Multi-Task Gaussian Process Upper Confidence Bound for Hyperparameter Tuning

Bo Shen^a, Raghav Gnanasambandam^a, Rongxuan Wang^a, and Zhenyu (James) Kong^{a*}

^aGrado Department of Industrial and Systems Engineering, Virginia Tech, Blacksburg, US

*Corresponding Author

Abstract

In many scientific and engineering applications, Bayesian optimization (BO) is a powerful tool for hyperparameter tuning of a machine learning model, materials design and discovery, etc. BO guides the choice of experiments in a sequential way to find a good combination of design points in as few experiments as possible. It can be formulated as a problem of optimizing a “black-box” function. Different from single-task Bayesian optimization, Multi-task Bayesian optimization is a general method to efficiently optimize multiple different but correlated “black-box” functions. The previous works in Multi-task Bayesian optimization algorithm queries a point to be evaluated for all tasks in each round of search, which is not efficient. For the case where different tasks are correlated, it is not necessary to evaluate all tasks for a given query point. Therefore, the objective of this work is to develop an algorithm for multi-task Bayesian optimization with automatic task selection so that only one task evaluation is needed per query round. Specifically, a new algorithm, namely, multi-task Gaussian process upper confidence bound (MT-GPUCB), is proposed to achieve this objective. The MT-GPUCB is a two-step algorithm, where the first step chooses which query point to evaluate, and the second step automatically selects the most informative task to evaluate. Under the bandit setting, a theoretical analysis is provided to show that our proposed MT-GPUCB is *no-regret* under some mild conditions. Our proposed algorithm is verified experimentally on a range of synthetic functions as well as real-world problems. The results clearly show the advantages of our query strategy for both design point and task.

Keywords: Multi-task Gaussian Process Upper Confidence Bound; Automatic Task Selection; *No-Regret*; Hyperparameter Tuning.

1 Introduction

In machine learning, the proper setting of hyperparameters in the algorithms (for example, regularization weights, learning rates, etc.) is crucial for achieving satisfying performance. A poor setting of hyperparameters may result in a useless model even when the model structure is correct. In materials design and discovery, how to choose the chemical structure, composition, or processing conditions of a material to meet design criteria is a key problem. There are many other examples of design problems in advertising, healthcare informatics,

manufacturing and so on. Any significant advances in automated design can result in immediate product improvements and innovation in a wide area of domains.

Bayesian optimization (BO) [Jones et al. \(1998\)](#); [Shahriari et al. \(2015\)](#) has emerged as a powerful tool for these various design problems. Fundamentally, it is a general method to efficiently optimize “black-box” functions. There is only weak prior knowledge available, typically characterized by expensive and noisy function evaluations, a lack of gradient information, and high levels of non-convexity. BO is impacting a wide range of areas, including combinatorial optimization [Williams et al. \(2000\)](#); [Hutter et al. \(2011\)](#), automatic machine learning [Bergstra et al. \(2011\)](#); [Snoek et al. \(2012\)](#), material design [Frazier and Wang \(2016\)](#), and reinforcement learning [Brochu et al. \(2010\)](#).

Bayesian optimization is a sequential model-based approach to solving the “black-box” optimization problem. For a given task, the method iterates over the following steps until the available computational budget is exhausted: 1) a set of evaluated points is used to learn a probabilistic regression model $p(f)$ of the objective function f [typically in the form of a Gaussian process (GP) [Rasmussen \(2003\)](#)]; 2) $p(f)$ is used to induce a proper acquisition function that leverages the uncertainty in the posterior to trade-off exploration and exploitation; 3) the acquisition function is optimized to determine the next query point to be evaluated; and 4) the regression data set in 1) is augmented with the newly evaluated point.

Different from single-task Bayesian optimization introduced above, Multi-task Bayesian optimization (MTBO) [Swersky et al. \(2013\)](#) is a general method to efficiently optimize multiple different but correlated “black-box” functions. The settings for multi-task Bayesian optimization widely exist in many real-world applications. For example, the K -fold cross-validation [Bengio and Grandvalet \(2004\)](#) is a widely used technique to estimate the generalization error of machine learning model for a given set of hyperparameters. However, it needs to retrain a model K times using all K training-validation splits. The validation errors of a model trained on K different training-validation splits can be treated as K “black-box” functions, which need to be minimized as K different tasks. These K tasks will be highly correlated since the data are randomly partitioned among K training-validation splits. The performance of our proposed method in the application of fast cross-validation [Swersky et al. \(2013\)](#); [Moss et al. \(2020\)](#) is presented in Section 6.1, which aims at mini-

mizing the average validation errors in K -fold cross-validation.

Another motivating example comes from additive manufacturing. That is, a material scientist might want to find the raw material properties (for example, thermal conductivity, laser absorptivity, surface tension, etc.), which are difficult to measure at high temperatures [Yan et al. \(2020\)](#). These raw material property values are the input of the Computational Fluid Dynamics (CFD) software [Flow Science \(2019\)](#) to simulate the melt pool dynamics for different printing conditions. While the printing conditions (for example, laser power and dwell time) can be varied for a given type of material, the raw material properties are generally provided in the forms of a suggested range, not specific values. Appropriate setting of these raw material properties are very critical to the quality of simulation output. For one specific printing setting, the task is to determine raw material property values to ensure the simulation quality. In the case of multiple printing settings, there are multiple tasks [that](#) need to be optimized. The performance of our proposed method in the application of raw material properties determination for multiple printing settings is presented in [Section 6.2](#).

Gaussian process upper confidence bound (GP-UCB) [Srinivas et al. \(2012\)](#) is one of the popular and efficient algorithms in Bayesian optimization using confidence bounds to deal with an exploitation-exploration trade-off. GP-UCB cannot be directly applied to the multi-task case, where one specific printing condition can be treated as one single task. Based on GP-UCB, [Dai et al. \(2020\)](#) developed a method to optimize each task separately, where the objective functions for all tasks are modeled by multi-output Gaussian process [Williams et al. \(2007\)](#). However, there are two drawbacks for this strategy: first, it is not sample-efficient since every task should be evaluated independently; second, for different tasks, the algorithm may find different values for the same raw material properties, which makes the results difficult to be used.

As an alternative, MTBO [Swersky et al. \(2013\)](#) aims to optimize all tasks simultaneously and make use of the shared information and structure to speed up the optimization process. Regardless of MTBO’s well-known efficiency, there is still space of improvement to better address the need of our applications of cross-validation and raw material properties determination identified above. Specifically, it is neither effective nor efficient for the MTBO algorithm to query a point to be evaluated for all tasks simultaneously in each

round since these tasks are correlated. Therefore, to fully utilize the correlations among tasks may provide a room for further improvement of the accuracy of MTBO algorithm.

The objective of this work is to develop an algorithm for multi-task Bayesian optimization with automatic task selection so that only one task evaluation is needed per query round. To achieve this objective, a new algorithm, namely, multi-task Gaussian process upper confidence bound (MT-GPUCB), is proposed. The MT-GPUCB is a two-step algorithm, where the first step chooses which query point to evaluate, and the second step automatically selects the most informative task to evaluate. The contributions of this paper are summarized as follows:

- Multi-task Gaussian process upper confidence bound is proposed to provide the query strategy for both design point and task in each round to balance exploration-exploitation across tasks.
- Under some mild conditions, multi-task Gaussian process upper confidence bound is proved to be a *no-regret* learning algorithm. That is, the algorithm will converge to the optimal solution in the end.

Remark 1 (Discussion on mean/integrated response BO). *The strategy of our method to select query point and task in each round is improved from MTBO Swersky et al. (2013), where our method selects the tasks by considering the task correlations but MTBO does not. It is conceptually similar to the mean/integrated response (BO) proposed by Williams et al. (2000); Janusevskis and Le Riche (2013); Toscano-Palmerin and Frazier (2018); Tan (2020). These reference papers consider that optimization problems (the objective function is the sum or integral of an output) depend on control and environmental variables. Those control and environmental variables are analogous to query points and tasks in our proposed method. The objective of the mean/integrated response BO is to find the optimal control variable under all environmental variables. Therefore, mean/integrated response BO can be considered as multi-task Bayesian optimization. However, there are distinct differences between our method and the above references in terms of modeling the relationship of tasks. For the mean/integrated response BO, they assume that the environmental variables (tasks) follow a specific distribution independently. As a result, these methods' performance deteriorates when a small number of tasks (≤ 10 for example) are highly correlated. This is*

because, in mean/integrated response BO, the correlation between tasks is not fully utilized by modeling them independently. In our paper, there is no distribution assumption on the tasks. Instead, our method uses a multi-task Gaussian process to model the correlation between tasks through a task correlation matrix, where the task correlation is better handled. Besides, the acquisition function in our method is more intuitive and simpler to implement without involving computationally expensive integration used by the above literature methods. In [Toscano-Palmerin and Frazier \(2018\)](#), their experiment results show that one of our benchmarks, namely, MTBO [Swersky et al. \(2013\)](#), had a competitive performance with their proposed method when the number of tasks is small ($m \leq 10$). In our experiments, our method performs much better than MTBO, where the number of tasks is small ($m \leq 10$). Due to this observation and implementation issue, the proposed method in [Toscano-Palmerin and Frazier \(2018\)](#) is not compared in our paper. Despite the difference in modeling, it is still necessary to compare with the state-of-the-art in mean/integrated response BO. EIQ from [Tan \(2020\)](#) is used as one of the benchmarks in Sections 5 and 6 since it is the most representative and effective method in the literature. In our paper, the case of a small number of tasks is the focus of our algorithm, where all experiments in our paper have less than ten tasks. For all experiments, our algorithm exhibits the best performance among all benchmarks because our algorithm can explore the task correlation explicitly.

The remainder of this paper is organized as follows. A brief review of theoretical foundations in single task Bayesian optimization is provided in Section 2. The proposed Multi-task Gaussian process upper confidence bound algorithm is introduced in Section 3. The regret analysis of our proposed algorithm is provided in Section 4, followed by the numerical studies and actual case studies in Sections 5 and 6 for testing and validation of the proposed algorithm. Finally, the conclusions and future work are discussed in Section 7.

2 Theoretical Foundations of Bayesian Optimization

To begin with, single task Bayesian optimization (BO) [Snoek et al. \(2012\)](#) considers the problem of sequentially optimizing an unknown function $f : \mathcal{X} \rightarrow \mathbb{R}$. The goal is to find

$$\mathbf{x}^* = \arg \max_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}) \quad (1)$$

as soon as possible. To model the “black-box” function f , the Gaussian process assumption on f is introduced, which is a Bayesian statistical approach for modeling functions. A standard gaussian process (GP) Rasmussen (2003) is a stochastic nonparametric approach for regression that extends the concept of multivariate Gaussian distributions to infinite dimensions. To enforce implicit properties like smoothness without relying on any parametric assumptions, a GP is used to model f , written as $\mathcal{GP}(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$, which is completely specified by its mean and covariance functions, where $\mu(\mathbf{x}) = \mathbb{E}(f(\mathbf{x}))$ represents the distribution mean that is often set to $\mathbf{0}$, and $k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - \mu(\mathbf{x}))(f(\mathbf{x}') - \mu(\mathbf{x}'))]$ is the kernel matrix. The kernel function k encodes smoothness properties of sample functions f drawn from the GP, which is the most critical component for GP. Finite-dimensional linear, squared exponential, and Matérn kernels are the common choices for the kernel functions Williams and Rasmussen (2006). Throughout this paper, we further assume bounded variance by restricting $k(\mathbf{x}, \mathbf{x}) \leq 1$, for all $\mathbf{x} \in \mathcal{X}$.

A major computational benefit of working with GPs is that posterior inference can be performed in closed form. Given collected observations $\mathbf{y}_T = [y_1, \dots, y_T]^\top$ at design inputs $A_T = \{\mathbf{x}_1, \dots, \mathbf{x}_T\}$, the posterior over f is a GP distribution again, with mean $\mu_T(\mathbf{x})$, covariance $k_T(\mathbf{x}, \mathbf{x}')$, and variance $\sigma_T^2(\mathbf{x})$:

$$\begin{aligned}\mu_T(\mathbf{x}) &= \mathbf{k}_T(\mathbf{x})^\top (\mathbf{K}_T + \sigma^2 \mathbf{I})^{-1} \mathbf{y}_T \\ k_T(\mathbf{x}, \mathbf{x}') &= k(\mathbf{x}, \mathbf{x}') - \mathbf{k}_T(\mathbf{x})^\top (\mathbf{K}_T + \sigma^2 \mathbf{I})^{-1} \mathbf{k}_T(\mathbf{x}') \\ \sigma_T^2(\mathbf{x}) &= k_T(\mathbf{x}, \mathbf{x}),\end{aligned}\tag{2}$$

where $\mathbf{k}_T(\mathbf{x}) = [k(\mathbf{x}_1, \mathbf{x}), \dots, k(\mathbf{x}_T, \mathbf{x})]^\top$ and $\mathbf{K}_T$ is the positive definite kernel matrix $[k(\mathbf{x}, \mathbf{x}')]_{\mathbf{x}, \mathbf{x}' \in A_T}$.

3 Multi-Task Gaussian Process Upper Confidence Bound

In our setting of multi-task Bayesian optimization, we can define a “black-box” function over the composite set $\mathcal{X} \times \mathcal{Z}$, namely, $f : \mathcal{X} \times \mathcal{Z} \rightarrow \mathbb{R}$, where $\mathcal{X} \in \mathbb{R}^d$ is the design space and $\mathcal{Z} = \{1, \dots, M\}$ is the set of task index. Assume that the (noisy) observations $A_T = \{(\mathbf{x}_t, m_t, y_t)\}_{t=1}^T$ for $y_t = f(\mathbf{x}_t, m_t) + \epsilon_t$ are available, where $f(\mathbf{x}, m)$ is the result of querying point $\mathbf{x}$ on task m , and ϵ_t is independent Gaussian noise following $N(0, \sigma^2)$. Our objective is to develop an algorithm that contains: Step 1. Choose the query point $\mathbf{x}_t \in \mathcal{X}$ that benefits for all tasks; Step 2. Select the most informative task (i.e., $m_t \in \mathcal{Z}$)

to evaluate for better sample efficiency.

To model the correlations between tasks, our underlying Gaussian process model over f must be extended across the task space. By defining a kernel over $\mathcal{X} \times \mathcal{Z}$, the posterior over f can be similarly calculated through (2). Although increasing the dimension of the kernel for $\mathcal{X}$ to incorporate $\mathcal{Z}$ provides a very flexible model, it is argued by Kandasamy et al. (2017) that overly flexible models can harm optimization speed by requiring too much learning, restricting the sharing of information across the task space. Therefore, it is common to use more restrictive separable kernels that better model specific aspects of the given problem. A common kernel for multi-task spaces is the intrinsic coregionalization kernel of Álvarez and Lawrence (2011). This kernel defines a co-variance between design parameter and task pairs of

$$k_{\text{multi}} = k_{\mathcal{X}}(\mathbf{x}, \mathbf{x}') \otimes k_{\mathcal{Z}}(m, m'),$$

where $\otimes$ denotes the Kronecker product, $k_{\mathcal{X}}$ measures the relationship between inputs, and $k_{\mathcal{Z}}$ measures the relationship between tasks, allowing the sharing of information across the task space. Once the composite kernel k_{multi} is determined, the prediction of f over the composite set $\mathcal{X} \times \mathcal{Z}$ follows (2).

The following assumption is made based on our applications.

Assumption 1. $\mathbf{x}^* = \arg \max_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}, m)$ for all task m . That is, $f(\mathbf{x}, m)$ has same set of optimal solutions (not necessary to be unique) for all different tasks. However, they may have a very different trend globally.

Assumption 1 is valid for our application of raw material properties determination. Different printing settings represent different tasks ($m = 1, \dots, M$), however, they share the same raw material properties ($\mathbf{x}^*$) that need to be identified. Our objective is to find the common optimal solution shared for different tasks

$$\mathbf{x}^* = \arg \max_{\mathbf{x} \in \mathcal{X}} \sum_{m=1}^M f(\mathbf{x}, m) \quad (3)$$

using as few samples as possible.

In the setting of single-task Bayesian optimization, Bayesian experimental design rule used in Chaloner and Verdinelli (1995)

$$\mathbf{x}_t = \arg \max_{\mathbf{x} \in \mathcal{X}} \sigma_{t-1}(\mathbf{x}), \quad (4)$$

can be wasteful since it aims at decreasing uncertainty globally, not just searching where maxima might be. Another approach to (1) is to pick points as $\mathbf{x}_t = \arg \max_{\mathbf{x} \in \mathcal{X}} \mu_{t-1}(\mathbf{x})$, maximizing the expected reward based on the posterior so far. However, this pure exploitation rule is too greedy and tends to get stuck in shallow local optima. Instead, GP-UCB Srinivas et al. (2012) is a combined strategy to choose

$$\mathbf{x}_t = \arg \max_{\mathbf{x} \in \mathcal{X}} \mu_{t-1}(\mathbf{x}) + \beta_t^{1/2} \sigma_{t-1}(\mathbf{x}), \quad (5)$$

where β_t are appropriate constants, $\mu_{t-1}(\mathbf{x})$ and $\sigma_{t-1}(\mathbf{x})$ can be calculated from the GP posterior defined in (2). This objective prefers both points $\mathbf{x}$ where f is uncertain (large $\sigma_{t-1}(\cdot)$, exploration) and such where we expect to achieve high rewards (large $\mu_{t-1}(\cdot)$, exploitation) since it implicitly negotiates the exploration-exploitation tradeoff.

However, GP-UCB is not ready to be used for multi-task Bayesian optimization since there is only one task in the objective. In our proposed multi-task Bayesian optimization, we would like to have an algorithm that automatically selects a task to evaluate for a given point to speed up the optimization because it is not necessary to evaluate all tasks for a given query point due to the correlations among multi-tasks. Motivated by GP-UCB and

Algorithm 1: MT-GPUCB

Input: Input space $\mathcal{X}, \mathcal{Z}$; GP prior $\mu_0 = 0, \sigma_0, k_{\mathcal{X}}$ and $k_{\mathcal{Z}}$

```

1 for  $t=1,2,\dots$  do
2   Step 1: Choose  $\mathbf{x}_t = \arg \max_{\mathbf{x} \in \mathcal{X}} \sum_{m=1}^M \mu(\mathbf{x}, m) + \beta_t^{1/2} \sigma_{t-1}(\mathbf{x}, m)$  // Maximize
   summation of UCB
3   Step 2: Choose  $m_t = \arg \max_{m \in [M]} \sigma_{t-1}(\mathbf{x}_t, m)$  // Maximize Information Gain
4   Observe  $y_t = f(\mathbf{x}_t, m_t) + \epsilon_t$ 
5   Perform GP and Bayesian update to obtain  $\mu_t(\mathbf{x}, m)$  and  $\sigma_t(\mathbf{x}, m)$  for all  $m$ 

```

Bayesian experimental design, our proposed MT-GPUCB algorithm 1 is to provide (1) a query strategy for design point and (2) automatic task selection to be performed at the query point selected in (1). Specifically, our proposed algorithm contains two main steps.

- Step 1: the algorithm selects query point $\mathbf{x}_t$ that has the largest summation of UCB

for all tasks (as shown in Line 2 algorithm 1), namely,

$$\mathbf{x}_t = \arg \max_{\mathbf{x} \in \mathcal{X}} \sum_{m=1}^M \mu(\mathbf{x}, m) + \beta_t^{1/2} \sigma_{t-1}(\mathbf{x}, m), \quad (6)$$

which means that the selected query point $\mathbf{x}_t$ has the “best” potential to perform good for all tasks simultaneously. The objective function in (6) is the summation form of (5) over different tasks so that it is a reasonable upper bound on the objective function in (3).

- Step 2: given the query point selected from step 1, the algorithm selects the task that has the largest information gain to play (as shown in Line 3 algorithm 1), namely,

$$m_t = \arg \max_{m \in [M]} \sigma_{t-1}(\mathbf{x}_t, m). \quad (7)$$

That is, the most informative task will be selected to represent all the other tasks. The above criteria in (7) origins from the criteria in (4). However, our criteria selects which task to evaluate given the query point selected in Step 1.

By querying one task for a given point in one round, our algorithm can achieve better sample efficiency than querying all tasks in one round. Despite the simplicity and easy interpretation of algorithm 1, theoretical performance guarantee can also be derived in next section, where the regret in the bandit setting will be analyzed.

4 Regret Analysis for Multi-Task Gaussian Process Upper Confidence Bound

A natural performance metric in the context of bandit [Srinivas et al. \(2012\)](#) is the cumulative regret, the loss in reward due to not knowing the optimal solution of f beforehand. Given the assumption 1, for our choice $\mathbf{x}_t$ in round t , define instantaneous multi-task regret as $r_t := \frac{1}{M} \sum_{m=1}^M f(\mathbf{x}^*, m) - f(\mathbf{x}_t, m)$. Even though only one task is evaluated each round, the regret for all tasks are incurred since all tasks are equally important. This is quite different from the definition in [Krause and Ong \(2011\)](#), where only the evaluated task is defined in the instantaneous regret. Our definition also add difficulties in our theoretical analysis. The cumulative regret after T rounds is the sum of instantaneous regrets: $R(T) := \sum_{t=1}^T r_t$. A desirable asymptotic property of an algorithm is to be *no-regret* (also called *sub-linear* convergence rate): $\lim_{T \rightarrow +\infty} \frac{R(T)}{T} = 0$

Similar to the GP-UCB regret analysis [Krause and Ong \(2011\)](#), our bound depends on a term capturing the information gain between query choices and “black-box” functions. Specifically, define

$$\gamma_T := \arg \max_{\{(\mathbf{x}'_t, m'_t) \in \mathcal{X} \times \mathcal{Z}\}_{t=1}^T} I(\mathbf{y}_T; \mathbf{f}_T) \quad (8)$$

is the maximum information gain between T rounds of queries, where $I(\mathbf{y}_T; \mathbf{f}_T)$ is information gain [Cover and Thomas \(1991\)](#). It is the mutual information between and observation $\mathbf{y}_T = \mathbf{f}_T + \epsilon_T$ at these points:

$$I(\mathbf{y}_T; \mathbf{f}_T) = H(\mathbf{y}_T) - H(\mathbf{y}_T | \mathbf{f}_T), \quad (9)$$

quantifying the reduction in uncertainty about f from revealing $\mathbf{y}_T$. Here, $\mathbf{f}_T = [f(\mathbf{x}, m)]_{(\mathbf{x}, m) \in \mathcal{X} \times \mathcal{Z}}$ and $\epsilon_T \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$. For a Gaussian, $H(\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})) = \frac{1}{2} \log |2\pi e \boldsymbol{\Sigma}|$, so that in our setting $I(\mathbf{y}_T; \mathbf{f}_T) = \frac{1}{2} \log |\mathbf{I} + \sigma^{-2} \mathbf{K}_T|$, where $\mathbf{K}_T = [k_{\mathcal{X}}(\mathbf{x}, \mathbf{x}') \times k_{\mathcal{Z}}(m, m')]_{(\mathbf{x}, m), (\mathbf{x}', m') \in \mathcal{X} \times \mathcal{Z}}$.

Three conditions are analyzed for regret bounds. Note that none of the results subsume each other, and so all cases may be of use. For the first two conditions, we assume a known GP prior and (1) a finite $\mathcal{X}$ and (2) infinite $\mathcal{X}$ with mild assumptions about k_{multi} . A third (and perhaps more “agnostic”) way to express assumptions about f is to require that f has low “complexity” as quantified in terms of the Reproducing Kernel Hilbert Space (RKHS, [Wahba \(1990\)](#)) norm associated with kernel k_{multi} . The following theorem shows that our MT-GPUCB algorithm converges to the optimal solution with *sub-linear* convergence rate.

Theorem 1. *Under assumption [1](#) and $k(\mathbf{x}, \mathbf{x}) \leq 1$, pick $\delta \in (0, 1)$, suppose one of the following conditions holds*

1. *$\mathcal{X}$ is finite, f is sampled from a multi-task gaussian process with known noise variance σ^2 , and $\beta_t = 2 \log(M|\mathcal{X}|t^2\pi^2/6\delta)$.*
2. *$\mathcal{X} \subset [0, r]^d$ be compact and convex, $d \in \mathbb{N}, r > 0$. Suppose that the kernel k_{multi} satisfies the following high probability bound on the derivatives of GP sample paths f : for some constants $a, b > 0$,*

$$\mathcal{P}\left(\sup_{\mathbf{x} \in \mathcal{X}} |\partial f(\mathbf{x}, m) / \partial \mathbf{x}_j| > L\right) \leq ae^{-(L/b)^2}, j = 1, \dots, d, m = 1, \dots, M.$$

Choose $\beta_t = 2 \log(2Mt^2\pi^2/(3\delta)) + 2d \log(t^2 r d b \sqrt{\log(4dMa/\delta)})$.

3. $\mathcal{X}$ is arbitrary. Let $\|f\|_{k_{\text{multi}}} \leq B$. The noise variables ϵ_t form an arbitrary martingale difference sequence, namely, $\mathbb{E}[\epsilon_t | \epsilon_1, \dots, \epsilon_{t-1}] = 0$ for all $t \in \mathbb{N}$, uniformly bounded by σ . Set $\beta_t = 2B^2 + 300\gamma_t \log^3(t/\delta)$.

Running algorithm 1, a regret bound can be obtained with high probability. Specifically,

$$\mathcal{P}(R_T \leq \sqrt{CT\beta_T\gamma_T} + 2, \forall T \geq 1) \geq 1 - \delta,$$

where $C = \frac{8}{\log(1+\sigma^{-2})}$.

Proof. See proof in Appendix A. □

Theorem 1 shows the regret bound under three independent conditions. The conditions (1) and (2) are corresponding to the cases of finite discrete and convex compact design space, respectively. Under condition (1) or (2), the cumulative regret is bounded in terms of the maximum information gain with respect to the multi-task GP defined over $\mathcal{X} \times \mathcal{Z}$. The smoothness assumption on $k(\mathbf{x}, \mathbf{x}')$ in conditions (2) disqualifies GPs with highly erratic sample paths. It holds for stationary kernels $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x} - \mathbf{x}')$ which are four times differentiable (Theorem 5 of Ghosal et al. (2006)) such as the Squared Exponential and Matérn kernels with $\nu > 2$. Under condition (3), a regret bound is obtained in a more agnostic setting, where no prior on f is assumed, and much weaker assumptions are made about the noise process. The theoretical results in Theorem 1 will be further verified experimentally using synthetic functions in Section 5.

Importantly, note that some upper bounds are needed for the information gain γ_T defined in (8) so that our algorithm 1 is a *no-regret*. γ_T is a problem-dependent quantity properties of both the kernel and the design input space that will determine the growth of regret. In Krause and Ong (2011) (section 5.2), the upper bound of γ_T has been derived for common kernels such as finite-dimensional linear, squared exponential, and Matérn kernels ($\nu > 1$) to guarantee that the *no-regret* conclusion is valid. For the Matérn kernels ($\nu > 1$), $\gamma_T \leq \mathcal{O}(MT^{d(d+1)/(2\nu+d(d+1))} \log T + M \log T)$.

Remark 2 (Discussion on mild conditions). *Under the Gaussian process assumption with proper kernel (for example, Matérn kernels with $\nu > 2$), condition (1) or (2) can be easily satisfied in many problems. For example, the number of layers and the number of neurons per layer in a deep neural network can only take discrete values Garrido-Merchán and*

Hernández-Lobato (2020); Maftouni et al. (2020), which is suitable for condition (1); the BoxConstraint and KernelScale in a Gaussian kernel support vector machine (SVM) Han et al. (2012) can form a compact convex design space, which is suitable for condition (2).

5 Numerical Study

To evaluate the performance of the proposed MT-GPUCB algorithm 1, a numerical illustration of our algorithm on six synthetic functions is conducted in this section. In all analyses, EIQ Tan (2020), MTBO Swersky et al. (2013), CGP-UCB Krause and Ong (2011), and GP-UCB Srinivas et al. (2012) are selected as benchmarks for comparison with the proposed algorithm, which are state-of-the-art methods in the related area. EIQ is one of the proposed acquisition functions that are designed for expected quadratic loss. MTBO is an entropy search based algorithm for multi-task BO while CGP-UCB is the algorithm assumed that the task to be evaluated is randomly selected. GP-UCB in the setting of multi-task BO represents the algorithm, where all tasks are evaluated in each round without considering the task correlation. Throughout this paper, the maximin Latin hypercube design Joseph and Hung (2008), which demonstrates good space filling properties and first-dimension projection properties, is implemented to obtain initial design points. The codes of MT-GPUCB are implemented in Matlab 2019a. The CPU of the computer used in the experiments is an Intel[®] Core[™] Processor i7-6820HQ (Quad Core 2.70 GHz, 3.60 GHz Turbo, 8MB 45W).

Choice of β_t in (6) for practical consideration: β_t , as specified by Theorem 1, has unknown constants and usually tends to be conservative in practice Srinivas et al. (2012). For better empirical performance, a more aggressive strategy is required. Following the recommendations in Kandasamy et al. (2015, 2017, 2019), we set it to be of the correct “order”, namely, $\beta_t = 0.2d \log(2t)$. It offers a good tradeoff between exploration and exploitation. Note that this captures the correct dependence on d and t in Theorem 1.

5.1 Experimental Settings

Ackley, Bohachevsky, Colville, Levy, Powell, and Rastrigin are selected from Surjanovic and Bingham (2020) for numerical comparison, where most of them have many local optimal solutions so that it is hard to optimize them. For each of those function, there are coefficients can be changed to simulate multiple functions with the same optimal solution to match the

Table 1: Optimal solution and coefficient vector for synthetic functions used in this numerical study.

Functions	$\mathbf{x}^*$	Task Encode Matrix ($\mathbf{E}$)
Ackley	$(0, 0)$	$\begin{pmatrix} 20 & 0.2 \\ 15 & 0.3 \end{pmatrix}$
Bohachevsky	$(0, 0)$	$\begin{pmatrix} 1 & 2 \\ 0.8 & 1.8 \\ 1.2 & 2.2 \end{pmatrix}$
Colville	$(1, 1, 1, 1)$	$\begin{pmatrix} 100 & 90 & 10.1 & 19.8 \\ 90 & 80 & 10.1 & 19.8 \\ 100 & 90 & 9.1 & 18 \\ 90 & 80 & 9.1 & 18 \end{pmatrix}$
Levy	$(1, 1)$	$\begin{pmatrix} 1 & 10 & 1 \\ 1.2 & 9 & 1.2 \end{pmatrix}$
Powell	$(0, 0, 0, 0)$	$\begin{pmatrix} 10 & 5 & 2 & 10 \\ 10 & 5 & 3 & 9 \\ 8 & 6 & 2 & 10 \end{pmatrix}$
Rastrigin	$(0, 0)$	$\begin{pmatrix} 8 & 10 \\ 10 & 8 \\ 9 & 9 \end{pmatrix}$

multi-task setting described in assumption 1. The forms of these functions are summarized as follows ($a, b, \dots$ are the coefficients can be tuned to generate different tasks except Bohachevsky):

- i. Ackley ($a, b > 0$): $a \exp(-b\sqrt{(x_1^2 + x_2^2)/2}) + \exp((\cos 2\pi x_1 + \cos 2\pi x_2)/2) - a - \exp(1)$
- ii. Bohachevsky: $f_1(\mathbf{x}) = -x_1^2 - 2x_2^2 + 0.3 \cos(3\pi x_1) + 0.4 \cos(4\pi x_2) - 0.7$;
 $f_2(\mathbf{x}) = -x_1^2 - 2x_2^2 + 0.3 \cos(3\pi x_1) \cos(4\pi x_2) - 0.3$;
 $f_3(\mathbf{x}) = -x_1^2 - 2x_2^2 + 0.3 \cos(3\pi x_1 + 4\pi x_2) - 0.3$.
- iii. Colville ($a, b, c, d > 0$): $-a(x_1^2 - x_2)^2 - (x_1 - 1)^2 - (x_3 - 1)^2 - b(x_3^2 - x_4)^2 - c((x_2 - 1)^2 + (x_4 - 1)^2) - d(x_2 - 1)(x_4 - 1)$

- iv. Levy ($a, b, c > 0$): $-a \sin^2(\pi w_1) - (w_1 - 1)^2[1 + b \sin^2(\pi w_1 + 1)] - c(w_2 - 1)^2[1 + \sin^2(2\pi w_2)]$, where $w_i = 1 + \frac{x_i - 1}{4}$ for all $i = 1, 2$.
- v. Powell ($a, b, c, d > 0$): $-(x_1 + ax_2)^2 - b(x_3 - x_4)^2 - (x_2 - cx_3)^4 - d(x_1 - x_4)^4$
- vi. Rastrigin ($a, b > 0$): $-(a + b) - (x_1^2 - a \cos(2\pi x_1)) - (x_2^2 - b \cos(2\pi x_2))$

In addition, the optimal solution and coefficient vector to encode different tasks for each function are listed in the Table 1. The visualization of 4 synthetic functions in Task 1 that lie in two-dimensional space is shown in Figure 1 to show the complex nature in optimization.

Figure 1: Visualization of 4 synthetic functions in two dimensional space (they are in the form of minimization problems).

The design space $\mathcal{X}$, the number of tasks, the number of initial design points, and the number of additional function evaluations for six synthetic functions are summarized in Table 2. The number of initial design points is calculated from $5Md$, where d and M are

Table 2: Summary of experimental settings in this numerical study.

Functions	$\mathcal{X}$	Number of Tasks (M)	Number of Initial Points	Number of Evaluations (T)
Ackley	$[-32.768, 32.768]^2$	2	20	60
Bohachevsky	$[-10, 10]^2$	3	30	30
Colville	$[-10, 10]^4$	4	80	80
Levy	$[-10, 10]^2$	2	20	40
Powell	$[-5, 5]^4$	3	60	80
Rastrigin	$[-5.12, 5.12]^2$	3	30	80

the dimensions of design space and the number of tasks, respectively. For each synthetic function, the number of additional function evaluations (T) is set to the number that one of the methods can achieve a near-optimal performance. Due to the different characteristics of

synthetic functions (for example, the number of tasks, the dimension of design space, and the complexity of optimization), the number of function evaluations (T) are quite different for different synthetic functions.

Once the task encode matrix is known, each row represents the coefficient for each task as the input to calculate the r in (10) for $k_Z(m, m') \in \mathbb{R}^{M \times M}$. Specifically, let $k_Z(m, m') = k_{\text{Mat}}(\mathbf{E}(m, :), \mathbf{E}(m', :))$. In terms of $k_X(x, x')$ and $k_Z(m, m') \in \mathbb{R}^{M \times M}$, Matérn ($\nu = 5/2$) kernels are selected, where the distance matrix in $k_Z(m, m')$ are constructed by the coefficients of different tasks. Note that the product of two Matérn ($\nu = 5/2$) kernels is still Matérn ($\nu = 5/2$). Thus, the smoothness assumption of the kernel in Theorem 1 under condition (2) is satisfied. The Matérn kernel [Genton \(2001\)](#) is given by

$$k_{\text{Mat}}(\mathbf{x}, \mathbf{x}') = (2^{1-\nu}/\Gamma(\nu))r^\nu B_\nu(r), r = (\sqrt{2\nu}/l)\|\mathbf{x} - \mathbf{x}'\|, \quad (10)$$

where ν controls the smoothness of sample paths (the smaller, the rougher) and B_ν is a modified Bessel function. The kernel parameters are learned through maximizing the marginal log likelihood. We maximized the acquisition function (6) by densely sampling 10000 points from a d -dimensional low-discrepancy Sobol sequence, and starting Matlab `fmincon` (a local optimizer) from the sampled point with highest value. All analyses in this section are repeated 50 times to get the mean and standard deviation for comparison.

5.2 Numerical Results

For quantitative comparison, the gap measure [Malkomes and Garnett \(2018\)](#) is reported, **which is** defined as $\frac{\sum_{m=1}^M f(\mathbf{x}_{\text{best}}, m) - \sum_{m=1}^M f(\mathbf{x}_{\text{first}}, m)}{\sum_{m=1}^M f(\mathbf{x}^*, m) - \sum_{m=1}^M f(\mathbf{x}_{\text{first}}, m)} \in [0, 1]$, where $\sum_{m=1}^M f(\mathbf{x}_{\text{first}}, m)$ is the maximum function value among the first initial design points, $\sum_{m=1}^M f(\mathbf{x}_{\text{best}}, m)$ is the best value found by the algorithm, and $\sum_{m=1}^M f(\mathbf{x}^*, m)$ is the optimum value. “0” means there is no improvement starting from the initial design points and “1” means the algorithm can find the optimum.

The results of the gap measure are summarized in Table 3. Our proposed algorithm has the best performance for all six synthetic functions in terms of mean and standard deviation. Specifically, our algorithm can find a better solution than other benchmarks within the same budget. CGP-UCB has the second-best performance while GPUCB has the worst performance. **EIQ has the second-worst performance.** This result demonstrates the effectiveness of the automatic task selection strategy of our proposed algorithm.

Table 3: Results for the gap measure performance across 50 repetitions for different synthetic functions and algorithms.

Functions	EIQ	MTBO	CGP-UCB	GPUCB	MT-GPUCB (Proposed)
Ackley	0.8408 (0.2652)	0.8915 (0.2968)	0.9915 (0.0188)	0.7030 (0.4479)	0.9936 (0.0106)
Bohachevsky	0.9625 (0.0199)	0.9943 (0.0021)	0.9957 (0.0035)	0.9832 (0.0127)	0.9975 (0.0006)
Colville	0.9076 (0.0130)	0.9657 (0.0172)	0.9271 (0.0415)	0.8060 (0.1410)	0.9767 (0.0126)
Levy	0.5370 (0.3414)	0.8363 (0.1341)	0.6754 (0.2329)	0.5142 (0.3034)	0.9979 (0.0027)
Powell	0.5280 (0.1817)	0.4717 (0.2534)	0.5729 (0.2778)	0.1524 (0.2371)	0.7606 (0.1690)
Rastrigin	0.2520 (0.3056)	0.3771 (0.3343)	0.9119 (0.1178)	0.0006 (0.0018)	0.9999 (0.0001)

Note: The numbers in (·) represent standard deviation;

The red bold numbers represent the best performance in each scenario.

Although our applications are not for online real-time, the computation efficiency of our algorithm is also provided in this study. Thereafter, the average computation time (in seconds) of each round is reported in Table 4. EIQ and MTBO are very time-consuming since they involve numerical interrogation, which is computationally expensive. Our proposed algorithm takes a similar computation time with CGP-UCB and GPUCB because they share the same acquisition function. In summary, the results from Tables 3 and 4 show that our MT-GPUCB has not only the best optimization accuracy but also the most efficient computation power.

Table 4: Results for the computation time of each round across 50 repetitions for different synthetic functions and algorithms.

Functions	EIQ	MTBO	CGP-UCB	GPUCB	MT-GPUCB (Proposed)
Ackley	9.78 (3.93)	12.79 (1.04)	5.57 (0.76)	6.39 (1.21)	5.49 (0.37)
Bohachevsky	165.88 (114.89)	15.95 (1.34)	8.59 (0.72)	7.98 (1.28)	8.50 (0.81)
Colville	28.87 (5.55)	30.25 (6.70)	12.01 (0.64)	14.61 (3.40)	12.07 (0.63)
Levy	17.24 (2.01)	15.34 (1.99)	6.47 (0.59)	7.35 (1.25)	7.18 (2.79)
Powell	18.81 (1.24)	27.37 (4.03)	9.91 (0.58)	13.77 (4.39)	9.90 (0.85)
Rastrigin	14.99 (6.62)	18.93 (2.94)	8.78 (0.87)	9.38 (2.83)	8.74 (0.86)

Furthermore, to show the convergence behavior for each algorithm, the simple regret,

Figure 2: Simple Regret via Number of Evaluations for different synthetic functions and methods (95% Confidence interval, Log means $\log_e$).

defined as $\min_{0 \leq t \leq T} r_t$, can be plotted via the number of evaluations in Figure 2. This measure is more relevant to pure search problems (i.e., no exploitation) and captures how quickly the algorithms find the optimal point. As shown in Figure 2, our algorithm can consistently lead to faster convergence trends than the benchmark algorithms. In most cases, our algorithm can reach the best performance achieved by other benchmarks within a much less number of evaluations. GPUCB is the worst across all the synthetic functions since it evaluated all tasks for a query point, which is very wasteful. **EIQ is ineffective for all the synthetic functions since it does not fully consider the correlation among tasks.**

6 Real-World Case Studies

In this section, two real-world case studies are used for evaluating the performance of the proposed algorithm. In Section 6.1, Application 1: Fast cross-validation to determine the

optimal set of hyperparameters for machine learning models. In Section 6.2, Application 2: Raw material properties determination using Flow3D Weld software in different print conditions. Same comparison methods, Matlab setting and acquisition function maximization method described in Section 5 are used in this section.

6.1 Application 1: Fast Cross-Validation for Hyperparameter Tuning

K -fold cross-validation [Bengio and Grandvalet \(2004\)](#) is a widely used technique for estimating the generalization error of machine learning models, but requires retraining a model K times. For a given set of hyperparameters, the generalization error can be usually obtained by taking the average validation errors based on the model training on K training-validation splits. This can be prohibitively expensive with complex models and large datasets. With a good GP model, we can very likely obtain a high quality estimation of generalization error by evaluating the model trained on one training-validation split. To speed up the process of hyperparameter tuning using cross-validation, the algorithm 1 is applied to dynamically determine which hyperparameters and training-validation split to query in each round. Datasets and machine learning models used in this application are as follows:

- (i) Train Gaussian kernel support vector machine (SVM) on Arcene Cancer Dataset [Guyon et al. \(2005\)](#) for two-class classification. Arcene Cancer Dataset is a mass-spectrometric dataset, which contains 7000 continuous input variables. In addition, there are 200 records in which 88 of 200 are cancer patients, 112 of 200 are healthy patients.
- (ii) Train convolutional neural network (CNN) [Lawrence et al. \(1997\)](#) on 2000 Handwritten digit samples (0~9) from MNIST dataset [LeCun et al. \(1998\)](#) for deep learning classification. Each image has size of 28×28 . The structure of CNN used in this article contains three convolutional layers and one fully connected layer. For i th convolutional layer ($i = 1, \dots, 3$), the number of filters is 2^{i+2} with size 3×3 , batch normalization and max pooling are used together with the ReLU activation function. For the final fully-connected layer, it uses the Softmax activation function for classification. The maximum number of epochs is set to 20, which makes CNN converges

for all the cases.

The hyperparameters **that** need to be tuned via cross-validation for the above two machine learning models are summarized below:

- (i) Determine **BoxConstraint** $\in [10^{-3}, 10^3]$ and **KernelScale** $\in [10^{-3}, 10^3]$ for training Gaussian kernel support vector machine (SVM) with 5-fold cross-validation. **BoxConstraint** and **KernelScale** are the two key components to the performance of SVM.
- (ii) Determine **InitialLearnRate** $\in [10^{-5}, 10^0]$, **L2Regularization** $\in [10^{-5}, 10^0]$ and **Momentum** $\in [0.4, 1]$ for training convolutional neural network (CNN) with 5-fold cross-validation. **InitialLearnRate** and **Momentum** are two most important parameters in stochastic gradient descent with momentum (SGDM) [Sutskever et al. \(2013\)](#) optimizer. The **L2Regularization** can help improving the ability of generalization of CNN.

In terms of $k_{\mathcal{X}}(x, x')$ and $k_{\mathcal{Z}}(m, m') \in \mathbb{R}^{M \times M}$, Matérn ($\nu = 5/2$) kernel with a separate length scale per predictor is selected for both kernels. One-hot encoding [Hastie et al. \(2009\)](#) is used in constructing the task features for each task so that $k_{\mathcal{Z}}(m, m')$ can be calculated as $k_{\mathcal{Z}}(m, m') = k_{\text{Mat}}(\mathbf{1}_m, \mathbf{1}_{m'})$, where $\mathbf{1}_m \in \mathbb{R}^M$ is the vector that only m -th position equals 1 and the remaining is 0.

For both SVM and CNN, 20 initial design points are selected by maximin Latin hypercube design. If a variable belongs to $[10^{-3}, 10^3]$, we will search from the exponents, that is, $[-3, 3]$. For 100 and 50 additional evaluations are conducted for SVM and CNN, respectively. All analyses in this section are repeated 20 times to get the mean and standard deviation for comparison. Since the optimal hyperparameters are unknown, the average accuracy of cross-validation is plotted against the number of evaluations in [Figure 3](#). For both SVM and CNN models, our algorithm can achieve the highest average accuracy in estimating the hyperparameters while it is very stable, as shown by the confidence interval. **EIQ and GP-UCB have the lowest average accuracy, while GP-UCB is unstable for both SVM and CNN.** CGP-UCB and MTBO demonstrate competitive performance at the early stage, while MT-GPUCB can keep improving as the number of evaluations goes on. **This**

(a) Average accuracy of cross-validation using SVM on Arcene (b) Average accuracy of cross-validation using CNN on Handwritten digits images

Figure 3: Average accuracy of cross-validation on SVM and CNN for different datasets and algorithms (95% Confidence interval).

experiment shows that our algorithm makes nontrivial decisions regarding which fold to query, which can steadily improve the average accuracy.

6.2 Application 2: Raw material Properties Determination for Flow-3D Weld

Flow-3D Weld [Flow Science \(2019\)](#) is a simulation software based on computational fluid dynamics (CFD). It provides powerful insights into laser welding processes such as electron beam melting (EBM) and selective laser melting (SLM) [Gokuldoss et al. \(2017\)](#), which are representative powder bed fusion additive manufacturing for machine learning applications in quality and reliability [Shen et al. \(2020, 2021\)](#). When we use the Flow-3D Weld, the material of metal powder (for example, Ti6Al4V) and its raw material property (for example, fluid absorption rate (FAR) and thermal conductivity (TC)), as well as printing conditions (for example, laser power and dwell time), are the input to the software. The output from Flow-3D Weld is the melt pool geometries (changing along time), where an example of one frame of the melt pool boundary is shown in Figure 4a. The melt pool information is a critical intermediate measure that reflects the outcome of a laser powder bed fusion process

Figure 4: Visualization of (a) output from Flow3d-Weld; (b) surrogate modeling of image loss.

Zhao et al. (2017). It is one of the critical problems being studied in the related area. For AM research, a challenge is determining the raw material properties of the metal powder to match with the actual AM experiment in terms of melt pool shape. These raw material properties, although known to us, are in the form of a range instead of specific values in the software due to the material property variability. Our proposed method in this paper is to determine the particular values of these raw material properties based on a number of AM experiments. After determining the raw material property values, we can use Flow-3D Weld with high accuracy. Otherwise, these values have to be randomly selected within the range, which cannot guarantee simulation accuracy using Flow-3D.

For the same raw material (Ti6Al4V powder in this paper), its properties should remain the same under different printing conditions (tasks defined in this paper). So, these raw material properties are considered hyperparameters in our proposed model. This paper aims to determine these hyperparameters based on a number of tasks (namely, actual AM experiments; in this paper, we used Flow-3D Weld to synthesize the AM experiments). The problem studied in this subsection has the following procedure.

- **Step 1 (AM experiments using Flow-3D simulation):** AM experiments are simulated using Flow-3D Weld. Ti6Al4V (power) is selected as the raw material with the

laser melting process. The entire process of one laser melting experiment takes 2 million seconds (ms), during which the laser is turned on. Two different printing conditions (tasks) are applied, namely, Condition 1: 50% laser power, 0.8 ms dwell time (time of laser on), and Condition 2: 40% laser power, 1 ms dwell time. We set the raw material properties of Ti6Al4V, namely, fluid absorption rate (FAR) = 0.3780 and thermal conductivity (TC) = $1.64e^6 \text{ cm}^*\text{g/s}^3\text{K}$ in all experiments to generate actual AM experiments data. These values are randomly chosen from the given ranges provided by the software, and they are treated as a BlackBox and unknown to our developed algorithm. They will be considered as the ground truth to test the performance of our proposed method. This is the way used in our paper to simulate the impact of the specific raw material properties on metal AM printing.

- **Step 2 (Multi-tasking learning to best determine the actual raw material properties):** Our algorithm is applied to guide a sequence of AM experiments (simulated using Flow-3D Weld) to determine the true values of FAR and TC used in **Step 1**. Specifically, our algorithm aims to minimize the image loss between the image from a queried simulation and the image from the AM data in **Step 1**, which is the ℓ_2 norm of the difference of two images. The image loss is used to construct the “black-box” function f in our multi-task BO.

The performance of different algorithms is evaluated by the distance between the values of FAR and TC queried in **Step 2** and the actual values of FAR and TC in **Step 1**. To let the reader better understand the difficulty of our problem, 50 points are selected by maximin Latin hypercube design, where 20 points are sampled near the optimal solution (FAR= 0.3780 and TC = $1.64e^6 \text{ cm}^*\text{g/s}^3\text{K}$). Together with the optimal solution, the visualization of our “black-box” function is shown in Figure 4b. It shows that our problem is very challenging with multiple local optimal solutions. Especially, the geometry of the surface is very complicated when it is close to the optimal solution.

In terms of $k_{\mathcal{X}}(x, x')$ and $k_{\mathcal{Z}}(m, m') \in \mathbb{R}^{M \times M}$, Matérn ($\nu = 5/2$) kernel with a separate length scale per predictor is selected for both kernels. The printing conditions are used in constructing the task features for each task. We search FAR from the range [0.34, 0.41] and TC from the range [1.4, 1.9] (e^6 will be multiplied when input the TC into the software). 20 initial design points are selected by maximin Latin hypercube design and 16 additional

Figure 5: The distance to optimal solution vs. number of evaluations for different algorithms (Log means $\log_e$).

evaluations are determined by BO algorithms. The distance to the optimal solution (ground truth) is plotted in Figure 5 for different algorithms. At the beginning, our algorithm learns at a relatively slow speed. After that, our algorithm converges to the optimal solution in a much faster speed than any other algorithms. In the end, our algorithm converges to a much better solution than any other benchmarks.

7 Conclusion

In this paper, a multi-task Gaussian process upper confidence bound (MT-GPUCB) algorithm is implemented for obtaining the solution from different but correlated tasks. The proposed MT-GPUCB considers query strategy for design point with automatic task selection in each round of multi-task Bayesian optimization to improve sample efficiency. Under some mild conditions, our algorithm is a *no-regret* algorithm, which converges to the optimal solution with *sub-linear* convergence rate. Experimentally, our algorithm is validated by synthetic functions and real-world case studies in fast cross-validation and raw material properties determination. Based on the convergence speed results of these case studies, it is evident that our MT-GPUCB outperforms state-of-the-art algorithms in related areas.

In addition, there are still some aspects of MT-GPUCB that deserve further investigations. First, the selection of kernel is critical to the performance of Gaussian process.

Therefore, how to select a proper kernel is one of the next steps of research. Especially, how to construct the task kernel k_Z is the key step. Second, the maximization of acquisition function is a non-trivial problem [Wilson et al. \(2018\)](#). Thereafter, how to efficiently optimize the acquisition function can be further investigated. Third, our query strategy to select both design point and task is not limited to UCB-based acquisition function. There are probability of improvement, expected improvement, and entropy-based acquisition functions can be explored in the future.

References

- Álvarez, M. A. and N. D. Lawrence (2011). Computationally efficient convolved multiple output gaussian processes. *The Journal of Machine Learning Research* 12, 1459–1500.
- Bengio, Y. and Y. Grandvalet (2004). No unbiased estimator of the variance of k-fold cross-validation. *Journal of machine learning research* 5(Sep), 1089–1105.
- Bergstra, J., R. Bardenet, Y. Bengio, and B. Kégl (2011). Algorithms for hyper-parameter optimization. *Advances in neural information processing systems* 24, 2546–2554.
- Brochu, E., V. M. Cora, and N. De Freitas (2010). A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599* 1, 1–49.
- Chaloner, K. and I. Verdinelli (1995). Bayesian experimental design: A review. *Statistical Science* 1(1), 273–304.
- Cover, T. M. and J. A. Thomas (1991). Information theory and statistics. *Elements of Information Theory* 1(1), 279–335.
- Dai, S., J. Song, and Y. Yue (2020). Multi-task bayesian optimization via gaussian process upper confidence bound. In *ICML 2020 Workshop on Real World Experiment Design and Active Learning*.
- Flow Science, I. (2019). *FLOW-3D, Version 12.0*. Santa Fe, NM: FLOW SCIENCE.
- Frazier, P. I. and J. Wang (2016). Bayesian optimization for materials design. In *Information Science for Materials Discovery and Design*, pp. 45–75. Springer.
- Garrido-Merchán, E. C. and D. Hernández-Lobato (2020). Dealing with categorical and integer-valued variables in bayesian optimization with gaussian processes. *Neurocomputing* 380, 20–35.
- Genton, M. G. (2001). Classes of kernels for machine learning: a statistics perspective. *Journal of machine learning research* 2(Dec), 299–312.
- Ghosal, S., A. Roy, et al. (2006). Posterior consistency of gaussian process prior for non-parametric binary regression. *The Annals of Statistics* 34(5), 2413–2429.

- Gokuldoss, P. K., S. Kolla, and J. Eckert (2017). Additive manufacturing processes: Selective laser melting, electron beam melting and binder jetting—selection guidelines. *Materials* 10(6), 672.
- Guyon, I., S. Gunn, A. Ben-Hur, and G. Dror (2005). Result analysis of the nips 2003 feature selection challenge. In *Advances in neural information processing systems*, pp. 545–552.
- Han, S., C. Qubo, and H. Meng (2012). Parameter selection in svm with rbf kernel function. In *World Automation Congress 2012*, pp. 1–4. IEEE.
- Hastie, T., R. Tibshirani, and J. Friedman (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.
- Hutter, F., H. H. Hoos, and K. Leyton-Brown (2011). Sequential model-based optimization for general algorithm configuration. In *International conference on learning and intelligent optimization*, pp. 507–523. Springer.
- Janusevskis, J. and R. Le Riche (2013). Simultaneous kriging-based estimation and optimization of mean response. *Journal of Global Optimization* 55(2), 313–336.
- Jones, D. R., M. Schonlau, and W. J. Welch (1998). Efficient global optimization of expensive black-box functions. *Journal of Global optimization* 13(4), 455–492.
- Joseph, V. R. and Y. Hung (2008). Orthogonal-maximin latin hypercube designs. *Statistica Sinica* 1(1), 171–186.
- Kandasamy, K., G. Dasarathy, J. Oliva, J. Schneider, and B. Póczos (2019). Multi-fidelity gaussian process bandit optimisation. *Journal of Artificial Intelligence Research* 66, 151–196.
- Kandasamy, K., G. Dasarathy, J. Schneider, and B. Póczos (2017). Multi-fidelity bayesian optimisation with continuous approximations. In *34th International Conference on Machine Learning, ICML 2017*, pp. 2861–2878. International Machine Learning Society (IMLS).
- Kandasamy, K., J. Schneider, and B. Póczos (2015). High dimensional bayesian optimisation and bandits via additive models. In *International conference on machine learning*, pp. 295–304. PMLR.
- Krause, A. and C. S. Ong (2011). Contextual gaussian process bandit optimization. In *Advances in neural information processing systems*, pp. 2447–2455.
- Lawrence, S., C. L. Giles, A. C. Tsoi, and A. D. Back (1997). Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks* 8(1), 98–113.
- LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11), 2278–2324.
- Maftouni, M., A. C. C. Law, B. Shen, Z. J. Kong, Y. Zhou, and N. A. Yazdi (2020). A robust ensemble-deep learning model for covid-19 diagnosis based on an integrated ct scan images database. In *IIE Annual Conference. Proceedings*, pp. 1–6. Institute of Industrial and Systems Engineers (IISE).

- Malkomes, G. and R. Garnett (2018). Automating bayesian optimization with bayesian optimization. *Advances in Neural Information Processing Systems* 31, 5984–5994.
- Moss, H. B., D. S. Leslie, and P. Rayson (2020). Mumbo: Multi-task max-value bayesian optimization. *arXiv preprint arXiv:2006.12093* 1(1), 1–23.
- Rasmussen, C. E. (2003). Gaussian processes in machine learning. In *Summer School on Machine Learning*, pp. 63–71. Springer.
- Shahriari, B., K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas (2015). Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE* 104(1), 148–175.
- Shen, B., R. Wang, A. C. C. Law, R. Kamath, H. Choo, and Z. Kong (2021). Super resolution for multi-sources image stream data using smooth and sparse tensor completion and its applications in data acquisition of additive manufacturing. *Technometrics* 1(1), 1–16.
- Shen, B., W. Xie, and Z. J. Kong (2020). Clustered discriminant regression for high-dimensional data feature extraction and its applications in healthcare and additive manufacturing. *IEEE Transactions on Automation Science and Engineering* 1(1), 1–13.
- Snoek, J., H. Larochelle, and R. P. Adams (2012). Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pp. 2951–2959.
- Srinivas, N., A. Krause, S. M. Kakade, and M. W. Seeger (2012). Information-theoretic regret bounds for gaussian process optimization in the bandit setting. *IEEE Transactions on Information Theory* 58(5), 3250–3265.
- Surjanovic, S. and D. Bingham (2020). Virtual library of simulation experiments: Test functions and datasets. Retrieved December 22, 2020, from <http://www.sfu.ca/~ssurjano>.
- Sutskever, I., J. Martens, G. Dahl, and G. Hinton (2013). On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pp. 1139–1147.
- Swersky, K., J. Snoek, and R. P. Adams (2013). Multi-task bayesian optimization. *Advances in neural information processing systems* 26, 2004–2012.
- Tan, M. H. (2020). Bayesian optimization of expected quadratic loss for multiresponse computer experiments with internal noise. *SIAM/ASA Journal on Uncertainty Quantification* 8(3), 891–925.
- Toscano-Palmerin, S. and P. I. Frazier (2018). Bayesian optimization with expensive integrands. *arXiv preprint arXiv:1803.08661* 1(1), 1–52.
- Wahba, G. (1990). *Spline models for observational data*. SIAM.
- Williams, B. J., T. J. Santner, and W. I. Notz (2000). Sequential design of computer experiments to minimize integrated response functions. *Statistica Sinica* 1(1), 1133–1152.

- Williams, C., E. V. Bonilla, and K. M. Chai (2007). Multi-task gaussian process prediction. *Advances in neural information processing systems 1*(1), 153–160.
- Williams, C. K. and C. E. Rasmussen (2006). *Gaussian processes for machine learning*, Volume 2. MIT press Cambridge, MA.
- Wilson, J., F. Hutter, and M. Deisenroth (2018). Maximizing acquisition functions for bayesian optimization. *Advances in Neural Information Processing Systems 31*, 9884–9895.
- Yan, X., Y. Xia, H. B. Blum, and T. Gernay (2020). Elevated temperature material properties of advanced high strength steel alloys. *Journal of Constructional Steel Research 174*, 106299.
- Zhao, C., K. Fezzaa, R. W. Cunningham, H. Wen, F. De Carlo, L. Chen, A. D. Rollett, and T. Sun (2017). Real-time monitoring of laser powder bed fusion process using high-speed x-ray imaging and diffraction. *Scientific reports 7*(1), 1–11.

A Proof of Theorem 1

In this section, we will show the details for the proof of Theorem 1. The proof of Theorem 1 under conditions (1) or (3) is shown in Section A.1. In Section A.2, the proof of Theorem 1 under condition (2) is provided.

A.1 Proof of Theorem 1 under Conditions (1) or (3)

To begin with, Lemmas 1 and 2 are introduced, which establish a bound on instantaneous regret in terms of posterior variance. Then, Lemma 3 connects the information gain with the posterior variances. At the end, the proof of Theorem 1 under conditions (1) or (3) is completed based on these three lemmas.

Lemma 1 (Lemma 5.1 and Theorem 6 Srinivas et al. (2012)). *Pick $\delta \in (0, 1)$. Suppose one of the following conditions holds*

- (i) $|\mathcal{X}| = N$, f is sampled from a known GP prior with noise varinacen σ^2 , and $\beta_t = 2 \log(MN\pi_t/\delta)$ where $\sum_{t \geq 1} \pi_t^{-1} = 1$, $\pi_t > 0$.
- (ii) The noise variables ϵ_t form an arbitrary martingale difference sequence, namely, $\mathbb{E}[\epsilon_t | \epsilon_1, \dots, \epsilon_{t-1}] = 0$ for all $t \in \mathbb{N}$, uniformly bounded by σ and $\beta_t = 2B^2 + 300\gamma_t \log^3(t/\delta)$.

Then

$$|f(\mathbf{x}, m) - \mu_{t-1}(\mathbf{x}, m)| \leq \beta_t^{1/2} \sigma_{t-1}(\mathbf{x}, m), \forall \mathbf{x} \in \mathcal{X}, t \geq 1, 1 \leq m \leq M \quad (11)$$

holds with probability $1 - \delta$.

Lemma 2. For a given $t \geq 1$, if $|f(\mathbf{x}, m) - \mu_{t-1}(\mathbf{x}, m)| \leq \beta_t^{1/2} \sigma_{t-1}(\mathbf{x}, m), \forall \mathbf{x} \in \mathcal{X}, 1 \leq m \leq M$, then the regret $r_t \leq \frac{2}{M} \beta_t^{1/2} \sum_{m=1}^M \sigma_{t-1}(\mathbf{x}_t, m)$.

Proof. By definition of $\mathbf{x}_t$: $\sum_{m=1}^M \mu(\mathbf{x}_t, m) + \beta_t^{1/2} \sigma_{t-1}(\mathbf{x}_t, m) \geq \sum_{m=1}^M \mu(\mathbf{x}^*, m) + \beta_t^{1/2} \sigma_{t-1}(\mathbf{x}^*, m) \geq \sum_{m=1}^M f(\mathbf{x}^*, m)$. Therefore,

$$\begin{aligned} r_t &= \frac{1}{M} \sum_{m=1}^M f(\mathbf{x}^*, m) - f(\mathbf{x}_t, m) \\ &\leq \frac{1}{M} \sum_{m=1}^M (\mu(\mathbf{x}_t, m) - f(\mathbf{x}_t, m)) + \beta_t^{1/2} \sigma_{t-1}(\mathbf{x}_t, m) \\ &\leq \frac{2}{M} \beta_t^{1/2} \sum_{m=1}^M \sigma_{t-1}(\mathbf{x}_t, m) \end{aligned}$$

where the first inequality is due to the definition of $\mathbf{x}_t$ and the second inequality comes from the inequality (11). $\square$

Next lemma connects the information gain with the posterior variances.

Lemma 3. The information gain from all queried points

$$I(\mathbf{y}_T; \mathbf{f}_T) = \sum_{t=1}^T \frac{1}{2} \log(1 + \sigma^{-2} \sigma_{t-1}^2(\mathbf{x}_t, m_t)).$$

Now we present the proof of Theorem 1 under conditions (1) or (3).

Proof of Theorem 1 under Conditions (1) or (3). By lemmas 1 and 2, we have that $r_t^2 \leq \frac{4}{M^2} \beta_t (\sum_{m=1}^M \sigma_{t-1}(\mathbf{x}_t, m))^2$ for any $t \geq 1$ with probability at least $1 - \delta$. From the Cauchy-Schwarz inequality, we have that

$$(\sum_{m=1}^M \sigma_{t-1}(\mathbf{x}_t, m))^2 \leq M \sum_{m=1}^M \sigma_{t-1}^2(\mathbf{x}_t, m). \quad (12)$$

It follows that

$$\begin{aligned} r_t^2 &= \frac{4}{M^2} \beta_t (\sum_{m=1}^M \sigma_{t-1}(\mathbf{x}_t, m))^2 \\ &\leq \frac{4}{M} \beta_t \sum_{m=1}^M \sigma_{t-1}^2(\mathbf{x}_t, m) \\ &\leq \frac{4}{M} \beta_t \sigma^2 C_1 \sum_{m=1}^M \log(1 + \sigma^{-2} \sigma_{t-1}^2(\mathbf{x}_t, m)) \\ &\leq 4 \beta_t \sigma^2 C_1 \log(1 + \sigma^{-2} \sigma_{t-1}^2(\mathbf{x}_t, m_t)) \end{aligned}$$

with $C_1 := \frac{\sigma^{-2}}{\log(1+\sigma^{-2})} \geq 1$, where the first inequality is based on the inequality (12), the second inequality is due to that $s^2 \leq C_1 \log(1+s^2)$ for $s^2 \in [0, \sigma^{-2}]$ and $\sigma^{-2} \sigma_{t-1}^2(\mathbf{x}_t, m) \leq \sigma^{-2} k_{\text{multi}}(\mathbf{x}_t, \mathbf{x}_t) \leq \sigma^{-2}$, and the last inequality is because of (7).

Based on the Cauchy-Schwarz inequality, we have

$$\begin{aligned} R_T^2 &\leq T \left(\sum_{t=1}^T r_t^2 \right) \\ &= 4\beta_T \sigma^2 C_1 T \sum_{t=1}^T \log(1 + \sigma^{-2} \sigma_{t-1}^2(\mathbf{x}_t, m_t)) \\ &= 8\beta_T \sigma^2 C_1 T I(\mathbf{y}_T; \mathbf{f}_T) \\ &\leq 8\beta_T \sigma^2 C_1 T \gamma_T, \end{aligned}$$

where the last inequality is because of the definition of γ_T . Therefore,

$$R_T \leq \sqrt{CT\beta_T\gamma_T}$$

where $C := 8\sigma^2 C_1 = \frac{8}{\log(1+\sigma^{-2})} \geq 8\sigma^2$. □

A.2 Proof of Theorem 1 under Condition (2)

Recall that the finite case proof is based on Lemma 1 paving the way for Lemma 2. However, Lemma 1 does not hold for infinite $\mathcal{X}$. From Lemma 4 to Lemma 7, the extension to infinite $\mathcal{X}$ is accomplished by the discretization trick. First, let us observe that we have confidence on all decisions actually chosen.

Lemma 4. *Pick $\delta \in (0, 1)$ and set $\beta_t = 2 \log(M\pi_t/\delta)$ where $\sum_{t \geq 1} \pi_t^{-1} = 1$, $\pi_t > 0$. Then,*

$$|f(\mathbf{x}_t, m) - \mu_{t-1}(\mathbf{x}_t, m)| \leq \beta_t^{1/2} \sigma_{t-1}(\mathbf{x}_t, m), \forall t \geq 1, 1 \leq m \leq M$$

holds with probability $1 - \delta$.

Proof. Directly followed by Lemma 1. □

Purely for the sake of analysis, we use a set of discretizations $\mathcal{X}_t \subset \mathcal{X}$, where $\mathcal{X}_t$ will be used at time t in the analysis

Lemma 5. *Pick $\delta \in (0, 1)$ and set $\beta_t = 2 \log(M|\mathcal{X}_t|\pi_t/\delta)$ where $\sum_{t \geq 1} \pi_t^{-1} = 1$, $\pi_t > 0$. Then,*

$$|f(\mathbf{x}_t, m) - \mu_{t-1}(\mathbf{x}_t, m)| \leq \beta_t^{1/2} \sigma_{t-1}(\mathbf{x}_t, m), \forall t \geq 1, 1 \leq m \leq M$$

holds with probability $1 - \delta$.

Now by assumption and the union bound, we have

$$\mathcal{P}(|\partial f(\mathbf{x}, m)/\partial \mathbf{x}_j| < L, \forall j, m, \mathbf{x} \in \mathcal{X}) \leq 1 - dMa e^{-(L/b)^2},$$

which implies that, with probability greater than $1 - dMa e^{-(L/b)^2}$, we have

$$|f(\mathbf{x}, m) - f(\mathbf{x}', m)| \leq L \|\mathbf{x} - \mathbf{x}'\|_1, \quad (13)$$

which allows us to obtain confidence on $\mathbf{x}^*$ as follows. Now let us choose a discretization $\mathcal{X}_t$ of size τ_t^d so that for all $\mathbf{x} \in \mathcal{X}$,

$$\|\mathbf{x} - [\mathbf{x}]_t\|_1 \leq rd/\tau_t,$$

where $[\mathbf{x}]_t$ denotes the closest point in $\mathcal{X}_t$ to $\mathbf{x}$. A sufficient discretization has each coordinate with τ_t uniformly spaced points.

Lemma 6. *Pick $\delta \in (0, 1)$ and set $\beta_t = 2 \log(2M\pi_t/\delta) + 2d \log(t^2 rdb \sqrt{\log(2dMa/\delta)})$, where $\sum_{t \geq 1} \pi_t^{-1} = 1$, $\pi_t > 0$. Let $\tau_t = t^2 rdb \sqrt{\log(2dMa/\delta)}$, and $[\mathbf{x}^*]_t$ denotes the closest point in $\mathcal{X}_t$ to $\mathbf{x}^*$. Then*

$$|f(\mathbf{x}^*, m) - \mu_{t-1}([\mathbf{x}^*]_t, m)| \leq \beta_t^2 \sigma_{t-1}([\mathbf{x}^*]_t, m) + \frac{1}{t^2}, \forall t \geq 1, m \quad (14)$$

holds with probability $1 - \delta$.

Proof. Using (13), we have that with probability greater than $1 - \delta$,

$$|f(\mathbf{x}, m) - f(\mathbf{x}', m)| \leq b \sqrt{\log(2dMa/\delta)} \|\mathbf{x} - \mathbf{x}'\|_1, \forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}, m.$$

Hence,

$$|f(\mathbf{x}, m) - f([\mathbf{x}]_t, m)| \leq rdb \sqrt{\log(2dMa/\delta)} / \tau_t, \forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}, m.$$

Now by choosing $\tau_t = t^2 rdb \sqrt{\log(2dMa/\delta)}$, we have

$$|f(\mathbf{x}, m) - f([\mathbf{x}]_t, m)| \leq \frac{1}{t^2}, \forall \mathbf{x} \in \mathcal{X}, m.$$

This implies that $|\mathcal{X}_t| = \left(t^2 rdb \sqrt{\log(2dMa/\delta)}\right)^d$. Using $\delta/2$ in Lemma 5, we can apply the confidence bound to $[\mathbf{x}]_t$ (as this lives in $\mathcal{X}_t$) to obtain the result. $\square$

Now we are ready to bound the regret.

Lemma 7. Pick $\delta \in (0, 1)$ and set $\beta_t = 2 \log(4M\pi_t/\delta) + 2d \log(t^2 r d b \sqrt{\log(4dMa/\delta)})$, where $\sum_{t \geq 1} \pi_t^{-1} = 1$, $\pi_t > 0$. Then

$$r_t \leq \frac{2}{M} \beta_t^{1/2} \sum_{m=1}^M \sigma_{t-1}(\mathbf{x}_t, m) + \frac{1}{t^2} \quad (15)$$

holds with probability $1 - \delta$.

Proof. We use $\delta/2$ in both Lemmas 4 and 6, so that these events hold with probability greater than $1 - \delta$. Note that the specification of β_t in the aforementioned lemma is greater than the specification used in Lemma 4 (with $\delta/2$), so this choice is valid.

By definition of $\mathbf{x}_t$: $\sum_{m=1}^M \mu(\mathbf{x}_t, m) + \beta_t^{1/2} \sigma_{t-1}(\mathbf{x}_t, m) \geq \sum_{m=1}^M \mu([\mathbf{x}^*]_t, m) + \beta_t^{1/2} \sigma_{t-1}([\mathbf{x}^*]_t, m)$. Also, by (14), we have that

$$\mu([\mathbf{x}^*]_t, m) + \beta_t^{1/2} \sigma_{t-1}([\mathbf{x}^*]_t, m) + \frac{1}{t^2} \geq f(\mathbf{x}^*, m), \forall m,$$

which implies that $\sum_{m=1}^M \mu(\mathbf{x}_t, m) + \beta_t^{1/2} \sigma_{t-1}(\mathbf{x}_t, m) \geq \sum_{m=1}^M f(\mathbf{x}^*, m) - \frac{1}{t^2}$

$$\begin{aligned} r_t &= \frac{1}{M} \sum_{m=1}^M f(\mathbf{x}^*, m) - f(\mathbf{x}_t, m) \\ &\leq \frac{1}{M} \sum_{m=1}^M (\mu(\mathbf{x}_t, m) - f(\mathbf{x}_t, m)) + \beta_t^{1/2} \sigma_{t-1}(\mathbf{x}_t, m) + \frac{1}{t^2} \\ &\leq \frac{2}{M} \beta_t^{1/2} \sum_{m=1}^M \sigma_{t-1}(\mathbf{x}_t, m) + \frac{1}{t^2} \end{aligned}$$

□

Now we are ready to complete the proof of Theorem 1 under condition (2).

Proof of Theorem 1 under Condition (2). As shown in the proof of Theorems 1 under conditions (1) or (3), the following

$$\sum_{t=1}^T \frac{2}{M} \beta_t^{1/2} \sum_{m=1}^M \sigma_{t-1}(\mathbf{x}_t, m) \leq \sqrt{CT\beta_T\gamma_T}$$

holds with probability $1 - \delta$. Take the summation of (15),

$$R_T \leq \sqrt{CT\beta_T\gamma_T} + \frac{\pi^2}{6}, \forall T \geq 1$$

since $\sum_{t \geq 1} \frac{1}{t^2} = \frac{\pi^2}{6}$.

□