Meta-Learning Priors for Safe Bayesian Optimization

Anonymous Author(s)
Affiliation
Address
email

Abstract: In robotics, optimizing controller parameters under safety constraints is an important challenge. Safe Bayesian optimization (BO) quantifies uncertainty in the objective and constraints to safely guide exploration in such settings. Hand-designing a suitable probabilistic model can be challenging however. In the presence of unknown safety constraints, it is crucial to choose reliable model hyper-parameters to avoid safety violations. Here, we propose a data-driven approach to this problem by *meta-learning priors* for safe BO from offline data. We build on a meta-learning algorithm, F-PACOH, capable of providing reliable uncertainty quantification in settings of data scarcity. As core contribution, we develop a novel framework for choosing *safety-compliant priors* in a data-riven manner via empirical uncertainty metrics and a *frontier search* algorithm. On benchmark functions and a high-precision motion system, we demonstrate that our meta-learnt priors accelerate convergence of safe BO approaches while maintaining safety.

Keywords: Meta-learning, Safety, Controller tuning, Bayesian Optimization

1 Introduction

Optimizing a black-box function with as few queries as possible is a ubiquitous problem in science and engineering. Bayesian Optimization (BO) is a promising paradigm, which learns a probabilistic surrogate model (often a Gaussian process, GP) of the unknown function to guide exploration. BO has been successfully applied for optimizing sensor configurations [1, 2] or tuning the parameters of robotic controllers [3, 4, 5, 6, 7]. However, such real-world applications are often subject to safety constraints which must not be violated in the process of optimization, e.g., the robot not getting damaged. Often, the dependence of the safety constraints on the query inputs is a priori unknown and can only be observed by measurement. To cope with these requirements, safe BO methods [8, 9, 10] model both the objective *and constraint* functions with GPs. The uncertainty in the constraint is used to approximate the feasible region from within, to guarantee that no safety violations occur. Therefore, a critical requirement for safe BO is the reliability of the uncertainty estimates of our models. Typically, it is assumed that a correct GP prior, upon which the uncertainty estimates hinge, is exogenously given [e.g., 8, 9, 10]. In practice, however, appropriate choices for the kernel variance and lengthscale are unknown and typically have to be chosen very conservatively or hand-tuned by trial and error, a problematic endeavour in a safety-critical setting. A too conservative choice dramatically reduces sample efficiency, whereas overestimating smoothness may risk safety violations.

Addressing these shortcomings, we develop an approach for *meta-learning informative, but safe, GP priors in a data-driven way* from related offline data. We build on the F-PACOH meta-learning method [11] which is capable of providing reliable uncertainty estimates, even in the face of data-scarcity and out-of-distribution data. However, their approach still relies on an appropriate kernel choice on which it falls back to in the absence of sufficient data. We propose a novel framework for choosing safety-compliant kernel hyper-parameters in a data-riven manner based on calibration and sharpness metrics of the confidence intervals. To optimize these uncertainty metrics, we devise a *frontier search* algorithm that efficiently exploits the monotone structure of the problem. The resulting *Safe Meta-Bayesian Optimization* (SAMBO) approach can be instantiated with existing safe BO

Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.

methods and utilize the improved, meta-learned GPs to perform safe optimization more efficiently. In our experiments, we evaluate and compare our proposed approach on benchmark functions as well as controller tuning for a high-precision motion system. Throughout, SAMBO significantly improves the query efficiency of popular safe BO methods, without compromising safety.

2 Related Work

Safe BO aims to efficiently optimize a black-box function under safety-critical conditions, where unknown safety constraints must not be violated. Constrained variants of standard BO methods [12, 13, 14] return feasible solutions, but do not reliably exclude unsafe queries. In contrast, SAFEOPT [8, 9] and related variants [15] guarantee safety at all times and have been used to safely tune controllers in various applications [e.g., 16, 17]. While SAFEOPT explores in an undirected manner, GOOSE [10, 18] by expanding the safe set in a goal-oriented fashion. All mentioned methods rely on GPs to model the target and constraint function and assume that correct kernel hyper-parameters are given. Our work is complementary: We show how to use related offline data to obtain informative and safe GP priors in a data-driven way that makes the downstream safe BO more query efficient.

Meta-Learning. Common approaches in meta-learning amortize inference by a recurrent model [19, 20, 21], learn a shared embedding space [22, 23, 24, 25] or a neural network initialization so it can be quickly adapted to new tasks [26, 27, 28]. Probabilistic modeling enables uncertainty quantification in meta-learning [29, 30, 31, 32]. However, it fails providing reliable uncertainty estimates when the available data is limited. Recent meta-learning methods aim at providing reliable confidence intervals even when data is scarce and non-i.i.d [11, 33, 34]. These methods extend meta-learning to interactive and life-long settings, however, they either make unrealistic model assumptions or hinge on hyper-parameters whose improper choice is critical in safety constraint settings. Our work uses F-PACOH [11] to meta-learn reliable GP priors, but removes the need for hand-specifying a correct hyper-prior by choosing its parameters in a data-driven, safety-aware manner.

3 Problem Statement and Background

3.1 Problem Statement

We consider the problem of safe Bayesian Optimization (safe BO), seeking the global minimizer

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) \quad \text{s.t.} \quad q(\mathbf{x}) \leq 0 \tag{1}$$

of a function $f : \mathcal{X} \to \mathbb{R}$ over a bounded domain $\mathcal{X}$ subject to a safety constraint $q(\mathbf{x}) \leq 0$ with constraint function $q : \mathcal{X} \to \mathbb{R}$. For instance, we may want to optimize the controller parameter of a robot without subjecting it to potentially damaging vibrations or collisions. During the optimization, we iteratively make queries $\mathbf{x}_1, ..., \mathbf{x}_T \in \mathcal{X}$ and observe noisy feedback $\tilde{f}_1, ..., \tilde{f}_T$ and $\tilde{q}_1, ..., \tilde{q}_T$, e.g., via $\tilde{f}_t = f(\mathbf{x}_t) + \epsilon_f$ and $\tilde{q}_t = q(\mathbf{x}_t) + \epsilon_q$ where ϵ_f, ϵ_q is σ^2 sub-Gaussian noise [35, 36]. In our setting, performing a query is assumed to be costly, e.g., running the robot and observing relevant measurements. Hence, we want to find a solution as close to the global minimum in as few iterations as possible without violating the safety constraint with any query we make.

Additionally, we assume to have access to datasets $\mathcal{D}_{1,T_1}, ..., \mathcal{D}_{n,T_n}$ with observations from n statistically similar but distinct data-generating systems, e.g., data of the same robotic platform under different conditions. Each dataset $\mathcal{D}_{i,T_i} = \{(\mathbf{x}_{i,1}, \tilde{f}_{i,1}, \tilde{q}_{i,1}), ..., (\mathbf{x}_{i,T_i}, \tilde{f}_{i,T_i}, \tilde{q}_{i,T_i})\}$ consists of T_i measurement triples, where $\tilde{f}_{i,t} = f_i(\mathbf{x}_{i,t}) + \epsilon_{f_i}$ and $\tilde{q}_{i,t} = q_i(\mathbf{x}_{i,t}) + \epsilon_{q_i}$ are the noisy target and constraints observations. We assume that the underlying functions $f_1, ..., f_n$ and $q_1, ..., q_n$ which generated the data are representative of our current target and constraint functions f and q, e.g., that they are all i.i.d. draws from the same stochastic process. However, the data within each dataset may be highly dependent (i.e., non-i.i.d.). For instance, each $\mathcal{D}_{i,T_i}$ may correspond to the queries and observations from previous safe BO sessions with the same robot under different conditions.

In this paper, we ask the question how we can harness such related data sources to make safe BO on our current problem of interest more query efficient, without compromising safety.

3.2 Safe Bayesian Optimization Methods

Safe BO methods construct a Bayesian *surrogate model* of the functions f and q based on previous observations $\mathcal{D}_t = \{(\mathbf{x}_{t'}, \tilde{f}_{t'}, \tilde{q}_{t'})\}_{t'<t}$. Typically, a Gaussian Process $\mathrm{GP}\left(f(\mathbf{x})|m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right)$ with mean $m(\mathbf{x})$ and kernel function $k(\mathbf{x}, \mathbf{x}')$ is employed to form posterior beliefs $p(f(\mathbf{x})|\mathcal{D}_t) = \mathcal{N}(\mu_t^f(\mathbf{x}), (\sigma_t^f(\mathbf{x}))^2)$ and $p(q(\mathbf{x})|\mathcal{D}_t) = \mathcal{N}(\mu_t^q(\mathbf{x}), (\sigma_t^q(\mathbf{x}))^2)$ over function values [37]. Based on the predictive posterior, we can form confidence intervals (CIs) to the confidence level $\alpha \in [0, 1]$

$$CI_\alpha^f(\mathbf{x}|\mathcal{D}_t) = \left[\mu_t^f(\mathbf{x}) \pm \beta^f(\alpha)\sigma_t^f(\mathbf{x})\right] \tag{2}$$

where $\beta^f(\alpha)$ the scaling of the standard deviation is set such that $f(x)$ is in the CI with probability α. For BO, we often employ a shift invariant kernel $k(\mathbf{x}, \mathbf{x}') = \nu\phi(||\mathbf{x}-\mathbf{x}'||/l)$, where ν is its variance, l the lengthscale and ϕ a positive function, e.g., squared exponential (SE) $\phi(z) = \exp(-z^2)$.

BO algorithms typically choose their query points by maximizing an acquisition function based on $p(f(\mathbf{x})|\mathcal{D}_t)$ which trades-off exploration and exploitation [38, 39, 40, 41]. When we have safety constraints, we need to maintain a *safe set* $\mathcal{S}_t(\alpha) = \{\mathbf{x} \in \mathcal{X} | \mu_t^q(\mathbf{x}) + \beta^q(\alpha)\sigma_t^q(\mathbf{x}) < 0\}$ which contains parts of the domain we know to be safe with high-probability α. To maintain safety, we can only query points within the current safe set. Additionally, we need to explore w.r.t. q so that we can expand the safe set. For instance, SafeOpt [8, 9, 16] computes a safe query candidate that optimizes the acquisition function for f as well as a query candidate that promises the best expansion of the safe set. Then it selects the one which has the highest uncertainty. While SafeOpt expands the safe set in an undirected way, GoOSE [10, 18] does so in a more directed manner to avoid unnecessary expansion queries that are irrelevant for minimizing the target function. To do so, in addition to the pessimistic $\mathcal{S}_t$, it also maintains an optimistic safe set which is used to calculate a query candidate $\mathbf{x}_t^{\mathrm{opt}}$ that maximizes the acquisition function for f. If $\mathbf{x}_t^{\mathrm{opt}}$ is outside of $\mathcal{S}_t$, it chooses safe query points aiming at expanding $\mathcal{S}_t$ in the direction of $\mathbf{x}_t^{\mathrm{opt}}$. See Appx. A for more details.

3.3 Meta-Learning GP Priors

In meta-learning [42, 43], we aim to extract prior knowledge (i.e., inductive bias) from a set of related learning tasks. Typically, the meta-learner is given such learning tasks in the form of n datasets $\mathcal{D}_{1,T_1}, ..., \mathcal{D}_{n,T_n}$ with $\mathcal{D}_{i,T_i} = \{(\mathbf{x}_{i,t}, y_{i,t})\}_{t=1}^{T_i}$ and outputs a refined prior distribution or hypothesis space which can then be used to accelerate inference on a new, but related, learning task.

Prior work propose to meta-learning GP priors [44, 45, 32], tough, fail to maintain reliable uncertainty estimates when data is scarce and/or non-i.i.d.. The recently introduced F-PACOH method [11] overcomes this issue, by using a regularization approach in the function space. As previous work [e.g., 46, 45], they use a learnable GP prior $\rho_{\boldsymbol{\theta}}(h) = \mathrm{GP}(h(\mathbf{x})|m_{\boldsymbol{\theta}}(\mathbf{x}), k_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{x}')$ where the mean and kernel function are neural networks with parameters $\boldsymbol{\theta}$ and employ the marginal log-likelihood to fit the $\rho_{\boldsymbol{\theta}}(h)$ to the meta-training data. However, during the meta-learning, they regularize $\rho_{\boldsymbol{\theta}}(h)$ towards a Vanilla GP hyper-prior $\rho(h) = \mathrm{GP}(h(\mathbf{x})|0, k(\mathbf{x}, \mathbf{x}'))$ with the SE kernel. To do so, they uniformly sample random measurement sets $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_m] \overset{\text{i.i.d.}}{\sim} \mathcal{U}(\mathcal{X})$ from the domain and compare the GPs finite marginals $\rho_{\boldsymbol{\theta}}(\mathbf{h}^{\mathbf{X}}) = p_{\boldsymbol{\theta}}(h(\mathbf{x}_1), ..., h(\mathbf{x}_m))$ and $\rho(\mathbf{h}^{\mathbf{X}})$ through their KL-divergence. The resulting meta-learning loss with the functional KL regularizer

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}\left(-\frac{1}{T_i}\underbrace{\ln Z(\mathcal{D}_{i,T_i}, \rho_{\boldsymbol{\theta}})}_{\text{marginal log-likelihood}} + \left(\frac{1}{\sqrt{n}} + \frac{1}{nT_i}\right)\underbrace{\mathbb{E}_{\mathbf{X}}\left[KL[\rho_{\boldsymbol{\theta}}(\mathbf{h}^{\mathbf{X}})||\rho(\mathbf{h}^{\mathbf{X}})]\right]}_{\text{functional KL-divergence}}\right) \tag{3}$$

makes sure that, in the absence of sufficient meta-training data, the learned GP behaves like a Vanilla GP. Overall, this allows us to meta-learn a more informative GP prior which still yields reliable confidence intervals, even if the meta-training data was collected via safe BO and is thus not i.i.d.

4 Choosing the Safe Kernel Hyper-Parameters

Important for safe BO is the reliability of the uncertainty estimates of our objective and constraint models. For GPs, the kernel hyper-parameters with the biggest influence on the CIs. If the kernel

variance ν is chosen too small and/or the lengthscale l to large, our models become over-confident and the corresponding BO unsafe. In the reverse case, the CIs become too conservative and safe BO requires many queries to progress. Despite the assumption commonly made in earlier work, [e.g., 8, 9, 10], appropriate choices for ν and l are unknown. In practice, they are typically chosen conservatively or hand-tuned by trial and error, problematic in safety-critical settings. Aiming to address this issue, we develop a framework for choosing the kernel hyper-parameters in a data-driven manner.

4.1 Assessing kernel hyper-parameters: Calibration and sharpness

Our approach is based on the *calibration* and *sharpness* of uncertainty estimates [47, 48, 49, 50]. Naturally, if we construct CIs to the confidence level α, we want that at least an α percentage of (unseen) observations to fall within these CIs. If this holds in expectation, we say that the uncertainty estimates are *calibrated*. If the empirical percentage is less than α, it indicates that our model's uncertainty estimates are over-confident and we are likely to underestimate the risk of safety violations. To empirically assess how calibrated a probabilistic regression model with hyper-parameters $\boldsymbol{\omega}$, conditioned on a training dataset $\mathcal{D}^{\text{tr}}$, is, we compute its *calibration frequency* on a test dataset $\mathcal{D}^{\text{test}}$:

$$\text{calib-freq}(\mathcal{D}^{\text{tr}}, \mathcal{D}^{\text{test}}, \boldsymbol{\omega}) = \frac{1}{|A|} \sum_{\alpha \in A} \mathbb{1} \left(\frac{1}{|\mathcal{D}^{\text{test}}|} \sum_{(\mathbf{x},y) \in \mathcal{D}^{\text{test}}} \left[\mathbb{1} \left(y \in CI_{\alpha}^{f} \left(\mathbf{x} | \mathcal{D}^{\text{tr}}, \boldsymbol{\omega} \right) \right) \right] \geq \alpha \right) \tag{4}$$

Here, $A \subset [0, 1]$ is a set of relevant confidence levels (in our case 20 values equally spaced between 0.8 and 1.0). Since the CIs of our model need to be calibrated at any iteration t during the BO and for any task we may face, we choose the best empirical estimate we can. We compute the average calibration frequency across all meta-training datasets and for any sub-sequence of points within a dataset. In particular, for any task $i = 1, ..., n$ and $t = 1, ..., T_i - 1$ we condition/train our model on the data points $\mathcal{D}_{i,\leq t} = \{(\mathbf{x}_{i,t'}, y_{i,t'})\}_{t'=1}^{t}$ and use the remaining data points $\mathcal{D}_{i,>t} = \{(\mathbf{x}_{i,t'}, y_{i,t'})\}_{t'=t+1}^{T_i}$ to compute the calibration frequency. Overall, this gives us

$$\text{calib}(\{\mathcal{D}_{i,T_i}\}_{i=1}^{n}, \boldsymbol{\omega}) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{T_i - 1} \sum_{t=1}^{T_i - 1} \text{calib-freq}(\mathcal{D}_{i,\leq t}, \mathcal{D}_{i,>t}, \boldsymbol{\omega}) \ . \tag{5}$$

While the calibration captures how *reliable* the uncertainty estimates are, it does not reflect how *useful* the confidence intervals are for narrowing down the range of possible function values. For instance, a predictor that always outputs a mean with sufficiently wide confidence intervals is calibrated, but useless for BO. Hence, we also consider the *sharpness* of the uncertainty estimates which we empirically quantify through the *average predictive standard deviation*. Similar to (5), we average over all tasks and data sub-sequences within each task:

$$\text{avg-std}(\{\mathcal{D}_{i,T_i}\}_{i=1}^{n}, \boldsymbol{\omega}) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{T_i - 1} \sum_{t=1}^{T_i - 1} \frac{1}{|\mathcal{D}_{i,>t}|} \sum_{(\mathbf{x},y) \in \mathcal{D}_{i,>t}} \sigma(\mathbf{x} | \mathcal{D}_{i,\leq t}, \boldsymbol{\omega}) \ . \tag{6}$$

4.2 Choosing good hyper-parameters via Frontier search

Based on the two empirical quantities introduced above, we can optimize the hyper-parameters $\boldsymbol{\omega}$ of our model as to maximize sharpness (i.e., minimize the avg-std) subject to calibration [47]:

$$\min_{\boldsymbol{\omega}} \ \text{avg-std}(\{\mathcal{D}_{i,T_i}\}_{i=1}^{n}, \boldsymbol{\omega}) \quad \text{s.t.} \ \ \text{calib}(\{\mathcal{D}_{i,T_i}\}_{i=1}^{n}, \boldsymbol{\omega}) \geq 1 \tag{7}$$

Since computing $\text{avg-std}(\{\mathcal{D}_{i,T_i}\}_{i=1}^{n}, \boldsymbol{\omega})$ and $\text{calib}(\{\mathcal{D}_{i,T_i}\}_{i=1}^{n}, \boldsymbol{\omega})$ requires solving the the GP inference problem many times, each query is computationally demanding. Hence, we need an optimization algorithm for (7) that requires as few queries as possible to get close to the optimal solution.

We develop an efficient *frontier search* algorithm that exploits the monotonicity properties of this optimization problem. Both avg-std and calib are monotonically increasing in the kernel variance ν and decreasing in the lengthscale l[1]. By setting $\mathbf{z} = (-l, \nu)$ and writing $s(\mathbf{z}) = \text{avg-std}(\{\mathcal{D}_{i,T_i}\}_{i=1}^{n}, l, \nu)$

[1]Note that the monotonicity of the calibation frequency in l is only an empirical heuristic that holds in almost all cases if ν is at least as big as the variance of the targets y in a dataset.

Algorithm 1 FRONTIERSEARCH

Input: Domain bounds $\mathbf{z}^l$, $\mathbf{z}^u$ s.t. $\mathbf{z}^l \leq \mathbf{z}^* \leq \mathbf{z}^u$

$\mathcal{Q}^u \leftarrow \{\mathbf{z}^u\}, \mathcal{Q}^l \leftarrow \{\mathbf{z}^l\}$
for $k = 1, ..., K$ **do**
  $(\mathbf{z}_r, \mathbf{z}'_r) \leftarrow$ LARGESTMAXMINRECT$(\mathcal{Q}^l, \mathcal{Q}^u)$ // Largest max-min rect betw. frontiers
  $\mathbf{z}_q \leftarrow$ BESTWORSTCASEQUERY$(\mathbf{z}_r, \mathbf{z}'_r, \mathcal{Q}^l, \mathcal{Q}^u)$ // Best query point to split rectangle
  if $c(\mathbf{z}_q) \geq 1$ **then**
    $\mathcal{Q}_u \leftarrow$ PRUNE$(\mathcal{Q}^u \cup \{\mathbf{z}_q\})$
  else
    $\mathcal{Q}_l \leftarrow$ PRUNE$(\mathcal{Q}^l \cup \{\mathbf{z}_q\})$

Return: $\arg\min_{\mathbf{z} \in \mathcal{Q}^u} s(\mathbf{z})$

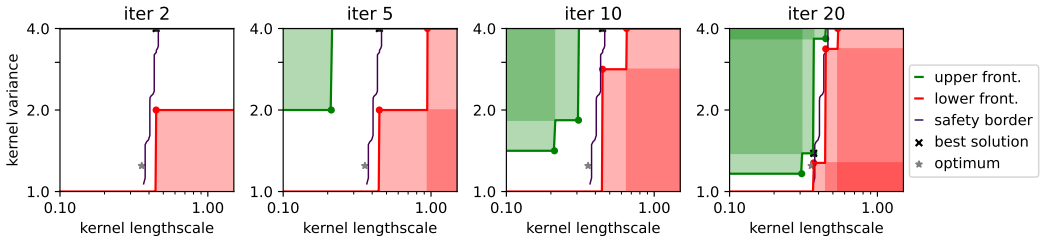

Figure 1: Frontier search (FS) on the kernel lengthscale and variance for the constraint model Argus robot. Red: areas ruled out, because unsafe. Green: Safe areas that are ruled out since dominated by better safe queries. After a few iterations, FS has already shrunk the set of possible optima (white area between fronts) to points close to the safety boarder and picked nearly optimal solution (cross).

and $c(\mathbf{z}) = \text{calib}(\{\mathcal{D}_{i,T_i}\}_{i=1}^n, l, \nu)$, we can turn (7) into a problem of the form

$$\min_{\mathbf{z}} \; s(\mathbf{z}) \quad \text{s.t.} \; c(\mathbf{z}) \geq 1 \quad \text{where } s(\mathbf{z}) : \mathbb{R}^2 \mapsto \mathbb{R} \text{ and } c(\mathbf{z}) : \mathbb{R}^2 \mapsto \mathbb{R} \text{ are monotone.} \tag{8}$$

We presume an upper and lower bound $(\mathbf{z}^u, \mathbf{z}^l)$ such that resulting search domain $\mathcal{Z} = [z_1^l, z_1^u] \times [z_2^l, z_2^u]$ contains the optimal solution $\mathbf{z}^* = \arg\min_{\mathbf{z}: c(\mathbf{z}) \geq 1} s(\mathbf{z})$. Since both $s(\mathbf{z})$ and $c(\mathbf{z})$ are monotone we know that $\mathbf{z}^*$ must lie on or directly above the constraint boundary $c(\mathbf{z}) = 1$ (c.f. Lemma 2).

In each iteration k of Algorithm 1 we query a point $\mathbf{z}_k^q \in \mathcal{Z}$ and observe the corresponding objective and constraint values $s(\mathbf{z}_k^q)$ and $c(\mathbf{z}_k^q)$. We separate the queries into two sets $\mathcal{Q}^u$ and $\mathcal{Q}^l$ based on whether they lie above or below the constraint boundary. That is, we add $\mathbf{z}_k^q$ to $\mathcal{Q}^u$ if $c(\mathbf{z}_k^q) \geq 1$ and to $\mathcal{Q}^l$ otherwise. Since the optimal solution lies on the constraint boundary and $c(\mathbf{z})$ is monotone, we can rule out entire corners of the search domain: For each $\mathbf{z}^q \in \mathcal{Q}^u$ we can rule all points $\mathbf{z}' > \mathbf{z}_k^q$ as candidates for the optimal solution and, similarly for all $\mathbf{z}^q \in \mathcal{Q}^l$, we can rule out all $\mathbf{z}' \leq \mathbf{z}_k^q$. This also allows us to *prune* the sets $\mathcal{Q}^u$ and $\mathcal{Q}^l$ by removing all the points from them that can be ruled out by a new query results. To keep track which parts of $\mathcal{Z}$ have not been ruled out yet, we construct an *upper and lower frontiers*, here expressed as functions $z_1 \mapsto z_2$ and $z_2 \mapsto z_1$,

$$F_2^u(z_1; \mathcal{Q}^u) = \min\{z_2' \mid z_1 \geq z_1', \mathbf{z}' \in \mathcal{Q}^u\}, \qquad F_1^u(z_2; \mathcal{Q}^u) = \min\{z_1' \mid z_2 \geq z_2', \mathbf{z}' \in \mathcal{Q}^u\} \tag{9}$$

$$F_2^l(z_1; \mathcal{Q}^l) = \max\{z_2' \mid z_1 \leq z_1', \mathbf{z}' \in \mathcal{Q}^l\}, \qquad F_1^l(z_2; \mathcal{Q}^l) = \max\{z_1' \mid z_2 \leq z_2', \mathbf{z}' \in \mathcal{Q}^l\} \tag{10}$$

such that the points $\Gamma = \{(z_1, z_2) \in \mathcal{Z} \mid F_2^l(z_1; \mathcal{Q}^l) \leq z_2 \leq F_2^u(z_1; \mathcal{Q}^u)\} \subseteq \mathcal{Z}$ between the frontiers are still plausible candidates for the optimal solution. For notational brevity, we define $\mathcal{F}^u = \{\mathbf{z} \in \mathcal{Z} | F_1^u(z_2; \mathcal{Q}^u) = z_1 \vee F_2^u(z_1; \mathcal{Q}^u) = z_2\}$ and $\mathcal{F}^l$ analogously as the set of points that lie on the upper and lower frontier respectively.

At any point, the best solution to (8) that we can give, is the best query we have made so far that fulfills the constraint, i.e., $\hat{\mathbf{z}} = \arg\min_{\mathbf{z} \in \mathcal{Q}^u} s(\mathbf{z})$. If we assume Lipschitz continuity for s, which holds in our case holds since $\mathcal{Z}$ is bounded and the avg-std is differentiable in l and ν, we can bound how much our best solution is away from the optimum, i.e., $s(\mathbf{z}^*)$:

Lemma 1. *Let $s(\mathbf{z})$ be L Lipschitz and $d(\Gamma, \mathcal{F}^u) := \max_{\mathbf{z}' \in \Gamma} \min_{\mathbf{z} \in \mathcal{F}^u} ||\mathbf{z} - \mathbf{z}'||$ the max-min distance between the frontiers. Then, the sub-optimality is bounded by $s(\hat{\mathbf{z}}) - s(\mathbf{z}^*) \leq L\, d(\Gamma, \mathcal{F}^u)$.*

Algorithm 2 Safe Meta-BO (SAMBO)

Input: Safe BO problem with f^{tar} and q^{tar}, set of datasets $\{\mathcal{D}^f_{i,T_i}\}_{i=1}^n,\ \{\mathcal{D}^q_{i,T_i}\}_{i=1}^n$

1: **for** $h \in \{f, q\}$ **do**
2: $\quad (l_h, \nu_h) \leftarrow$ FRONTIERSEARCH$(\{\mathcal{D}^h_{i,T_i}\}_{i=1}^n)$
3: $\quad \rho_{\boldsymbol{\theta}_h}(h) \leftarrow$ F-PACOH$(\{\mathcal{D}^h_{i,T_i}\}_{i=1}^n,\ \rho_{l_h,\nu_h}(h))$

Return: $\hat{\mathbf{z}}^* \leftarrow$ SAFEBO$(f^{tar},\ q^{tar},\ \rho_{\boldsymbol{\theta}_f}(f),\ \rho_{\boldsymbol{\theta}_q}(q))$

Here, the key insight is that we can bound the sub-optimality $s(\hat{\mathbf{z}}) - s(\mathbf{z}^*)$ with the maximum distance of any point between the frontiers from its closest point on the upper frontier instead of $\hat{\mathbf{z}}$. This is the case because $\hat{\mathbf{z}}$ dominates any point on the upper frontier (i.e., $s(\hat{\mathbf{z}}) \leq s(\mathbf{z}')\ \forall \mathbf{z} \in \mathcal{F}^u$).

Hence, we want to choose the next query so that we can shrink the max-min distance $d(\Gamma, \mathcal{F}^u)$ between the frontiers the most. For this purpose, we select the largest max-min rectangle between the frontiers (c.f. Definition 3). Then, we choose the query point, that reduce the max-min distance within the rectangle the best. For this we need to consider two scenarios that will affect the max-min distance differently: either the query point satisfies the constraint $(c(\mathbf{z}^q) \geq 1)$ or it does not $(c(\mathbf{z}^q) < 1)$. We compute the rectangle's max-min distance for both scenarios and choose the query-point that gives us the lowest max-min distance in the less-favorable (worst-case) scenario. For more details we refer to Appendix C. Finally, we provide worst-case rates for the proposed frontier search algorithm:

Theorem 1. *Under the assumptions of Lemma 1, Algorithm 1 needs no more than* $k \leq 3^{\lceil \log_2(1/\epsilon) \rceil} = \mathcal{O}\left((1/\epsilon)^{1.59}\right)$ *iterations to have a sub-optimality of less than* $s(\hat{\mathbf{z}}_k) - s(\mathbf{z}^*) \leq L||\mathbf{z}^u - \mathbf{z}^l||\,(1/\epsilon)$.

The $\mathcal{O}\left((1/\epsilon)^{1.59}\right)$ query complexity is more efficient than those of similar algorithms that do not make use of the monotonicity properties, e.g., that of grid search: $\mathcal{O}\left((1/\epsilon + 1)^2\right)$.

5 Safe BO with meta-learned GP priors

In Sec. 4 we discuss how to efficiently choose the kernel variance and lengthscale, so that we obtain calibrated and yet sharp uncertainty estimates. Now, we go one step further and use the related datasets $\mathcal{D}_{1,T_1}, ..., \mathcal{D}_{n,T_n}$ to meta-learn GP priors, aiming to give the downstream safe BO algorithm more prior knowledge about the optimization problem at hand so that it can be more query efficient.

For this, we use the F-PACOH [11] meta-learning approach, introduced in Sec 3. We choose this method since it regularizes the meta-learned model in the function space and, thus, is able to maintain reliability of the uncertainty estimates, even for out-of-distribution data. F-PACOH requires a Vanilla GP $\rho(h) = \mathcal{GP}(h(\mathbf{x})|0, k(\mathbf{x}, \mathbf{x}'))$ as a hyper-prior, towards which it regularizes the meta-learned GP $\rho_{\boldsymbol{\theta}}(h) = \mathcal{GP}(h(\mathbf{x})|m_{\boldsymbol{\theta}}(\mathbf{x}), k_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{x}')$. Hence, we face a similar kernel hyper-parameter choice problem as addressed in Section 4: If the kernel parameters of the hyper-prior are chosen such that the Vanilla GP itself yields over-confident uncertainty estimates, safe BO with the meta-learned prior will most likely turn out to be unsafe as well. Hence, we use the calibration-sharpness based frontier search to choose the kernel variance and lengthscale of the hyper-prior.

Since safe BO maintains separate models for the target $f(\mathbf{x})$ and constraint $q(\mathbf{x})$, we meta-learn a prior for each of the functions. We split the sets of data triplets $\mathcal{D}_{i,T_i} = \{(\mathbf{x}_{i,t}, \tilde{f}_{i,t}, \tilde{q}_{i,t})\}_{i=1}^{T_i}$ into separate datasets $\mathcal{D}^f_{i,T_i} = \{(\mathbf{x}_{i,t}, \tilde{f}_{i,t})\}_{t=1}^{T_i}$ and $\mathcal{D}^q_{i,T_i} = \{(\mathbf{x}_{i,t}, \tilde{q}_{i,t})\}_{t=1}^{T_i}$. First, we use frontier search to find kernel parameters (l_f, ν_f) and (l_q, ν_q) that give good sharpness subject to calibration on the respective set of datasets. Then, we meta-learn GP priors $\rho_{\boldsymbol{\theta}_f}(f)$ and $\rho_{\boldsymbol{\theta}_q}(q)$ for f and q with F-PACOH, while using the Vanilla GPs $\rho_{l_f,\nu_f}(f))$ and $\rho_{l_q,\nu_q}(q))$ with the chosen kernel hyper-parameters as hyper-prior. Finally, we run a safe BO algorithm (either SAFEOPT or GOOSE) with the meta-learned GP priors and perform safe Bayesian optimization on our target problem of interest. This procedure is summarized in Algorithm 2. We refer to our *Safe Meta-BO* (SaMBO) algorithm with GOOSE, as SAMBO-G and, when instantiated with SAFEOPT as SAMBO-S.

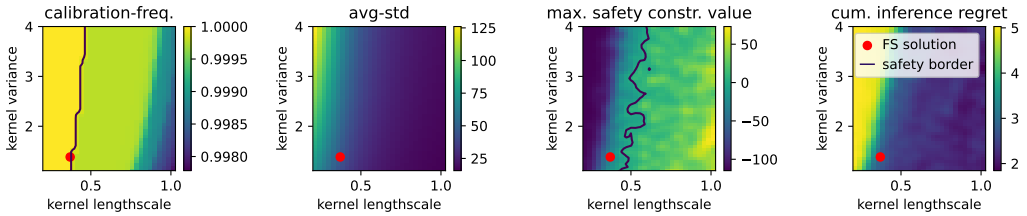

Figure 2: Left: calib and avg-std across a grid of kernel lengthscales l_q and variances ν_q for the constraint model. Right: max safety constraint value and cumulative regret for 200 iterations of safe BO. Calibration and sharpness are good proxies for safety and efficiency of the downstream safe BO.

6 Experiments

First, we investigate if the calibration & sharpness based frontier search chooses good and safe kernel parameters. Second, we evaluate the the full SAMBO algorithm in a range of safe BO environments.

6.1 Experiment Setup

In the following, we describe our experiment setup and methodology. See Appx. E for more details.

Synthetic safe BO benchmark environments. We consider two synthetic environments, based on popular benchmark functions with a two-dimensional domain. The fist is based on the *Camelback* function [51] overlaid with products of sinusoids of varying amplitude, phase and shifts. The constraint function is similar, with an additional random quadratic component to ensure connection between most of the safe regions of the domain. The second environment is based on the challenging Eggholder function [52] with many local minima. The function's shape can be varied by random sampling of three parameters. The constraint is a quadratic function, overlayed with sinusoids of varying frequencies. A task corresponds to a pair of randomly drawn target and constraint functions.

Controller Tuning for a high-precision linear robot. As robotic case study, we tune the controller of a linear axis in an Argus linear motion system by Schneeberger Linear Technology, a high-precision/speed robot for wafer inspection. To goal is to tune the three gain parameters of a cascaded PI controller to achieve minimal position error. We minimize the total variation (TV) of the position error signal, while constraining its maximum frequency in the FFT which measures (potentially damaging) instabilities/vibrations in the system. Different tasks correspond to different step sizes, ranging from $10\mu m$ to $10mm$. At different scales, the robot behaves differently in response to the controller parameters, resulting in different target and constraint functions. The experiments are conducted with a simulation of the robot, so we can explore unsafe kernel hyper-parameters – a key element for finding and visualizing the safety boundary in Figure 2.

Meta-training data. We generate the benchmark meta-training by running SAFEOPT with conservative kernel hyper-parameters choices on each task. The resulting datasets are non-i.i.d. and only consist of observations where $q(\mathbf{z}) < 0$, much more realistic than sampling data uniformly and i.i.d.. With this, we aim to mimic a practical scenario where we have performed various safe BO on related tasks in the past and now want to harness the corresponding data. For the synthetic environments, we use $n = 40$ tasks with $T_i = 100$ samples in case of the Camelback-Sin and $T_i = 200$ samples for the Random Eggholder environment. For the Argus controller tuning, we use $n = 20$ and $T_i = 400$.

Evaluation metrics. To evaluate the performance of various methods, we run safe BO with them on at least 4 unseen test tasks with each 5 seeds. To measure a methods' query efficiency, we report the (safe) inference regret $r_t = f(\hat{\mathbf{x}}_t^*) - f(\mathbf{x}^*)$ where $\hat{\mathbf{x}}_t^* = \arg\min_{\mathbf{z} \in \mathcal{S}_t} \mu_t(\mathbf{x})$ is a method's best current guess for the safe minimizer of (1). In Fig. 2, we also report the cumulative regret $R_T := \sum_{t=1}^{T} r_t$.

6.2 Choosing the kernel parameters via calibration & sharpness based frontier search

We investigate how well the calibration and sharpness based frontier search (FS) for finding kernel hyper-parameters works in practice. For that, we consider the Argus controller tuning problem. Generally, we perform FS in the log-space of l and ν since we found this to work better in practice.

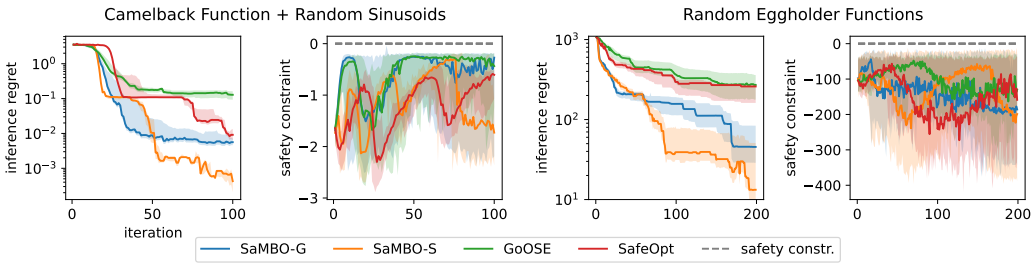

Figure 3: Safe BO regret and safety constraint on the synthetic benchmarks. SAMBO converges significantly faster towards the global optimum than the baselines, without violating the constraints.

Fig. 1 displays the frontier search at various iterations. FS quickly shrinks the set of possible safe optima (area between upper frontier (green) and lower frontier (red)) to points close to the safety border. After only 20 iterations, it already found a solution that is very close to the safe optimum. This showcases the efficiency we gain thanks to taking into account the monotonicity of the problem.

Furthermore, we investigate how well calibration and sharpness reflect what we care about: 1) No safety constraint violations and 2) query efficiency. We compute the maximum constraint value across 200 iterations with GOOSE as well as the cumulative regret across a grid of kernel length-scales and variances. Fig. 2 holds the results, together with the calib and avg-std. Overall, calibration and sharpness of the GPs' uncertainty estimates are a good proxy for the downstream safety during BO and, respectively, the regret. Importantly, all parameters that fulfill the calibration constraint $\text{calib}(\{\mathcal{D}_{i,T_i}\}_{i=1}^n, \omega) \geq 1$ lead to safe BO runs without constraint violations. Finally, the kernel parameters, chosen by FS, are both safe and lead to a small cumulative regret. Overall, this empirically supports the validity of our data-driven approach for choosing good, but safe, kernel parameters.

6.3 Safe Bayesian Optimization Benchmark & Controller Tuning

We compare the two SAMBO instantiations, SAMBO-S and SAMBO-G, with their corresponding safe BO baseline methods SAFEOPT and GOOSE. For the baselines, we use the GP kernel parameters found by FS. Fig. 3 displays the results for the synthetic benchmark functions and Fig. 4 for the Argus controller tuning. Note, that in case of the regret, the shaded areas correspond to confidence intervals while for the safety constraint values they correspond to the entire range of values (i.e., max - min). Overall, SAMBO converges to near optimal solutions much faster than the baselines without meta-learning. Across all environments, there are no safety violations which demonstrates that 1) the kernel parameters by FS are safe and 2) SAMBO maintains safety thanks to principled regularization in the function space during meta-learning. The improved query efficiency without compromising safety in the controller tuning setting, where the constraint (maximum frequency in the signal) is highly non-smooth, demonstrates the applicability of SAMBO to challenging real-world robotics problems.

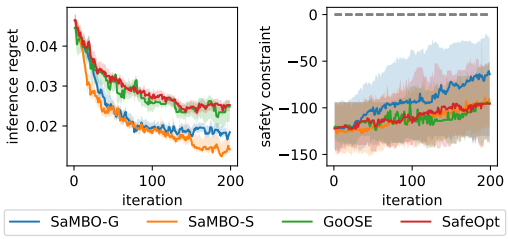

Figure 4: Safe controller tuning for the Argus robot. SAMBO safely finds good controller parameters faster than safe BO baselines.

7 Summary and Discussion of Limitations

We have introduced a data-driven framework for choosing kernel parameters or even meta-learning GP priors that are both informative and reliable. When combined with a safe BO algorithm, the resulting SAMBO speeds up the optimization of, e.g., controller parameters, without compromising safety. Except for the observation noise variance, which is often known or easy to estimate, our framework makes safe BO free of hyper-parameters and, thus, more robust and practical. However, it relies on the availability of offline data that is both sufficient in quantity and representative of the target task. As our approach relies on empirical estimates of the calibration, it may fail to ensure safety when given too little data or tasks that are systematically different to the target task.

References

[1] R. Garnett, M. A. Osborne, and S. J. Roberts. Bayesian optimization for sensor set selection. In *Proceedings of the 9th ACM/IEEE international conference on information processing in sensor networks*, pages 209–219, 2010.

[2] E. Cisbani, A. Del Dotto, C. Fanelli, M. Williams, M. Alfred, F. Barbosa, L. Barion, V. Berdnikov, W. Brooks, T. Cao, et al. Ai-optimized detector design for the future electron-ion collider: the dual-radiator rich case. *Journal of Instrumentation*, 15(05):P05009, 2020.

[3] A. Marco, P. Hennig, J. Bohg, S. Schaal, and S. Trimpe. Automatic lqr tuning based on gaussian process global optimization. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 270–277. IEEE, 2016.

[4] M. Neumann-Brosig, A. Marco, D. Schwarzmann, and S. Trimpe. Data-efficient autotuning with bayesian optimization: An industrial control study. *IEEE Transactions on Control Systems Technology*, 28(3):730–740, 2019.

[5] M. Rowold, A. Wischnewski, and B. Lohmann. Constrained bayesian optimization of a linear feed-forward controller. *IFAC-PapersOnLine*, 52(29):1–6, 2019. ISSN 2405-8963. 13th IFAC Workshop on Adaptive and Learning Control Systems ALCOS 2019.

[6] M. Khosravi, V. Behrunani, R. S. Smith, A. Rupenyan, and J. Lygeros. Cascade control: Data-driven tuning approach based on bayesian optimization. *IFAC-PapersOnLine*, 53(2):382–387, 2020.

[7] A. Rupenyan, M. Khosravi, and J. Lygeros. Performance-based trajectory optimization for path following control using bayesian optimization. In *60th IEEE conference on Decision and Control (CDC 2021)*, 2021.

[8] Y. Sui, A. Gotovos, J. W. Burdick, and A. Krause. Safe exploration for optimization with Gaussian processes. In *ICML*, 2015.

[9] F. Berkenkamp, A. P. Schoellig, and A. Krause. Safe controller optimization for quadrotors with gaussian processes. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 491–496, 2016.

[10] M. Turchetta, F. Berkenkamp, and A. Krause. Safe exploration for interactive machine learning. In *Advances in Neural Information Processing Systems*, 2019.

[11] J. Rothfuss, D. Heyn, J. Chen, and A. Krause. Meta-learning reliable priors in the function space. *Advances in Neural Information Processing Systems*, 34, 2021.

[12] J. M. Hernández-Lobato, M. A. Gelbart, M. W. Hoffman, R. Adams, and Z. Ghahramani. Predictive entropy search for Bayesian optimization with unknown constraints. In *ICML*, 2015.

[13] M. Khosravi, C. Koenig, M. Maier, R. S. Smith, J. Lygeros, and A. Rupenyan. Safety-aware cascade controller tuning using constrained bayesian optimization. *IEEE Transactions on Industrial Electronics*, 2022.

[14] M. Fiducioso, S. Curi, B. Schumacher, M. Gwerder, and A. Krause. Safe contextual bayesian optimization for sustainable room temperature pid control tuning. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 5850–5856. AAAI Press, December 2019.

[15] Y. Sui, vincent Zhuang, J. Burdick, and Y. Yue. Stagewise safe Bayesian optimization with Gaussian processes. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 4781–4789. PMLR, 10–15 Jul 2018.

[16] F. Berkenkamp, A. Krause, and A. P. Schoellig. Bayesian optimization with safety constraints: safe and automatic parameter tuning in robotics. *Machine Learning*, pages 1–35, 2021.

[17] M. Khosravi, A. Eichler, N. A. Schmid, R. S. Smith, and P. Heer. Controller tuning by bayesian optimization an application to a heat pump. *2019 18th European Control Conference (ECC)*, pages 1467–1472, 2019.

[18] C. König, M. Turchetta, J. Lygeros, A. Rupenyan, and A. Krause. Safe and efficient model-free adaptive control via bayesian optimization. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9782–9788. IEEE, 2021.

[19] S. Hochreiter, A. S. Younger, and P. R. Conwell. Learning To Learn Using Gradient Descent. In *International Conference on Artificial Neural Networks*, 2001.

[20] M. Andrychowicz, M. Denil, S. G. Colmenarejo, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. De Freitas. Learning to learn by gradient descent by gradient descent. *arXiv*, 2016.

[21] Y. Chen, M. W. Hoffman, S. G. Colmenarejo, M. Denil, T. P. Lillicrap, M. Botvinick, and N. De Freitas. Learning to Learn without Gradient Descent by Gradient Descent. In *International Conference on Machine Learning*, 2017.

[22] J. Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 2000.

[23] J. Snell, K. Swersky, and R. Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, 2017.

[24] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, 2016.

[25] J. Harrison, A. Sharma, and M. Pavone. Meta-learning priors for efficient online bayesian regression. In *International Workshop on the Algorithmic Foundations of Robotics*, pages 318–337. Springer, 2018.

[26] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, 2017.

[27] J. Rothfuss, D. Lee, I. Clavera, T. Asfour, and P. Abbeel. ProMP: Proximal Meta-Policy Search. In *International Conference on Learning Representations*, 2019.

[28] A. Nichol, J. Achiam, and J. Schulman. On First-Order Meta-Learning Algorithms. *arXiv*, 2018.

[29] A. Pentina and C. Lampert. A PAC-Bayesian bound for lifelong learning. In *International Conference on Machine Learning*, 2014.

[30] M. Garnelo, D. Rosenbaum, C. Maddison, T. Ramalho, D. Saxton, M. Shanahan, Y. W. Teh, D. Rezende, and S. M. A. Eslami. Conditional neural processes. In *International Conference on Machine Learning*, volume 80, pages 1704–1713, 2018.

[31] T. Kim, J. Yoon, O. Dia, S. Kim, Y. Bengio, and S. Ahn. Bayesian model-agnostic meta-learning. In *Advances in Neural Information Processing Systems*, 2018.

[32] J. Rothfuss, V. Fortuin, M. Josifoski, and A. Krause. PACOH: Bayes-optimal meta-learning with PAC-guarantees. In *International Conference for Machine Learning (ICML)*, 2021.

[33] L. Cella, K. Lounici, and M. Pontil. Meta representation learning with contextual linear bandits. *arXiv preprint arXiv:2205.15100*, 2022.

[34] P. Kassraie, J. Rothfuss, and A. Krause. Meta-learning hypothesis spaces for sequential decision-making. In *International Conference on Machine Learning*, 2022.

[35] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. Taking the Human Out of the Loop: A Review of Bayesian Optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016. doi:10.1109/JPROC.2015.2494218.

[36] P. I. Frazier. A Tutorial on Bayesian Optimization. *arXiv preprint arXiv:1807.02811*, 2018.

[37] C. E. Rasmussen and C. K. I. Williams. *Gaussian processes in machine learning*. MIT Press, 2006.

[38] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256, 2002.

[39] N. Srinivas, A. Krause, S. Kakade, and M. Seeger. Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. In *International Conference on Machine Learning*, pages 1015–1022, 07 2010.

[40] W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933. ISSN 00063444. URL http://www.jstor.org/stable/2332286.

[41] Z. Wang and S. Jegelka. Max-value entropy search for efficient bayesian optimization. In *International Conference on Machine Learning*, pages 3627–3635. PMLR, 2017.

[42] J. Schmidhuber. *Evolutionary principles in self-referential learning. On learning how to learn: The meta-meta-... hook*. PhD thesis, Technische Universitaet Munchen, 1987.

[43] S. Thrun and L. Pratt, editors. *Learning to Learn*. Kluwer Academic Publishers, 1998.

[44] V. Perrone, R. Jenatton, M. W. Seeger, and C. Archambeau. Scalable Hyperparameter Transfer Learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31, 2018.

[45] V. Fortuin and G. Rätsch. Deep mean functions for meta-learning in gaussian processes. *arXiv preprint arXiv:1901.08098*, 2019.

[46] A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing. Deep kernel learning. In *International Conference on Artificial Intelligence and Statistics*, pages 370–378. PMLR, 2016.

[47] T. Gneiting, F. Balabdaoui, and A. E. Raftery. Probabilistic forecasts, calibration and sharpness. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(2):243–268, 2007.

[48] T. Gneiting and A. E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

[49] V. Kuleshov, N. Fenner, and S. Ermon. Accurate Uncertainties for Deep Learning Using Calibrated Regression. In *International Conference on Machine Learning*, 2018.

[50] M.-O. Pohle. The murphy decomposition and the calibration-resolution principle: A new perspective on forecast evaluation. *arXiv preprint arXiv:2005.01835*, 2020.

[51] M. Molga and C. Smutnicki. Test functions for optimization needs. *Test functions for optimization needs*, 101:48, 2005.

[52] M. Jamil and X.-S. Yang. A literature survey of benchmark functions for global optimisation problems. *International Journal of Mathematical Modelling and Numerical Optimisation*, 4 (2):150–194, 2013.