Bayesian Optimization with Switching Cost: Regret Analysis and Lookahead Variants

Peng Liu[1] , **Haowei Wang**[2] , **Wei Qiyu**[3]
[1]Singapore Management University
[2]Rice-Rick Digitalization
[3]Shanghai University
liupeng@smu.edu.sg, haowei_wang@ricerick.com, qywei@shu.edu.cn

Abstract

Bayesian Optimization (BO) has recently received increasing attention due to its efficiency in optimizing expensive-to-evaluate functions. For some practical problems, it is essential to consider the path-dependent switching cost between consecutive sampling locations given a total traveling budget. For example, when using a drone to locate cracks in a building wall or search for lost survivors in the wild, the search path needs to be efficiently planned given the limited battery power of the drone. Tackling such problems requires a careful cost-benefit analysis of candidate locations and balancing exploration and exploitation. In this work, we formulate such a problem as a constrained Markov Decision Process (MDP) and solve it by proposing a new distance-adjusted multi-step lookahead acquisition function, the distUCB, and using rollout approximation. We also provide a theoretical regret analysis of the distUCB-based Bayesian optimization algorithm. In addition, the empirical performance of the proposed algorithm is tested based on both synthetic and real data experiments, and it shows that our cost-aware non-myopic algorithm performs better than other popular alternatives.

1 Introduction

Suppose we want to locate a global maximum $f^* = f(\mathbf{x}^*)$ from a continuous, compact, and bounded region $\mathcal{X} \subset \mathbb{R}^p$. The unknown objective function $f : \mathcal{X} \to \mathbb{R}$ is a black-box function that lacks closed-form expression and gradient information, making it difficult to optimize. In addition, evaluations of the function are often subject to a budget, thus necessitating a quick search of the global maximum in as few steps as possible. To obtain $\mathbf{x}^* = \text{argmax}_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x})$, a sample-efficient global optimization approach called Bayesian optimization is often adopted. A typical BO approach uses a surrogate model to approximate the unknown objective function $f(\mathbf{x})$ and an acquisition function to guide the search. The surrogate model, using Gaussian processes in most practical applications, provides the most probable posterior estimates of functional values and uncertainties across the full search domain. The acquisition function is then used to maximize the expected marginal gain in the utility of the collected dataset to proceed with the sequential decision-making under uncertainty.

In this sequential optimization process, each new data point in $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ is revealed online, possibly perturbed by random noise. Given a limited sampling budget, the BO policy must consciously reason about exploring new regions with potentially high values and exploiting a known promising region with high confidence. In real-life applications such as crack detection in a wall or survivor search in the wild, a physical drone is often used to perform the search guided by a specific algorithm, which also needs to account for the total air time when moving the drone between locations and taking pictures. A total budget, such as the available battery power, must be considered in this case. The search policy needs to consider possible future paths and associated switching costs to make an optimal decision at the current step, a typical Dynamic Programming (DP) problem. However, exactly solving this intractable DP is NP-hard, especially in a continuous search space with infinite decision variables. Deciding on the continual search requires justifying a higher gain in the expected utility over the potential traveling cost.

In this paper, motivated by the aforementioned real-life applications, we consider the problem of sample-efficient optimization inclusive of a second criterion for success, which involves the cumulative distance traveled between subsequent tests. We formulate the problem as a constrained MDP and propose an efficient rollout algorithm based on a distance-adjusted Upper Confidence Bound (distUCB) heuristic to obtain the approximate DP solution. The proposed non-myopic BO policy effectively weighs the transition cost and gains over several future steps to determine the optimal sampling location at the current step. Our formulation highlights an automatic stopping criterion: the search terminates when the cumulative switching cost at a future candidate location exceeds the remaining budget, thus making it a finite-horizon MDP. In addition, we provide a theoretical analysis on the convergence guarantee of the cost-aware base policy using an information gain approach, as motivated by the sub-linear regret analysis in [Srinivas *et al.*, 2009].

1.1 Literature Review

Bayesian optimization was first proposed by [Mockus and others, 1978] to optimize expensive, black-box functions using expected improvement (EI) as the acquisition function. Representative candidates in this increasingly crowded space of novel acquisition functions include upper confidence bound (UCB) [Srinivas *et al.*, 2009], predictive entropy search (PES) [Hernández-Lobato *et al.*, 2014], and knowledge gradient (KG) [Scott *et al.*, 2011], among others.

The rising interest in non-myopic BO acquisition functions started from [Lam and Willcox, 2017], which considered the constrained BO problem and proposed a lookahead approach to maximize the long-term gain using a particular approximate DP algorithm called rollout. The rollout algorithm is a suboptimal control method used in the BO context to determine the following sampling location by maximizing an approximate long-term reward over a rolling horizon. It dates back to [Bertsekas *et al.*, 1997] in the deterministic optimization case and was later extended to stochastic dynamic programs by [Goodson *et al.*, 2016]. Recent work by [Yue and Kontar, 2020] characterizes the rollout improving nature in non-myopic policy under properly selected heuristics. It guides the stagewise rolling horizon based on the negative effect from a misspecified surrogate model. Since a typical rollout algorithm suffers from exponential growth in computational time as the rolling horizon increases, [Jiang *et al.*, 2020] further accelerated the computation by jointly optimizing all decision variables in a multi-step tree. Besides, [Lee *et al.*, 2021] proposed a cost-constrained BO framework using a non-myopic acquisition function and rollout approximation with EI per unit cost as the base policy.

On another front, the path-dependent switching cost in the sequential decision-making process was first explicitly considered in [Marchant and Ramos, 2012], where the distance traveled by a moving robot is incorporated as an additive penalty in the UCB acquisition function, resulting in the so-called distance-based upper confidence bound (DUCB). However, the two weighting parameters introduced by the variance and distance terms are challenging to be tuned simultaneously. The original convergence guarantee for UCB in [Srinivas *et al.*, 2009] no longer exists in the presence of an additional distance term. To our knowledge, none of the existing work has considered explicitly incorporating switching costs in the non-myopic BO framework. To obtain substantial acceleration, we also employ an efficient rollout algorithm with quasi-Monte Carlo, sample average approximation (SAA) initially used in [Balandat *et al.*, 2019], and the one-shot optimization scheme in [Jiang *et al.*, 2020].

1.2 Summary of Contributions

We summarize our main contributions as follows:

- We provide the first analysis on BO with path-dependent switching cost formulated as a constrained MDP and develop an efficient rollout approximation algorithm that involves long-term cost-benefit analysis.
- Our framework offers an optimal policy (based on Bellman's principle of optimality) that automatically considers the termination condition if the cost of the additional search is uniformly higher than the remaining budget across the whole domain, which further results in an episodic simulation over future steps and saves the need to set the size of the rolling horizon.
- We perform a theoretical analysis of the convergence rate of the proposed algorithm and conduct both synthetic and real-data experiments to show the potential benefits of our proposed methodology compared with other baseline approaches.

2 Background

2.1 Bayesian Optimization

In a maximization setting, the objective is to find the global maximizer $\mathbf{x}^* = \text{argmax}_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x})$. Suppose we have now collected a set of n observations $\mathcal{D}_n = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where the sampled location $\mathbf{x}_i \in \mathbb{R}^p$ is a p-dimensional vector in a compact set $\mathcal{X} \subset \mathbb{R}^p$, and $y_i \in \mathbb{R}$ is a noise-perturbed scalar observation with $y_i = f(\mathbf{x}_i) + \epsilon_i$. We assume a Gaussian noise $\epsilon_i \sim N(0, \sigma^2)$ with a homogenous variance σ^2 across the whole domain. Under the GP framework, any finite set of random variables within the domain follows a multivariate Gaussian distribution, resulting in $y_{1:n} \sim \mathcal{N}(\mathbf{0}, \mathbf{K}_n)$, where we assume a constant zero mean function $\mu = 0$ and $\mathbf{K}_n = \mathbf{K}_n(\mathbf{x}_{1:n}, \mathbf{x}_{1:n})$ denotes the covariance matrix, noting $\mathbf{K}(\mathbf{x}_{1:n}, \mathbf{x}_{1:n})_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$. Specifically, for a new sampling location $\mathbf{x}_* \in \mathcal{X}$, its corresponding observation y_* again follows a normal distribution, i.e., $p(y_*; \mathbf{x}_*, \mathcal{D}_n) = \mathcal{N}(y_*|\mu_*, \sigma_*^2)$, where $\mu_* = \mathbf{k}(\mathbf{x}_*, \mathbf{x}_{1:n})^T(\mathbf{K}_n + \sigma^2\mathbf{I})^{-1}y_{1:n}$ and $\sigma_*^2 = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}(\mathbf{x}_*, \mathbf{x}_{1:n})^T(\mathbf{K}_n + \sigma^2\mathbf{I})^{-1}\mathbf{k}(\mathbf{x}_*, \mathbf{x}_{1:n})$, and $\mathbf{k}(\mathbf{x}_*, \mathbf{x}_{1:n}) = [k(\mathbf{x}_*, \mathbf{x}_1), ..., k(\mathbf{x}_*, \mathbf{x}_n)]^T$. We refer the readers to [Rasmussen and Williams, 2006] for a more comprehensive review of Gaussian processes.

In each BO iteration, the GP posterior is updated by refreshing its governing hyperparameters based on the available training set $\mathcal{D}_n$, then choosing the next sampling location where the specified auxiliary acquisition function obtains its maximum. Locating the maximum of the acquisition function is also called the inner optimization, which happens per iteration in contrast to the outer optimization that seeks the global optimum of the unknown objective function. The newly acquired location and its observation are then appended to form $\mathcal{D}_{n+1} = \{\mathcal{D}_n \cup (\mathbf{x}_{n+1}, y_{n+1})\}$, completing one round of iteration and proceeding until the sampling budget exhausts.

2.2 Distance-Constrained Bayesian Optimization

The distance can be considered an additional cost constraint imposed on the optimizer. For example, a specific location may not be reachable due to the limited battery of a robot. In such cases, the optimizer should display a tradeoff between the utility of a sampling location and the cost it will incur if it decides to sample in that particular location, preferably considering future evolution should the learning proceeds.

There are different approaches to modeling the distance penalty by embedding it into the acquisition function to discount the utility of a candidate sampling location. Examples include subtracting from UCB the distance traveled from the current location to the candidate location. Specifically, in the

DUCB formulation used in [Marchant and Ramos, 2012], the acquisition function for the next candidate location $\mathbf{x}_{n+1}$ with $y_{n+1} \sim \mathcal{N}(\mu_n, \sigma_n^2)$ takes the following form:

$$\begin{aligned}\alpha(\mathbf{x}_{n+1}; \mathcal{D}_n) =& \mu_n(\mathbf{x}_{n+1}) + \beta_{n+1}\sigma_n(\mathbf{x}_{n+1}) \\ & -\rho_{n+1} d(\mathbf{x}_{n+1}, \mathbf{x}_n)\end{aligned} \tag{1}$$

where $d(\mathbf{x}_{n+1}, \mathbf{x}_n) = \sqrt{\sum_{i=1}^{p}(\mathbf{x}_{n+1}^i - \mathbf{x}_n^i)^2}$ denotes the general Euclidean distance between the current location $\mathbf{x}_n$ of the optimizer and the next candidate location $\mathbf{x}_{n+1}$, β_{n+1} is a stage-wise weighting factor to set the level of confidence in the face of uncertainty, and ρ_{n+1} is another weighting factor for the distance-induced penalty. Although being an intuitive formulation, this data-driven approach requires tuning two hyper-parameters simultaneously, which is both stage and domain-dependent and lacks the theoretical justification for setting the optimal hyper-parameters, as compared to the case of choosing $\beta_n = 2\log(n^2 2\pi^2/(3\delta)) + 2p\log(n^2 pbr\sqrt{\log(4pa/\delta)})$ in [Srinivas *et al.*, 2009], where we use p to denote the dimensionality of the feature space.

Other heuristics include multiplying the original acquisition function with an indicator function that specifies the feasible region of the optimizer at the current step, as originally proposed in [Gardner *et al.*, 2014]. However, explicitly confining the optimizer within the feasible region is not conducive to the cost-performance tradeoff in the early stages; the optimizer displays no foresight on budget consumption when the feasible region covers the whole domain. In addition to the myopic nature of the acquisition function, the multiplying indicator function also makes it inconvenient for theoretical analysis.

More recent work by [Lee *et al.*, 2021] divides the acquisition function (using EI) by the corresponding cost to get the EI per unit cost (EIpu) in a multi-step lookahead setting, which is most similar to our work. Although [Lee *et al.*, 2021] uses the rollout algorithm to obtain an approximate DP solution with a theoretical guarantee on its sequential improving nature, no analysis on the convergence rate of the base policy is given. In our experiment section later, we also show the superior performance of our proposed distUCB policy over EIpu and other alternatives.

2.3 Regret Analysis Using Information Gain

To better characterize the performance of the online optimizer in BO setting, we denote $r_i = f(\mathbf{x}^*) - f(\mathbf{x}_i)$ as the instantaneous regret (or simple regret) of the online optimizer at step i due to not knowing the global optimum f^* in advance. The cumulative regret R_N after N steps is derived by summing all instantaneous regrets: $R_N = \sum_{i=1}^{N} r_i$.

In order to bound the cumulative regret R_N, we follow a similar approach in [Srinivas *et al.*, 2009], using the maximum information gain $\gamma_N = \max \mathrm{I}(y_{1:N}; f_{1:N})$ after N steps to provide an explicit sub-linear convergence rate. Here, $\mathrm{I}(y_{1:N}; f_{1:N}) = \mathrm{H}(y_{1:N}) - \mathrm{H}(y_{1:N}|f_{1:N})$ denotes the reduction in the uncertainty of f after obtaining N noisy measurements $y_{1:N}$, where $\mathrm{H}(y_{1:N}) = \frac{1}{2}\log|2\pi e(\mathbf{K}_N + \sigma^2\mathbf{I})|$ is the marginal entropy and $\mathrm{H}(y_{1:N}|f_{1:N}) = \frac{1}{2}\log|2\pi e(\sigma^2\mathbf{I})|$ is the conditional entropy under a multivariate Gaussian distribution with Gaussian noise. Plugging in, we have $\mathrm{I}(y_{1:N}; f_{1:N}) = \frac{1}{2}\log|\sigma^{-1}\mathbf{K}_N + \mathbf{I}|$.

Being a submodular function, the information gain $\mathrm{I}(y_{1:N}; f_{1:N})$ reveals a property of diminishing return as N increases, which plays an important role in bounding γ_N as shown in [Srinivas *et al.*, 2009].

2.4 Multi-Step Lookahead Acquisition Function

The sequential process in Bayesian optimization can be modeled as a Markov Decision Process (MDP), where the learning environment is characterized by a state space $\mathcal{S}$, an action space $\mathcal{A}$, a transition probability between different states $\mathcal{P}$, and the reward R. In the context of BO, the state space $\mathcal{S}$ is the observed set of data $\mathcal{D}_n$ until the current time step n, and the action space $\mathcal{A}$ is the whole continuous domain $\mathcal{X}$. A policy $\pi = \{\pi_1, \ldots, \pi_N\}$ specifies the sampling decisions for a total of N steps. Specifically, π_n at any step n is a function that maps the current state $\mathcal{D}_n$ to the next sampling location $\mathbf{x}_{n+1}$, i.e., $\pi_n(\mathcal{D}_n) = \mathbf{x}_{n+1}$. The only transition probability is the probability of sampling y_{n+1} from its posterior:

$$P(y_{n+1}|\mathbf{x}_{n+1}, \mathcal{D}_n) = \mathcal{N}(\mu_n(\mathbf{x}_{n+1}; \mathcal{D}_n), \sigma_n^2(\mathbf{x}_{n+1}; \mathcal{D}_n)) \tag{2}$$

An optimal policy can be derived by maximizing the sum of the current reward $R(\mathcal{D}_n, \mathbf{x}_{n+1}, \mathcal{D}_{n+1})$ and the long-term return that accounts for all possible randomness in both the locations and the functional evaluations, i.e., $\pi^* = \mathrm{argmax}_{\pi\in\Pi} \sum_{i=0}^{\tau-1} R(\mathcal{D}_{n+i}, \pi_{n+i}(\mathcal{D}_{n+i}), \mathcal{D}_{n+i+1})$. Following Bellman's principle of optimality, the optimal policy not only selects the current action (i.e., sampling location) that maximizes the reward but also assumes all future actions are made optimally.

Specifically, denote $u(\mathcal{D}_n)$ as the utility of the current training set $\mathcal{D}_n$. Typically, $u(\mathcal{D}_n) = \max\{y_{1:n}\}$ in an improvement-based heuristic. The expected utility after acquiring an additional putative pair $(\S_{n+1}, y_{n+1})$ can then be expressed as

$$\begin{aligned}&\mathbb{E}_{y_{n+1}}(u(\mathbf{x}_{n+1}, y_{n+1}, \mathcal{D}_n)|\mathbf{x}_{n+1}, \mathcal{D}_n) = \\ &\int u(\mathbf{x}_{n+1}, y_{n+1}, \mathcal{D}_n)|\mathbf{x}_{n+1}, \mathcal{D}_n) p(y_{n+1}|\mathbf{x}_{n+1}, \mathcal{D}_n) dy_{n+1}\end{aligned} \tag{3}$$

where the expectation is taken over the random variable y_{n+1} at the candidate location $\mathbf{x}_{n+1}$. The optimal sampling location at time step $n+1$ is:

$$\mathbf{x}_{n+1}^* = \mathrm{argmax}_{\mathbf{x}_{n+1}\in\mathcal{X}} \alpha_1(\mathbf{x}_{n+1}; \mathcal{D}_n) \tag{4}$$

where,

$$\begin{aligned}\alpha_1(\mathbf{x}_{n+1}; \mathcal{D}_n) = & \mathbb{E}_{y_{n+1}}(u(\mathbf{x}_{n+1}, y_{n+1}, \mathcal{D}_n)|\mathbf{x}_{n+1}, \mathcal{D}_n) \\ & - u(\mathcal{D}_n)\end{aligned} \tag{5}$$

denotes the one-step expected marginal gain in utility, with the subscript indicating the number of lookahead steps into the future.

Now, assume a total lookahead horizon of τ steps into the future, forming a putative dataset $\mathcal{D}_{n+\tau} = \mathcal{D}_n \cup$

$\{(\mathbf{x}_{n+1}, y_{n+1})\} \cup \cdots \cup \{(\mathbf{x}_{n+\tau}, y_{n+\tau})\}$. The multi-step lookahead policy then chooses the next sampling location via:

$$\mathbf{x}^*_{n+1} = \text{argmax}_{\mathbf{x}_{n+1} \in \mathcal{X}} \alpha_\tau(\mathbf{x}_{n+1}; \mathcal{D}_n) \tag{6}$$

where $\alpha_\tau(\S_{n+1}; \mathcal{D}_n) = \mathbb{E}(u(\mathcal{D}_{n+\tau})|\mathbf{x}_{n+1}, \mathcal{D}_n) - u(\mathcal{D}_n)$ is the τ-step lookahead expected marginal gain in utility, with the expectation taken over all possible values of y_{n+1} as well as following pairs $\{(\mathbf{x}_{n+i}, y_{n+i})\}_{i=2}^{\tau}$.

The τ-step nonmyopic acquisition function can also be decomposed into the sum of an immediate expected one-step lookahead gain in utility as well as the expected marginal gain for the remaining $\tau-1$ steps. Adding and subtracting the one-step utility function gives the following recursive form:

$$\begin{aligned}\alpha_\tau(\mathbf{x}_{n+1}; \mathcal{D}_n) &= \alpha_1(\mathbf{x}_{n+1}; \mathcal{D}_n) \\ &+ \mathbb{E}[\alpha_{\tau-1}(\mathbf{x}_{n+2}; \mathcal{D}_{n+1})|\mathbf{x}_{n+1}, \mathcal{D}_n]\end{aligned} \tag{7}$$

Selecting the best sampling location with $\alpha^*_\tau(\mathbf{x}_{n+1}; \mathcal{D}_n) = \max_{\mathbf{x}_{n+1} \in \mathcal{X}} \alpha_\tau(\mathbf{x}_{n+1}; \mathcal{D}_n)$ amounts to directly maximizing the average-case long-term utility and follows the Bellman's principle of optimality: a sequence of optimal decisions starts with making the first optimal decision, followed by a series of optimal decisions conditioned on the previous outcome. The optimal multi-step lookahead acquisition function could be written as a DP formulation that consists of a series of nested maximization and expectation operations:

$$\begin{aligned}&\alpha^*_\tau(x_{n+1}; \mathcal{D}_n) \\ &= \max_{x_{n+1} \in \mathcal{X}} \{\alpha_1(x_{n+1}; \mathcal{D}_n) \\ &+ \mathbb{E}[\max_{x_{n+2} \in \mathcal{X}} \alpha_{\tau-1}(x_{n+2}; \mathcal{D}_{n+1})|x_{n+1}, \mathcal{D}_n)]\}\end{aligned} \tag{8}$$

Choosing the next sampling action by maximizing the multi-step lookahead acquisition function gives the optimal policy π^*. However, solving this intractable DP problem is computationally challenging when the lookahead horizon becomes large. In most lookahead problems in the BO setting, the rollout algorithm has become a popular approximate DP solution that provides a good-quality solution in a reasonable computational time.

The rollout algorithm breaks the forward-backward pass in solving an intractable DP problem and instead completes a single forward pass. It starts by simulating a set of future values over the future τ horizon and uses a base policy with averaged results to approximate the value of the multi-step acquisition function. In this paper, we use a new distance-based UCB as the one-step base policy in the rollout algorithm to perform a multi-step lookahead selection of sampling locations. As it turns out, such a lookahead policy delivers a natural sense of impact: as the policy treks from point to point in the search space, a constant tradeoff between seeking the maximum function value and minimizing the switching cost exists. Considering such a tradeoff, even for 1 or 2 steps ahead, makes a difference in the overall search performance.

3 Methodology

Given a total traveling budget B, the goal of our moving robot is to seek the global maximum before its power drains. When the remaining budget is insufficient to support an additional sampling at the suggested location, the optimizer terminates and returns a recommendation based on the previously sampled locations. Mathematically, for a total of N steps, we have

$$\begin{aligned}&\max_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}) \\ &\text{s.t.} \sum_{i=1}^{N-1} d(\mathbf{x}_{i+1}, \mathbf{x}_i) \leq B\end{aligned} \tag{9}$$

where $d(\mathbf{x}_{i+1}, \mathbf{x}_i) = ||\mathbf{x}_{i+1} - \mathbf{x}_i||_2$ for $i = 2, ..., N$. We propose a new distance-adjusted UCB (distUCB) base policy to quantify the utility of a candidate location $\mathbf{x}_{n+1}$ across the whole domain:

$$\alpha_{distUCB}(\mathbf{x}_{n+1}; \mathcal{D}_n) = \mu_n(\mathbf{x}_{n+1}) + \beta_{n+1} \frac{\sigma_n(\mathbf{x}_{n+1})}{d(\mathbf{x}_{n+1}, \mathbf{x}_n)} \tag{10}$$

where distUCB intuitively scales down the utility of the uncertain long-distance regions. Such formulation avoids another tuning parameter and permits theoretical analysis of its convergence rate. To facilitate the analysis of the regret bound, we absorb the distance-based scaling factor into β_{n+1} and denote it as $\beta_{n+1,d}$ to highlight its dependence on both the current step n and the switching cost. Thus,

$$\alpha_{distUCB}(\mathbf{x}_{n+1}; \mathcal{D}_n) = \mu_n(\mathbf{x}_{n+1}) + \beta_{n+1,d} \sigma_n(\mathbf{x}_{n+1}) \tag{11}$$

which is also the reward function R in the constrained MDP (CMDP) setting. Next, we define the expected total return and cost of a policy π. For any policy π with a lookahead horizon of τ, the expected total return and cost are respectively defined as:

$$\begin{aligned}&V^\pi_\tau(\mathbf{x}_{n+1}; \mathcal{D}_n) \\ &= \mathbb{E}\big[\sum_{i=0}^{\tau-1} (\mu_{n+i}(\mathbf{x}_{n+i+1}) + \beta_{n+1+i,d} \sigma_{n+i}(\mathbf{x}_{n+i+1}))\big]\end{aligned} \tag{12}$$

$$C^\pi_\tau(\mathbf{x}_{n+1}; \mathcal{D}_n) = \mathbb{E}\big[\sum_{i=0}^{\tau-1} d(\mathbf{x}_{n+i}, \mathbf{x}_{n+i+1})\big] \tag{13}$$

Here, $V^\pi_\tau(\mathbf{x}_{n+1}; \mathcal{D}_n)$ is the action value at location $\mathbf{x}_{n+1}$, defined as the expected sum of current and future rewards till horizon τ, given training set $\mathcal{D}_n$ and following the policy π. $C^\pi_\tau(\mathbf{x}_{n+1}; \mathcal{D}_n)$ denotes the associated expected total cost. The problem can thus be modeled as seeking the optimal policy π^* (which maximizes $\alpha_{distUCB}(\mathbf{x}_{n+i+1}; \mathcal{D}_{n+i})$ in each lookahead stage) in the context of CMDP as follows:

$$\begin{aligned}\pi^* &= \text{argmax}_{\pi \in \Pi} V^\pi_\tau(\mathbf{x}_{n+1}; \mathcal{D}_n) \\ &\text{s.t.} C^\pi_\tau(\mathbf{x}_{n+1}; \mathcal{D}_n) \leq B\end{aligned} \tag{14}$$

where Π is the space of all admissible policies. When simulating future trajectories using rollout, only feasible trajectories are allowed if the remaining budget permits. Denote a base policy $\tilde{\pi} = \{\tilde{\pi}_1, \ldots, \tilde{\pi}_N\}$ used to approximate the optimal policy. When given a dataset $\mathcal{D}_n$ and simulating at a future stage $i \in \{0, \ldots, \tau - 1\}$, the corresponding decision rule $\tilde{\pi}_{n+i}$ follows the base heuristic of maximizing the stage-wise acquisition function $\alpha_{distUCB}(\mathbf{x}_{n+i+1}; \mathcal{D}_{n+i})$:

Algorithm 1: distUCB: distance-adjusted UCB

Data: GP prior μ_0 and σ_0, search domain $\mathcal{X}$
Result: Estimated location of global maximum x^*
while $\sum_{i=1}^{n} d_i \leq B$ **do**
  Select next sampling location
  $\mathbf{x}_{n+1} = argmax_{x \in \chi} \mu_n + \beta_{n,d}\sigma_n$;
  Query the corresponding observation
  $y_{n+1} = f(\mathbf{x}_{n+1}) + \epsilon_{n+1}$;
  Update the training set
  $\mathcal{D}_{n+1} = \mathcal{D}_n \cup (\mathbf{x}_{n+1}, y_{n+1})$;
  Update GP posterior to obtain μ_{n+1} *and* σ^2_{n+1};
Output: $x^* = \arg\max_{\mathbf{x} \in \mathcal{D}_n} y(\mathbf{x})$

$$\tilde{\pi}_{n+i} = \text{argmax}_{\mathbf{x}_{n+i+1} \in \mathcal{X}} \mathbb{E}[\alpha_{distUCB}(\mathbf{x}_{n+i+1}; \mathcal{D}_{n+i})] \tag{15}$$

Rolling out this heuristic policy till budget exhaustion gives the rollout policy, where the horizon τ differs for each feature path. When calculating the value function of the rollout policy, we use Monte-Carlo integration to approximate the expectation operator, giving:

$$V^{\tilde{\pi}}_{\tau}(\mathbf{x}_{n+1}; \mathcal{D}_n) \approx \frac{1}{M} \sum_{j=1}^{M} \big[\sum_{i=0}^{\tau-1} \alpha_{distUCB}(\mathbf{x}_{n+i+1}; \mathcal{D}_{n+i}) \big] \tag{16}$$

To accelerate and stabilize the simulations, we use quasi-Monte-Carlo based on the Sobol sequence and common random variables when using gradient-based updates based on the re-parameterization trick.

Since budget B is a limited scalar, we note that the non-myopic acquisition function using rollout essentially solves a constrained finite-horizon dynamic program. The overall procedure is given in Algorithm 1.

4 Theoretical Analysis

To facilitate theoretical analysis, we make the following assumptions.

Assumption 1. *Given constants a, b, and L, we assume that the kernel function $k(\mathbf{x}, \mathbf{x}')$ satisfies the following Lipschitz continuity for the confidence bound of the derivatives of GP sample paths f:*

$$P\Big(sup_{\mathbf{x} \in \mathcal{X}} |\frac{\partial f}{\partial \mathbf{x}_j}| > L\Big) \leq a e^{-L^2/b^2} \quad j = 1, \ldots, p \tag{17}$$

For example, the popular squared exponential kernel $k(x, x') = \sigma^2 exp(-\frac{(x-x')^2}{2l^2})$ with the length-scale parameter l and noise variance σ^2 satisfies this assumption.

Of note, the assumption on Lipschitz continuity is standard as in the regret analysis of BO [Srinivas *et al.*, 2009], which also includes the popular Matérn kernel. We introduce the main theorem on the cumulative regret bound of the single-stage distUCB algorithm.

4.1 Regret Bound Analysis (The Main Theorem)

Theorem 1. *Let $\mathcal{X} \subset [0, r]^p$ be compact and convex, $p \in \mathbb{N}$, $r > 0$. Under Assumption 1, for any arbitrarily small $\delta \in (0, 1)$, choose $\beta_{n,d} = \frac{2}{d^2} log \frac{4\pi_n}{\delta} + \frac{2p}{d^2} log\Big(n^2 brp\sqrt{log(\frac{4pa}{\delta})}\Big)$, where $\sum_{n \geq 1} \pi_n^{-1} = 1$, $\pi_n > 0$. As $n \to \infty$, we obtain a regret bound of $\mathcal{O}^*(\sqrt{pN\gamma_N})$. Specifically, with $C_1 = \frac{8}{log(1+\sigma^{-2})}$, we have:*

$$P(R_N \leq \sqrt{C_1 N \beta_{N,d} \gamma_N}) \geq 1 - \delta \tag{18}$$

We put the sketch proof for the main theorem in the appendix.

5 Experiments

We compare distUCB with EI, EIpu, and UCB rollout via both synthetic and real-world experiments. Among these benchmark methods, EI was first proposed in [Mockus and others, 1978], EIpu was proposed in [Lee *et al.*, 2021] using non-myopic acquisition function and rollout approximation with EI per unit cost as the base policy, and UCB rollout was constructed by us in order to compare the effectiveness of considering the switching cost, which was obtained by combining UCB [Srinivas *et al.*, 2009] method and rollout algorithm. Note that we did not consider DUCB due to its myopic nature and dependence on multiple hyperparameters involved; instead, our main benchmark under a similar setting is the EIpu acquisition function.

Our constrained Bayesian optimization problem essentially highlights a tradeoff between the search quality and distance traveled. An aggressive policy may achieve a higher function value (lower regret) but incur a significant switching cost, while a more cost-constrained policy will likely end up with a lower cost but at the expense of a higher regret. To this end, we study how the switching cost affects the test performance in multiple experiments, involving both synthetic and real data, and use the Euclidean distance to measure the distance between two points in the search space.

5.1 Synthetic Experiments

To make the comparison fair, we follow the setting of EIpu, using a Matérn $5/2$ kernel for the GP with its hyperparameters learned via maximum likelihood estimation. These hyper-parameters are re-estimated by maximizing the evidence after each iteration, a common practice used in [Srinivas *et al.*, 2009] and [Lee *et al.*, 2021] .

We compare the distUCB algorithm with other baseline policies over classical functions used for global optimization: Ackley-2, Branin-2, and Hartmann-6, detailed in Table 1. We also repeat each experiment 30 times, reporting the mean and standard deviation across all repetitions. The total iteration budget N for each experiment is set to $N = n_0 + 100$, where n_0 denotes the number of initial design points and is set as 20, 40, and 60 for the three synthetic functions, respectively. All experiments are evaluated on a Latin hypercube after normalization.

In addition, we add a homogeneous noise with a standard normal distribution with a standard deviation of 0.1, and

Functions	Domain
Ackley-2	$[-32.768, 32.768]^2$
Branin-2	$[-5, 10] \times [0, 15]$
Hartmann-6	$[-1, 1]^6$

Table 1: Functions used in the synthetic experiments

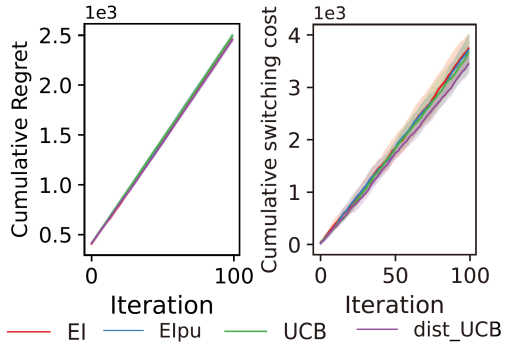

Figure 1: Performance of the four algorithms on Hartmann-6. Left: cumulative regret. Right: cumulative switching cost. distUCB reports a competing cumulative regret but achieves the lowest cumulative cost compared with other alternative policies.

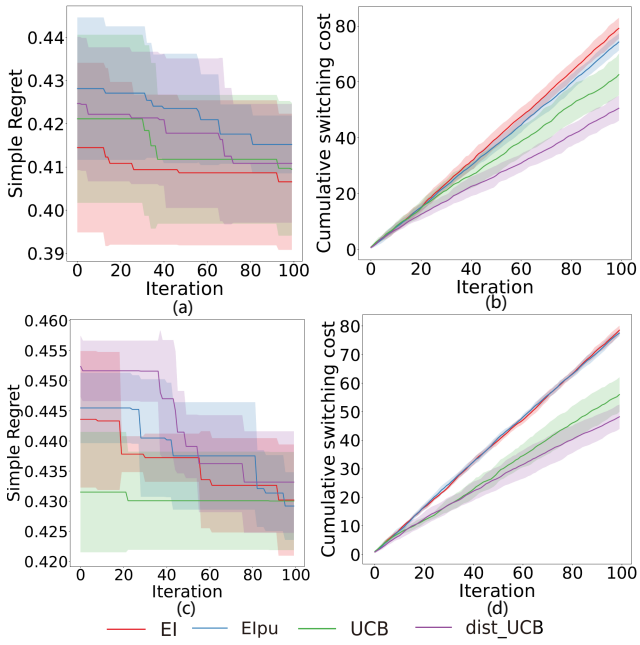

Figure 2: Simple regret and cumulative switching cost of different BO algorithms in the neural network hyperparameter tuning experiment. (a): Simple regret on breast cancer dataset. (b): Cumulative switching cost on breast cancer dataset. (c): Simple regret on spam dataset. (d): Cumulative switching cost on spam dataset. As the number of iterations increases, distUCB reports an increasingly closer simple regret than other policies but consistently obtains the lowest cumulative switching cost.

set up 32 simulations in each Monte Carlo approximation to keep the same setting as EIpu. For a fair comparison, each method's number of lookahead steps is set to 3, although the horizon could be determined automatically based on the remaining budget.

In general, we observed that distUCB displays a better cost awareness in the tradeoff between cumulative regret and switching costs, as shown by the lowest cumulative switching cost together with a competing cumulative regret. As shown in Figure 1, distUCB performs nearly identically to UCB rollout on cumulative regret after 100 iterations, which is only marginally larger than EI and EIpu. However, distUCB incurs a much smaller cumulative switching cost among all the policies. This suggests that our method achieves competitive results in cumulative regret compared to baseline methods while consuming a much smaller budget. See the appendix for the experimental results of the other two synthetic functions for a similar observation.

The experiment results also suggest that our method works better on relatively high-dimensional problems, which is possibly due to the fast convergence in low-dimensional test functions. The switching cost is thus potentially a more cost-inducing factor in high-dimensional space. We choose two relatively high-dimensional real cases in the next section to validate our hypothesis.

5.2 Real Data Experiments

Many real-world experiments prefer a small jump (i.e., a lower switching cost) between test configurations. For example, search over a domain with a prior preference region identified by domain expertise or previous experience. The cost consideration in distUCB naturally fits such scenarios. Also, to make the tradeoff more explicit, we show the empirical dynamics between cumulative switching cost and simple regret, which is a better metric to measure the quality of the current proposal. Such comparison allows us to assess how the limited traversal distance impacts the quality of the optimization in a multi-objective sense.

Using a neural network model, we empirically evaluate distUCB on two hyperparameter tuning tasks for training a two-layer feed-forward neural network on two popular UCI datasets: breast cancer and spam, both commonly used in hyperparameter tuning experiments (see [Hu *et al.*, 2022]). The breast cancer Wisconsin dataset has 569 data and 32 columns, where most features are continuous and nucleus-related. The label of each sample is the result of the diagnosis, including 357 benign cases and 212 malignant cases. The spam database contains 4601 samples (1813 of which are spam) and 58 attributes. For both data sets, we allocate 70% to training and 30% to the test set.

We consider tuning four hyperparameters: batch size, initial learning rate, learning rate schedule, and the number of hidden dimensions. For each of the four hyperparameters to be adjusted, the adjustment range is given: $[32, 128]$, $[1e-6, 1.0]$, $[1e-6, 1.0]$, $[0.5, 4]$. We specify these four hyperparameters in each iteration before training a Multi-layer Perceptron (MLP) prediction model on the training set. The trained model is then scored over the test set, where the test set classification error is registered for the current iteration. Each iteration repeats 5 times.

The results of the four methods on two hyperparameter tuning experiments are shown in Figure 2. After 100 iterations, all four policies deliver a similar simple regret, showing a decreasing impact of the cost constraint for the distance-aware policy. However, distUCB algorithm consistently incurs a lower cumulative switching cost than the other three methods. All results reflect the central theme of a continuous tradeoff between simple regret, as a measure of optimization quality, and the cumulative switching cost, as an indication of total distance traveled. We further observe that such a tradeoff is more pronounced only in the early stage of the search; as iteration proceeds, all policies report a smaller optimality gap, yet only the cost-aware policy (in this case, distUCB) demonstrates the long-term benefit of a consistently lower cumulative switch cost.

6 Conclusion

In this paper, we proposed a cost-aware multi-step acquisition function to perform sequential sampling under uncertainty. Our proposed algorithm (distUCB) can deliver similar or better overall performance in the cumulative regret and switching cost. We provide a theoretically no-regret analysis of the base policy and show its superior empirical performance in a rollout policy compared with standard benchmark methods. Using synthetic and real-data experiments, we highlight the explicit tradeoff between global maximization (searching anywhere within the domain) and cost minimization (penalizing long-distance moves). Our work thus paves the way for further studies in cost-aware BO with non-myopic policies.

A Sketch Proof of The Main Theorem

Our proof relies on the confidence bound on the selected decision $\mathbf{x}_n$ at stage n for an adequately selected hyperparameter $\beta_{n,d}$, as well as a stage-wise discretization $\mathcal{X}_n \subset \mathcal{X}$ used to obtain a bound of $f(\mathbf{x}^*)$. We use $\mathcal{O}^*$, a variant of the $\mathcal{O}$ notation to suppress the log factors.

Proof. Following a similar argument in [Srinivas *et al.*, 2009], we have $r_n^2 \leq 4d^2\beta_{n,d}\sigma_{n-1}^2(\mathbf{x}_n)$ with probability $\geq 1-\delta$ as $n \to \infty$. Since $\beta_{n,d} = \frac{2}{d^2}\log\frac{4\pi_n}{\delta} + \frac{2p}{d^2}\log\left(n^2brp\sqrt{\log(\frac{4pa}{\delta})}\right)$ and is non-decreasing in n, we can upper bound it by the final stage $\beta_{N,d}$:

$$\begin{aligned} &4d^2\beta_{n,d}\sigma_{n-1}^2(\mathbf{x}_n) \\ &\leq 4\beta_{N,d}\sigma^2(\sigma^{-2}\sigma_{n-1}^2(\mathbf{x}_n)) \\ &= 4\beta_{N,d}\sigma^2\frac{\sigma^{-2}}{\log(1+\sigma^{-2})}(\sigma_{n-1}^2(\mathbf{x}_n)\log(1+\sigma^{-2})) \end{aligned} \tag{19}$$

where we rely on the fact that $k(\mathbf{x}_n, \mathbf{x}_n) = 1$ for the SE kernel, thus $\sigma_{n-1}^2(\mathbf{x}_n) \leq k(\mathbf{x}_n, \mathbf{x}_n) = 1$. Note that $(1+\sigma^{-2})^{\sigma_{n-1}^2(\mathbf{x}_n)}$ is a concave function for a given $\sigma_{n-1}^2(\mathbf{x}_n) \in [0,1]$ with different values of $\sigma^{-2} > 0$. Using the first-order Taylor approximation to bound the concave function, giving $\sigma_{n-1}^2(\mathbf{x}_n)\log(1+\sigma^{-2}) \leq \log(1+\sigma^{-2}\sigma_{n-1}^2(\mathbf{x}_n))$. Setting $C_2 = \frac{\sigma^{-2}}{\log(1+\sigma^{-2})}$, we have:

$$4d^2\beta_{n,d}\sigma_{n-1}^2(\mathbf{x}_n) \leq 4\beta_{N,d}\sigma^2 C_2\log(1+\sigma^{-2}\sigma_{n-1}^2(\mathbf{x}_n)) \tag{20}$$

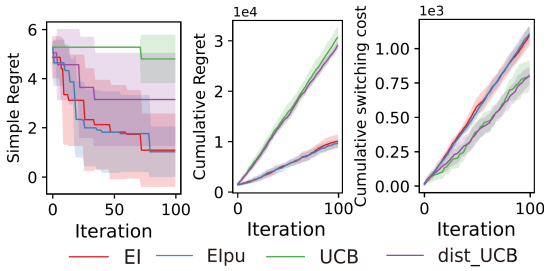

Figure 3: Performance on Branin function. Left: simple regret of four algorithms on the Branin test function. Middle: cumulative regret of four algorithms on the Branin test function. Right: cumulative switching cost of four algorithms on the Branin test function. As the number of iteration rounds increases, distUCB strikes a decent tradeoff between regret and switching cost.

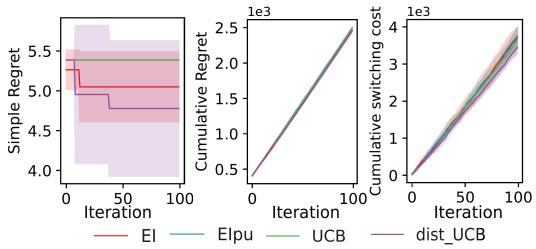

Figure 4: Performance on Ackley function. Left: simple regret of four algorithms on the Ackley test function. Middle: cumulative regret of four algorithms on the Ackley test function. Right: cumulative switching cost of four algorithms on the Ackley test function. distUCB incurs the lowest cumulative switching cost while delivering a similar cumulative regret.

Using Cauchy-Schwarz inequality, we have:

$$\begin{aligned} R_N^2 &\leq N\sum_{n=1}^{N} r_n^2 \\ &\leq N\sum_{n=1}^{N} 4\beta_{N,d}\sigma^2 C_2\log(1+\sigma^{-2}\sigma_{n-1}^2(\mathbf{x}_n)) \\ &= 8N\beta_{N,d}\sigma^2 C_2\mathrm{I}(y_{1:N}; f_{1:N}) \\ &= C_1 N\beta_{N,d}\mathrm{I}(y_{1:N}; f_{1:N}) \\ &\leq C_1 N\beta_{N,d}\gamma_N \end{aligned} \tag{21}$$

where $C_1 = 8\sigma^2 C_2$ and $\gamma_N = \max\mathrm{I}(y_{1:N}; f_{1:N})$ is the maximum information gain after N steps of sampling. Thus,

$$P(R_N \leq \sqrt{C_1 N\beta_{N,d}\gamma_N}) \geq 1-\delta \tag{22}$$

Note that the form of our main theorem is quite similar with [Srinivas *et al.*, 2009], although our stage-wise constant $\beta_{n,d}$ is different and includes a distance term.

□

Experiment Details

The experiment results on the Branin and Ackley functions are shown in Figure 3 and Figure 4. We observe a similar tradeoff in distUCB compared with other policies.

Ethical Statement

There are no ethical issues.

Acknowledgments

We thank Professor Chung-Piaw Teo for his constructive suggestions while preparing this manuscript.

References

[Balandat *et al.*, 2019] Maximilian Balandat, Brian Karrer, Daniel R Jiang, Samuel Daulton, Benjamin Letham, Andrew Gordon Wilson, and Eytan Bakshy. Botorch: Programmable bayesian optimization in pytorch. *arXiv preprint arXiv:1910.06403*, 117, 2019.

[Bertsekas *et al.*, 1997] Dimitri Bertsekas, John Tsitsiklis, and Cynara Wu. Rollout algorithms for combinatorial optimization. *Journal of Heuristics*, 3:245–262, 11 1997.

[Gardner *et al.*, 2014] Jacob R Gardner, Matt J Kusner, Zhixiang Eddie Xu, Kilian Q Weinberger, and John P Cunningham. Bayesian optimization with inequality constraints. In *ICML*, volume 2014, pages 937–945, 2014.

[Goodson *et al.*, 2016] Justin Goodson, Barrett Thomas, and Jeffrey Ohlmann. A rollout algorithm framework for heuristic solutions to finite-horizon stochastic dynamic programs. *European Journal of Operational Research*, 258, 10 2016.

[Hernández-Lobato *et al.*, 2014] José Hernández-Lobato, Matthew Hoffman, and Zoubin Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. *Advances in Neural Information Processing Systems*, 1, 06 2014.

[Hu *et al.*, 2022] Shouri Hu, Haowei Wang, Zhongxiang Dai, Bryan Kian Hsiang Low, and Szu Hui Ng. Adjusted expected improvement for cumulative regret minimization in noisy bayesian optimization. *arXiv preprint arXiv:2205.04901*, 2022.

[Jiang *et al.*, 2020] Shali Jiang, Daniel Jiang, Maximilian Balandat, Brian Karrer, Jacob Gardner, and Roman Garnett. Efficient nonmyopic bayesian optimization via one-shot multi-step trees. *Advances in Neural Information Processing Systems*, 2020.

[Lam and Willcox, 2017] Remi Lam and Karen Willcox. Lookahead bayesian optimization with inequality constraints. *Advances in neural information processing systems*, 12 2017.

[Lee *et al.*, 2021] Eric Hans Lee, David Eriksson, Valerio Perrone, and Matthias W. Seeger. A nonmyopic approach to cost-constrained bayesian optimization. *37th Conference on Uncertainty in Artificial Intelligence*, 2021.

[Marchant and Ramos, 2012] Roman Marchant and Fabio Ramos. Bayesian optimisation for intelligent environmental monitoring. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 2242–2249, 10 2012.

[Mockus and others, 1978] J. Mockus et al. The application of bayesian methods for seeking the extremum. *Towards Global Optimization*, 2, 1978.

[Rasmussen and Williams, 2006] Carl E. Rasmussen and Christopher Williams. Gaussian processes for machine learning. *MIT*, 2006.

[Scott *et al.*, 2011] Warren Scott, Peter Frazier, and Warren Powell. The correlated knowledge gradient for simulation optimization of continuous parameters using gaussian process regression. *SIAM Journal on Optimization*, 21:996–1026, 07 2011.

[Srinivas *et al.*, 2009] 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.

[Yue and Kontar, 2020] Xubo Yue and Raed AL Kontar. Why non-myopic bayesian optimization is promising and how far should we look-ahead? a study via rollout. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2808–2818. PMLR, 26–28 Aug 2020.