Alternating Optimisation and Quadrature for Robust Control

Supratik Paul[1], Konstantinos Chatzilygeroudis[2], Kamil Ciosek[1],

Jean-Baptiste Mouret[2], Michael A. Osborne[3], Shimon Whiteson[1]

[1]Department of Computer Science, University of Oxford

[2]Inria, Villers-lès-Nancy, France; CNRS/Université de Lorraine, Loria, UMR 7503, Vandœuvre-lès-Nancy, France

[3]Department of Engineering Science, University of Oxford

[1]{supratik.paul,kamil.ciosek,shimon.whiteson}@cs.ox.ac.uk

[2]{konstantinos.chatzilygeroudis,jean-baptiste.mouret}@inria.fr

[3]mosb@robots.ox.ac.uk

Abstract

Bayesian optimisation has been successfully applied to a variety of reinforcement learning problems. However, the traditional approach for learning optimal policies in simulators does not utilise the opportunity to improve learning by adjusting certain environment variables: state features that are unobservable and randomly determined by the environment in a physical setting but are controllable in a simulator. This paper considers the problem of finding a robust policy while taking into account the impact of environment variables. We present *Alternating Optimisation and Quadrature* (ALOQ), which uses Bayesian optimisation and Bayesian quadrature to address such settings. ALOQ is robust to the presence of significant rare events, which may not be observable under random sampling, but play a substantial role in determining the optimal policy. Experimental results across different domains show that ALOQ can learn more efficiently and robustly than existing methods.

1 Introduction

A key consideration when applying *reinforcement learning* (RL) to a physical setting is the risk and expense of running trials, e.g., while learning the optimal policy for a robot. Another consideration is the robustness of the learned policies. Since it is typically infeasible to test a policy in all contexts, it is difficult to ensure it works as broadly as intended. Fortunately, policies can often be tested in a simulator that exposes key *environment variables* – state features that are unobserved and randomly determined by the environment in a physical setting but are controllable in the simulator. This paper considers how to use environment variables to help learn robust policies.

Although running trials in a simulator is cheaper and safer than running physical trials, the computational cost of each simulated trial can still be quite high. The challenge then is to develop algorithms that are sample efficient, i.e., that minimise the number of such trials. In such settings, *Bayesian Optimisation* (BO) (Brochu, Cora, and de Freitas 2010) is a sample-efficient approach that has been successfully applied to RL in multiple domains (Lizotte et al. 2007; Martinez-Cantin et al. 2007; 2009; Cully et al. 2015; Calandra et al. 2015).

A naïve approach would be to randomly sample values for the environment variables in each trial, so as to estimate expected performance. However, this approach (1) often requires testing each policy in a prohibitively large number of scenarios, and (2) is not robust to *significant rare events* (SREs), i.e., it fails any time there are rare events that substantially affect expected performance. For example, rare localisation errors may mean that a robot is much nearer to an obstacle than expected, increasing the risk of a collision. Since collisions are so catastrophic, avoiding them is key to maximising expected performance, even though the factors contributing to the collision occur only rarely. In such cases, the naïve approach will not see such rare events often enough to learn an appropriate response.

Instead, we propose a new approach called *alternating optimisation and quadrature* (ALOQ) specifically aimed towards learning policies that are robust to these rare events while being as sample efficient as possible. The main idea is to *actively* select the environment variables (instead of sampling them) in a simulator thanks to a *Gaussian Process* (GP) that models returns as a function of *both* the policy and the environment variables and then, at each time-step, to use BO and *Bayesian Quadrature* (BQ) in turn to select a policy and environment setting, respectively, to evaluate.

We apply ALOQ to a number of problems and our results demonstrate that ALOQ learns better and faster than multiple baselines. We also demonstrate that the policy learnt by ALOQ in a simulated hexapod transfers successfully to the real robot.

2 Related Work

Frank, Mannor, and Precup (2008) also consider the problems posed by SREs. In particular, they propose an approach based on importance sampling (IS) for efficiently evaluating policies whose expected value may be substantially affected by rare events. While their approach is based on *temporal difference* (TD) methods, we take a BO-based policy search approach. Unlike TD methods, BO is well suited to settings in which sample efficiency is paramount and/or where assumptions (e.g., the Markov property) that underlie TD methods cannot be verified. More importantly, they assume prior knowledge of the SREs, such that they can directly alter the probability of such events during policy evaluation. By contrast, a key strength of ALOQ is that it requires

only that a set of environment variables can be controlled in the simulator, without assuming any prior knowledge about whether SREs exist, or about the settings of the environment variables that might trigger them.

More recently, Ciosek and Whiteson (2017) also proposed an IS based algorithm, OFFER, where the setting of the environment variable is gradually changed based on observed trials. Since OFFER is a TD method, it suffers from all the disadvantages mentioned earlier. It also assumes that the environment variable only affects the initial state as otherwise it leads to unstable IS estimates.

Williams, Santner, and Notz (2000) consider a problem setting they call the *design of computer experiments* that is similar to our setting, but does not specifically consider SREs. Their proposed GP-based approach marginalises out the environment variable by alternating between BO and BQ. However, unlike ALOQ, their method is based on the EI acquisition function, which makes it computationally expensive for reasons discussed in Section 4, and is applicable only to discrete environment variables. We include their method as a baseline in our experiments. Our results presented in Section 5 show that, compared to ALOQ, their method is unsuitable for settings with SREs. Further, their method is far more computationally expensive and fails even to outperform a baseline that randomly samples the environment variable at each step.

Krause and Ong (2011) also address optimising performance in the presence of environment variables. However, they address a fundamentally different contextual bandit setting in which the learned policy conditions on the observed environment variable.

PILCO (Deisenroth and Rasmussen 2011) is a model-based policy search method that achieves remarkable sample efficiency in robot control (Deisenroth, Fox, and Rasmussen 2015). PILCO superficially resembles ALOQ in its use of GPs but the key difference is that in PILCO the GP models the transition dynamics while in ALOQ it models the returns as a function of the policy and environment variable. PILCO is fundamentally ill suited to our setting. First, it assumes that the transition dynamics are Gaussian and can be learned with a few hundred observed transitions, which is often infeasible in more complex environments (i.e., it scales poorly as the dimensionality of the state/action space increases). Second, even in simple environments, PILCO will not be able to learn the transition dynamics because in our setting the environment variable is not observed in physical trials, leading to major violations of the Gaussian assumption when those environment variables can cause SREs.

Policies found in simulators are rarely optimal when deployed on the physical agent due to the reality gap that may exist due to the inability of any simulator to model reality perfectly. EPOpt (Rajeswaran et al. 2016) tries to address this by finding policies that are robust to simulators with different settings of its parameters. First, multiple instances of the simulator are generated by drawing a random sample of the simulator parameter settings. Trajectories are then sampled from each of these instances and used by a batch policy optimisation algorithm (e.g. TRPO (Schulman et al. 2015)). While ALOQ finds a risk-neutral policy, EPOpt finds a risk-averse solution based on maximising the conditional value at risk (CVaR) by feeding the policy optimisation only the sampled trajectories whose returns are lower than the CVaR. In a risk-neutral setting, EPOpt reduces to the underlying policy optimisation algorithm with trajectories randomly sampled from different instances of the simulator. This approach will not see SREs often enough to learn an appropriate response, as we demonstrate in our experiments.

Pinto et al. (2017) also suggest a method to address the problem of finding robust policies. Their method learns a policy by training in a simulator that is adversarial in nature, i.e., the simulator settings are dynamically chosen to minimise the returns of the policy. This method requires significant prior knowledge to be able to set the simulator settings such that it provides just the right amount of challenge to the policy. Furthermore, it does not consider any settings with SREs.

3 Background

GPs provide a principled way of quantifying uncertainties associated with modelling unknown functions. A GP is a distribution over functions, and is fully specified by its mean function $m(\mathbf{x})$ and covariance function $k(\mathbf{x}, \mathbf{x}')$ (see Rasmussen and Williams (2005) for an in-depth treatment) which encode any prior belief about the nature of the function. The prior can be combined with observed values to update the belief about the function in a Bayesian way to generate a posterior distribution.

The prior mean function of the GP is often assumed to be 0 for convenience. A popular choice for the covariance function is the class of stationary functions of the form $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x} - \mathbf{x}')$, which implies that the correlation between the function values of any two points depends only on the distance between them.

In GP regression, it is assumed that the observed function values $\{f(\mathbf{x}_i)\}_{i=1}^N$ is a sample from a multivariate Gaussian distribution. The prediction for a new point $\mathbf{x}^*$ is connected with the observations through the mean and covariance functions. By conditioning on the observed data, this can be computed analytically as a Gaussian $\mathcal{N}\big(\mu\big(f(\mathbf{x}^*)\big), \sigma^2\big(f(\mathbf{x}^*)\big)\big)$:

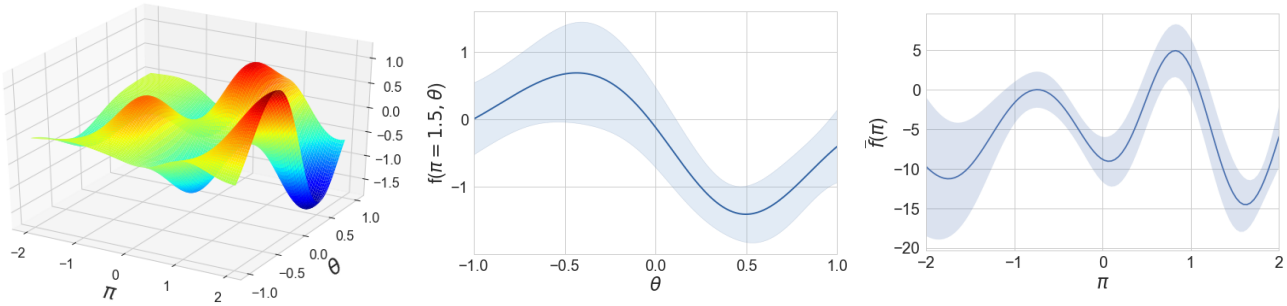
$$\mu\big(f(\mathbf{x}^*)\big) = k(\mathbf{x}^*, \mathbf{X})(\mathbf{K} + \sigma^2_{noise}\mathbf{I})^{-1} f(\mathbf{X}) \tag{1a}$$

$$\sigma^2\big(f(\mathbf{x}^*)\big) = k(\mathbf{x}^*, \mathbf{x}^*) \tag{1b}$$

$$- k(\mathbf{x}^*, \mathbf{X})(\mathbf{K} + \sigma^2_{noise}\mathbf{I})^{-1} k(\mathbf{X}, \mathbf{x}^*), \tag{1c}$$

where $\mathbf{X}$ denotes the vector of observed inputs, $f(\mathbf{X})$ the vector of corresponding function values, and $\mathbf{K}$ is the matrix with entries $k(\mathbf{x_i}, \mathbf{x_j})$.

This property of generating estimates of the uncertainty associated with any prediction makes it particularly suited for finding the optimum of $f(\mathbf{x})$ using BO. BO requires an *acquisition function* to guide the search and balance exploitation (searching the space expected to have the optimum) and exploration (searching the space which has not been explored well). Given a set of observations, the next point for evaluation is actively chosen as the $\mathbf{x}$ that maximises the acquisition function.



(a) (b) (c)

Figure 1: ALOQ models the return f as a function of (π, θ); (a) the predicted mean based on some observed data; (b) the predicted return of $\pi = 1.5$ for different θ, together with the uncertainty associated with them, given $p(\theta)$; (c) ALOQ marginalises out θ and computes $\bar{f}(\pi)$ and its associated uncertainty, which is used to actively select π.

Two commonly used acquisition functions are *expected improvement* (EI) (Močkus 1975; Jones, Schonlau, and Welch 1998) and *upper confidence bound* (UCB) (Cox and John 1992; 1997). Defining $\mathbf{x}^+$ as the current optimal evaluation, i.e., $\mathbf{x}^+ = \text{argmax}_{\mathbf{x}_i} f(\mathbf{x}_i)$, EI seeks to maximise the expected improvement over the current optimum $\alpha_{EI}(\mathbf{x}) = \mathbb{E}[I(\mathbf{x})]$, where $I(\mathbf{x}) = \max\{0, f(\mathbf{x}) - f(\mathbf{x}^+)\}$. By contrast, UCB does not depend on $\mathbf{x}^+$ but directly incorporates the uncertainty in the prediction by defining an upper bound: $\alpha_{UCB}(\mathbf{x}) = \mu(\mathbf{x}) + \kappa\sigma(\mathbf{x})$, where κ controls the exploration-exploitation tradeoff.

BQ (O'Hagan 1991; Rasmussen and Ghahramani 2003) is a sample-efficient technique for computing integrals of the form $\bar{f} = \int f(\mathbf{x})p(\mathbf{x})d\mathbf{x}$, where $p(\mathbf{x})$ is a probability distribution. Using GP regression to compute the prediction for any $f(\mathbf{x})$ given some observed data, $\bar{f}$ is a Gaussian whose mean and variance can be computed analytically for particular choices of the covariance function and $p(\mathbf{x})$ (Briol et al. 2015b). If no analytical solution exists, we can approximate the mean and variance via Monte Carlo quadrature by sampling the predictions of various $f(\mathbf{x})$.

Given some observed data $\mathcal{D}$, we can also devise acquisition functions for BQ to actively select the next point $\mathbf{x}^*$ for evaluation. A natural objective here is to select $\mathbf{x}$ that minimises the uncertainty of $\bar{f}$, i.e., $\mathbf{x}^* = \text{argmin}_{\mathbf{x}} \mathbb{V}(\bar{f}|\mathcal{D}, \mathbf{x})$ (Osborne et al. 2012). Due to the nature of GPs, $\mathbb{V}(\bar{f}|\mathcal{D}, \mathbf{x})$ does not depend on $f(\mathbf{x})$ and is thus computationally feasible to evaluate. *Uncertainty sampling* (Settles 2010) is an alternative acquisition function that chooses the $\mathbf{x}^*$ with the maximum posterior variance: $\mathbf{x}^* = \text{argmax}_{\mathbf{x}} \mathbb{V}(f(\mathbf{x})|\mathcal{D})$. Although simple and computationally cheap, it is not the same as reducing uncertainty about $\bar{f}$ since evaluating the point with the highest prediction uncertainty does not necessarily lead to the maximum reduction in the uncertainty of the estimate of the integral.

Monte Carlo (MC) quadrature simply samples $(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N)$ from $p(\mathbf{x})$ and estimates the integral as $\bar{f} \approx \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}_i)$. This typically requires a large N and so is less sample efficient than BQ: it should only be used if f is cheap to evaluate. The many merits of BQ over MC, both philosophically and practically, are brought out by O'Hagan (1987) and Hennig, Osborne, and Girolami (2015). Below, we will describe an active Bayesian quadrature scheme (that is, selecting points according to an acquisition function), inspired by the empirical improvements offered by those of Osborne et al. (2012) and Gunter et al. (2014).

4 Problem Setting & Method

We assume access to a computationally expensive simulator that takes as input a policy $\pi \in \mathcal{A}$ and environment variable $\theta \in \mathcal{B}$ and produces as output the return $f(\pi, \theta) \in \mathbb{R}$, where both $\mathcal{A}$ and $\mathcal{B}$ belong to some compact sets in $\mathbb{R}^{d_\pi}$ and $\mathbb{R}^{d_\theta}$, respectively.

We also assume access to $p(\theta)$, the probability distribution over θ. $p(\theta)$ may be known a priori, or it may be a posterior distribution estimated from whatever physical trials have been conducted. Note that we do not require a perfect simulator: any uncertainty about the dynamics of the physical world can be modelled in $p(\theta)$, i.e., some environment variables may just be simulator parameters whose correct fixed setting is not known with certainty.

Defining $f_i = f(\pi_i, \theta_i)$, we assume we have a dataset $\mathcal{D}_{1:l} = \{(\pi_1, \theta_1, f_1), (\pi_2, \theta_2, f_2), \ldots, (\pi_l, \theta_l, f_l)\}$. Our objective is to find an optimal policy π^*:

$$\pi^* = \underset{\pi}{\text{argmax}}\, \bar{f}(\pi) = \underset{\pi}{\text{argmax}}\, \mathbb{E}_\theta[f(\pi, \theta)]. \qquad (2)$$

First, consider a naïve approach consisting of a standard application of BO that disregards θ, performs BO on $\tilde{f}(\pi) = f(\pi, \theta)$ with only one input π, and attempts to estimate π^*. Formally, this approach models $\tilde{f}$ as a GP with a zero mean function and a suitable covariance function $k(\pi, \pi')$. For any given π, the variation in f due to different settings of θ is treated as noise. To estimate π^*, the naïve approach applies BO, while sampling θ from $p(\theta)$ at each timestep. This approach will almost surely fail due to not sampling SREs often enough to learn a suitable response.

By contrast, our method ALOQ (see Alg. 1) models $f(\pi, \theta)$ as a GP: $f \sim GP(m, k)$, acknowledging both its inputs. The main idea behind ALOQ is, given $\mathcal{D}_{1:l}$, to use

a BO acquisition function to select π_{l+1} for evaluation and then use a BQ acquisition function to select θ_{l+1}, conditioning on π_{l+1}.

Selecting π_{l+1} requires maximising a BO acquisition function (6) on $\bar{f}(\pi)$, which requires estimating $\bar{f}(\pi)$, together with the uncertainty associated with it. Fortunately BQ is well suited for this since it can use the GP to estimate $\bar{f}(\pi)$ together with the uncertainty associated with it. This is illustrated in Figure 1.

Once π_{l+1} is chosen, ALOQ selects θ_{l+1} by minimising a BQ acquisition function (7) quantifying the uncertainty about $\bar{f}(\pi_{l+1})$. After $(\pi_{t+1}, \theta_{l+1})$ is selected, ALOQ evaluates it on the simulator and updates the GP with the new datapoint $(\pi_{l+1}, \theta_{l+1}, f_{l+1})$. Our estimate of π^* is thus:

$$\hat{\pi}^* = \operatorname*{argmax}_{\pi} \mathbb{E}[\bar{f}(\pi) | \mathcal{D}_{1:l+1}]. \tag{3}$$

Although the approach described so far actively selects π and θ through BO and BQ, it is unlikely to perform well in practice. A key observation is that the presence of SREs, which we seek to address with ALOQ, implies that the scale of f varies considerably, e.g., returns in case of collision vs no collision. This nonstationarity cannot be modelled with our stationary kernel. Therefore, we must transform the inputs to ensure stationarity of f. In particular, we employ *Beta warping*, i.e., transform the inputs using Beta CDFs with parameters (α, β) (Snoek et al. 2014). The CDF of the beta distribution on the support $0 < x < 1$ is given by:

$$\text{BetaCDF}(x, \alpha, \beta) = \int_0^x \frac{u^{\alpha-1}(1-u)^{\beta-1}}{B(\alpha, \beta)} \mathrm{d}u, \tag{4}$$

where $B(\alpha, \beta)$ is the beta function. The beta CDF is particularly suitable for our purpose as it is able to model a variety of warpings based on the settings of only two parameters (α, β). ALOQ transforms each dimension of π and θ independently, and treats the corresponding (α, β) as hyperparameters. We assume that we are working with the transformed inputs for the rest of the paper.

While the resulting algorithm should be able to cope with SREs, the $\hat{\pi}^*$ that it returns at each iteration may still be poor, since our BQ evaluation of $\bar{f}(\pi)$ leads to a noisy approximation of the true expected return. This is particularly problematic in high dimensional settings. To address this, *intensification* (Bartz-Beielstein, Lasarczyk, and Preuss 2005; Hutter et al. 2009), i.e., re-evaluation of selected policies in the simulator, is essential. Therefore, ALOQ performs two simulator calls at each timestep. In the first evaluation, $(\pi_{l+1}, \theta_{l+1})$ is selected via the BO/BQ scheme described earlier. In the second stage, $(\hat{\pi}^*, \theta^*)$ is evaluated, where $\hat{\pi}^* \in \pi_{1:l+1}$ is selected using (3) and $\theta^* | \hat{\pi}^*$ using the BQ acquisition function (7).

Computing $\bar{f}(\pi)$: For discrete θ with support $\{\theta_1, \theta_2, \ldots, \theta_{N_\theta}\}$, the estimate of the mean μ and variance σ^2 for $\bar{f}(\pi) \mid \mathcal{D}_{1:l}$ is straightforward:

$$\mu = \frac{1}{N_\theta} \sum_{i=1}^{N_\theta} \mathbb{E}[f(\pi, \theta_i) | \mathcal{D}_{1:l}] \tag{5a}$$

$$\sigma^2 = \frac{1}{N_\theta^2} \sum_{i=1}^{N_\theta} \sum_{j=1}^{N_\theta} Cov[f(\pi, \theta_i) | \mathcal{D}_{1:l}, f(\pi, \theta_j) | \mathcal{D}_{1:l}], \tag{5b}$$

where $f(\pi, \theta)$ is the prediction from the GP with mean and covariance computed using (1). For continuous θ, we apply Monte Carlo quadrature. Although this requires sampling a large number of θ and evaluating the corresponding $f(\pi, \theta) \mid \mathcal{D}_{1:l}$, it is feasible since we evaluate $f(\pi, \theta) \mid \mathcal{D}_{1:l}$, not from the expensive simulator, but from the computationally cheaper GP.

BO acquisition function for π: A modified version of the UCB acquisition function is a natural choice since using (5) we can compute it easily as

$$\alpha_{ALOQ}(\pi) = \mu(\bar{f}(\pi) \mid \mathcal{D}_{1:l}) + \kappa\sigma(\bar{f}(\pi) \mid \mathcal{D}_{1:l}), \tag{6}$$

and set $\pi_{l+1} = \operatorname{argmax}_\pi \alpha_{ALOQ}(\pi)$.

Note that although it is possible to define an EI-based acquisition function: $\alpha = \mathbb{E}_{\bar{f}(\pi)|\mathcal{D}_{1:l}}[I(\pi)]$, where $I(\pi) = \max\{0, \bar{f}(\pi) - \bar{f}(\pi^+)\}$, as an alternative choice for ALOQ, it is prohibitively expensive to compute in practice. The stochastic $\bar{f}(\pi^+) \mid \mathcal{D}_{1:l}$ renders this analytically intractable. Approximating it using Monte Carlo sampling would require performing predictions on $l \times N_\theta$ points, i.e., all the l observed π's paired with all the N_θ possible settings of the environment variable, which is infeasible even for moderate l as the computational complexity of GP predictions scales quadratically with the number of predictions.

BQ acquisition function for θ: BQ can be viewed as performing policy evaluation in our approach. Since the presence of SREs leads to high variance in the returns associated with any given policy, it is of critical importance that we minimise the uncertainty associated with our estimate of the expected return of a policy. We formalise this objective through our BQ acquisition function for θ: ALOQ selects $\theta_{l+1} \mid \pi_{l+1}$ by minimising the posterior variance of $\bar{f}(\pi_{l+1})$, yielding:

$$\theta_{l+1} | \pi_{l+1} = \operatorname*{argmin}_{\theta} \mathbb{V}(\bar{f}(\pi_{l+1}) | \mathcal{D}_{1:l}, \pi_{l+1}, \theta). \tag{7}$$

We also tried uncertainty sampling in our experiments. Unsurprisingly it performed worse as it is not as good at reducing the uncertainty associated with the expected return of a policy as explained in Section 3.

Properties of ALOQ: Thanks to convergence guarantees for BO using α_{UCB} (Srinivas et al. 2010), ALOQ converges if the BQ scheme on which it relies also converges. Unfortunately, to the best of our knowledge, existing convergence guarantees (Kanagawa, Sriperumbudur, and Fukumizu 2016; Briol et al. 2015a) apply only to BQ methods that do not actively select points, as (7) does. Of course, we expect such active selection to only improve the rate of convergence of our algorithms over non-active versions. However, our empirical results in Section 5 show that in practice ALOQ efficiently optimises policies in the presence of SREs across a variety of tasks.

ALOQ's computational complexity is dominated by an $\mathcal{O}(l^3)$ matrix inversion, where l is the sample size of the dataset $\mathcal{D}$. This cubic scaling is common to all BO methods involving GPs. The BQ integral estimation in each iteration requires only GP predictions, which are $\mathcal{O}(l^2)$.

Algorithm 1 ALOQ

input A simulator that outputs $f = f(\pi, \theta)$, initial dataset $\mathcal{D}_{1:l}$, the maximum number of function evaluations L, and a GP prior.

1: **for** $n = l+1, l+3, ..., L-1$ **do**
2: Update the Beta warping parameters and transform the inputs.
3: Update the GP to condition on the (transformed) dataset $\mathcal{D}_{1:l}$
4: Use (5) to estimate $p(\bar{f}|\mathcal{D}_{1:n-1})$
5: Use the BO acquisition function (6) to select $\pi_n = \text{argmax}_\pi\, \alpha_{ALOQ}(\pi)$
6: Use the BQ acquisition function (7) to select $\theta_n|\pi_n = \text{argmin}_\theta\, \mathbb{V}(\bar{f}(\pi_n)|\mathcal{D}_{1:n-1}, \pi_n, \theta)$
7: Perform a simulator call with (π_n, θ_n) to obtain f_n and update $\mathcal{D}_{1:n-1}$ to $\mathcal{D}_{1:n}$
8: Find $\hat{\pi}^* = \text{argmax}_{\pi_i}\, \bar{f}(\pi_i)|\mathcal{D}_{1:n}$ and $\theta^*|\hat{\pi}^*$ using the BQ acquisition function (7).
9: Perform a second simulator call with $(\hat{\pi}^*, \theta^*)$ to obtain f_{n+1} and update $\mathcal{D}_{1:n}$ to $\mathcal{D}_{1:n+1}$
10: **end for**

output $\pi^* = \text{argmax}_{\pi_i}\, \bar{f}(\pi_i) \mid \mathcal{D}_{1:L} \quad i = 1, 2, ..., L$

5 Experimental Results

To evaluate ALOQ we applied it to 1) a simulated robot arm control task, including a variation where $p(\theta)$ is not known a priori but must be inferred from data, and 2) a hexapod locomotion task (Cully et al. 2015).

We compare ALOQ to several baselines: 1) the *naïve* method described in the previous section; 2) the method of Williams, Santner, and Notz (2000), which we refer to as *WSN*; 3) the simple policy gradient method Reinforce (Williams 1992), and 4) the state-of-the-art policy gradient method TRPO (Schulman et al. 2015). To show the importance of each component of ALOQ, we also perform experiments with ablated versions of ALOQ, namely: 1) *Random Quadrature ALOQ* (RQ-ALOQ), in which θ is sampled randomly from $p(\theta)$ instead of being chosen actively; 2) *unwarped ALOQ*, which does not perform Beta warping of the inputs; and 3) *one-step ALOQ*, which does not use intensification. All plotted results are the median of 20 independent runs.

5.1 Robotic Arm Simulator

In this experiment, we evaluate ALOQ's performance on a robot control problem implemented in a kinematic simulator. The goal is to configure each of the three controllable joints of a robot arm such that the tip of the arm gets as close as possible to a predefined target point.

Collision Avoidance In the first setting, we assume that the robotic arm is part of a mobile robot that has localised itself near the target. However, due to localisation errors, there is a small possibility that it is near a wall and some joint angles may lead to the arm colliding with the wall and incurring a large cost. Minimising cost entails getting as close to the target as possible while avoiding the region where the wall may be present. The environment variable in this setting is the distance to the wall.

Figures 2a and 2b show the expected cost (lower is better) of the arm configurations after each timestep for each method. ALOQ, unwarped ALOQ, and RQ-ALOQ greatly outperform the other baselines. Reinforce and TRPO, being relatively sample inefficient, exhibit a very slow rate of improvement in performance, while WSN fails to converge at all.

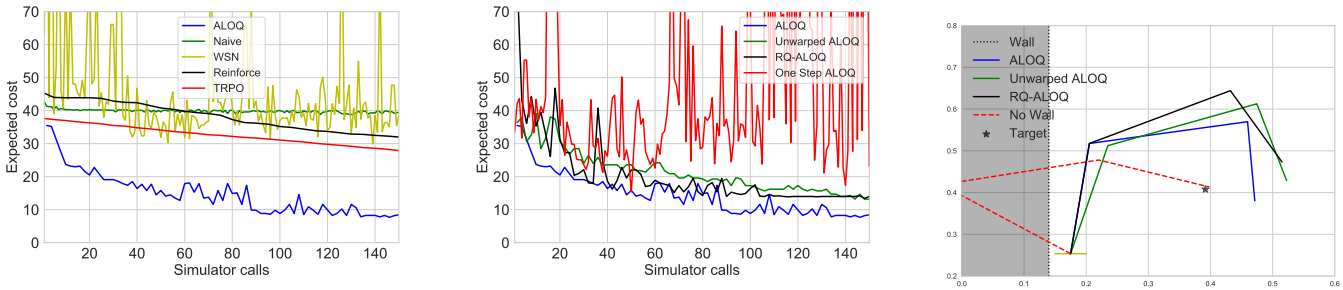
Figure 2c shows the learned arm configurations, as well as the policy that would be learned by ALOQ if there was no wall (No Wall). The shaded region represents the possible locations of the wall. This plot illustrates that ALOQ learns a policy that gets closest to the target. Furthermore, while all the BO based algorithms learn to avoid the wall, active selection of θ allows ALOQ to do so more quickly: smart quadrature allows it to more efficiently observe rare events and accurately estimate their boundary. For readability we have only presented the arm configurations for algorithms which have performance comparable to ALOQ.

Joint Breakage Next we consider a variation in which instead of uncertainty introduced by localisation, some settings of the first joint carry a 5% probability of it breaking, which consequently incurs a large cost. Minimising cost thus entails getting as close to the target as possible, while minimising the probability of the joint breaking.

Figures 3a and 3b shows the expected cost (lower is better) of the arm configurations after each timestep for each method. Since θ is continuous in this setting, and WSN requires discrete θ, it was run on a slightly different version with θ discretised by 100 equidistant points. The results are similar to the previous experiment, except that the baselines perform worse. In particular, the Naïve baseline, WSN, and Reinforce seem to have converged to a suboptimal policy since they have not witnessed any SREs.

Figure 3c shows the learned arm configurations together with the policy that would be learned if there were no SREs ('No break'). The shaded region represents the joint angles that can lead to failure. This figure illustrates that ALOQ learns a qualitatively different policy than the other algorithms, one which avoids the joint angles that might lead to a breakage while still getting close to the target faster than the other methods. Again for readability we only present the arm configurations for the most competitive algorithms.

Performance of Reinforce and TRPO Both these baselines are relatively sample inefficient. However, one question that arises is whether these methods eventually find the optimal policy. To check this, we ran them for 2000 iterations with a batch size of 5 trajectories (thus a total of 10000 simulator calls). We repeated this for both the Collision Avoidance and Joint Breakage settings. The expected cost of the arm configurations after each iteration are presented in Figure 4 (we only present the results up to 1000 simulator calls for readability - there is no improvement beyond what can be seen in the plot). Both baselines can solve the tasks in settings without SREs, i.e. where there is no possibility of a collision or a breakage ('No Wall' and 'No Break' in the figures). However, in settings with SREs they converge rapidly

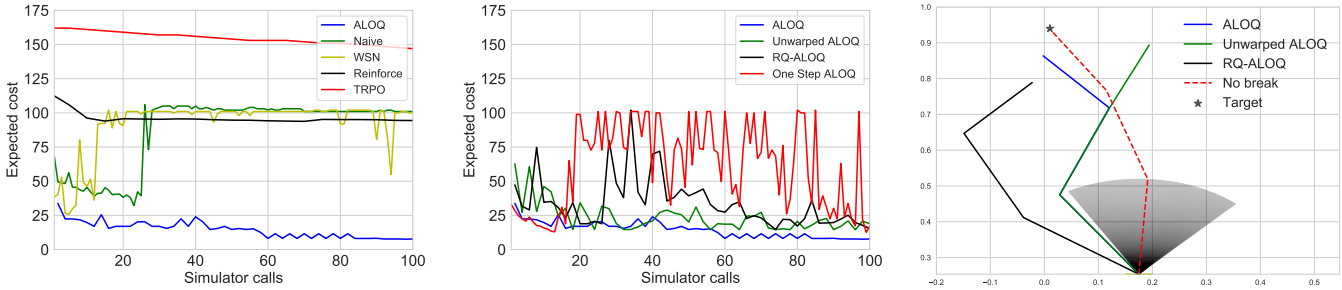


(a) Expected costs of different $\hat{\pi}^*$ - Baselines

(b) Expected costs of different $\hat{\pi}^*$ - Ablations

(c) Learned arm configurations

Figure 2: Performance and learned configurations on the robotic arm collision avoidance task.



(a) Expected costs of different $\hat{\pi}^*$ - Baselines

(b) Expected costs of different $\hat{\pi}^*$ - Ablations

(c) Learned arm configurations

Figure 3: Performance and learned configurations on the robotic arm joint breakage task.

to a suboptimal policy from which they are unable recover even if run for much longer, since they don't experience the SREs often enough. This is especially striking in the collision avoidance task where TRPO converges to a policy that has a relatively high probability of leading to a collision.

Setting with unknown $p(\theta)$ Now we consider the setting where $p(\theta)$ is not known a priori, but must be approximated using trajectories from some baseline policy. In this setting, instead of directly setting the robot arm's joint angles, we set the torque applied to each joint (π). The final joint angles are determined by the torque and the unknown friction between the joints (θ). Setting the torque too high can lead to the joint breaking, which incurs a large cost.

We use the simulator as a proxy for both real trials as well as the simulated trials. In the first case, we simply sample θ from a uniform prior, run a baseline policy, and use the observed returns to compute an approximate posterior over θ. We then use ALOQ to compute the optimal policy over this posterior ('ALOQ policy'). For comparison, we also compute the MAP of θ and the corresponding optimal policy ('MAP policy'). To show that active selection of θ is advantageous, we also compare against the policy learned by RQ-ALOQ.

Since we are approximating the unknown $p(\theta)$ with a set of samples, it makes sense to keep the sample size relatively low for computational efficiency when finding the ALOQ policy (50 samples in this instance). However, to show that ALOQ is robust to this approximation, when comparing the performance of the ALOQ and MAP policies, we used a much larger sample size of 400 for the posterior distribution.

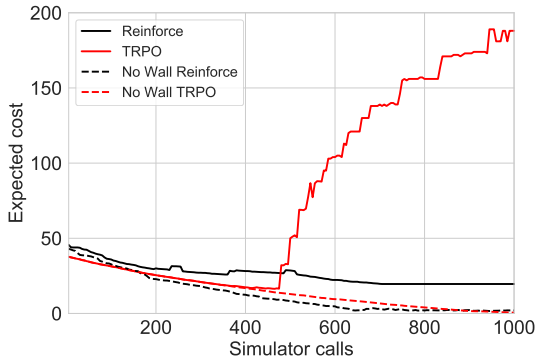
For evaluation, we drew 1000 samples of θ from the more granular posterior distribution and measured the returns of the three policies for each of the samples. The average cost incurred by the ALOQ policy (presented in Table 1) was 31% lower than that incurred by the MAP policy and 23.6% lower than the RQ-ALOQ policy. This is because ALOQ finds a policy that slightly underperforms the MAP policy in some of cases but avoids over 95% of the SREs (cost $\geq$70 in Table 1) experienced by the MAP and RQ-ALOQ policies.

Table 1: Comparison of the performance of ALOQ, MAP and RQ-ALOQ policies when $p(\theta)$ must be estimated

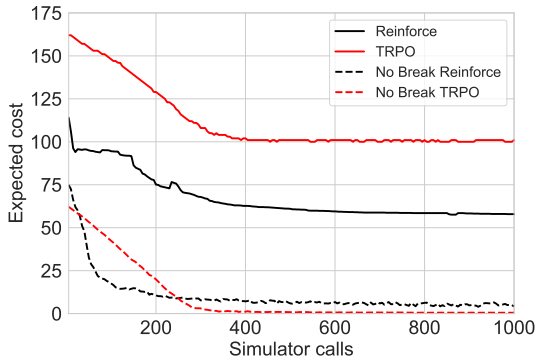
	Average Cost	% Episodes in Cost Range 0-20	20-70	$\geq$70
ALOQ Policy	19.82	61.3%	38.5%	0.2%
MAP Policy	28.76	67.1%	28.7%	4.2%
RQ-ALOQ	25.95	-	94.5%	5.5%

5.2 Hexapod Locomotion Task

As robots move from fully controlled environments to more complex and natural ones, they have to face the inevitable risk of getting damaged. However, it may be expensive or even impossible to decommission a robot whenever any damage condition prevents it from completing its task.

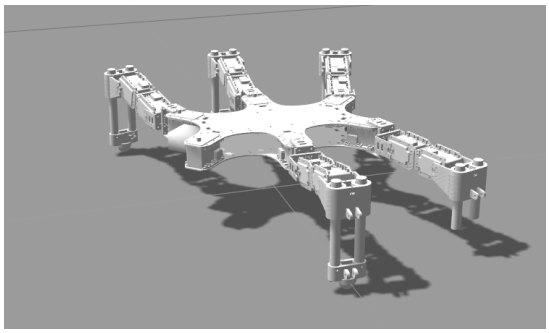


(a) Collision avoidance task

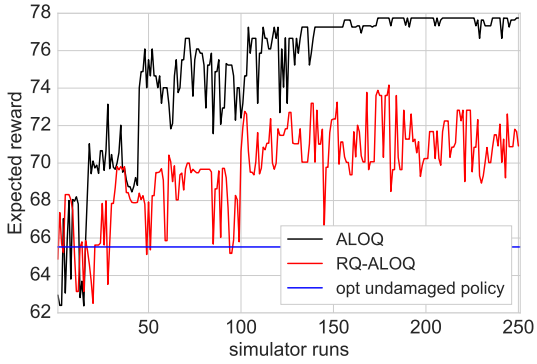


(b) Arm breakage task

Figure 4: Performance of Reinforce and TRPO on the Robotic Arm Simulator experiments.



(a) Hexapod with a shortened and a missing leg.



(b) Expected value of $\hat{\pi}^*$

Figure 5: Hexapod locomotion problem.

Hence, it is desirable to develop methods that enable robots to recover from failure.

Intelligent trial and error (IT&E) (Cully et al. 2015) has been shown to recover from various damage conditions and thereby prevent catastrophic failure. Before deployment, IT&E uses the simulator to create an archive of diverse and locally high performing policies for the intact robot that are mapped to a lower dimensional *behaviour space*. If the robot becomes damaged after deployment, it uses BO to quickly find the policy in the archive that has the highest performance on the damaged robot. However, it can only respond after damage has occurred. Though it learns quickly, performance may still be poor while learning during the initial trials after damage occurs. To mitigate this effect, we propose to use ALOQ to learn in simulation the policy with the highest expected performance across the possible damage conditions. By deploying this policy, instead of the policy that is optimal for the intact robot, we can minimise in expectation the negative effects of damage in the period before IT&E has learned to recover.

We consider a hexapod locomotion task with a setup similar to that of (Cully et al. 2015) to demonstrate this experimentally. The objective is to cross a finish line a fixed distance from its starting point. Failure to cross the line leads to a large negative reward, while the reward for completing the task is inversely proportional to the time taken.

It is possible that a subset of the legs may be damaged or broken when deployed in a physical setting. For our experiments we assume that, based on prior experience, any of the front two or back two legs can be shortened or removed with probability of 10% and 5% respectively, independent of the other legs, leading to 81 possible configurations. We excluded the middle two legs from our experiment as their failure had a relatively lower impact on the hexapod's movement. The configuration of the six legs acts as our environment variable. Figure 5a shows one such setting.

We applied ALOQ to learn the optimal policy given these damage probabilities, but restricted the search to the policies in the archive created by (Cully et al. 2015). Figure 5b shows that ALOQ finds a policy with much higher expected reward than RQ-ALOQ. It also shows the policy that generates the maximum reward when none of the legs are damaged or broken ('opt undamaged policy').

To demonstrate that ALOQ learns a policy that can be applied to a physical environment, we also deployed the best ALOQ policy on the real hexapod. In order to limit the number of physical trials required to evaluate ALOQ, we limited the possibility of damage to the rear two legs. The learnt policy performed well on the physical robot because it optimised performance on the rare configurations that matter most for expected return (e.g., either leg shortened).

6 Conclusions

This paper proposed ALOQ, a novel approach to using BO and BQ to perform sample-efficient RL in a way that is robust to the presence of significant rare events. We empirically evaluated ALOQ on different simulated tasks involving a robotic arm simulator, and a hexapod locomotion task and showed how it can be also be applied to settings where the distribution of the environment variable is unknown a priori, and that it successfully transfers to a real robot. Our results demonstrated that ALOQ outperforms multiple baselines, including related methods proposed in the literature. Further, ALOQ is computationally efficient and does not require any restrictive assumptions to be made about the environment variables.

Acknowledgments

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreements #637713 and #637972).

References

Bartz-Beielstein, T.; Lasarczyk, C. W. G.; and Preuss, M. 2005. Sequential parameter optimization. In *2005 IEEE Congress on Evolutionary Computation*, 773–780 Vol.1.

Briol, F.-X.; Oates, C. J.; Girolami, M.; Osborne, M. A.; and Sejdinovic, D. 2015a. Probabilistic Integration: A Role for Statisticians in Numerical Analysis? *ArXiv e-prints*.

Briol, F.-X.; Oates, C. J.; Girolami, M.; Osborne, M. A.; and Sejdinovic, D. 2015b. Probabilistic Integration: A role for statisticians in numerical analysis? *arXiv:1512.00933 [cs, math, stat]*. arXiv: 1512.00933.

Brochu, E.; Cora, V. M.; and de Freitas, N. 2010. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. eprint arXiv:1012.2599, arXiv.org.

Calandra, R.; Seyfarth, A.; Peters, J.; and Deisenroth, M. 2015. Bayesian optimization for learning gaits under uncertainty. *Annals of Mathematics and Artificial Intelligence*.

Ciosek, K., and Whiteson, S. 2017. Offer: Off-environment reinforcement learning. In *AAAI 2017: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*.

Cox, D. D., and John, S. 1992. A statistical method for global optimization. In *Systems, Man and Cybernetics, 1992., IEEE International Conference on*.

Cox, D. D., and John, S. 1997. Sdo: A statistical method for global optimization. In *in Multidisciplinary Design Optimization: State-of-the-Art*.

Cully, A.; Clune, J.; Tarapore, D.; and Mouret, J.-B. 2015. Robots that can adapt like animals. *Nature* 521.

Deisenroth, M. P., and Rasmussen, C. E. 2011. Pilco: A model-based and data-efficient approach to policy search. In *ICML*.

Deisenroth, M. P.; Fox, D.; and Rasmussen, C. E. 2015. Gaussian processes for data-efficient learning in robotics and control. *IEEE Trans. Pattern Anal. Mach. Intell.* 37(2):408–423.

Frank, J.; Mannor, S.; and Precup, D. 2008. Reinforcement learning in the presence of rare events. In *ICML*.

Gunter, T.; Osborne, M. A.; Garnett, R.; Hennig, P.; and Roberts, S. 2014. Sampling for inference in probabilistic models with fast bayesian quadrature. In *NIPS*.

Hennig, P.; Osborne, M. A.; and Girolami, M. 2015. Probabilistic numerics and uncertainty in computations. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*.

Hutter, F.; Hoos, H. H.; Leyton-Brown, K.; and Murphy, K. P. 2009. An experimental investigation of model-based parameter optimisation: Spo and beyond. In *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, 271–278.

Jones, D.; Schonlau, M.; and Welch, W. 1998. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*.

Kanagawa, M.; Sriperumbudur, B. K.; and Fukumizu, K. 2016. Convergence guarantees for kernel-based quadrature rules in misspecified settings. In *Advances in Neural Information Processing Systems 29*.

Krause, A., and Ong, C. S. 2011. Contextual gaussian process bandit optimization. In *NIPS*.

Lizotte, D. J.; Wang, T.; Bowling, M.; and Schuurmans, D. 2007. Automatic gait optimization with gaussian process regression. In *IJCAI*.

Martinez-Cantin, R.; de Freitas, N.; Doucet, A.; and Castellanos, J. 2007. Active policy learning for robot planning and exploration under uncertainty. In *Robotics: Science and Systems*.

Martinez-Cantin, R.; de Freitas, N.; Brochu, E.; Castellanos, J.; and Doucet, A. 2009. A bayesian exploration-exploitation approach for optimal online sensing and planning with a visually guided mobile robot. *Autonomous Robots* 27(2).

Močkus, J. 1975. On bayesian methods for seeking the extremum. In *Optimization Techniques IFIP Technical Conference Novosibirsk, July 1–7, 1974*.

O'Hagan, A. 1987. Monte carlo is fundamentally unsound. *Journal of the Royal Statistical Society. Series D (The Statistician)* 36(2/3):pp. 247–249.

O'Hagan, A. 1991. Bayes-hermite quadrature. *Journal of Statistical Planning and Inference*.

Osborne, M.; Garnett, R.; Ghahramani, Z.; Duvenaud, D. K.; Roberts, S. J.; and Rasmussen, C. E. 2012. Active learning of model evidence using bayesian quadrature. In *NIPS*.

Pinto, L.; Davidson, J.; Sukthankar, R.; and Gupta, A. 2017. Robust adversarial reinforcement learning. *CoRR* abs/1703.02702.

Rajeswaran, A.; Ghotra, S.; Levine, S.; and Ravindran, B. 2016. Epopt: Learning robust neural network policies using model ensembles. *CoRR* abs/1610.01283.

Rasmussen, C. E., and Ghahramani, Z. 2003. Bayesian monte carlo. *NIPS* 15.

Rasmussen, C. E., and Williams, C. K. I. 2005. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press.

Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; and Moritz, P. 2015. Trust region policy optimization. In Bach, F., and Blei, D., eds., *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*. Lille, France: PMLR.

Settles, B. 2010. Active learning literature survey. *University of Wisconsin, Madison* 52(55-66):11. 00000.

Snoek, J.; Swersky, K.; Zemel, R.; and Adams, R. 2014. Input warping for bayesian optimization of non-stationary functions. In *ICML*.

Srinivas, N.; Krause, A.; Kakade, S. M.; and Seeger, M. 2010. Gaussian process optimization in the bandit setting: no regret and experimental design. In *ICML*.

Williams, B. J.; Santner, T. J.; and Notz, W. I. 2000. Sequential design of computer experiments to minimize integrated response functions. *Statistica Sinica* 10(4).

Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8(3):229–256.