Knowing The What But Not The Where in Bayesian Optimization

Vu Nguyen [1] **Michael A Osborne** [1]

Abstract

Bayesian optimization has demonstrated impressive success in finding the optimum input $\mathbf{x}^*$ and output $f^* = f(\mathbf{x}^*) = \max f(\mathbf{x})$ of a black-box function f. In some applications, however, the optimum output f^* is known in advance and the goal is to find the corresponding optimum input $\mathbf{x}^*$. In this paper, we consider a new setting in BO in which the knowledge of the optimum output f^* is available. Our goal is to exploit the knowledge about f^* to search for the input $\mathbf{x}^*$ efficiently. To achieve this goal, we first transform the Gaussian process surrogate using the information about the optimum output. Then, we propose two acquisition functions, called confidence bound minimization and expected regret minimization. We show that our approaches work intuitively and give quantitatively better performance against standard BO methods. We demonstrate real applications in tuning a deep reinforcement learning algorithm on the CartPole problem and XGBoost on Skin Segmentation dataset in which the optimum values are publicly available.

1. Introduction

Bayesian optimization (BO) (Brochu et al., 2010; Shahriari et al., 2016; Oh et al., 2018; **?**) is an efficient method for the global optimization of a black-box function. BO has been successfully employed in selecting chemical compounds (Hernández-Lobato et al., 2017), material design (Frazier & Wang, 2016; Li et al., 2018), algorithmic assurance (**?**), and in search for hyperparameters of machine learning algorithms (Snoek et al., 2012; Klein et al., 2017; Chen et al., 2018). These recent results suggest BO is more efficient than manual, random, or grid search.

Bayesian optimization finds the global maximizer $\mathbf{x}^* = \arg\max_{\mathbf{x}\in\mathscr{X}} f(\mathbf{x})$ of the black-box function f by incorporating prior beliefs about f and updating the prior with evaluations where $\mathscr{X} \subset \mathbb{R}^d$ is the search domain. The model used for approximating the black-box function is called the surrogate model. A popular choice for a surrogate model is the Gaussian process (GP) (Rasmussen, 2006) although there are existing alternative options, such as random forests (Hutter et al., 2011), deep neural networks (Snoek et al., 2015), Bayesian neural networks (Springenberg et al., 2016) and Mondrian trees (Wang et al., 2018). This surrogate model is then used to define an acquisition function which determines the next query of the black-box function.

In some settings, the optimum output $f^* = f(\mathbf{x}^*)$ is known in advance. For example, the optimal reward is available for common reinforcement learning benchmarks or we know the optimum accuracy is 100 in tuning classification algorithm for specific datasets. As another example in inverse optimization, we retrieve the input resulting the given target (**??**). The question is how to efficiently utilize such prior knowledge to find the optimal inputs using the fewest number of queries.

In this paper, we give the first BO approach to this setting in which we know what we are looking for, but we do not know where it is. Specifically, we know the optimum output $f^* = \max_{\mathbf{x}\in\mathscr{X}} f(\mathbf{x})$ and aim to search for the unknown optimum input $\mathbf{x}^* = \arg\max_{\mathbf{x}\in\mathscr{X}} f(\mathbf{x})$ by utilizing f^* value.

We incorporate the information about f^* into Bayesian optimization in the following ways. First, we use the knowledge of f^* to build a transformed GP surrogate model. Our intuition in transforming a GP is based on the fact that the black-box function value $f(\mathbf{x})$ should not be above the threshold f^* (since $f^* \geq f(\mathbf{x}), \forall \mathbf{x} \in \mathscr{X}$, by definition). As a result, the GP surrogate should also follow this property. Second, we propose two acquisition functions which make decisions informed by the f^* value, namely *confidence bound minimization* and *expected regret minimization*.

We validate our model using benchmark functions and tuning a deep reinforcement learning algorithm where we observe the optimum value in advance. These experiments demonstrate that our proposed framework works both intuitively better and experimentally outperforms the baselines. Our main contributions are summarized as follows:

[1]University of Oxford, UK. Correspondence to: Vu Nguyen <vu@robots.ox.ac.uk>.

Proceedings of the 37th International Conference on Machine Learning, Online, PMLR 119, 2020.

- a first study of Bayesian optimization for exploiting the known optimum output f^*;
- a transformed Gaussian process surrogate using the knowledge of f^*; and
- two novel acquisition functions to efficiently select the optimum location given f^*.

2. Preliminaries

In this section, we review some of the existing acquisition functions from the Bayesian optimization literature which can readily incorporate the known f^* value. Then, we summarize the possible transformation techniques used to control the Gaussian process using f^*.

2.1. Available acquisition functions for the known f^*

Bayesian optimization uses an acquisition function to make a query. Among many existing acquisition functions (Hennig & Schuler, 2012; Hernández-Lobato et al., 2014; Wang et al., 2016; Letham et al., 2019; Astudillo & Frazier, 2019; **?**), we review two acquisition functions which can incorporate the known optimum output f^* directly in their forms. We then use the two acquisition functions as the baselines for comparison.

Expected improvement with known incumbent f^*. EI (Mockus et al., 1978) considers the expectation over the improvement function which is defined over the *incumbent* ξ as $\mathbb{E}[I_t(\mathbf{x})] = \mathbb{E}[\max\{0, f(\mathbf{x}) - \xi\}]$. One needs to define the incumbent to improve upon. Existing research has considered modifying this incumbent with various choices (Wang & de Freitas, 2014; Berk et al., 2018). The typical choice of the incumbent is the best observed value so far in the observation set $\xi = \max_{y_i \in \mathscr{D}_{t-1}} y_i$ where $\mathscr{D}_{t-1}$ is the dataset upto iteration t. Given the known optimum output f^*, one can readily use it as the incumbent, i.e., setting $\xi = f^*$ to have the following forms:

$$\alpha^{\text{EI}^*}(\mathbf{x}) = \sigma(\mathbf{x})\,\phi(z) + [\mu(\mathbf{x}) - f^*]\,\Phi(z) \qquad (1)$$

where $\mu(\mathbf{x})$ is the GP predictive mean, $\sigma(\mathbf{x})$ is the GP predictive variance, $z = \frac{\mu(\mathbf{x}) - f^*}{\sigma(\mathbf{x})}$, ϕ is the standard normal p.d.f. and Φ is the c.d.f.

Output entropy search with known f^*. The second group of acquisition functions, which are readily to incorporate the known optimum, include several approaches gaining information about the output, such as output-space PES (Hoffman & Ghahramani, 2015), MES (Wang & Jegelka, 2017) and FITBO (Ru et al., 2018). These approaches consider different ways to gain information about the optimum output f^*. When f^* is not known in advance, Hoffman & Ghahramani (2015); Wang & Jegelka (2017) utilize Thompson sampling to sample f^*, or a collection of $f^*_m, \forall m \leq M$, while Ru et al. (2018) consider f^* as a hyperparameter. After generating optimum value samples, the above approaches consider different approximation strategies.

Since the optimum output f^* is available in our setting, we can use it directly within the above approaches. We select to review the MES due to its simplicity and closed-form computation. Given the known f^* value, MES approximates $I(\mathbf{x}, y; f^*)$ using a truncated Gaussian distribution such that the distribution of y needs to satisfy $y < f^*$, to obtain,

$$I(\mathbf{x}, y; f^*) = H[p(y|D_t, \mathbf{x})] - \mathbb{E}[H(p(y|D_t, \mathbf{x}, f^*))]p(f^*|D_t).$$

Let $\gamma(\mathbf{x}, f^*) = \frac{f^* - \mu(\mathbf{x})}{\sigma(\mathbf{x})}$, we have the MES* as

$$\alpha^{\text{MES}^*}(\mathbf{x} \mid f^*) = \frac{\gamma(\mathbf{x}, f^*)\phi[\gamma(\mathbf{x}, f^*)]}{2\Phi[\gamma(\mathbf{x}, f^*)]} - \log\Phi[\gamma(\mathbf{x}, f^*)].$$

2.2. Gaussian process transformation for $f \leq f^*$

We summarize several transformation approaches which can be potentially used to enforce that the function f is everywhere below f^*, given the upper bound $f^* = \max_{\forall \mathbf{x}} f(\mathbf{x})$.

The first category is to use functions such as sigmoid and tanh. However, there are two problems with such functions. The first problem is that they both require the knowledge of the lower bound, $\min f(\mathbf{x})$, and the upper bound, $\max f(\mathbf{x})$, for the normalization to the predefined ranges, i.e. $[0, 1]$ for sigmoid and $[-1, 1]$ for tanh. However, we do not know the lower bound in our setting. The second problem is that exact inference for a GP is analytically intractable under these transformations. Particularly, this will become the Gaussian process classification problem (Nickisch & Rasmussen, 2008) where approximation must be made, such as using expectation propagation (Kuss & Rasmussen, 2005; Riihimäki et al., 2013; Hernández-Lobato & Hernández-Lobato, 2016).

The second category is to transform the output of a GP using warping (MacKay, 1998; Snelson et al., 2004). However, the warped GP is less efficient in the context of Bayesian optimization. This is because a warped GP requires more data points[1] to learn the mapping from original to transformed space while we only have a small number of observations in BO setting.

The third category makes use of a linearization trick (Osborne et al., 2012; Gunter et al., 2014) as GPs are closed under linear transformations. This linearization ensures that we arrive at another GP after transformation given our existing GP. In this paper, we shall follow this linearization trick to transform the surrogate model given f^*.

[1] using the datasets with 800 to 1000 samples for learning.

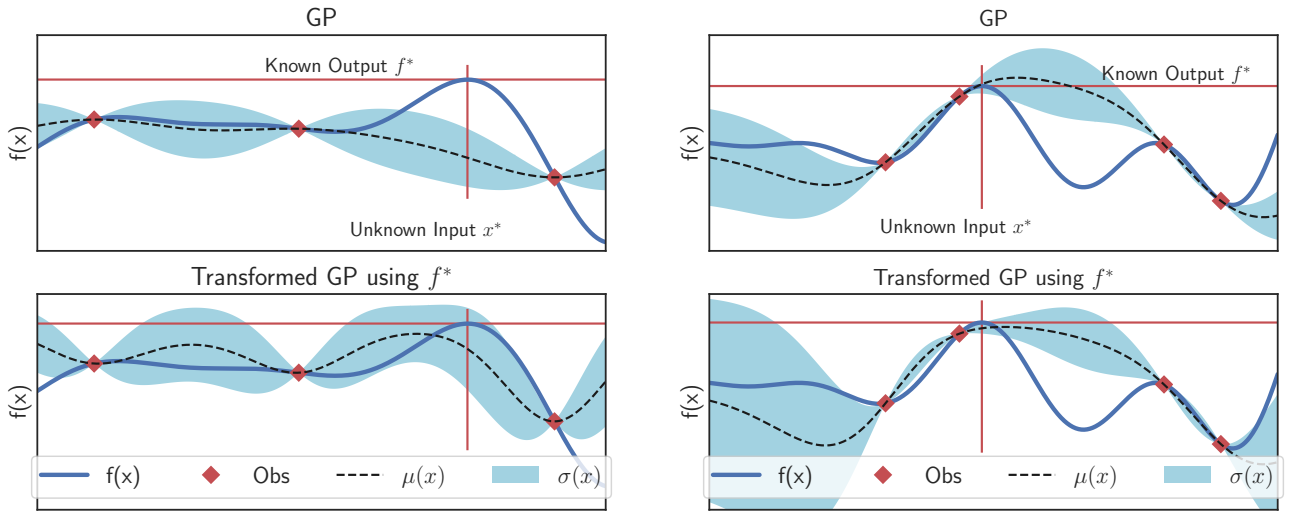

Figure 1. Comparison of the transformed GP with the GP using two different functions in left and right. The known f^* output and unknown input $\mathbf{x}^*$ are highlighted by horizontal and vertical red lines respectively. Top: the GP allows $\mu(\mathbf{x})$ to go above and below f^*. Bottom: the transformed GP will lift up the surrogate model closer to the known optimum output f^* (left) and not go above f^* (right).

3. Bayesian Optimization When The True Optimum Output Is Known

We present a new approach for Bayesian optimization given situations where the knowledge of optimum output (value) $f^* = \max_{\mathbf{x}\in\mathscr{X}} f(\mathbf{x})$ is available. Our goal is to utilize this knowledge to improve BO performance in finding the unknown optimum input (location) $\mathbf{x}^* = \arg\max_{\mathbf{x}\in\mathscr{X}} f(\mathbf{x})$. We first encode f^* to build an informed GP surrogate model through transformation and then we propose two acquisition functions which effectively exploit knowledge of f^*.

3.1. Transformed Gaussian process

We make use of the knowledge about the optimum output to control the GP surrogate model through transformation. Our transformation starts with two key observations that firstly the function value $f(\mathbf{x})$ should reach the optimum output; but secondly never be greater than the optimal value f^*, by definition of f^* being a maximum value. Therefore, the desired GP surrogate should not go above this threshold. Based on this intuition, we propose the GP transformation given f^* as follows

$$f(\mathbf{x}) = f^* - \frac{1}{2}g^2(\mathbf{x}) \qquad g(\mathbf{x}) \sim GP(m_0, K).$$

Our above transformation avoids the potential issues described in Sec. 2.2. That is we don't need a lot of samples to learn the transformation mapping for the desired property that the function is always held $f^* \geq f(\mathbf{x}), \forall \mathbf{x} \in \mathscr{X}$ as $g^2(\mathbf{x}) \geq 0$. The prior mean for $g(\mathbf{x})$ can be used either $m_0 = 0$ or $m_0 = \sqrt{2f^*}$. These choices will bring two different effects. A zero mean prior $m_0 = 0$ will tend to lift up the surrogate model closer to f^* as $f(\mathbf{x}) = f^*$ when $g(\mathbf{x}) = 0$. On the other hand, non-zero mean $m_0 = \sqrt{2f^*}$ will encourage the mean prior of f closer to zero – as a common practice in GP modeling where the output is standardized around zero $y \sim \mathscr{N}(0,1)$.

Given the observations $\mathscr{D}_f = (\mathbf{x}_i, y_i)_{i=1}^N$, we can compute the observations for g, i.e., $\mathscr{D}_g = (\mathbf{x}_i, g_i)_{i=1}^N$ where $g_i = \sqrt{2(f^* - y_i)}$. Then, we can write the posterior of $p(g \mid \mathscr{D}_g, f^*) \sim \mathscr{N}(\mu_g(\mathbf{x}), \sigma_g(\mathbf{x}))$ as $\mu_g(\mathbf{x}) = m_0 + \mathbf{k}_* \mathbf{K}^{-1}(\mathbf{g} - m_0)$ and $\sigma_g^2(\mathbf{x}) = k_{**} - \mathbf{k}_* \mathbf{K}^{-1} \mathbf{k}_*^T$ where m_0 is the prior mean of $g(\mathbf{x})$.

We don't introduce any extra parameter for the above transformation. However, the transformation causes the distribution for any f to become a non-central χ^2 process, making the analysis intractable. To tackle this problem and obtain a posterior distribution $p(f \mid \mathscr{D}_f, f^*)$ that is also Gaussian, we employ an approximation technique presented in Gunter et al. (2014); Ru et al. (2018). That is, we perform a local linearization of the transformation $h(g) = f^* - \frac{1}{2}g^2(\mathbf{x})$ around g_0 and obtain $f \approx h(g_0) + h'(g_0)(g - g_0)$ where the gradient $h'(g_0) = -g$. Following Gunter et al. (2014); Ru et al. (2018), we set $g_0 = \mu_g$ to the mode of the posterior distribution $p(g \mid .)$ and obtain an expression for f as

$$\begin{aligned} f(\mathbf{x}) &\approx f^* - \frac{1}{2}\mu_g^2(\mathbf{x}) - \mu_g(\mathbf{x})\left[g(\mathbf{x}) - \mu_g(\mathbf{x})\right] \\ &= f^* + \frac{1}{2}\mu_g^2(\mathbf{x}) - \mu_g(\mathbf{x})g(\mathbf{x}). \end{aligned}$$

We have considered the mode g_0 of linear approximation to be the multivariate function $\mu_g(\mathbf{x}), \forall \mathbf{x}$. As the property of Taylor expansion, the approximation is very good at the

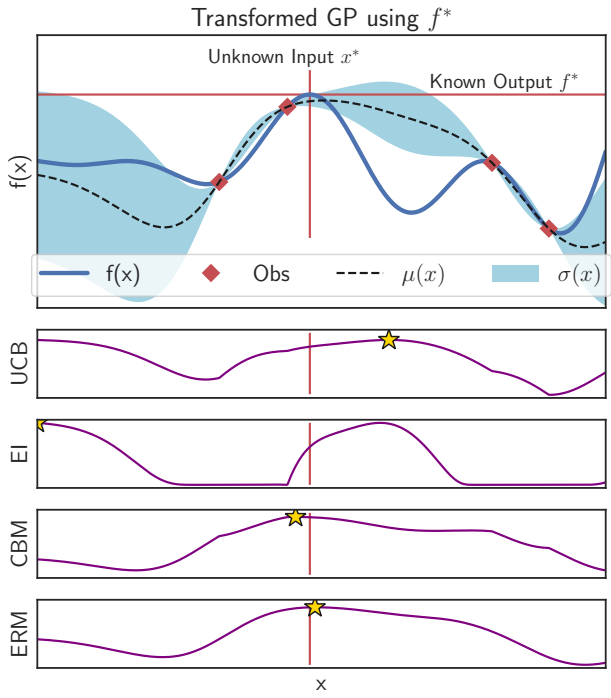

Figure 2. Illustration of the proposed acquisition functions CBM and ERM. A yellow star indicates the maximum of the acquisition function and thus is the selected point. Using the knowledge of f^*, CBM and ERM can better identify $\mathbf{x}^*$ while EI and UCB cannot.

mode g_0 and thus μ_g. Since the linear transformation of a Gaussian process remains Gaussian, the predictive posterior distribution for f now has a closed form for $p(f \mid .) = \mathcal{N}(f \mid \mu, \sigma)$ where the predictive mean and variance are given by

$$\mu(\mathbf{x}) = f^* - \frac{1}{2}\mu_g^2(\mathbf{x}), \tag{2}$$

$$\sigma(\mathbf{x}) = \mu_g(\mathbf{x})\sigma_g(\mathbf{x})\mu_g(\mathbf{x}). \tag{3}$$

These Eqs. (2) and (3) are the key to compute our acquisition functions in the next sections. As the effect of transformation, the predictive uncertainty $\sigma(\mathbf{x})$ of the transformed GP becomes larger than in the case of vanilla GP at the location where $\mu(\mathbf{x})$ is low. This is because $\mu_g(\mathbf{x})$ is high when $\mu(\mathbf{x})$ is low and thus $\sigma(\mathbf{x})$ is high in Eq. (3). This property may let other acquisition functions (e.g., UCB, EI) explore more aggressively than they should. We further examine these effects in the supplement.

We visualize the property of our transformed GP and compare with the vanilla GP in Fig. 1. By transforming the GP using f^*, we encode the knowledge about f^* into the surrogate model, and thus are able to enforce that the surrogate model gets close to but never above f^*, as desired, unlike the vanilla GP. In the supplement, we provide further illustration that transforming the surrogate model can help to find the optimum faster. We present quantitative comparison of our transformed GP and vanilla GP in Fig. 3 and in the supplement.

3.2. Confidence bound minimization

In this section, we introduce confidence bound minimization (CBM) to efficiently select the (unknown) optimum location $\mathbf{x}^*$ given $f^* = f(\mathbf{x}^*)$. Our idea is based on the underlying concept of GP-UCB (Srinivas et al., 2010). We consider the GP surrogate at any location $\mathbf{x} \in \mathcal{X}$ w.h.p.

$$\mu(\mathbf{x}) - \sqrt{\beta_t}\sigma(\mathbf{x}) \leq f(\mathbf{x}) \leq \mu(\mathbf{x}) + \sqrt{\beta_t}\sigma(\mathbf{x}) \tag{4}$$

where β_t is a hyperparameter. Given the knowledge of f^*, we can express this property at the optimum location $\mathbf{x}^*$ where $f^* = f(\mathbf{x}^*)$ to have w.h.p.

$$\mu(\mathbf{x}^*) - \sqrt{\beta_t}\sigma(\mathbf{x}^*) \leq f^* \leq \mu(\mathbf{x}^*) + \sqrt{\beta_t}\sigma(\mathbf{x}^*).$$

This is equivalent to write $|\mu(\mathbf{x}^*) - f^*| \leq \sqrt{\beta_t}\sigma(\mathbf{x}^*)$. Therefore, we can find the next point $\mathbf{x}_t$ by balancing the posterior mean being close to the known optimum f^* with having low variance. That is

$$\alpha_t^{\text{CBM}}(\mathbf{x}) = |\mu(\mathbf{x}) - f^*| + \sqrt{\beta_t}\sigma(\mathbf{x})$$

where $\mu(\mathbf{x})$ and $\sigma(\mathbf{x})$ are the GP mean and variance from Eq. (2) and Eq. (3) respectively. We select the next point by taking

$$\mathbf{x}_{t+1} = \underset{\mathbf{x}\in\mathcal{X}}{\text{argmin}}\ \alpha_t^{\text{CBM}}(\mathbf{x}). \tag{5}$$

In the above objective function, we aim to quickly locate the area potentially containing an optimum. Since the acquisition function is non-negative, $\alpha^{\text{CBM}}(\mathbf{x}) \geq 0, \forall \mathbf{x} \in \mathcal{X}$, it takes the minimum value at the ideal location where $\mu(\mathbf{x}_t) = f^*$ and $\sigma(\mathbf{x}_t) = 0$. When these two conditions are met, we can conclude that $f(\mathbf{x}_t) = f(\mathbf{x}^*)$ and thus $\mathbf{x}_t$ is what we are looking for, as the property of Eq. (4).

Because the CBM involves a hyperparameter β to which performance can be sensitive, we below propose another acquisition function incorporating the knowledge of f^* using no hyperparameter.

3.3. Expected regret minimization

We next develop our second acquisition function using f^*, called expected regret minimization (ERM). We start with the regret function $r(\mathbf{x}) = f^* - f(\mathbf{x})$. The probability of regret $r(\mathbf{x})$ on a normal posterior distribution is as follows

$$p(r) = \frac{1}{\sqrt{2\pi}\sigma(\mathbf{x})} \exp\left(-\frac{1}{2}\frac{[f^* - \mu(\mathbf{x}) - r(\mathbf{x})]^2}{\sigma^2(\mathbf{x})}\right). \tag{6}$$

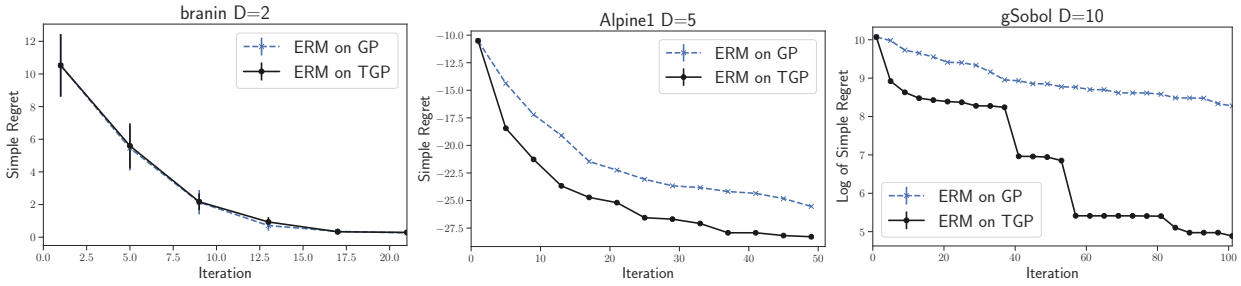

Figure 3. We show that our model performs much better using transformed Gaussian process (TGP) than the vanilla GP. The knowledge of f^* is useful to inform the surrogate model for better optimization, especially in high dimensional functions.

Algorithm 1 BO with known optimum output.

Input: #iter T, optimum value $f^* = \max_{\mathbf{x}\in\mathscr{X}} f(\mathbf{x})$

1: **while** $t \leq T$ and $f^* > \max_{\forall y_i \in D_t} y_i$ **do**
2: Construct a transformed Gaussian process surrogate model from $\mathscr{D}_t$ and f^*.
3: Estimating μ and σ from Eqs. (2) and (3).
4: Select $\mathbf{x}_t = \arg\min_{\mathbf{x}\in\mathscr{X}} \alpha_t^{\text{ERM}}(\mathbf{x})$, or $\alpha_t^{\text{CBM}}(\mathbf{x})$, using the above transformed GP model.
5: Evaluate $y_t = f(\mathbf{x}_t)$, set $g_t = \sqrt{2(f^* - y_t)}$ and augment $\mathscr{D}_t = \mathscr{D}_{t-1} \cup (\mathbf{x}_t, y_t, g_t)$.
6: **end while**

As the end goal in optimization is to minimize the regret, we consider our acquisition function to minimize this expected regret as $\alpha^{\text{ERM}}(\mathbf{x}) = \mathbb{E}[r(\mathbf{x})]$. Using the likelihood function in Eq. (6), we write the expected regret minimization acquisition function as

$$\mathbb{E}[r(\mathbf{x})] = \int \frac{r}{\sqrt{2\pi}\sigma(\mathbf{x})} \exp\left(-\frac{1}{2}\frac{[f^* - \mu(\mathbf{x}) - r(\mathbf{x})]^2}{\sigma^2(\mathbf{x})}\right) dr.$$

Let $z = \frac{f^* - \mu(\mathbf{x})}{\sigma(\mathbf{x})}$, we obtain the closed-form computation as

$$\alpha^{\text{ERM}}(\mathbf{x}) = \sigma(\mathbf{x})\,\phi(z) + [f^* - \mu(\mathbf{x})]\,\Phi(z) \qquad (7)$$

where $\phi(z)$ and $\Phi(z)$ are the standard normal p.d.f. and c.d.f., respectively. To select the next point, we minimize this acquisition function which is equivalent to minimizing the expected regret,

$$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x}\in\mathscr{X}} \alpha^{\text{ERM}}(\mathbf{x}) = \arg\min_{\mathbf{x}\in\mathscr{X}} \mathbb{E}[r(\mathbf{x})]. \qquad (8)$$

Our choice in Eq. (8) is where to minimize the expected regret. We can see that this acquisition function is always positive $\alpha^{\text{ERM}}(\mathbf{x}) \geq 0, \forall \mathbf{x} \in \mathscr{X}$. It is minimized at the ideal location $\mathbf{x}_t$, i.e., $\alpha^{\text{ERM}}(\mathbf{x}_t) = \mathbb{E}[r(\mathbf{x})] = 0$, when $f^* - \mu(\mathbf{x}_t) = 0$ and $\sigma(\mathbf{x}_t) = 0$. This case happens at the desired location where the GP predictive value is equal to the true f^* with zero GP uncertainty.

Although our ERM is inspired by the EI in the way that we define the regret function and take the expectation, the resulting approach is different in the following. The original EI strategy is to balance exploration and exploitation, i.e., prefers high GP mean and high GP variance. On the other hand, ERM will not encourage such trade-off directly. Instead, ERM selects the point to minimize the expected regret $\mathbb{E}[f^* - f(\mathbf{x})]$ with $\mu(\mathbf{x})$ being closer to the known f^* while having low variance to make sure that the GP estimation at our chosen location is correct. Then, if the chosen location turns out to be not expected (e.g., poor function value), the GP is updated and ERM will move to another place which minimizes the new expected regret. Therefore, these behaviors of EI and our ERM are radically different.

Algorithm. We summarize all steps in Algorithm 1. Given the original observation $\{\mathbf{x}_i, y_i\}_{i=1}^N$ and f^*, we compute $g_i = \sqrt{2(f^* - y_i)}$, then build a transformed GP using $\{\mathbf{x}_i, g_i\}_{i=1}^N$. Using a transformed GP, we can predict the mean $\mu(\mathbf{x})$ and uncertainty $\sigma(\mathbf{x})$ at any location $\mathbf{x}$ from Eqs. (2) and (3) which are used to compute the CBM and ERM acquisition functions in Eq. (5) and Eq. (8). Our formulas are in closed-forms and the algorithm is easy to implement. In addition, our computational complexity is as cheap as the GP-UCB and EI.

Illustration of CBM and ERM

We illustrate in Fig. 2 our proposed CBM and ERM comparing to the standard UCB and EI with both vanilla GP and transformed GP settings. Our acquisition functions make use of the knowledge about f^* to make an informed decision about where we should query. That is, CBM and ERM will select the location where the GP mean $\mu(\mathbf{x})$ is close to the optimal value f^* and we are highly certain about it – or low $\sigma(\mathbf{x})$. On the other hand, GP-UCB and EI will always keep exploring as the principle of explore-exploit without using the knowledge of f^*. As the results, GP-UCB and EI can not identify the unknown location $\mathbf{x}^*$ efficiently as opposed to our acquisition functions.

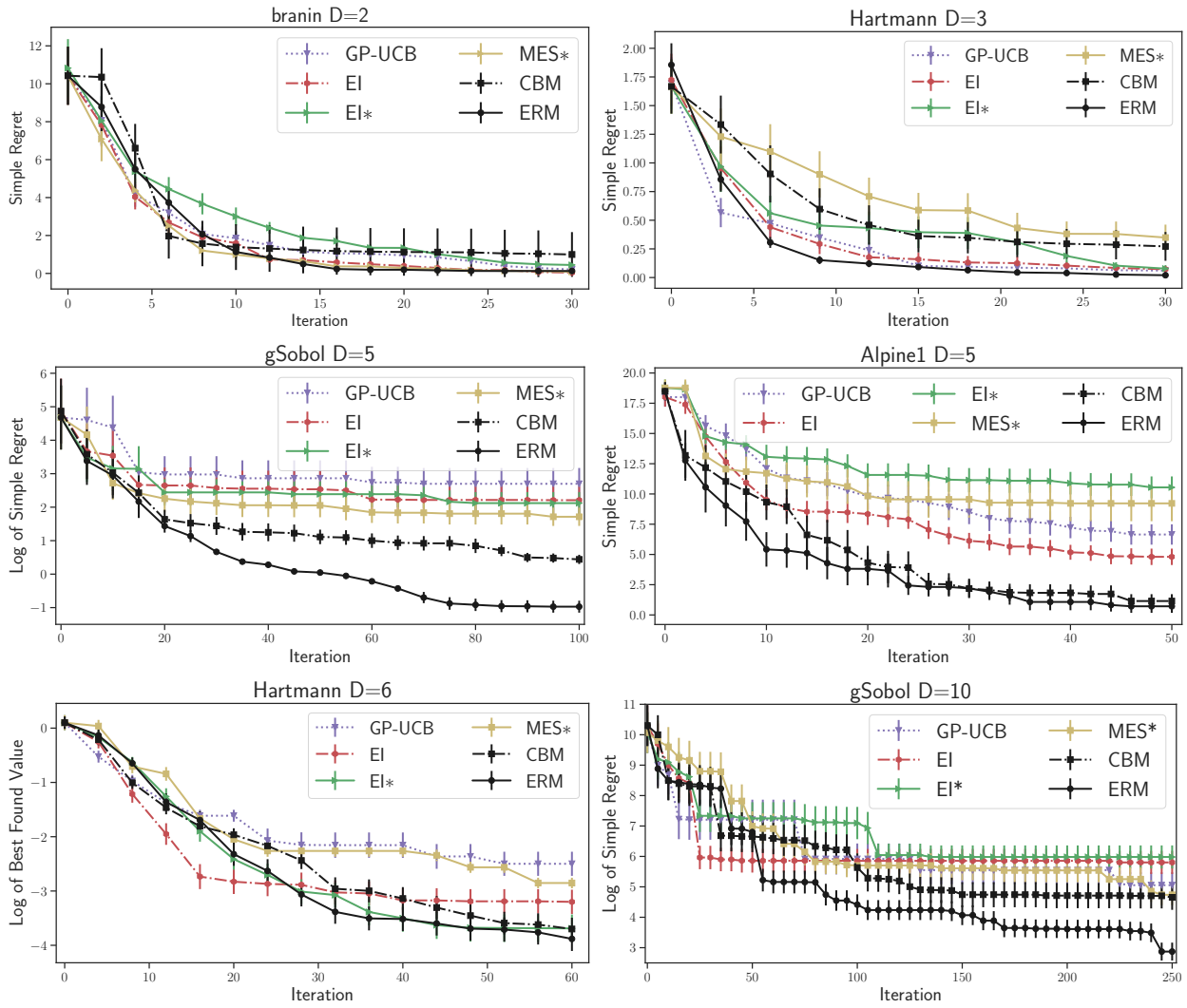

Figure 4. Optimization comparison using benchmark functions from $D = 2$ to $D = 10$ dimensions. We demonstrate that the known optimum output f^* will significantly boost the performances in high dimensions, such as in Alpine1 $D = 5$, gSobol $D = 5$ and $D = 10$.

4. Experiments

The main goal of our experiments is to show that we can effectively exploit the known optimum output to improve Bayesian optimization performance. We first demonstrate the efficiency of our model on benchmark functions. Then, we perform hyperparameter optimization for a XGBoost classification on Skin Segmentation dataset and a deep reinforcement learning task on CartPole problem where the optimum values are publicly available. We provide additional experiments in the supplement and the code is released at.[2]

Settings. All implementations are in Python. The experiments are independently performed 20 times. We use the squared exponential kernel $k(x,x') = \exp\left(-||x-x'||^2/\sigma_l\right)$ where σ_l is optimized from the GP marginal likelihood, the input is scaled $x \sim [0,1]^d$ and the output is standardized $y \sim \mathcal{N}(0,1)$ for robustness. We follow Theorem 3 in Srinivas et al. (2010) to specify $\beta_t = 2f^* + 300\log^3(t/\delta)$.

To avoid our algorithms from the early exploitation, we use a standard BO (with GP and EI) at the earlier iterations and our proposed algorithms at later iterations once the f^* value has been reached by the upper confidence bound. The reaching f^* condition can be checked in each BO iteration using a global optimization toolbox, i.e., $\exists \mathbf{x} \mid f^* \leq \mu(\mathbf{x}) + \sqrt{\beta_t}\sigma(\mathbf{x})$.

Our CBM and ERM use a transformed Gaussian process (Sec. 3.1) in all experiments. We learn empirically that using a transformed GP as a surrogate will boost the performance for our CBM and ERM significantly against the case of using vanilla GP. For other baselines, we use both surrogates

[2] github.com/ntienvu/KnownOptimum_BO

Table 1. Hyperparameters for XGBoost.

Known $f^* = 100$ (Accuracy)			
Variables	Min	Max	Found $\mathbf{x}^*$
min child weight	1	20	4.66
colsample bytree	0.1	1	0.99
max depth	5	15	9.71
subsample	0.5	1	0.77
alpha	0	10	0.82
gamma	0	10	0.51

and report the best performance. We present further details of experiments in the supplement.

Baselines. To the best of our knowledge, there is no baseline in directly using the known optimum output for BO. We select to compare our model with the vanilla BO without knowing the optimum value including the GP-UCB (Srinivas et al., 2010) and EI (Mockus et al., 1978). In addition, we use two other baselines using f^* described in Sec. 2.1.

4.1. Comparison on benchmark function given f^*

We perform optimization tasks on 6 common benchmark functions.[3] For these functions, we assume that the optimum value f^* is available in advance which will be given to the algorithm. We use the simple regret for comparison, defined as $f^* - \max_{\forall i \le t} f(x_i)$ for maximization problem.

The experimental results are presented in Fig. 4 which shows that our proposed CBM and ERM are among the best approaches over all problems considered. This is because our framework has utilized the additional knowledge of f^* to build an informed surrogate model and decision functions. Especially, ERM outperforms all methods by a wide margin. While CBM can be sensitive to the hyperparameter β_t, ERM has no parameter and is thus more robust.

Particularly, our approaches with f^* perform significantly better than the baselines in gSobol and Alpine1 functions. The results indicate that the knowledge of f^* is particularly useful for high dimensional functions.

4.2. Tuning machine learning algorithms with f^*

A popular application of BO is for hyperparameter tuning of machine learning models. Some machine learning tasks come with the known optimal value in advance. We consider tuning (1) a classification task using XGBoost on a Skin dataset and (2) a deep reinforcement learning task on a CartPole problem (Barto et al., 1983). Further detail of the experiment is described in the supplement.

[3] https://www.sfu.ca/~ssurjano/optimization.html

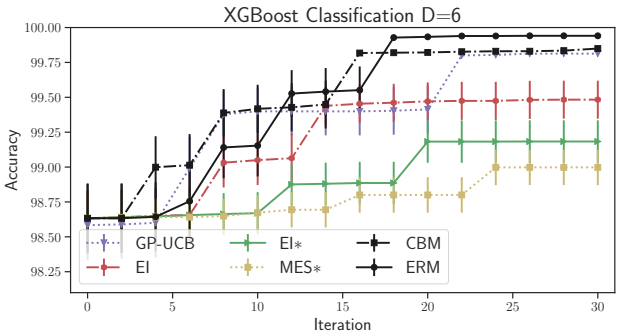

Figure 5. Tuning performance on Skin dataset.

XGBoost classification. We demonstrate a classification task using XGBoost (Chen & Guestrin, 2016) on a Skin Segmentation dataset[4] where we know the best accuracy is $f^* = 100$, as shown in Table 1 of Le et al. (2016).

The Skin Segmentation dataset is split into 15% for training and 85% for testing for a classification problem. There are 6 hyperparameters for XGBoost (Chen & Guestrin, 2016) which is summarized in Table 1. To optimize the integer (ordinal) variables, we round the scalars to the nearest values in the continuous space. We present the result in Fig. 5. Our proposed ERM is the best approach, outperforming all the baselines by a wide margin. This demonstrates the benefit of exploiting the optimum value f^* in BO.

Deep reinforcement learning. CartPole is a pendulum with a center of gravity above its pivot point. The goal is to keep the cartpole balanced by controlling a pivot point. The reward performance in CartPole is often averaged over 100 consecutive trials. The maximum reward is known from the literature[5] as $f^* = 200$.

We then use a deep reinforcement learning (DRL) algorithm to solve the CartPole problem and use Bayesian optimization to optimize the hyperparameters. In particular, we use the advantage actor critic (A2C) (Sutton & Barto, 1998) which possesses three sensitive hyperparameters, including the discount factor γ, the learning rate for actor model, α_1, and the learning rate for critic model, α_2. We choose not to optimize the deep learning architecture for simplicity. We use Bayesian optimization given the known optimum output of 200 to find the best hyperparameters for the A2C algorithm. We present the results in Fig. 6 where our ERM reaches the optimal performance after 20 iterations outperforming all other baselines. In Fig. 6 Left, we visualize the selected point $\{\mathbf{x}_t\}_{t=1}^T$ by our ERM acquisition function. Our ERM initially explores at several places and then exploits in the high value region (yellow dots).

[4] https://archive.ics.uci.edu/ml/datasets/skin+segmentation
[5] https://gym.openai.com/envs/CartPole-v0/

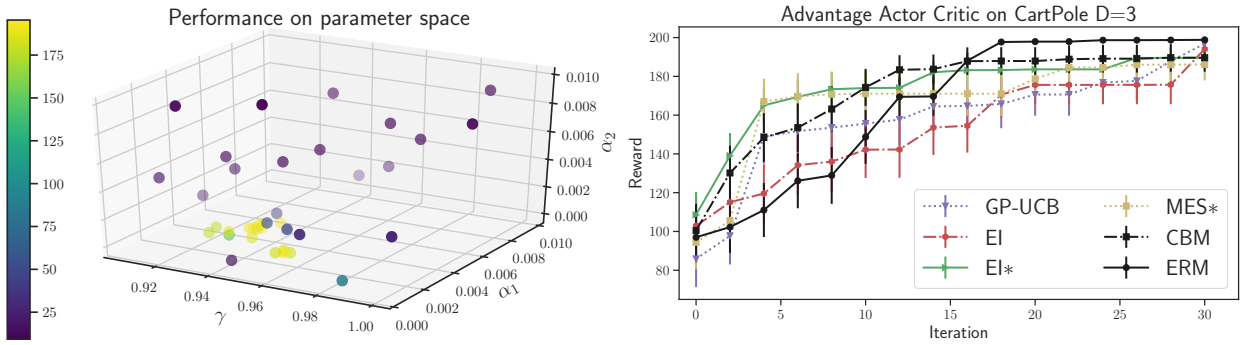

Figure 6. Hyperparameter tuning for a deep reinforcement learning algorithm. The optimum value is available $f^* = 200$. Left: Selected points by our algorithm on tuning DRL. Color indicates the reward $f(x)$ value. Right: Performance comparison with the baselines.

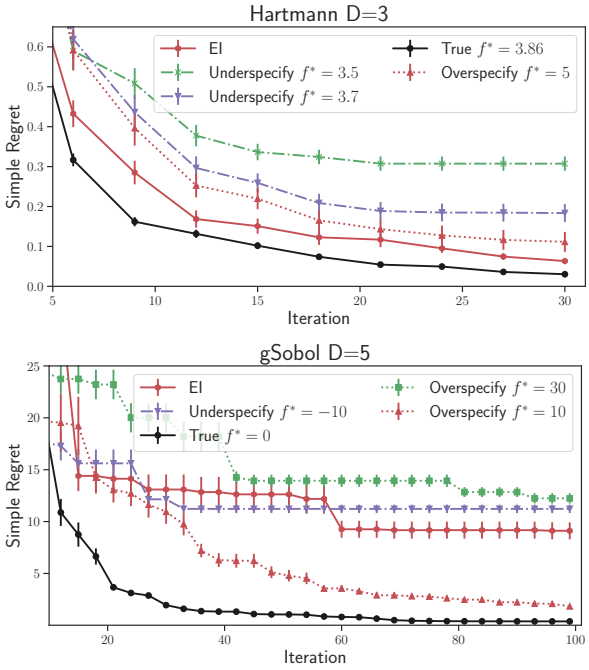

Figure 7. Experiments with ERM in the maximization problem. Over-specifying is when the value of f^* is larger than the true optimal value and under-specifying is when the value of f^* is smaller than the true. Top: the true $f^* = 3.86$ for Hartmann. Bottom: the true $f^* = 0$ for gSobol. Both cases of misspecifying f^* will degrade the performance.

4.3. What happens if we misspecify the optimum value

We now consider setting the f^* to a value which is not the true optimum of the black-box function. We show that our model's performance will drop with misspecified value of f^* with different effects. Specifically, we both set f^* larger (over-specify) and smaller (under-specify) than the true value in a maximization problem.

We experiment with our ERM using this misspecified setting of f^* in Fig. 7. The results suggest that our algorithm using the true value ($f^* = 3.86$ for Hartmann and $f^* = 0$ for gSobol) will have the best performance. Both over-specifying and under-specifying f^* will return worse performance. These misspecified settings slightly perform worse than the standard EI in Hartmann while it still performs better than EI for overspecifying $f^* = 10$ in gSobol. In particular, the under-specifying case will result in worse performance than over-specifying. This is because our acquisition function will get stuck at the area once being found wrongly as the optimal. On the other hand, if we over-specify f^*, our algorithm continues exploring to find the optimum because it can not find the point where both conditions are met $\sigma(\mathbf{x}_t) = 0$ and $f^* = \mu(\mathbf{x}_t)$.

Discussion. We make the following observations. If we know the true value f^*, ERM will return the best result. If we do not know the exact f^* value, the performance of our approach is degraded. Thus, we should use the existing BO approaches, such as EI, for the best performance.

5. Conclusion and Future Work

In this paper, we have considered a new setting in Bayesian optimization with known optimum output. We present a transformed Gaussian process surrogate to model the objective function better by exploiting the knowledge of f^*. Then, we propose two decision strategies which can exploit the function optimum value to make informed decisions. Our approaches are intuitively simple and easy to implement. By using extra knowledge of f^*, we demonstrate that our ERM can converge quickly to the optimum in benchmark functions and real-world applications.

In future work, we can expand our algorithm to handle batch setting for parallel evaluations or extend this work to other classes of surrogate functions such as Bayesian neural networks (Neal, 2012) and deep GP (Damianou & Lawrence, 2013). Moreover, we can extend the model to handle f^* within a range of ε from the true output.

References

Astudillo, Raul and Frazier, Peter. Bayesian optimization of composite functions. In *International Conference on Machine Learning*, pp. 354–363, 2019.

Barto, Andrew G, Sutton, Richard S, and Anderson, Charles W. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5):834–846, 1983.

Berk, Julian, Nguyen, Vu, Gupta, Sunil, Rana, Santu, and Venkatesh, Svetha. Exploration enhanced expected improvement for Bayesian optimization. In *Machine Learning and Knowledge Discovery in Databases*. Springer, 2018.

Brochu, Eric, Cora, Vlad M, and De Freitas, Nando. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.

Chen, Tianqi and Guestrin, Carlos. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794. ACM, 2016.

Chen, Yutian, Huang, Aja, Wang, Ziyu, Antonoglou, Ioannis, Schrittwieser, Julian, Silver, David, and de Freitas, Nando. Bayesian optimization in AlphaGo. *arXiv preprint arXiv:1812.06855*, 2018.

Damianou, Andreas and Lawrence, Neil. Deep gaussian processes. In *Artificial Intelligence and Statistics*, pp. 207–215, 2013.

Frazier, Peter I. A tutorial on Bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

Frazier, Peter I and Wang, Jialei. Bayesian optimization for materials design. In *Information Science for Materials Discovery and Design*, pp. 45–75. Springer, 2016.

Gunter, Tom, Osborne, Michael A, Garnett, Roman, Hennig, Philipp, and Roberts, Stephen J. Sampling for inference in probabilistic models with fast Bayesian quadrature. In *Advances in neural information processing systems*, pp. 2789–2797, 2014.

Hennig, Philipp and Schuler, Christian J. Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, 13:1809–1837, 2012.

Hernández-Lobato, Daniel and Hernández-Lobato, José Miguel. Scalable Gaussian process classification via expectation propagation. In *Artificial Intelligence and Statistics*, pp. 168–176, 2016.

Hernández-Lobato, José Miguel, Hoffman, Matthew W, and Ghahramani, Zoubin. Predictive entropy search for efficient global optimization of black-box functions. In *Advances in Neural Information Processing Systems*, pp. 918–926, 2014.

Hernández-Lobato, José Miguel, Requeima, James, Pyzer-Knapp, Edward O, and Aspuru-Guzik, Alán. Parallel and distributed Thompson sampling for large-scale accelerated exploration of chemical space. *In International Conference on Machine Learning*, pp. 1470–1479, 2017.

Hoffman, Matthew W and Ghahramani, Zoubin. Output-space predictive entropy search for flexible global optimization. 2015.

Hutter, Frank, Hoos, Holger H, and Leyton-Brown, Kevin. Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization*, pp. 507–523. Springer, 2011.

Klein, Aaron, Falkner, Stefan, Bartels, Simon, Hennig, Philipp, and Hutter, Frank. Fast Bayesian optimization of machine learning hyperparameters on large datasets. In *Artificial Intelligence and Statistics*, pp. 528–536, 2017.

Kuss, Malte and Rasmussen, Carl Edward. Assessing approximate inference for binary Gaussian process classification. *Journal of machine learning research*, 6(Oct): 1679–1704, 2005.

Le, Trung, Nguyen, Vu, Nguyen, Tu Dinh, and Phung, Dinh. Nonparametric budgeted stochastic gradient descent. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pp. 654–572, 2016.

Letham, Benjamin, Karrer, Brian, Ottoni, Guilherme, Bakshy, Eytan, et al. Constrained Bayesian optimization with noisy experiments. *Bayesian Analysis*, 14(2):495–519, 2019.

Li, Cheng, Santu, Rana, Gupta, Sunil, Nguyen, Vu, Venkatesh, Svetha, Sutti, Alessandra, Leal, David Rubin De Celis, Slezak, Teo, Height, Murray, Mohammed, Mazher, and Gibson, Ian. Accelerating experimental design by incorporating experimenter hunches. In *IEEE International Conference on Data Mining (ICDM)*, pp. 257–266, 2018.

MacKay, David JC. Introduction to Gaussian processes. *NATO ASI Series F Computer and Systems Sciences*, 168: 133–166, 1998.

Mockus, Jonas, Tiesis, Vytautas, and Zilinskas, Antanas. The application of Bayesian methods for seeking the extremum. *Towards global optimization*, 2(117-129): 2, 1978.

Neal, Radford M. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Nickisch, Hannes and Rasmussen, Carl Edward. Approximations for binary Gaussian process classification. *Journal of Machine Learning Research*, 9(Oct):2035–2078, 2008.

Oh, ChangYong, Gavves, Efstratios, and Welling, Max. Bock: Bayesian optimization with cylindrical kernels. In *International Conference on Machine Learning*, pp. 3865–3874, 2018.

Osborne, Michael, Garnett, Roman, Ghahramani, Zoubin, Duvenaud, David K, Roberts, Stephen J, and Rasmussen, Carl E. Active learning of model evidence using Bayesian quadrature. In *Advances in neural information processing systems*, pp. 46–54, 2012.

Rasmussen, Carl Edward. Gaussian processes for machine learning. 2006.

Riihimäki, Jaakko, Jylänki, Pasi, and Vehtari, Aki. Nested expectation propagation for Gaussian process classification with a multinomial probit likelihood. *Journal of Machine Learning Research*, 14(Jan):75–109, 2013.

Ru, Binxin, McLeod, Mark, Granziol, Diego, and Osborne, Michael A. Fast information-theoretic Bayesian optimisation. In *International Conference on Machine Learning*, pp. 4381–4389, 2018.

Shahriari, Bobak, Swersky, Kevin, Wang, Ziyu, Adams, Ryan P, and de Freitas, Nando. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.

Snelson, Edward, Ghahramani, Zoubin, and Rasmussen, Carl E. Warped Gaussian processes. In *Advances in neural information processing systems*, pp. 337–344, 2004.

Snoek, Jasper, Larochelle, Hugo, and Adams, Ryan P. Practical Bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pp. 2951–2959, 2012.

Snoek, Jasper, Rippel, Oren, Swersky, Kevin, Kiros, Ryan, Satish, Nadathur, Sundaram, Narayanan, Patwary, Mostofa, Prabhat, Mr, and Adams, Ryan. Scalable Bayesian optimization using deep neural networks. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 2171–2180, 2015.

Springenberg, Jost Tobias, Klein, Aaron, Falkner, Stefan, and Hutter, Frank. Bayesian optimization with robust bayesian neural networks. In *Advances in Neural Information Processing Systems*, pp. 4134–4142, 2016.

Srinivas, Niranjan, Krause, Andreas, Kakade, Sham, and Seeger, Matthias. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the 27th International Conference on Machine Learning*, pp. 1015–1022, 2010.

Sutton, Richard S and Barto, Andrew G. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Wang, Zi and Jegelka, Stefanie. Max-value entropy search for efficient Bayesian optimization. In *International Conference on Machine Learning*, pp. 3627–3635, 2017.

Wang, Zi, Zhou, Bolei, and Jegelka, Stefanie. Optimization as estimation with Gaussian processes in bandit settings. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pp. 1022–1031, 2016.

Wang, Zi, Gehring, Clement, Kohli, Pushmeet, and Jegelka, Stefanie. Batched large-scale Bayesian optimization in high-dimensional spaces. In *International Conference on Artificial Intelligence and Statistics*, pp. 745–754, 2018.

Wang, Ziyu and de Freitas, Nando. Theoretical analysis of Bayesian optimisation with unknown Gaussian process hyper-parameters. *arXiv preprint arXiv:1406.7758*, 2014.