieves the cost-adjusted regret close to hindsight optimal adaptive stopping and hindsight optimal fixed-iteration stopping when paired with the PBGI or TS acquisition function, particularly with PBGI.

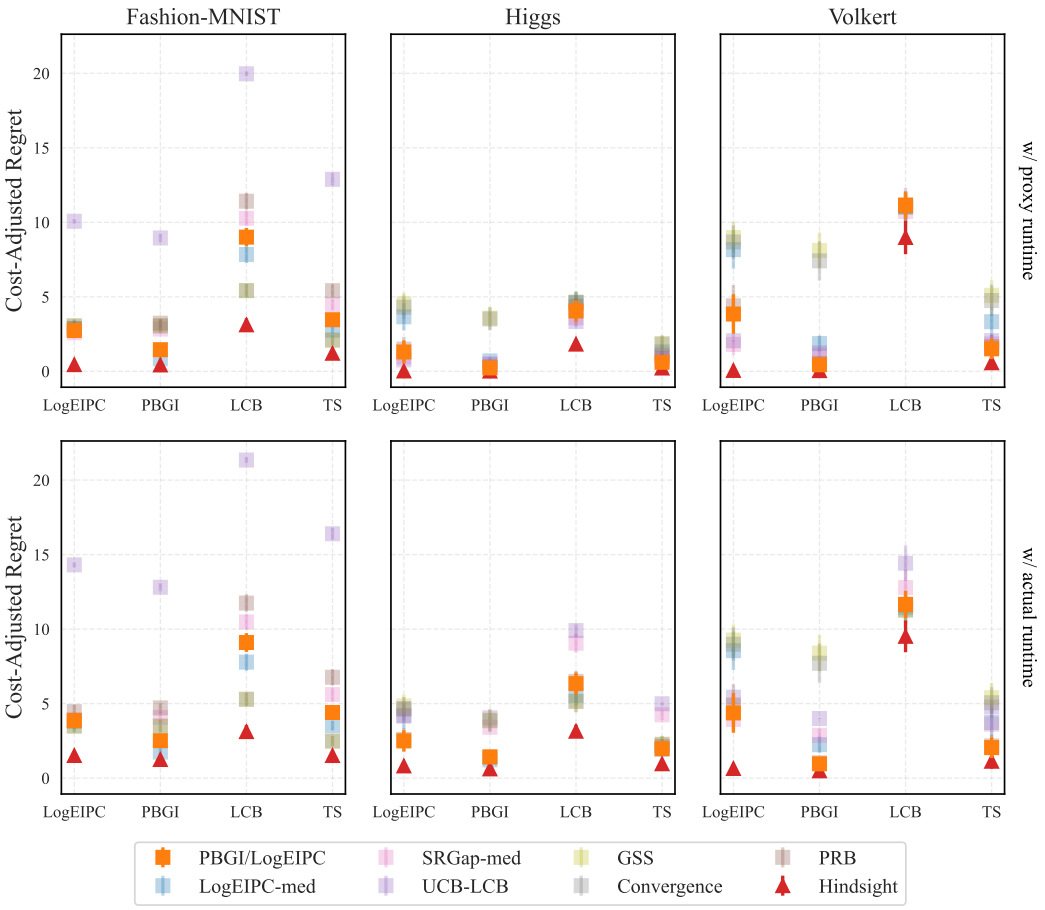


Figure 15: Comparison of cost-adjusted simple regret when using scaled proxy runtime versus scaled actual runtime as the evaluation cost on LCBench. While our PBGI/LogEIPC stopping rule performs slightly worse under actual runtime (e.g., dropping from best to second-best on the *higgs* dataset), likely due to cost model mismatch, it remains close to the hindsight optimal in all cases.

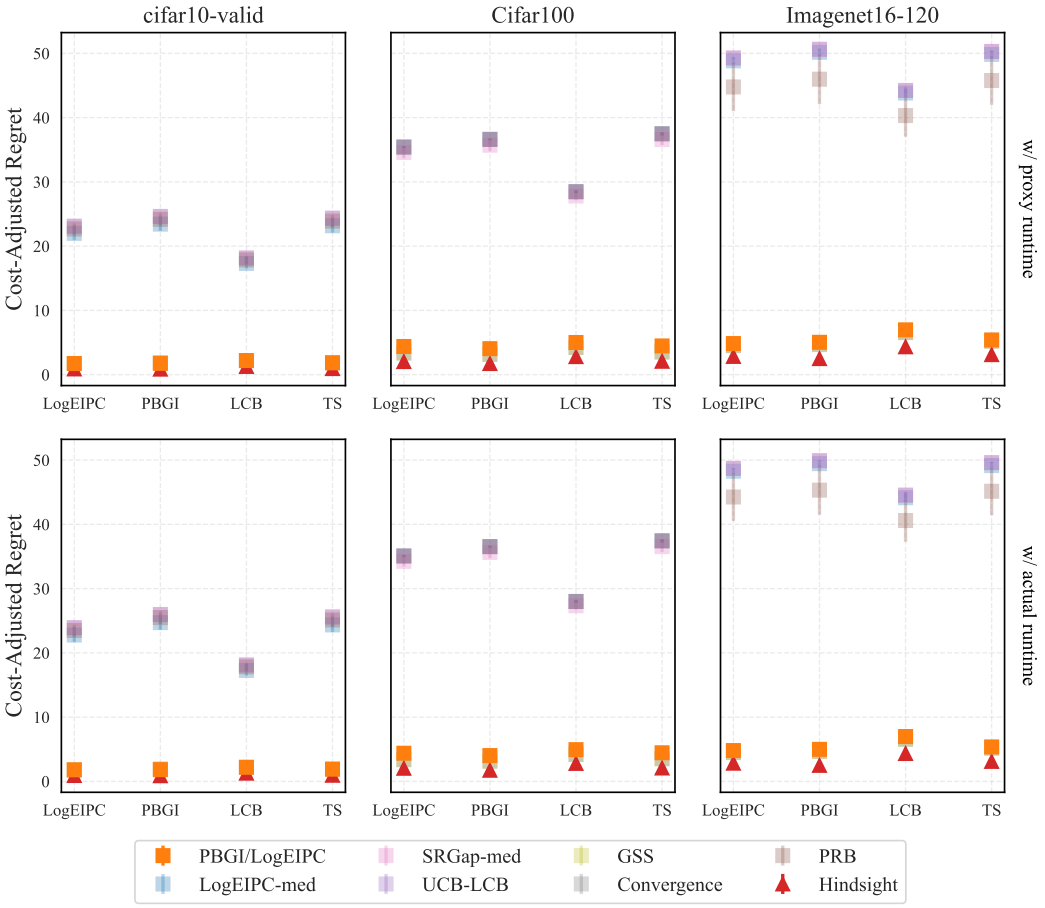


Figure 16: Comparison of cost-adjusted simple regret when using scaled proxy runtime versus scaled actual runtime as the evaluation cost on NATS-Bench. The results are nearly identical to the cost-model-match setting.

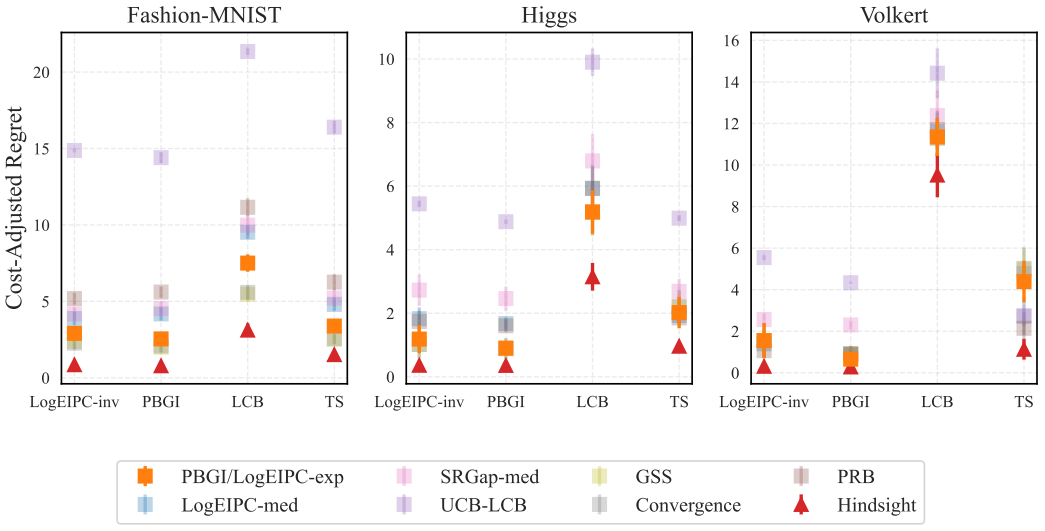


Figure 17: Cost-adjusted regret under the unknown-cost setting. The results are similar to those in the known-cost setting.

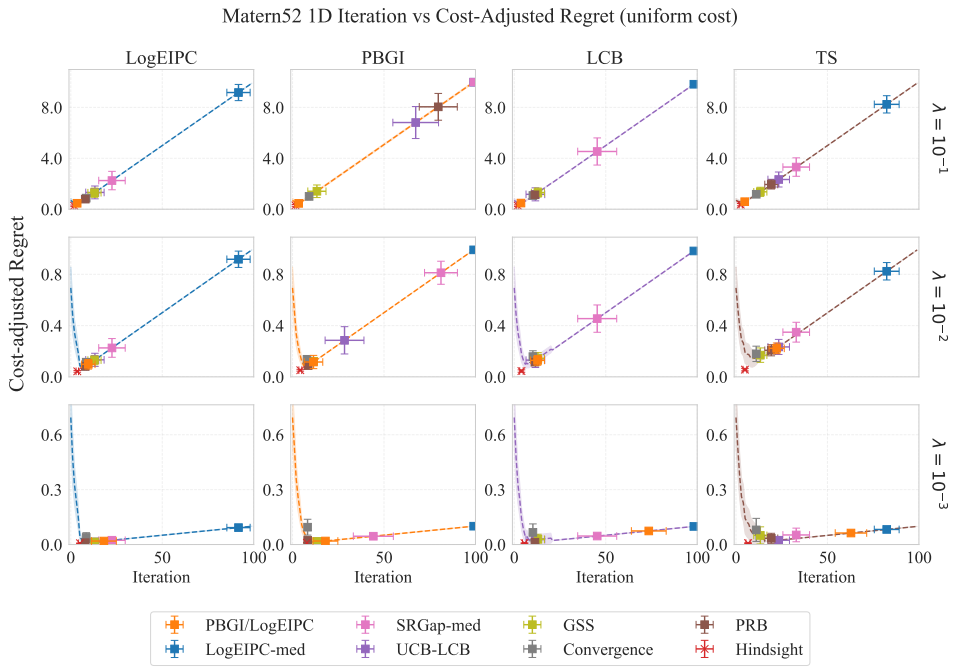


Figure 18: Comparison of cost-adjusted simple regret of different combinations of acquisition functions and stopping rules in the 1D Synthetic experiments, with cost-scaling factor $\lambda = 10^{1}, 10^{-2}, 10^{-3}$ and under uniform cost. The dashed line in each subplot represent fixed iteration stopping (e.g., the y-axis value of the line at iteration 50 represent the cost-adjusted regret when always stopping at iteration 50).

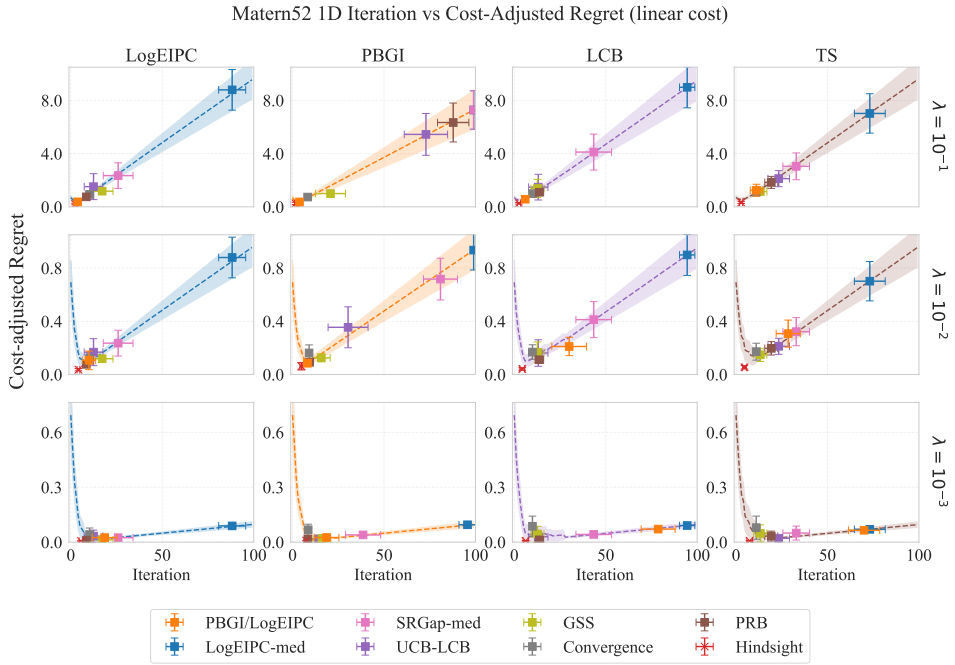


Figure 19: Comparison of cost-adjusted simple regret of different combinations of acquisition functions and stopping rules in the 1D Synthetic experiments, with cost-scaling factor $\lambda = 10^{1}, 10^{-2}, 10^{-3}$ and under linear cost. The dashed line in each subplot represent fixed iteration stopping (e.g., the y-axis value of the line at iteration 50 represent the cost-adjusted regret when always stopping at iteration 50).

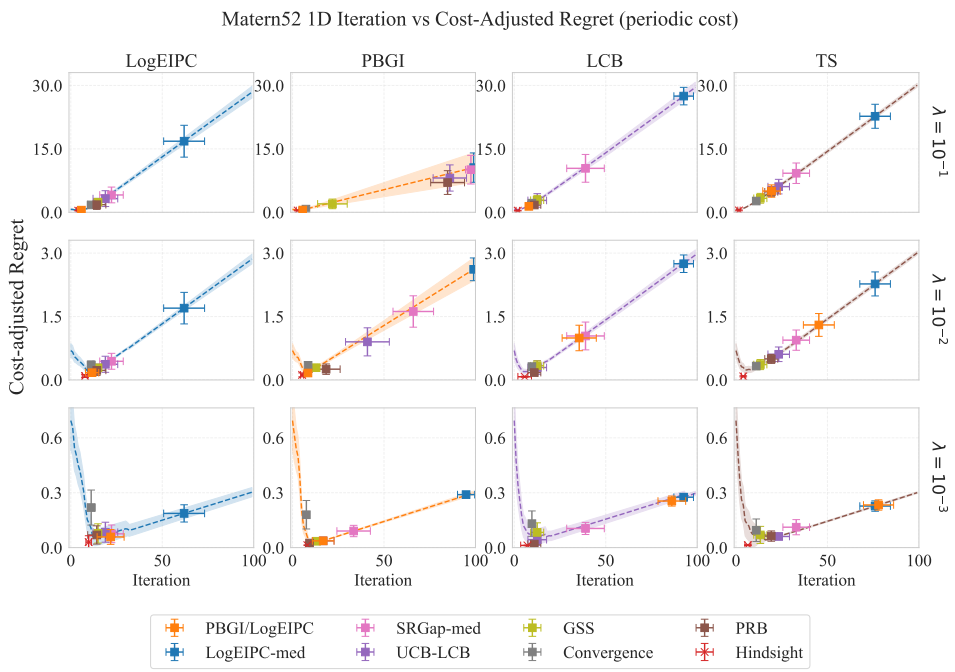


Figure 20: Comparison of cost-adjusted simple regret of different combinations of acquisition functions and stopping rules in the 1D Synthetic experiments, with cost-scaling factor $\lambda = 10^{1}, 10^{-2}, 10^{-3}$ and under periodic cost. The dashed line in each subplot represent fixed iteration stopping (e.g., the y-axis value of the line at iteration 50 represent the cost-adjusted regret when always stopping at iteration 50).

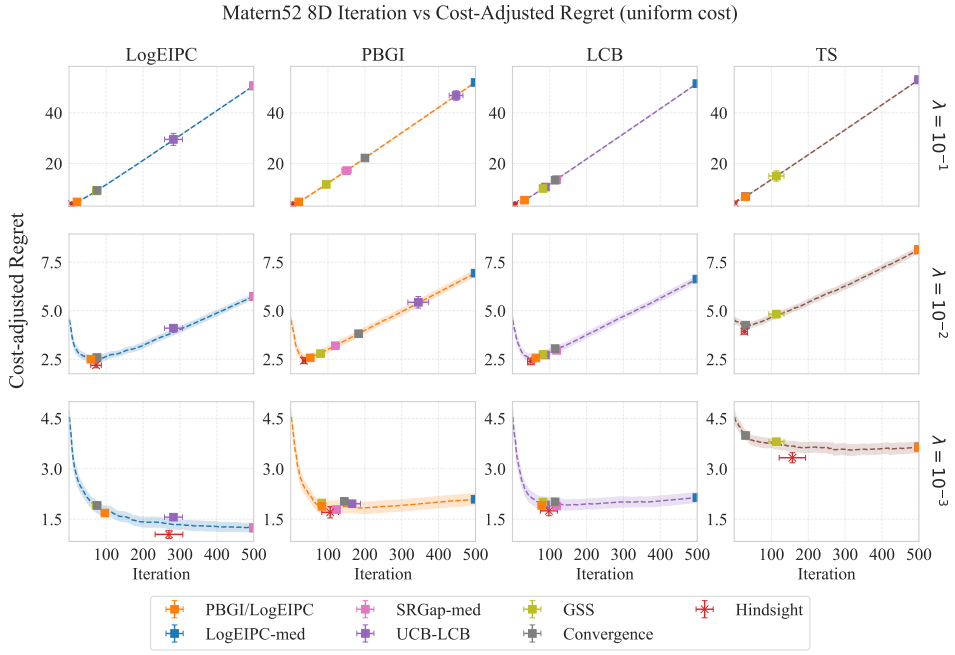


Figure 21: Comparison of cost-adjusted simple regret of different combinations of acquisition functions and stopping rules in the 8D Synthetic experiments, with cost-scaling factor $\lambda = 10^{1}, 10^{-2}, 10^{-3}$ and under uniform cost. The dashed line in each subplot represent fixed iteration stopping (e.g., the y-axis value of the line at iteration 50 represent the cost-adjusted regret when always stopping at iteration 50).

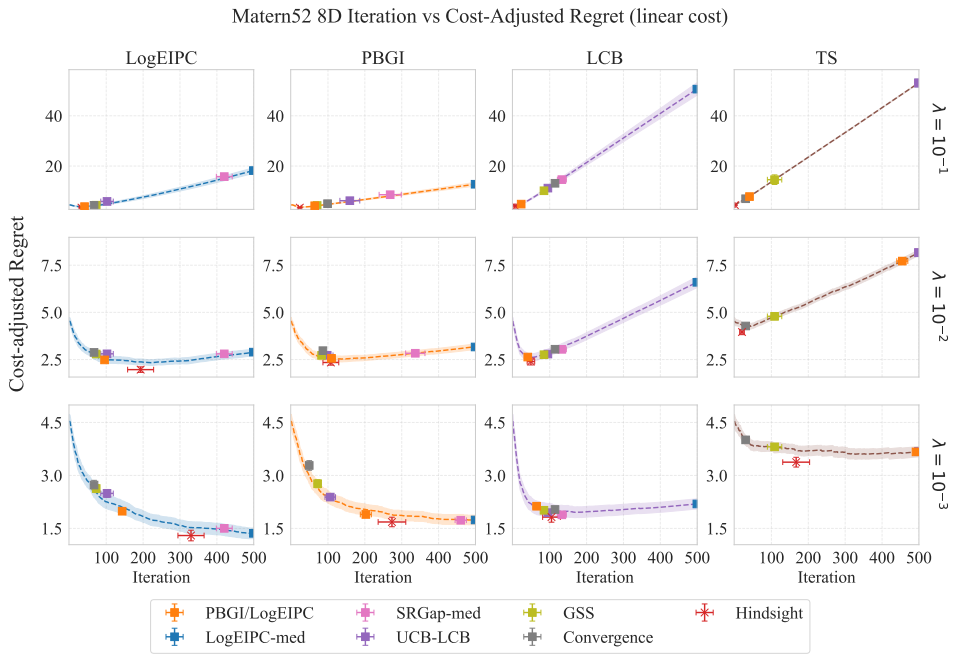


Figure 22: Comparison of cost-adjusted simple regret of different combinations of acquisition functions and stopping rules in the 8D Synthetic experiments, with cost-scaling factor $\lambda = 10^{1}, 10^{-2}, 10^{-3}$ and under linear cost. The dashed line in each subplot represent fixed iteration stopping (e.g., the y-axis value of the line at iteration 50 represent the cost-adjusted regret when always stopping at iteration 50).

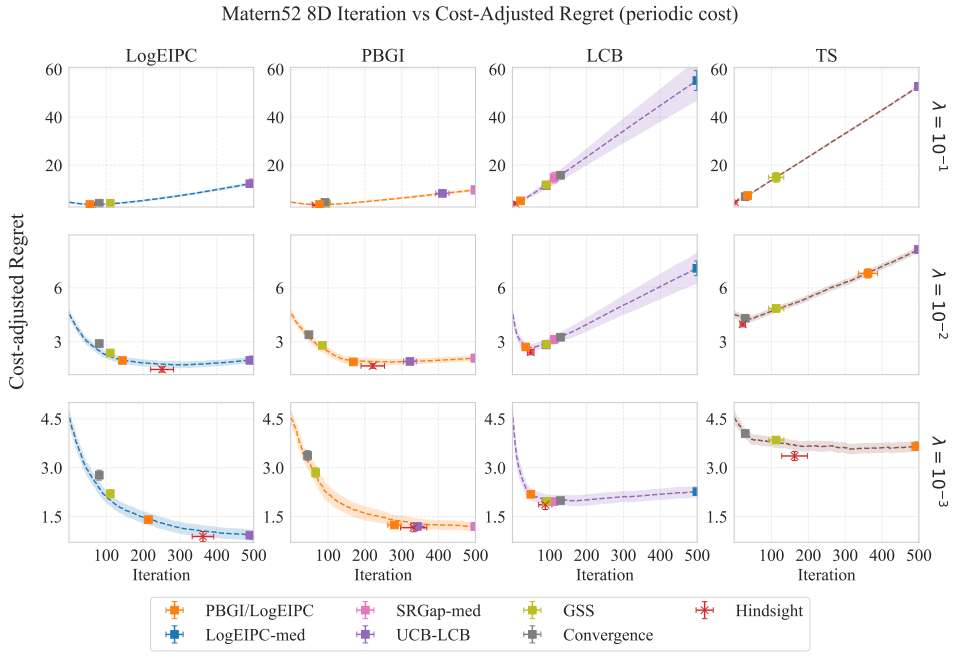


Figure 23: Comparison of cost-adjusted simple regret of different combinations of acquisition functions and stopping rules in the 8D Synthetic experiments, with cost-scaling factor $\lambda = 10^{1}, 10^{-2}, 10^{-3}$ and under periodic cost. The dashed line in each subplot represent fixed iteration stopping (e.g., the y-axis value of the line at iteration 50 represent the cost-adjusted regret when always stopping at iteration 50).