Cautious Bayesian Optimization
for Efficient and Scalable Policy Search

Lukas P. Fröhlich[*]   LUKASFRO@ETHZ.CH
Melanie N. Zeilinger   MZEILINGER@ETHZ.CH
Institute for Dynamic Systems and Control, ETH Zurich, Switzerland

Edgar D. Klenske[*]   EDGAR.KLENSKE@GAUSS-ML.COM
Gauss Machine Learning, Stuttgart, Germany

Abstract

Sample efficiency is one of the key factors when applying policy search to real-world problems. In recent years, Bayesian Optimization (BO) has become prominent in the field of robotics due to its sample efficiency and little prior knowledge needed. However, one drawback of BO is its poor performance on high-dimensional search spaces as it focuses on global search. In the policy search setting, local optimization is typically sufficient as initial policies are often available, e.g., via meta-learning, kinesthetic demonstrations or sim-to-real approaches. In this paper, we propose to constrain the policy search space to a sublevel-set of the Bayesian surrogate model's predictive uncertainty. This simple yet effective way of constraining the policy update enables BO to scale to high-dimensional spaces ($>$100) as well as reduces the risk of damaging the system. We demonstrate the effectiveness of our approach on a wide range of problems, including a motor skills task, adapting deep RL agents to new reward signals and a sim-to-real task for an inverted pendulum system.

Keywords: Local Bayesian Optimization, Policy Search, Robot Learning

1. Introduction

Policy search has established itself as a powerful method for reinforcement learning (RL) with numerous successful applications in robotics (Deisenroth et al., 2013; Kober et al., 2013; Stulp and Sigaud, 2013). By using parameterized policies, the RL problem can be simplified, e.g., via problem-specific policy structures and/or informed parameter initialization from simulations or kinesthetic demonstrations (Ijspeert et al., 2003; Khansari-Zadeh and Billard, 2011). For real-world applications, two key criteria have to be fulfilled by a policy search algorithm: 1) *Sample efficiency* in terms of required experiments, as each interaction with a real system requires some form of human supervision and potentially inflicts wear and tear on the robot. 2) *Safety considerations* to not harm the robot's environment and potentially itself.

In recent years, Bayesian optimization (BO) has seen rising interest in the robotics community as noted by recent surveys on policy search (Sigaud and Stulp, 2019; Chatzilygeroudis et al., 2019). The most important advantage of BO is its efficiency in terms of function evaluations due to the active choice of new sample points based on a Bayesian surrogate model of the objective. Further, additional model knowledge can be included, e.g., via prior mean functions (Wilson et al., 2014), task-specific kernels (Antonova et al., 2017), multiple information sources from low-fidelity simulations and real experiments (Marco et al., 2017) as well as safety aspects (Berkenkamp et al., 2016). However, one drawback of BO is its focus on global search and as such *"[BO] does not scale well to large*

---

[*] Work done while at Bosch Center for Artificial Intelligence, Renningen, Germany.

policy parameter spaces" (Sigaud and Stulp, 2019, Sec. 3). For many applications, however, a global search of the parameters is not required as a good initial solution often exists, e.g., via simulations, demonstrations, model-based considerations or similar previously solved tasks.

In this paper, we introduce *confidence region Bayesian optimization (CRBO)*, a scalable extension to the well-known BO algorithm. Starting with an initial guess for a policy, we propose to locally constrain the parameter space based on the Bayesian surrogate model's predictive uncertainty. In particular, we only allow policies to be evaluated for which the surrogate model is confident about their outcome. The benefits of this approach are numerous: 1) We avoid over-exploration of the parameter space and thus retain the proven sample-efficiency of BO, even in high-dimensional settings. 2) By staying close to previously evaluated policies, i.e., being cautious, we reduce the risk of damaging the system or its environment. Additionally, we provide a worst-case analysis of the expected outcome for the policy to be evaluated. 3) The hyperparameter governing the cautiousness of CRBO has an intuitive explanation and is easy to tune in practice. 4) Apart from Lipschitz continuity for the worst-case analysis, we make no assumptions about either the underlying system or the structure of the objective function as is common in order to scale BO to higher-dimensional parameter spaces. The general applicability of CRBO is demonstrated on a range of problems from robotics and continuous control. We believe that CRBO satisfies the need for an efficient and scalable policy search method that can additionally leverage the benefits from numerous works in the low-dimensional BO setting for robotics. An implementation of CRBO is publicly available at www.crbo.org.

2. Related Work

Scaling BO to high-dimensional parameter spaces is an active field of research. A common assumption to alleviate the curse of dimensionality is that the objective function only varies along a low-dimensional subspace of the full parameter space (Wang et al., 2016; Nayebi et al., 2019), which in general is unknown. Therefore, Wang et al. (2016) sample a random projection matrix and Nayebi et al. (2019) use a hashing function to construct an embedding that maps to the lower-dimensional space. Both methods are limited to linear subspaces and the true effective dimensionality of the subspace is typically also unknown and thus treated as an additional hyperparameter. Kirschner et al. (2019) on the other hand, iteratively solve the optimization problem on 1-dimensional subspaces to alleviate the issue of unknown subspaces. However, the focus of this work is on global and not local search. Another approach to deal with high-dimensional search spaces is to assume an additive structure of the objective function (Kandasamy et al., 2015; Gardner et al., 2017) and the key issue lies in inferring the a-priori unknown decomposition of optimization variables. Kandasamy et al. (2015) show that knowledge of the true structure improves convergence for BO, however, in practice the best of randomly sampled decompositions is chosen. If no such structure exists in the original objective, these approaches can lead to suboptimal solutions. In the context of robotics, the objective function is typically known and the influence of certain optimization variables can be exploited. Yuan et al. (2019) use domain specific knowledge of a whole-body-control formulation in order to identify a suitable subspace during optimization. Fröhlich et al. (2019) make use of a probabilistic dynamics model to sample approximate solutions and reduce the search space via principal component analysis. However, this approach is limited to linear systems.

Constraining the parameter space during optimization is not uncommon, for example by assuming unknown constraints (Hernández-Lobato et al., 2015) or by the notion of a critical region of the search space (Berkenkamp et al., 2016; Marco et al., 2019). Unlike these methods, which assume

the constraints to be part of the optimization problem itself, in CRBO the domain is constrained by means of the surrogate model's uncertainty. However, the ideas from Hernández-Lobato et al. (2015) and Marco et al. (2019) can be included in our framework. A discussion about the difference between CRBO and the SafeOpt framework (Berkenkamp et al., 2016) is provided in Sec. 4. Adapting the search space during the optimization has also been proposed before, but both approaches are limited to rectangular domains and do not consider cautiousness when increasing the search space (Fröhlich et al., 2019; Ha et al., 2019).

While standard BO is typically used for global optimization, there have been efforts to combine it with the advantages of local optimization, e.g., faster convergence in the vicinity of optima. Wabersich and Toussaint (2016) propose a non-stationary kernel function that exploits the locally convex region around local and global minima. McLeod et al. (2018) introduced the idea to use a local optimizer once the objective's surrogate model is certain about the location of the global optimum, however, this approach is limited to the noiseless setting. Recently, Eriksson et al. (2019) proposed to use multiple, rectangular trust-regions that only locally approximate the objective function. These trust-regions are then updated based on heuristics instead of considering cautiousness and next evaluation points are suggested by posterior sampling of the surrogate model. Similarly, Akrour et al. (2017) use a sampling based approach where a Gaussian search distribution is updated in an (approximately) optimal manner adhering to information-theoretic constraints. In the experimental section, we compare CRBO to both trust region Bayesian optimization (TuRBO) (Eriksson et al., 2019) and Local BO (Akrour et al., 2017).

3. Preliminaries

In this section, we will formally introduce the policy search problem and briefly review BO as a solution strategy for the resulting optimization problem. The goal of reinforcement learning is to find a policy $\pi : \mathcal{S} \to \mathcal{A}$ mapping states $s \in \mathcal{S}$ to actions $a \in \mathcal{A}$ that maximizes a predefined reward signal (see, e.g., Sutton and Barto (1998)). In the (episodic) policy search setting, the policy $\pi(s; \theta)$ is parameterized by a vector $\theta \in \Theta \subseteq \mathbb{R}^d$, e.g., the weights of a neural network (NN). The optimal policy is then found via maximization of an episode's expected return with respect to θ,

$$\theta^* = \underset{\theta \in \Theta}{\operatorname{argmax}} J(\theta), \quad \text{with } J(\theta) = \mathbb{E}\left[R(\tau)|\theta\right] = \int R(\tau)p_\theta(\tau)d\tau, \quad (1)$$

where $\tau = \{s_t, a_t\}_{t=1:T}$ denotes a state-action trajectory and the return is given by the sum of immediate rewards $R(\tau) = \sum_t r(s_t, a_t)$ with $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. The stochasticity of the return is the result from varying initial state conditions as well as the stochastic nature of the environment itself. In this paper, we consider deterministic policies and address (1) by means of Bayesian optimization (BO) based on noisy observations of the episodic returns.

BO is a method for global optimization of stochastic black-box functions designed to be highly sample efficient (Frazier, 2018). The efficiency of BO stems from two main ingredients: 1) a Bayesian surrogate model that approximates the objective function and 2) choosing the next evaluation point by maximizing a so-called acquisition function $\alpha(\theta)$. A common choice for the Bayesian surrogate model are Gaussian processes (GPs), which define a prior distribution over functions, such that any finite number of function values are normally distributed with mean $\mu_0(\theta)$ and covariance specified by the kernel function $k(\theta, \theta')$ for any $\theta, \theta' \in \Theta$ (w.l.o.g. we assume $\mu_0(\theta) \equiv 0$) (Rasmussen and Williams, 2006). In the regression setting, the goal is to predict an unknown latent function $J(\theta) : \Theta \to \mathbb{R}$ based on noisy observations $\mathcal{D}_n = \{(\theta_i, y_i = J(\theta_i) + \epsilon)\}_{i=1:n}$ with $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$.

Given a Gaussian likelihood model, conditioning the GP's prior distribution on data $\mathcal{D}_n$ leads to the closed-form posterior predictive distribution,

$$\mu_n(\boldsymbol{\theta}) = \boldsymbol{k}(\boldsymbol{\theta})^\top \boldsymbol{K}^{-1}\boldsymbol{y}, \quad \sigma_n^2(\boldsymbol{\theta}) = k(\boldsymbol{\theta}, \boldsymbol{\theta}) - \boldsymbol{k}(\boldsymbol{\theta})^\top \boldsymbol{K}^{-1}\boldsymbol{k}(\boldsymbol{\theta}), \tag{2}$$

at any $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ with $[\boldsymbol{k}(\boldsymbol{\theta})]_i = k(\boldsymbol{\theta}, \boldsymbol{\theta}_i)$, $[\boldsymbol{K}]_{ij} = k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) + \delta_{ij}\sigma_\epsilon^2$, $[\boldsymbol{y}]_i = y_i$ and δ_{ij} denotes the Kronecker delta. Typical choices for the kernel function in the context of BO are the squared exponential (SE) and Matérn kernels (Rasmussen and Williams, 2006, Chapter 4).

Given the probabilistic model of the objective function, the next evaluation point is chosen by maximizing the acquisition function $\alpha(\boldsymbol{\theta})$ quantifying the exploration-exploitation trade-off between regions with large predicted values (exploitation) and regions of high uncertainty (exploration). Numerous acquisition functions have been proposed in the literature, e.g., upper confidence bound (UCB) (Cox and John, 1992), expected improvement (EI) (Močkus, 1975) as well as information-theoretic variants (Hennig and Schuler, 2012; Hernández-Lobato et al., 2014; Wang and Jegelka, 2017). One of the major challenges associated with BO is the optimization of the acquisition function, which can be non-convex. However, unlike the original objective function, the acquisition function is cheap to evaluate and often has analytical gradients such that local solvers can be employed.

4. Confidence Region Bayesian Optimization

In standard Bayesian optimization (BO), the user defines fixed box-constraints with lower and upper bounds $\boldsymbol{\theta}_l, \boldsymbol{\theta}_u$ for the optimization (policy) parameters $\boldsymbol{\theta}$. If these bounds are chosen too tight, one might miss out on promising regions of the parameter space, but loosening the bounds leads to an increase of the search volume, which scales exponentially with the dimensionality of the parameters. This issue is especially critical due to BO's tendency of *over-exploring* the domain boundaries at the beginning of the optimization process (Siivola et al., 2018). In the context of policy search, this leads to many rollouts with poor performance, potentially damaging the system without gaining much information about the optimal parameters.

In practice, we often do not need to start an optimization process from scratch, but a suitable initial guess exists in many scenarios. As a matter of fact, many approaches in robotics can provide such an initial solution: optimal control using model-based considerations, learning from demonstrations, meta-learning by previously solved tasks, transferring knowledge from simulation to hardware, etc. Thus, the considered goal is to *locally improve on the initial policy* instead of globally searching the parameter space as done in standard BO. Akin to trust-region methods (Nocedal and Wright, 2006, Chapter 4.4), we constrain our next optimization iterate based on the agreement between the objective function and its surrogate model. In other words, we only want to search for new parameters in regions where we are confident in our model, i.e., the GP approximating the objective. Being a Bayesian model, GPs naturally provide this notion of confidence in terms of the predictive uncertainty given by Eq. (2). More formally, we propose to constrain the parameter space during optimization of the acquisition function such that

$$\boldsymbol{\theta}_{n+1} = \underset{\boldsymbol{\theta} \in \mathcal{C}_n}{\operatorname{argmax}}\, \alpha(\boldsymbol{\theta}), \quad \text{and } \mathcal{C}_n = \left\{ \boldsymbol{\theta} \in \mathbb{R}^d \,|\, \sigma_n(\boldsymbol{\theta}) \leq \gamma\sigma_f \right\} \cap \boldsymbol{\Theta}, \tag{3}$$

where $\mathcal{C}_n$ is referred to as the *confidence region*, σ_f denotes the signal standard deviation of the GP's kernel and $\gamma \in (0, 1]$ is a tunable parameter, governing the effective size of $\mathcal{C}_n$. For small values of γ, the confidence region is confined to be close to the data, whereas in the limit $\gamma = 1$, we obtain the standard BO formulation. Based on this constraint for the parameter space, we name our method
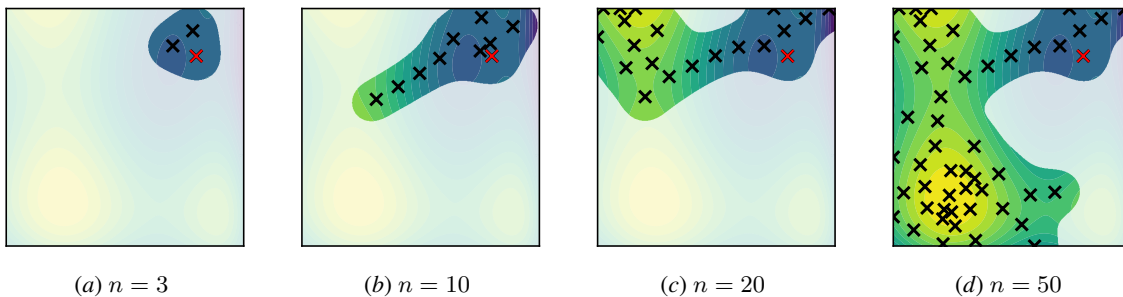
| $(a)\ n = 3$ | $(b)\ n = 10$ | $(c)\ n = 20$ | $(d)\ n = 50$ |

Figure 1: Two-dimensional example for the growing confidence region $\mathcal{C}_n$ (see Eq. (3)) during the optimization process. The evaluated policies are denoted by the crosses, with red being the initial policy. In the beginning, CRBO follows the gradient direction for efficient exploration and then continues to explore and exploit local optima. Note that the algorithm does not get stuck in the local optimum in the upper left corner but continues to explore towards the global optimum instead.

confidence region Bayesian optimization (CRBO). Throughout the paper, we keep γ fixed during optimization, however, one could increase the parameter over time as to retain a theoretical no-regret bound akin to Srinivas et al. (2010).

Note that the GP's predictive uncertainty depends on the observed data, such that the confidence region $\mathcal{C}_n$ can grow adaptively after each rollout of the system. An exemplary run of CRBO is depicted in Fig. 1. After a local exploration phase around the initial policy (Fig. 1(a)), CRBO follows the gradient (Fig. 1(b)) where the relative stepsize is governed by the confidence parameter γ. Especially this greedy behavior in the beginning of the optimization is one of the key aspects that makes CRBO highly efficient. Once the algorithm finds a local optimum, it explores the surrounding region (Fig. 1(c)). Note that CRBO does not get stuck in the local optimum, but instead continues exploring and finds the global optimum which it then exploits (Fig. 1(d)).

One of the main benefits of CRBO lies in its simplicity. By only constraining the search space via Eq. (3), this approach is applicable to all acquisition functions, which enables CRBO to be used in virtually any BO framework. Further, the idea behind CRBO can be used to improve other sophisticated variants such as multi-task BO (Swersky et al., 2013), multi-fidelity BO (Marco et al., 2017) or manifold BO (Jaquier et al., 2019).

4.1. Quantifying Cautiousness

Constraining the search space by Eq. (3) implies that we stay close to previously evaluated parameters and, as a consequence, the parameter space is explored in a *cautious* manner. Consequently, we can estimate the worst-case outcome for the next rollout assuming the objective function is Lipschitz-continuous in the parameters, i.e., $\|J(\boldsymbol{\theta}_i) - J(\boldsymbol{\theta}_j)\| \le L_J \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_j\|$ for all $\boldsymbol{\theta}_i, \boldsymbol{\theta}_j \in \mathcal{C}_n$. For this analysis, we first approximate the maximum distance of any evaluated policy to the boundary of the confidence region $\partial \mathcal{C}_n$ and second, we estimate the objective's Lipschitz constant L_J.

Note that if only given one data point $\boldsymbol{\theta}_0$, the confidence region is defined by a ball centered around $\boldsymbol{\theta}_0$ with radius $r_0 = \|\boldsymbol{\theta} - \boldsymbol{\theta}_0\|$ determined by the condition $\sigma_0(\boldsymbol{\theta}; r_0) = \gamma \sigma_f$. For the SE kernel and $\sigma_\epsilon^2 \ll \sigma_f^2$, we can compute r_0 analytically; for other kernels, r_0 can be found numerically. We then approximate the confidence region by a union of balls centered at the evaluated parameters,

$$\mathcal{C}_n \approx \tilde{\mathcal{C}}_n = \cup_{i=1}^n \{\boldsymbol{\theta} \in \boldsymbol{\Theta} | \|\boldsymbol{\theta} - \boldsymbol{\theta}_i\| \le r_0\}, \quad \text{with } r_0^2 = -2\ell^2 \log \sqrt{1 - \gamma^2}. \tag{4}$$

Algorithm 1 Hit-and-Run sampler (Bélisle et al., 1993) used for optimization of the acquisition function in the confidence region $\mathcal{C}_n$.

Choose starting point $\tilde{\boldsymbol{\theta}}_0 \in \mathcal{C}_n$
for $k = 1{:}K$ **do**
    Generate uniformly random direction: $\boldsymbol{d} \in \mathbb{R}^d$
    Find line set: $L_k = \mathcal{C}_n \cap \{\boldsymbol{\theta} | \boldsymbol{\theta} = \boldsymbol{\theta}_{k-1} + \lambda \boldsymbol{d}, \lambda \in \mathbb{R}\}$
    Sample next point $\tilde{\boldsymbol{\theta}}_k \sim \mathcal{U}(\boldsymbol{\theta} \in L_k)$
end

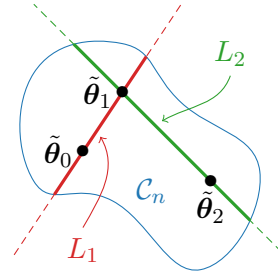**return** Set of sample points $\{\tilde{\boldsymbol{\theta}}_k\}_{k=1:K}$

Figure 2: Visualization of Alg. 1.

Thus, every possible next parameter $\boldsymbol{\theta}_{n+1}$ is within a distance of r_0 to at least one previously evaluated parameter in the dataset $\mathcal{D}_n$. Given this maximum step size and the bound on the change in output space via the Lipschitz constant, we can estimate the worst possible outcome for the next policy. More specifically, the expected outcome of the policy parameterized by $\boldsymbol{\theta}_{n+1}$ is lower-bounded by

$$J(\boldsymbol{\theta}_{n+1}) \gtrsim J(\boldsymbol{\theta}_k) - r_0(\gamma) \cdot L_J, \quad \text{with } \boldsymbol{\theta}_k = \operatorname*{argmin}_{\boldsymbol{\theta}_i \in \mathcal{D}_n} \|\boldsymbol{\theta}_{n+1} - \boldsymbol{\theta}_i\|, \tag{5}$$

where we made the dependence of r_0 on the confidence region parameter γ explicit. In general, however, the true Lipschitz constant of the objective is not known a-priori and needs to be estimated from data, e.g., as the maximum of the norm of the GP mean gradient $L_J \approx \max_{\boldsymbol{\theta}} \|\nabla \mu_n(\boldsymbol{\theta})\|$. While underestimating the Lipschitz constant can lead to a violation of (5), this analysis offers valuable insight into the effect of the confidence parameter γ on the cautiousness of CRBO. In practice, one could also adaptively tune the confidence parameter γ such that a certain minimum performance threshold $J_{\min}$ is not violated, i.e., $J(\boldsymbol{\theta}_{n+1}) \geq J_{\min}$.

Note that the introduced concept of cautiousness is related albeit different from safety as presented by Berkenkamp et al. (2016). In particular, we do not assume an unknown safe set, e.g., defined by a performance threshold. Further, by relaxing the strict safety assumption, the optimization of the acquisition function becomes scalable to high-dimensional spaces as we do not rely on (adaptive) grid search (Berkenkamp et al., 2016; Duivenvoorden et al., 2017). Lastly, CRBO offers a parameter to directly tune the cautiousness whereas in the SafeOpt framework safety is only guaranteed under strict assumptions on the class of objective functions and no tunable parameter is provided.

4.2. Optimizing the Acquisition Function

Efficiently optimizing the acquisition function is crucial for the success of BO. Even in rectangular optimization domains for standard BO, this is already a difficult problem due to the function's non-convex nature. In practice, many BO toolboxes use numerical optimization methods, e.g., L-BFGS (Liu and Nocedal, 1989), with different starting points to find the acquisition function's optimum. For CRBO the problem is even more difficult, as the confidence region can become non-convex (Fig. 1(d)). We propose to evaluate the acquisition function at many locations which are uniformly distributed across $\mathcal{C}_n$ and then initialize a local optimizer at the location with the highest function value. To deal with the nonlinearity of the constraints, we employ sequential quadratic programming. While this procedure does not guarantee convergence to the global optimum, it works well in practice and is computationally efficient.

In order to initialize the local optimizer, we need to sample the confidence region uniformly, which for a non-convex bounded region is a hard problem in itself. Simple methods, such as
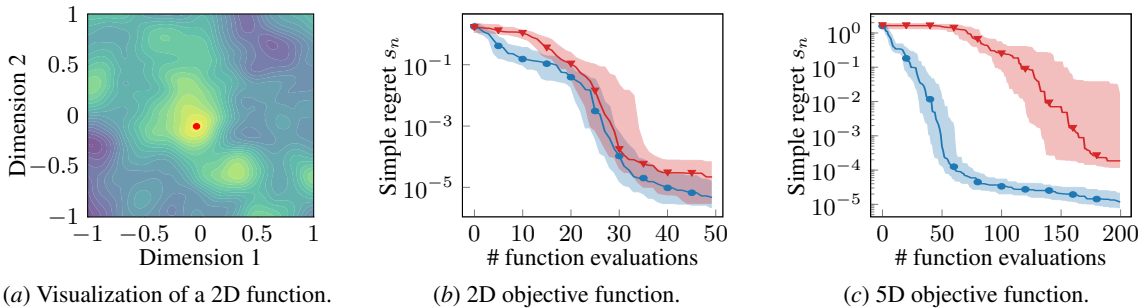
(*a*) Visualization of a 2D function.   (*b*) 2D objective function.   (*c*) 5D objective function.

Figure 3: Comparison of CRBO (—•—) and standard BO (—▼—) on 2- and 5-dimensional synthetic benchmark functions, (b) and (c), respectively. For each experiment, the initial data point is chosen randomly with a distance of 0.3 to the global optimum.

rejection sampling, do not scale well to the high-dimensional setting as the rejection rate increases exponentially with $\dim(\boldsymbol{\theta})$. We therefore employ the so-called Hit-and-Run sampler (Smith, 1984), which works by iteratively choosing random one-dimensional subspaces on which rejection sampling is highly efficient. Pseudo-code of the sampler is presented in Alg. 1 as well as an exemplary visualization of three consecutive samples in Fig. 2. For more efficient coverage of the sample space, we use the fact that the observed data is already well distributed across the confidence region and thus start multiple parallel sampling chains from each of the data points.

5. Experimental Results

We begin the experimental section with a comparison of standard BO and CRBO on synthetic test functions (Fig. 3) to emphasize the difference between a global and local exploration approach when a sufficiently good initial guess exists. Further, we investigate a variety of control tasks. In particular, we consider a complex motor-skill task where the initial policy stems from a (simulated) kinesthetic demonstration (Fig. 4), different environments from the OpenAI gym for which pre-trained RL agents are adapted to new reward signals (Fig. 5), and a sim-to-real task, where an initial policy is trained in simulation and fine-tuned on hardware (Fig. 6). Throughout all experiments, we use UCB as acquisition function for CRBO, the Matérn kernel as covariance function and we infer the MAP estimate of the GP hyperparameters after each iteration. If not stated otherwise, each experiment was repeated 20 times. The plots show the median as solid line and the interquartile range as shaded area.

5.1. Low-dimensional Synthetic Functions

The first experiment investigates the different exploration strategies of standard BO and CRBO. As objective functions, we draw 50 random samples from a GP with zero mean prior and a Matérn kernel ($\sigma_f^2 = 1.0, \ell = 0.3$) on the domain $[-1, 1]^d$. In order to avoid the optimum to be close to domain boundaries, we condition the GP on one training point in the center with $y = 3$. A visualization of a resulting sample is shown in Fig. 3(a). For each objective, the initial data point was sampled randomly with a distance of 0.3 to the global optimum and the observation noise was fixed to $\sigma_\epsilon = 10^{-3}$. The results in terms of simple regret, $s_n = \min_t |y_t - y^*|$, are depicted in Fig. 3. In the 2-dimensional setting, locally constraining the search space shows no significant advantage over the global exploration strategy. As the search space is relatively small, standard BO quickly explores the domain and converges to the global optimum. Even for a moderately sized problem with 5 optimization parameters, however, the global exploration approach leads to slow convergence in the beginning and only after around 100 function evaluations, standard BO begins exploiting the
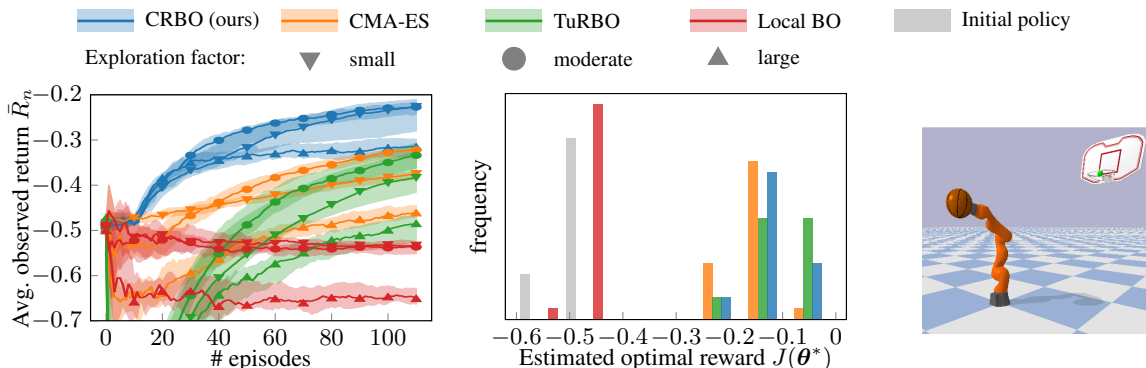
Figure 4: Results for Basketball task. (left) We investigate the influence of each method's exploration parameter. (center) The final performance is shown for the variants with moderate exploration.

region around the global optimum. The local exploration approach is in stark contrast to this behavior. Specifically, CRBO follows the gradient direction from the beginning and thus requires significantly less function evaluations compared to standard BO.

5.2. Policy Search Tasks

For the high-dimensional policy search tasks, we compare CRBO to the following methods: 1) CMA-ES (Hansen and Ostermeier, 2001) as the gold standard for stochastic optimization, using the official Python implementation, 2) TuRBO (Eriksson et al., 2019) as current state of the art for high-dimensional BO, using the provided Python implementation, and lastly 3) Local BO (Akrour et al., 2017) as another BO approach based on locality, using a Matlab implementation.[1] For fair comparison, each algorithm starts with the same initial policy $\boldsymbol{\theta}_0$.

We measure an algorithm's performance with respect to two metrics: first, as measure for an algorithm's final performance, the expected return after optimization, $J(\boldsymbol{\theta}^*) = \mathbb{E}[R(\boldsymbol{\tau})|\boldsymbol{\theta}^*]$, which we estimate using 10 rollouts. Second, we consider the average of the observed returns during the optimization, $\bar{R}_n = \frac{1}{n}\sum_{i=1:n} R(\boldsymbol{\tau}_i)$, akin to the cumulative regret in the BO literature, with $\boldsymbol{\tau}_i$ being the trajectory of the i-th episode. While a good final performance $J(\boldsymbol{\theta}^*)$ is desired, $\bar{R}_n$ essentially quantifies the cautiousness during the search for a good policy.

To find the optimal value for each method's exploration parameter, we repeat the simulated experiments for different values of the exploration parameters in pre-defined ranges. Clearly, this is not a viable approach for the hardware experiment. For CRBO, we therefore use the exploration parameter that performed best in the simulated experiments. For a fair comparison, we tried three different values for the exploration parameter of CMA-ES in a truncated experiment, i.e., less episodes per experiment and fewer repetitions. Only for the best performing parameter value we then conducted the full experiment. To keep the figures interpretable, we only present a subset of the results in the main paper.

Learning from Demonstrations – Basketball Task In this experiment, the goal is for a 7-DoF Kuka robotic arm to throw a basketball into a hoop as depicted in Fig. 4. Specifically, the reward is chosen as the minimum distance between the ball's trajectory and the center of the hoop. With this experiment, we want to investigate the influence of each method's exploration parameter. For CMA-ES and Local BO, this corresponds to the initial variance of the search distribution, for TuRBO this corresponds to the size of the initial rectangular trust region and for CRBO we vary the confidence

---

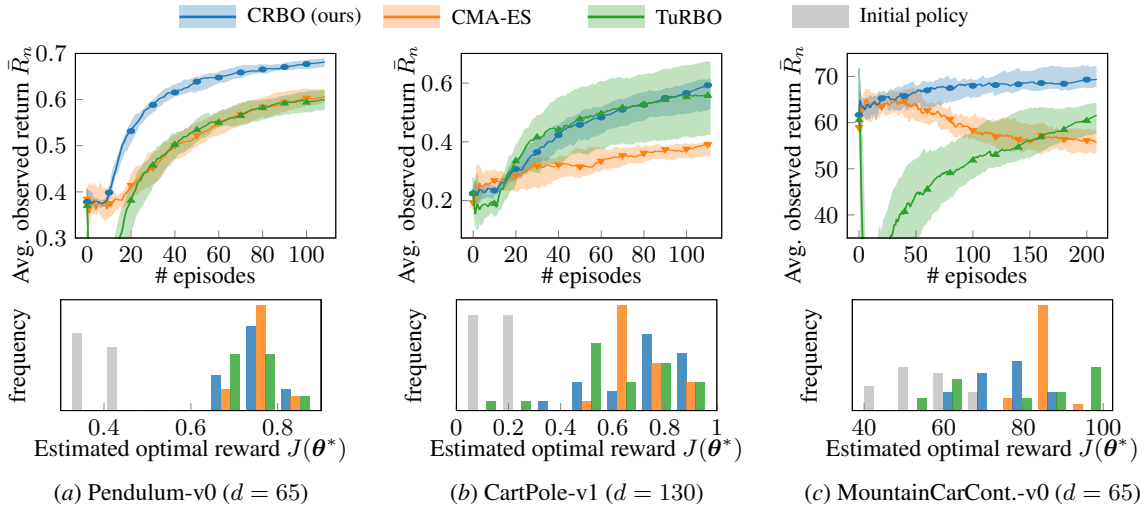1. Source code obtained from the authors via personal communication.

Figure 5: Results for OpenAI gym environments with adapted rewards signals.

parameter γ as to govern the confidence region's effective size. For each method, we choose three different values ranging from small to large (exact values are given in the supplementary material). The policy for the robot is parameterized by a dynamic movement primitive (DMP) (Ijspeert et al., 2003) for each degree of freedom with 15 basis functions each. Additionally, the goal attractor for each joint angle was optimized, resulting in a total of 112 free parameters. The initial policy was learned by fitting the DMPs to a heuristically designed trajectory going through three via-points similar to a kinesthetic demonstration from a human. For the simulation of the robot we use the PyRoboLearn framework (Delhaisse et al., 2019). The results are shown in Fig. 4. Common for all methods is that a large exploration factor leads to the smallest average observed return. However, small exploration does not necessarily result in large average return as the algorithms can be stuck in local optima. CRBO is the only algorithm that shows monotonic improvement in terms of the average observed reward while all other methods are less cautious in the beginning of the optimization. This is especially true for TuRBO due to its focus on global search. Local BO demonstrates no significant improvement over the first 100 episodes which is in accordance with the originally presented results on a similarly complex task. We therefore refrain from further comparison in other experiments. In terms of final performance, CRBO is slightly subpar compared to TuRBO, which can be seen as a trade-off for cautiousness. For CRBO, we chose $\gamma \in \{0.4, 0.6, 0.9\}$.

Adapting Deep RL Agents to New Rewards Signals Recently, de Bruin et al. (2020) argued that deep RL algorithms are well suited to efficiently learn the feature extractor part of a NN policy, however, the so-called policy-head can be fine-tuned via gradient-free methods to further increase an agent's performance on the original reward signal. In the next experiment, we investigate if an agent can quickly adapt to a different reward signal if the feature extractor part of the network is kept fixed. Specifically, we use the agents trained with proximal policy optimization (PPO) (Schulman et al., 2017) provided by Raffin (2018). The policies are 2-layer NNs with $n_h = 64$ hidden units each, but we only optimize the weights of the linear mapping from the last hidden layer to the actions resulting in $(n_h + 1) \times |\mathcal{A}|$ optimization parameters. We consider the following three environments from the OpenAI gym: Pendulum-v0, Cartpole-v1 and MountainCarContinuous-v0. For each environment, we adjust the reward signal to fine-tune and improve the agent's behavior. For the pendulum, we change the reward signal to $r_t = (1 - |\alpha_t|/2°)/T$ to promote a zero steady-state error of the pendulum's
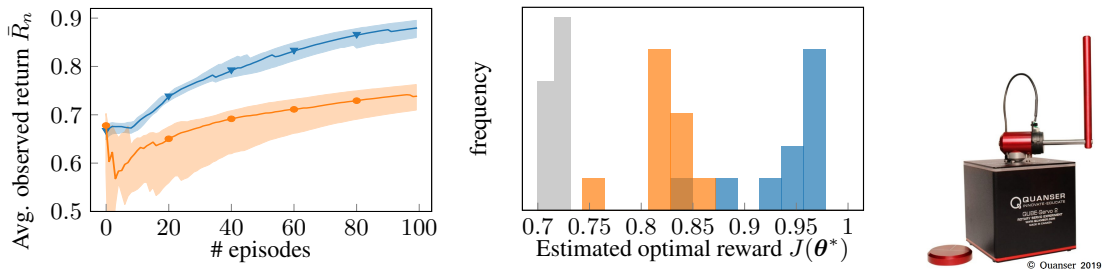
Figure 6: Results for the the Furuta pendulum; CRBO (━▼━), CMA-ES (━●━), initial policy (▬).

angle α at the end of an episode of length T. The reward signal for the cartpole is changed to $r_t = (1 - |x_t|)/T$ with x_t being the cart's position such that the system stays in the center. Lastly, we penalize the action of the mountain car more strictly compared to the original reward to optimize for more energy efficient trajectories such that $r_t = -0.5a_t^2$ and additional $+100$ reward if the car reaches the goal state. The results are shown in Fig. 5. Across all environments, CRBO exhibits the best average observed return. For the three environments, we chose $\gamma = 0.3, 0.6$ and 0.5, respectively. The final performance of all methods is similar for the Pendulum and CartPole environments, but for the MountainCar environment, the objective is extremely noisy which slows down CRBO. Note that CMA-ES inflates the covariance of its search distribution for increased exploration, leading to a decline in the average performance. TuRBO evaluates many policies with poor performance in the beginning. This is especially prominent for the Pendulum and MountainCar environments, for which the return is sensitive to changes in the parameter space.

Sim-to-Real – Stabilizing a Furuta Pendulum   The next task considers the transfer of a policy trained in simulation to the hardware. The Furuta pendulum consists of an unactuated pendulum attached to the end of a rotary arm that is actuated via a motor (see Fig. 6); the dynamics of this system are similar to that of the cart-pole. Similar to the previous experiment, we parameterize the policy by a 2-layer NN and optimize the weights of the last layer. The initial policy is obtained by imitation learning in simulation with a stabilizing controller as expert policy using 30 seconds of demonstration time and state/action pairs sampled at 250 Hz. The goal is to track a sinusoidal reference trajectory of the horizontal arm's angle while keeping the pendulum in the upright position. The reward is given by $r_t = (1 - |\alpha_t - \alpha_{t,\text{ref}}|/90°)/T$ and an episode is terminated if the pendulum tips over or after 20 seconds are reached. We repeat the experiment 10 times for each method. Due to the unpredictable behavior of TuRBO in the initial optimization phase, we do not include this method in the comparison. On this task, CRBO with $\gamma = 0.5$ consistently outperforms CMA-ES both in terms of average and final performance by a large margin.

6. Conclusion

In this paper we have introduced confidence region Bayesian optimization (CRBO), a scalable extension to the well-known BO algorithm. As such, CRBO can be incorporated into a variety of existing BO methods. The main goal of CRBO is to retain the sample efficiency of standard BO by locally constraining the parameters which allows for optimization of higher-dimensional parameter spaces. Specifically, we achieve this goal by constraining the search space to a sublevel-set of the surrogate model's predictive uncertainty. The resulting algorithm can easily be tuned by an interpretable parameter that determines the cautiousness of the optimization. We have demonstrated superior performance of CRBO compared to state-of-the-art methods across many different tasks.

Acknowledgments

References

Riad Akrour, Dmitry Sorokin, Jan Peters, and Gerhard Neumann. Local bayesian optimization of motor skills. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 41–50, 2017.

Rika Antonova, Akshara Rai, and Christopher G Atkeson. Deep kernels for optimizing locomotion controllers. *arXiv preprint arXiv:1707.09062*, 2017.

Claude JP Bélisle, H Edwin Romeijn, and Robert L Smith. Hit-and-run algorithms for generating multivariate distributions. *Mathematics of Operations Research*, 18(2):255–266, 1993.

Felix Berkenkamp, Angela P Schoellig, and Andreas Krause. Safe controller optimization for quadrotors with Gaussian processes. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 491–496. IEEE, 2016.

Konstantinos Chatzilygeroudis, Vassilis Vassiliades, Freek Stulp, Sylvain Calinon, and Jean-Baptiste Mouret. A survey on policy search algorithms for learning robot controllers in a handful of trials. *IEEE Transactions on Robotics*, 2019.

Dennis D. Cox and Susan John. A statistical method for global optimization. In *IEEE Transactions on Systems, Man, and Cybernetics*, pages 1242–1246, 1992.

Tim de Bruin, Jens Kober, Karl Tuyls, and Robert Babuška. Fine-tuning deep RL with gradient-free optimization. In *Proceedings of the IFAC World Congress*, 2020.

Marc Peter Deisenroth, Gerhard Neumann, and Jan Peters. *A survey on policy search for robotics*. now publishers, 2013.

Brian Delhaisse, Leonel Rozo, and Darwin Caldwell. Pyrobolearn: A Python framework for robot learning practitioners. In *Proceedings of the Conference on Robot Learning (CoRL)*, 2019.

Rikky RPR Duivenvoorden, Felix Berkenkamp, Nicolas Carion, Andreas Krause, and Angela P Schoellig. Constrained bayesian optimization with particle swarms for safe adaptive controller tuning. *Proceedings of the IFAC World Congress*, pages 11800–11807, 2017.

David Eriksson, Michael Pearce, Jacob Gardner, Ryan D Turner, and Matthias Poloczek. Scalable global optimization via local Bayesian optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5496–5507, 2019.

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

Lukas P. Fröhlich, Edgar D. Klenske, Christian G. Daniel, and Melanie N. Zeilinger. High-dimensional Bayesian optimization for policy search via automatic domain selection. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

Jacob Gardner, Chuan Guo, Kilian Weinberger, Roman Garnett, and Roger Grosse. Discovering and exploiting additive structure for bayesian optimization. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1311–1319, 2017.

Huong Ha, Santu Rana, Sunil Gupta, Thanh Nguyen, Svetha Venkatesh, et al. Bayesian optimization with unknown search space. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 11795–11804, 2019.

Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation*, 9(2):159–195, 2001.

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

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

José Miguel Hernández-Lobato, Michael A. Gelbart, Matthew W. Hoffman, Ryan P. Adams, and Zoubin Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2015.

Auke J Ijspeert, Jun Nakanishi, and Stefan Schaal. Learning attractor landscapes for learning motor primitives. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1547–1554, 2003.

N Jaquier, L Rozo, S Calinon, and M Buerger. Bayesian optimization meets Riemannian manifolds in robot learning. In *Proceedings of the Conference on Robot Learning (CoRL)*, 2019.

Kirthevasan Kandasamy, Jeff Schneider, and Barnabás Póczos. High dimensional Bayesian optimisation and bandits via additive models. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 295–304, 2015.

Ss Mohammad Khansari-Zadeh and Aude Billard. Learning stable nonlinear dynamical systems with gaussian mixture models. *IEEE Transactions on Robotics*, 27(5):943–957, 2011.

Johannes Kirschner, Mojmir Mutny, Nicole Hiller, Rasmus Ischebeck, and Andreas Krause. Adaptive and safe Bayesian optimization in high dimensions via one-dimensional subspaces. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

Dong C Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

Alonso Marco, Felix Berkenkamp, Philipp Hennig, Angela Schoellig, Andreas Krause, Stefan Schaal, and Sebastian Trimpe. Virtual vs. real: Trading off simulations and physical experiments in reinforcement learning with Bayesian optimization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1557–1563, 2017.

Alonso Marco, Dominik Baumann, Philipp Hennig, and Sebastian Trimpe. Classified regression for bayesian optimization: Robot learning with unknown penalties. *arXiv preprint:1907.10383*, 2019.

Mark McLeod, Stephen Roberts, and Michael A. Osborne. Optimization, fast and slow: optimally switching between local and Bayesian optimization. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

Jonas Močkus. On Bayesian methods for seeking the extremum. In *Optimization Techniques IFIP Technical Conference*, pages 400–404, 1975.

Amin Nayebi, Alexander Munteanu, and Matthias Poloczek. A framework for bayesian optimization in embedded subspaces. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 4752–4761, 2019.

Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

Antonin Raffin. RL baselines zoo. https://github.com/araffin/rl-baselines-zoo, 2018.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Olivier Sigaud and Freek Stulp. Policy search in continuous action domains: an overview. *Neural Networks*, 2019.

Eero Siivola, Aki Vehtari, Jarno Vanhatalo, Javier González, and Michael Riis Andersen. Correcting boundary over-exploration deficiencies in bayesian optimization with virtual derivative sign observations. In *Proceedings of the IEEE International Workshop on Mach Learning for Signal Processing (MLSP)*, pages 1–6, 2018.

Robert L Smith. Efficient Monte Carlo procedures for generating points uniformly distributed over bounded regions. *Operations Research*, 32(6), 1984.

Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2010.

Freek Stulp and Olivier Sigaud. Robot skill learning: From reinforcement learning to evolution strategies. *Paladyn, Journal of Behavioral Robotics*, 4(1):49–61, 2013.

Richard S Sutton and Andrew G Barto. *Introduction to Reinforcement Learning*, volume 135. MIT Press, 1998.

Kevin Swersky, Jasper Snoek, and Ryan P Adams. Multi-task Bayesian Optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2004–2012, 2013.

Kim Peter Wabersich and Marc Toussaint. Advancing Bayesian optimization: The mixed-global-local (MGL) kernel and length-scale cool down. In *Proceedings of the NeurIPS Workshop on Bayesian Optimization*, 2016.

Zi Wang and Stefanie Jegelka. Max-value entropy search for efficient Bayesian optimization. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 3627–3635, 2017.

Ziyu Wang, Frank Hutter, Masrour Zoghi, David Matheson, and Nando de Feitas. Bayesian optimization in a billion dimensions via random embeddings. *Journal of Artificial Intelligence Research*, 55:361–387, 2016.

Aaron Wilson, Alan Fern, and Prasad Tadepalli. Using trajectory data to improve Bayesian optimization for reinforcement learning. *Journal of Machine Learning Research*, 15(1):253–282, 2014.

Kai Yuan, Iordanis Chatzinikolaidis, and Zhibin Li. Bayesian optimization for whole-body control of high-degree-of-freedom robots through reduction of dimensionality. *IEEE Robotics and Automation Letters*, 4(3):2268–2275, 2019.