Proceedings of the 2020 Winter Simulation Conference
K.-H. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, eds.

BAYESIAN OPTIMIZATION SEARCHING FOR ROBUST SOLUTIONS

Hoai Phuong Le
Juergen Branke

University of Warwick
Gibbet Hill Road
Coventry, CV4 7AL, UK

ABSTRACT

This paper considers the use of Bayesian optimization to identify robust solutions, where robust means having a high expected performance given disturbances over the decision variables and independent noise in the output. We propose a variant of the well-known Knowledge Gradient acquisition function that has been proposed for the case of optimizing integrals. We empirically evaluate our method on one and two dimensional functions and demonstrate that it significantly outperforms the uniform allocation of sampling points and an alternative approach that estimates each function value by averaging over a random sample.

1 INTRODUCTION

Global optimization is the search for maxima or minima of a function over a set of inputs. Often when optimizing complex systems, it is not possible to guarantee that the implemented solution exactly follows the design specification, or there is uncertainty about the environment in which the solution operates. For example in engineering, manufacturing tolerances may mean that the actually produced solution deviates from the designed solution. Or a schedule should tolerate small deviations in estimated processing times. In such cases, one is often looking for solutions that are not only good but also robust. Robustness in this context means some degree of insensitivity to small disturbances of the environment or the design variables (Paenke et al. 2006). In this paper we focus on solutions which should have a good expected performance despite disturbances of the decision variables or noise in the output. In other words, the solution should not only be good, but its neighborhood should also have high average quality (Branke 1998).

To quantify a solution's robustness performance, one way is to sample from the uncertain distribution, and estimate its expected performance directly by averaging. This is obviously computationally very expensive, especially in higher dimensional search spaces and if the output is noisy, as many samples would be needed for an accurate estimate.

If evaluating a solution is expensive and the number of function evaluations is very limited, Bayesian optimization has been shown to be a powerful black-box optimization technique. Bayesian optimization is a machine learning approach to optimization, where an iteratively built response surface is used to determine the solution that, if added to the current dataset, would provide the largest additional value in information. In our paper, we adapt a method which has been proposed for unknown input parameter distributions of a simulation model, known as "input uncertainty", in Pearce and Branke (2017), to find the robust solution of black-box functions.

The paper is organized as follows. We begin with an overview of related literature on robust optimization in Section 2. We formally define the problem in Section 3 and explain our method for simulation optimization for robust solutions in Section 4, followed by empirical tests and comparison with some benchmarks in Section 5. Finally Section 6 consists of conclusions and suggestions for future work.

2 LITERATURE REVIEW

Bayesian optimization has recently become popular for various types of problems involving expensive-to-evaluate functions. In each iteration, a surrogate model, usually a Gaussian process (GP), is fitted, and the next sampling point is selected based on the information provided by the surrogate model, by maximizing a so-called acquisition function or infill criterion. The most popular acquisition function is Expected Improvement (EI) from Efficient Global Optimization (EGO) (Jones et al. 1998), other examples are Knowledge Gradient (Scott et al. 2011) or Entropy Search (Hennig and Schuler 2012).

There is a large number of publications regarding robust global optimization, but mostly based on evolutionary algorithms (Beyer and Sendhoff 2007). The use of Bayesian optimization to solve that problem has not been explored very much so far. The few existing papers can be divided into papers searching for solutions robust with respect to the worst case given a compact set of disturbances of the decision variable, and those that search for good expected performance given a probability distribution of disturbances, as we do in our paper.

Among the papers looking for worst-case robustness, Marzat et al. (2013) suggest to combine Bayesian optimization with an iterative relaxation procedure. The basic idea is to maintain a set of disturbances, and then alternate between finding the robust optimal solution given the set of disturbances, and finding a new worst case and adding the corresponding disturbance to the set of disturbances. Expected improvement-based algorithms are used in each of the two alternating optimization steps. One advantage of this method is that the disturbances do not need to be of the decision variables, but could also be uncertainty over input parameters of the simulation model.

ur Rehman et al. (2014) and ur Rehman and Langelaar (2017) adapt the EI acquisition function to the case of searching for a worst-case robust solution. First, the reference solution is not the best found solution, but the solution with the best worst-case prediction according to the constructed metamodel. As acquisition function, for any potential sampling point x, the worst case x_w in the disturbance region is first determined according to the metamodel, and then the predicted performance distribution at that worst case location x_w is used to compute the modified EI at location x. The solution sampled is then solution x_w. In an empirical comparison on some benchmark problems, the proposed approach was much more sample-efficient than the method from Marzat et al. (2013). The difference between ur Rehman et al. (2014) and ur Rehman and Langelaar (2017) is that the former considers unconstrained problems, whereas the latter considers constrained problems.

Sanders et al. (2019) also use the estimated posterior mean of the GP to identify the current best robust solution (the solution that has the best worst case within the disturbance region). To identify the next disturbance region to sample in, a number of realizations (functions) are drawn from the Gaussian process, and for each realization, the actual improvement over the current best solution is computed. The overall expected improvement of a solution is then the average of the improvements over all function realizations. An evolutionary algorithm is used to search for the solution with the largest expected improvement. The actual sample is then taken within the disturbance region of this solution, at the location with the largest variance as predicted by the Gaussian process. An empirical evaluation on a number of benchmark problems shows significantly better performance compared to the algorithm proposed by ur Rehman et al. (2014). While the authors focus on worst-case performance in their paper, they mention that with a minor modification, their algorithm could also be used to optimize expected performance over a disturbance region.

A very recent paper that optimizes expected performance over the input disturbances has been published by Fröhlich et al. (2020). This paper is quite similar to ours but adapts the Entropy Search acquisition function (Hennig and Schuler 2012) and tests with normally distributed disturbances, whereas our algorithm is based on the Knowledge Gradient idea and tested with uniformly distributed disturbances. Our method is similar to the case of simulation optimization with input uncertainty (Pearce and Branke 2017) or optimising expensive integrands (Toscano-Palmerin and Frazier 2018) but applied to the case of searching for robust solutions.

3 PROBLEM DEFINITION

Given a black-box function f, defined over a box-constrained input domain $X \subset \mathbb{R}^D$ subject to independent white noise $\varepsilon \sim \mathcal{N}(0, \sigma_\varepsilon^2)$ with constant variance across the input domain. Additionally defined a probability distribution $\mathbb{P}[\delta]$ of the disturbance $\delta \in [x-\Delta, x+\Delta]$ around each solution $x \in X$. While we test our approach with uniformly distributed disturbance in this paper, the algorithm applies also to other distributions of the disturbance, although one may have to revert to Monte Carlo estimation rather than analytical derivation of the acquisition function.

Our objective is to search for the robust solution that maximizes the expected performance across its Δ-neighborhood. More specifically, we aim to maximize the following robustness function.

$$\max_{x \in X} F(x) = \int_{-\Delta}^{\Delta} \int_{-\infty}^{\infty} (f(x+\delta) + \varepsilon)\, \mathbb{P}[\delta] d\varepsilon d\delta.$$

The quality of the method is measured by the *opportunity cost*, which is the difference in robustness function value between the true optimal robust solution and the solution recommended by the algorithm. Denote the final solution that the algorithm returns by x_r, the opportunity cost is then

$$OpportunityCost = \max_{x'} F(x') - F(x_r).$$

We assume f can be approximated reasonably well by a Gaussian Process. We have a limited budget of N samples (N evaluations of the latent function f), and in each iteration n, the algorithm can choose the solution x^n to be sampled, dependent on the information collected so far. The goal is to sample solutions in a way that minimizes the opportunity cost at the end of the optimization.

The algorithm needs to answer two questions: Where to sample next (i.e., what acquisition function to use), and what solution to return at the end.

4 METHODOLOGY

We propose a Bayesian optimization algorithm to efficiently search for robust solutions. The method uses a Gaussian process as surrogate model, and a new acquisition function to iteratively decide where to sample next. The surrogate model and the acquisition function are described in the following subsections.

4.1 Gaussian Process

There are some options for building a surrogate model, such as Gaussian process (GP) or Tree Parzen estimators. We chose a Gaussian process.

A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution (Rasmussen and Williams 2006). Gaussian processes are characterized by a prior *mean function* $\mu_0(x)$ and the *covariance function* or *kernel* $\Sigma_0(x, x')$.

Formulas of several types of mean functions and kernels are introduced by Rasmussen and Williams (2006), the choices are also discussed by Frazier (2018). We use the standard constant prior mean and squared exponential kernel.

We model the black-box function f by a Gaussian process, i.e.,

$$f(x) \sim \mathcal{GP}(\mu_0(x), \Sigma_0(x', x)).$$

The conditional distribution of $f(x)$ given the observations of f at n points $x^1, \ldots, x^n$ is computed as

$$f(x) | f(x^1), \ldots, f(x^n) \sim \mathcal{N}(\mu^n(x), \sigma^n(x)^2),$$

where μ^n and σ^n are respectively the posterior mean and variance of the Gaussian process built on the observations at $x^1, \ldots, x^n$ (Frazier 2018).

With the notation as follows

- $x^{1:n} = (x^1, \ldots, x^n)^T$,
- $f(x^{1:n}) = (f(x^1), \ldots, f(x^n))^T$,
- $\Sigma_0(x, x^{1:n}) = (\Sigma_0(x, x^1), \ldots, \Sigma_0(x, x^n))$,
- $\Sigma_0(x^{1:n}, x) = (\Sigma_0(x^1, x), \ldots, \Sigma_0(x^n, x))^T$,
- $\Sigma_0(x^{1:n}, x^{1:n}) = [\Sigma_0(x^1, x^{1:n}), \ldots, \Sigma_0(x^n, x^{1:n})]$,

for the case of noisy latent function where the noise is independent white noise with constant variance σ_ε, the posterior mean μ^n and posterior covariance Σ^n can be computed by

$$\mu^n(x) = \Sigma_0(x, x^{1:n})(\Sigma_0(x^{1:n}, x^{1:n}) + \sigma_\varepsilon^2 I_n)^{-1}(f(x^{1:n}) - \mu_0(x^{1:n})) + \mu_0(x), \tag{1}$$

$$\Sigma^n(x', x) = \Sigma_0(x', x) - \Sigma_0(x', x^{1:n})(\Sigma_0(x^{1:n}, x^{1:n}) + \sigma_\varepsilon^2 I_n)^{-1}\Sigma_0(x^{1:n}, x), \tag{2}$$

where I_n denotes the identity matrix of size n.

4.2 Standard Knowledge Gradient

In Bayesian optimization, at each iteration, the next sampling point is chosen as the point that maximizes a so-called acquisition function.

Knowledge Gradient (KG) for optimizing over a discrete and finite set was introduced by Frazier et al. (2009), and maximizes the expected improvement of the maximal value of posterior means conditioned on sampling once more at a specific point. With the assumption that the next sample x^{n+1} will be at x and $\mu^n(x)$ denoting the posterior mean after n samples have been taken, KG can be written as

$$KG_n(x) := \mathbb{E}[\max_{x'} \mu^{n+1}(x') - \max_{x''} \mu^n(x'') | x^{n+1} = x].$$

Scott et al. (2011) present the KG for continuous parameters, an extension of KG for correlated belief, which can be approximated by the maximization over a finite subset of the input space, for instance KG_n is approximated by discretizing X over a subset $X_{n_X} = \{x_1, \ldots, x_{n_X}\} \subset X$ as follows

$$KG_n(x) := \mathbb{E}[\max\{\mu_1 + Z\sigma_1, \ldots, \mu_{n_X+1} + Z\sigma_{n_X+1}\} | x^{n+1} = x] \tag{3}$$

where $Z \sim \mathcal{N}(0,1)$ and

$$\mu_i = \mu^n(x_i) - \max_{x' \in X_{n_X} \cup \{x\}} \mu^n(x'), \qquad i = \overline{1, n_X}, \qquad \mu_{n_X+1} = \mu^n(x) - \max_{x' \in X_{n_X} \cup \{x\}} \mu^n(x')$$

$$\sigma_i = \tilde{\sigma}^n(x_i, x) = \frac{\Sigma^n(x_i, x)}{\sigma^n(x)}, \qquad i = \overline{1, n_X}, \qquad \sigma_{n_X+1} = \tilde{\sigma}^{n_X+1} = \frac{\Sigma^n(x, x)}{\sigma^n(x)}.$$

4.3 Robust Knowledge Gradient

For the problem of identifying robust solutions we suggest the robust Knowledge Gradient *rKG*. With $\mu^n(x)$ estimating the underlying function $f(x)$, we approximate the robustness function F by

$$M^n(x) = \int_{-\Delta}^{\Delta} \mu^n(x + \delta)\mathbb{P}[\delta] d\delta, x \in X$$

The robust Knowledge Gradient is then simply the expected value of the increase in maximal value of the posterior robustness means conditioned on sampling once more at a specific location, i.e., the main difference between the standard and robust Knowledge Gradients is that we look at the change of robustness mean instead of the posterior mean, if we sample one more point.

Hence *rKG* after n samples for continuous parameters can be written as follows:

$$rKG_n(x) := \mathbb{E}\left[\max_{x'} M^{n+1}(x') - \max_{x''} M^n(x'') | x^{n+1} = x\right]. \tag{4}$$

Similarly to the standard Knowledge Gradient, we can rewrite the conditional mean as $\mu^{n+1}(x') = \mu^n(x') + \tilde{\sigma}^n(x',x)Z|x^{n+1} = x$ with $Z \sim \mathcal{N}(0,1)$. Thus

$$\begin{aligned} M^{n+1}(x') &= \int_{-\Delta}^{\Delta} (\mu^n(x'+\delta) + \tilde{\sigma}^n(x'+\delta,x)Z)\mathbb{P}[\delta]d\delta \\ &= \int_{-\Delta}^{\Delta} \mu^n(x'+\delta)\mathbb{P}[\delta]d\delta + \int_{-\Delta}^{\Delta} \tilde{\sigma}^n(x'+\delta,x)Z\mathbb{P}[\delta]d\delta \\ &= M^n(x') + \tilde{\Sigma}^n(x',x)Z, \end{aligned}$$

where

$$\tilde{\Sigma}^n(x',x) = \int_{-\Delta}^{\Delta} \tilde{\sigma}^n(x'+\delta,x)\mathbb{P}[\delta]d\delta, x' \in X. \tag{5}$$

The robust Knowledge Gradient $rKG(x)$ still has a formula similar to (3)

$$rKG_n(x) := \mathbb{E}[\max\{M_1 + Z\tilde{\Sigma}_1, \ldots, M_{n_X+1} + Z\tilde{\Sigma}_{n_X+1}\}]$$

but with the adjusted components $M_i = M^n(x_i) - \max_{x' \in X_{n_X} \cup \{x\}} M^n(x')$, $M_{n_X+1} = M^n(x) - \max_{x' \in X_{n_X} \cup \{x\}} M^n(x')$ and $\tilde{\Sigma}_i = \tilde{\Sigma}^n(x_i,x)$, $\tilde{\Sigma}_{n_X+1} = \tilde{\Sigma}^n(x,x)$, $i = \overline{1,n_X}$.

We now derive the analytical formulas of each element M_i and $\tilde{\Sigma}_i$ for the choice of squared-exponential kernel, i.e., $\Sigma_0(x',x) = \alpha_0 exp\left(-\frac{\|x'-x\|^2}{2l_x^2}\right)$ and constant prior mean μ_0.

For **uniformly distributed** disturbance δ, we have $\mathbb{P}[\delta] = \frac{1}{2\Delta}$.

For each $i = \overline{1,n_X}$, from (4) and (1) we have

$$\begin{aligned} M^n(x_i) = &\left(\int_{-\Delta}^{\Delta} \Sigma_0(x_i+\delta,x^1)\mathbb{P}[\delta]d\delta, \ldots, \int_{-\Delta}^{\Delta} \Sigma_0(x_i+\delta,x^n)\mathbb{P}[\delta]d\delta\right). \\ &\left(\Sigma_0(x^{1:n},x^{1:n}) + \sigma_\varepsilon^2 I_n\right)^{-1}\left(f(x^{1:n}) - \mu_0(x^{1:n})\right) + \int_{-\Delta}^{\Delta} \mu_0(x_i+\delta)\mathbb{P}[\delta]d\delta \end{aligned} \tag{6}$$

It is easy to see that in (6) the last integral is just μ_0 and the k^{th} element of first the vector will be

$$\int_{-\Delta}^{\Delta} \Sigma_0(x_i+\delta,x^k)\mathbb{P}[\delta]d\delta = \frac{\alpha_0}{2\Delta}\int_{-\Delta}^{\Delta} exp\left(-\frac{\|x_i - x^k + \delta\|^2}{2l_x^2}\right)d\delta.$$

As can be seen easily with the change of variables, it is nothing but the truncated Gaussian integral. Thus

$$\int_{-\Delta}^{\Delta} \Sigma_0(x_i+\delta,x^k)\mathbb{P}[\delta]d\delta = \frac{\alpha_0\sqrt{2\pi}l_x}{2\Delta}\left(\Phi\left(\frac{x_i - x^k + \Delta}{l_x}\right) - \Phi\left(\frac{x_i - x^k - \Delta}{l_x}\right)\right). \tag{7}$$

Similarly, from (2) and (5) we can compute $\tilde{\Sigma}_i$ for $i = \overline{1,n_X}$.

$$\begin{aligned} \tilde{\Sigma}^n(x_i,x) &= \int_{-\Delta}^{\Delta} \frac{\Sigma^n(x_i+\delta,x)}{\sigma^n(x)}\mathbb{P}[\delta]d\delta \\ &= \int_{-\Delta}^{\Delta} \frac{\Sigma_0(x_i+\delta,x) - \Sigma_0(x_i+\delta,x^{1:n})(\Sigma_0(x^{1:n},x^{1:n}) + \sigma_\varepsilon^2 I_n)^{-1}\Sigma_0(x^{1:n},x)}{2\Delta\sigma^n(x)}d\delta \\ &= \frac{1}{2\Delta\sigma^n(x)}\int_{-\Delta}^{\Delta} \Sigma_0(x_i+\delta,x)d\delta \\ &\quad -\frac{1}{2\Delta\sigma^n(x)}\left(\int_{-\Delta}^{\Delta} \Sigma_0(x_i+\delta,x^{1:n})d\delta\right)\left(\Sigma_0(x^{1:n},x^{1:n}) + \sigma_\varepsilon^2 I_n\right)^{-1}\Sigma_0(x^{1:n},x). \end{aligned}$$

The vector as the integral in brackets in the last equality is calculated just like in computation of $M^n(x_i)$. The first term can be computed similarly to the k^{th} element of that vector, replacing x^k with x so

$$\int_{-\Delta}^{\Delta} \Sigma_0(x_i+\delta,x)d\delta = \sqrt{2\pi}\alpha_0 l_x \left(\Phi(\frac{x_i-x+\Delta}{l_x}) - \Phi(\frac{x_i-x-\Delta}{l_x})\right).$$

M_{n_X+1} and $\tilde{\Sigma}_{n_X+1}$ can be computed similarly, replacing x_i with x.

For the higher dimensional case, $\mathbb{P}[\delta] = \frac{1}{(2\Delta)^D}$ and the only difference in the formula of the kernel is

$$\Sigma_0(x',x) = \alpha_0 exp\left(-(x'-x)^T diag(1/l_1^2,\ldots,1/l_D^2)(x'-x)/2\right),$$

where D is number of dimensions. For the purpose of numerical experiments, we only compute *rKG* for the two-dimensional case. Thus the k^{th} element of the first vector in (6) will be

$$\begin{aligned}\int_{-\Delta}^{\Delta} \Sigma_0(x_i+\delta,x^k)d\delta &= \int_{-\Delta}^{\Delta}\int_{-\Delta}^{\Delta} \frac{\alpha_0}{(2\Delta)^2} exp\left(-\frac{(x_{i_1}-x_1^k+\delta_1)^2}{2l_1^2} + \frac{(x_{i_2}-x_2^k+\delta_2)^2}{2l_2^2}\right) d\delta_1 d\delta_2 \\ &= \frac{\alpha_0}{(2\Delta)^2}\int_{-\Delta}^{\Delta} exp\left(-\frac{(x_{i_1}-x_1^k+\delta_1)^2}{2l_1^2}+\right) d\delta_1 \int_{-\Delta}^{\Delta} exp\left(\frac{(x_{i_2}-x_2^k+\delta_2)^2}{2l_2^2}\right) d\delta_2,\end{aligned}$$

and so for 2 dimensional input, the k^{th} element is

$$\int_{-\Delta}^{\Delta} \Sigma_0(x_i+\delta,x^k)d\delta = \alpha_0 \prod_{d=1}^{2} \frac{\sqrt{2\pi}l_d}{2\Delta}\left(\Phi(\frac{x_{i_d}-x_d^k+\Delta}{l_d}) - \Phi(\frac{x_{i_d}-x_d^k-\Delta}{l_d})\right). \tag{8}$$

From (7) and (8) we can generalize the formula for the k^{th} element of the vector in (6) to D dimensional input as follows

$$\int_{-\Delta}^{\Delta} \Sigma_0(x_i+\delta,x^k)d\delta = \alpha_0 \prod_{d=1}^{D} \frac{\sqrt{2\pi}l_d}{2\Delta}\left(\Phi(\frac{x_{i_d}-x_d^k+\Delta}{l_d}) - \Phi(\frac{x_{i_d}-x_d^k-\Delta}{l_d})\right).$$

For **normally distributed** disturbances, analytical formulas for M_i and $\tilde{\Sigma}_i$ can be derived similarly.

The expectation in (3) can be computed using Algorithm 1 in Scott et al. (2011).

To decide what solution to return at the end of optimization, we fit a GP model over all sampled points and find the point that maximizes the robustness performance of that model, i.e.,

$$x_r^N = \underset{x}{\operatorname{argmax}}\, M^N(x).$$

The details of our method of determining sampling points are summarized in Algorithm 1.

Algorithm 1: Pseudo-code for robustness optimization

Place a Gaussian process prior on f
Update distribution on f at n_0 initially sampled points.
Set the number of sampled points n to n_0.
while $n \leq N$ **do**
  Fit a GP on n samples.
  Calculate rKG, replacing mean and kernel of standard KG with their robustness counterparts.
  Return the point with the largest value of rKG.
  Add it to the set of sampled points and increase n by 1.
end
Result: $x_r^N = \text{argmax}_x M^N(x)$

5 EXPERIMENTS

5.1 Benchmark Problems

We test our rKG method on several one- and two- dimensional benchmark problems.

1. A simple function with two peaks, with the right one having a higher undisturbed function value, but the left one having a higher robustness value.

   $$\max f_1(x) = -0.5(x+1)\sin(\pi x^2)$$

   with $X = [0.1, 2.1]$ and $\Delta = 0.15$. See Figure 1(a) for an illustration. This function is also considered in a two-dimensional version by simply adding up over the two dimensions:

   $$\max f_3(x_1, x_2) = -0.5(x_1+1)\sin(\pi x_1^2) - 0.5(x_2+1)\sin(\pi x_2^2)$$

   with search space $[0.1, 2.1]$x$[0.1, 2.1]$ and $\Delta = (0.15, 0.15)$. For an illustration see Figure 5(a).
2. A function from Paenke et al. (2006)

   $$\min f_2(x) = 2\sin(10e^{-0.2x}x)e^{-0.25x}$$

   for robust minimum over the interval $[0, 10]$ and $\Delta = 0.5$. Figure 4(a) shows an illustration.

The disturbance distribution is uniform, capped at the boundary of the search space, i.e.,

$$\mathbb{P}[\delta] \quad = \quad \begin{cases} \frac{1}{\min\{ub, x+\Delta\} - \max\{lb, x-\Delta\}} & \delta \in [\max\{lb, x-\Delta\}, \min\{ub, x+\Delta\}] \\ 0 & \text{otherwise} \end{cases},$$

where *lb* and *ub* are the lower and upper bound of the search space, respectively.

Unless stated otherwise, we set the output noise to have standard deviation 0.1, i.e., $\varepsilon \sim \mathcal{N}(0, 0.1^2)$, although we also run some experiments without output noise.

5.2 Benchmark Methods

We compare our *rKG* algorithm with two alternative approaches.

- *Direct Robustness Approximation* (DRA) at each sample location. The idea here is to use the GP to directly approximate the robustness function, so every observation is taken as a sample of $F(x)$ with random disturbance and output noise. In order to reduce the observation error, we examine the use of averaging each observation over k independent replications. The method is then denoted $DRA(k)$. Standard KG is used as acquisition function. The GP model always allows for observation noise even if the underlying function f is deterministic, as observations are still stochastic due to the random disturbance. The method returns the solution with best posterior mean of the approximated robustness function.
- *Uniform Allocation* which allocates the samples by Latin Hypercube Sampling (McKay et al. 1979) rather than by our acquisition function. Apart from the acquisition function, the algorithm is identical to the algorithm used with *rKG*, and the method returns the solution with best robust posterior mean $x_r^N = \text{argmax}_x M^N(x)$.

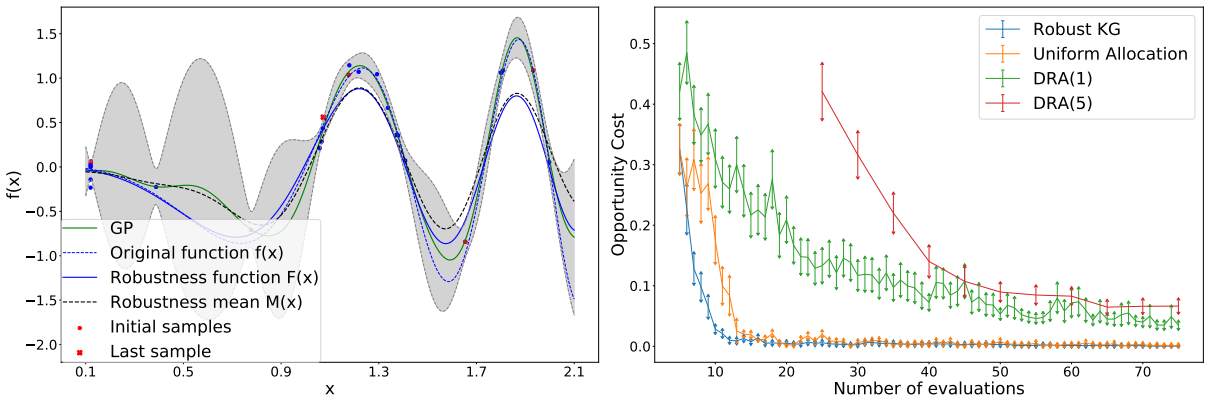
5.3 Experimental Setup

The number of initial observations was chosen as 5 for the simple one-dimensional test problem, 10 for the more complicated function f_2, and 25 for the two-dimensional test problem. Initial observations are chosen using stratified sampling. At each step, the hyperparameters of the GP are tuned by maximizing the log marginal likelihood, using the functions from TensorFlow library (Abadi et al. 2016). All results are averaged over 100 independent runs.

5.4 Results

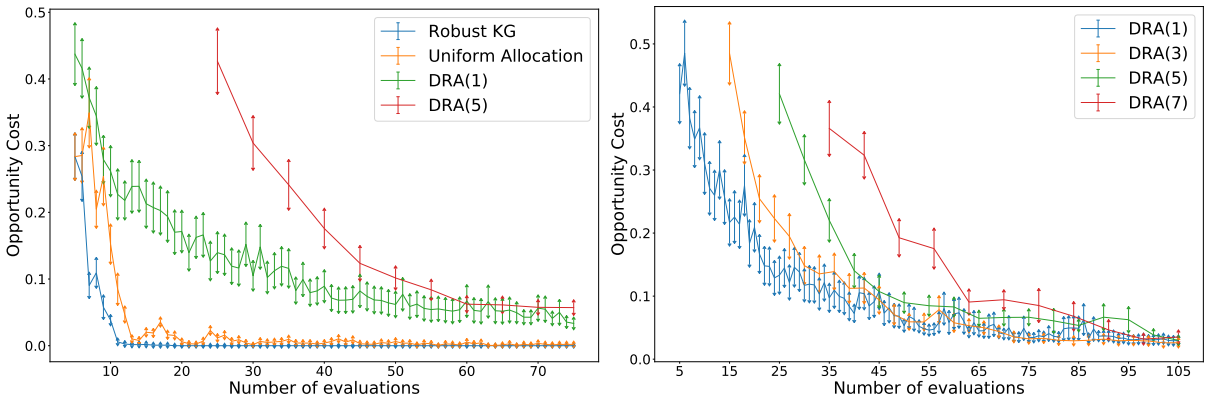
5.4.1 Simple One-Dimensional Test Function

In this section we consider the simple one-dimensional test function $f_1(x)$ displayed in Figure 1(a). Standard Knowledge Gradient optimising function f would return the solution near the point $x = 1.873$, which however has lower robustness performance. The robust optimum is approximately at the point $x = 1.22$. We start



(a) (b)

Figure 1: (a) GP built on 25 samples defined by rKG, (b) Opportunity cost over 75 evaluations.



(a) (b)

Figure 2: (a) Opportunity cost in deterministic case, (b) Opportunity cost when using *DRA*.

our experiments with an investigation of DRA method's parameter k, the number of replications to average over for a single observation. Increasing k reduces observation noise, but requires significantly more computational time. The result can be seen in Figure 2(b), which compares the opportunity cost over the total number of evaluations used so far. Obviously, with $k = 1$, DRA starts improving after the initial 5 solutions were evaluated, whereas for example the run with $k = 7$ replications requires $5 \times 7 = 35$ evaluations just to initialize. Eventually, all runs converge to a similarly good solution with low opportunity cost, but when k is smaller, convergence in terms of the total number of evaluations used is faster. Thus we conclude that at least for this test problem, it is better to use the noise handling capability of the GP rather than multiple replications to evaluate a single solution. While this would have been expected for problems with just output noise, this was not so clear for problems with disturbance of decision variables, as the

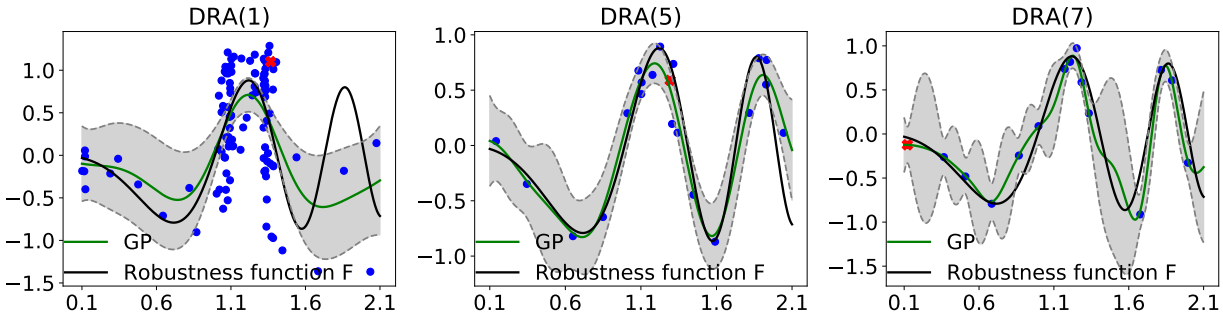


Figure 3: The GPs built on 105 sampled points as the results of using DRA(k).

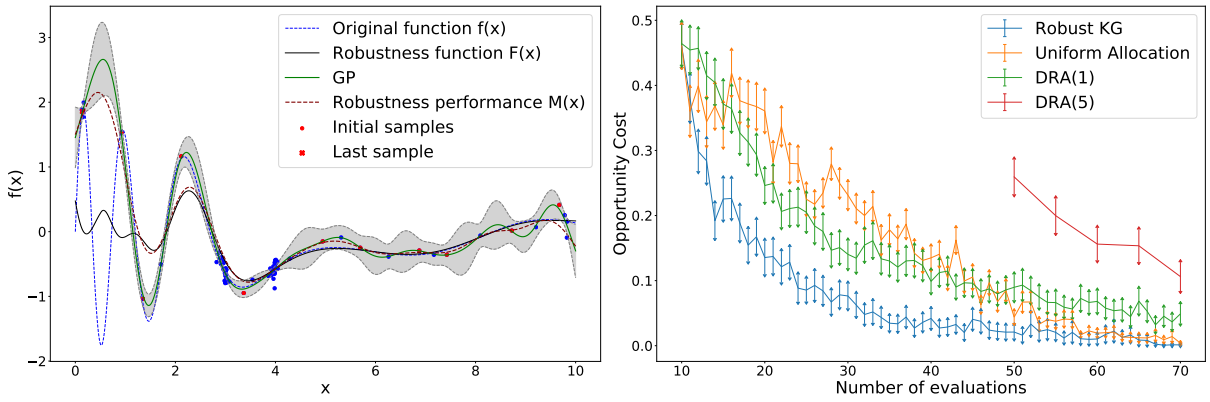
disturbance leads to heteroscedastic observation noise. Some examples for the GP built based on different k can be found in Figure 3. Figure 1(b) compares the results of *rKG*, *Uniform Allocation*, *DRA(1)* and *DRA(5)* on the simple one-dimensional test function. Clearly, our Robust KG method outperforms all other methods. On this simple function, *Uniform Allocation* is not far behind. *DRA* is significantly worse throughout the run. It is interesting to note that *DRA(1)* has a worse opportunity cost already after the initial 5 samples, i.e., using the same information as *rKG*. It seems that making the relationship between f and F explicit by learning f and deriving F through integration leads to a better model than approximating F directly, although this observation is not repeated on other benchmark problems.

Figure 1a shows an example for the resulting GP and the robustness performance, which are built on the set of samples suggested by the *rKG* method. The initial samples are coloured red, subsequent samples are coloured blue, and the red cross is the last sampled point. As can be seen, the estimated posterior robust mean $M(x)$ very closely matches the true robustness function $F(x)$ in the most promising areas.

Finally, we tested the algorithms on a deterministic version of f_1, see Figure 2a. The results on this simple function are very similar to what we have already observed on the noisy function, so the implicit averaging of the Kernel function seems to make all approaches very robust to additional output noise.

5.4.2 More Complicated One-dimensional Test Function

This test function taken from Paenke et al. (2006) is a minimization problem, it has several local minima, and they are concentrated on the left side of the search space.



(a) (b)

Figure 4: (a) GP built on 70 samples defined by rKG, (b) Opportunity cost over number of evaluations.

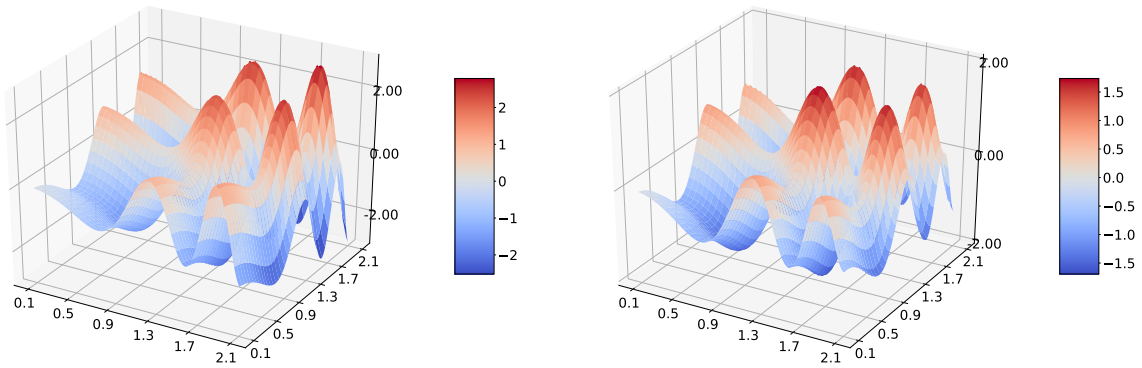
Uniform Allocation may struggle here, as allocating many samples to the right side of the search space is intuitively inefficient.

This is confirmed in Figure 4 that shows the opportunity cost over the number of evaluations. On this function, allocating samples uniformly is occasionally even worse than *DRA(1)*. Our *rKG* acquisition function again converges significantly faster than all the other approaches. The approach that uses 5 replications for each observation is particularly slow, and *rKG* has identified the optimum even before this method has properly started.

5.4.3 Two-Dimensional Test Function

For the two dimensional benchmark problem, we need more initial sampling points (we use 25), which means *DRA(5)* would need 125 evaluations to start with and renders this method not applicable. Therefore we only compare *rKG*, *Uniform allocation* and *DRA(1)*.

Figure 6(a) illustrates the learned robustness performance prediction after 100 samples using the *rKG* acquisition function. In the most relevant area around the robust peak (middle one), it approximates the true robustness performance (Figure 5(b)) very well. The opportunity cost over the number of evaluations



(a) Original function f(x)

(b) Robustness function F(x)

Figure 5: Two-dimensional benchmark problem.

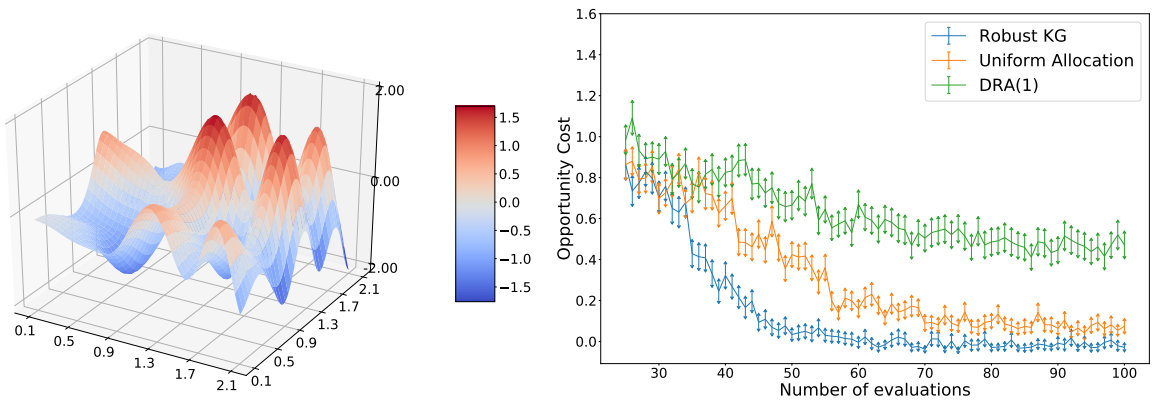
is shown in Figure 6b with error bars. While the *rKG* acquisition function once more quickly converges to the correct robust optimum (opportunity cost of zero), *Uniform Allocation* and *DRA(1)* are significantly slower.

6 CONCLUSION

In many real-world optimization problems, the solutions are affected by disturbances to the decision variables, e.g. because of manufacturing tolerances. Finding a robust optimum solution becomes critical to lessen the sensitivity to the disturbances and obtaining a high expected performance.

This paper has introduced an algorithm based on the Knowledge Gradient idea for finding the robust optimum of costly-to-evaluate functions, adapting the technique used for Bayesian optimization with input uncertainty in Pearce and Branke (2017) and Toscano-Palmerin and Frazier (2018). The key point of the algorithm is the use of a novel acquisition function, the robust Knowledge Gradient, to iteratively identify the next sampling point. We have demonstrated that robust Knowledge Gradient is efficient in achieving the same solution quality as alternative approaches at a much lower number of required function evaluations.

There are several avenues for future work. The algorithm should be tested with more problems (in particular real-world problems) and for higher dimensional cases. Also other measures of robustness should



(a) (b)

Figure 6: (a) Robustness performance of GP built on 100 samples defined by rKG, (b) Opportunity cost over evaluations.

be tested such as with a normally distributed disturbance or where one is interested not in the expected performance, but in a quantile or the worst case.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Michael Pearce for his technical assistance in completing the experiments. The first author would like to acknowledge funding from EPSRC through grant EP/L015374/1.

REFERENCES

Abadi, M., P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng. 2016. "TensorFlow: A System for Large-Scale Machine Learning". In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, 265–283.

Beyer, H.-G., and B. Sendhoff. 2007. "Robust Optimization – A Comprehensive Survey". *Computer Methods in Applied Mechanics and Engineering* 196(33):3190 – 3218.

Branke, J. 1998. "Creating Robust Solutions by Means of Evolutionary Algorithms". In *Parallel Problem Solving from Nature — PPSN V*, edited by A. E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, 119–128. Berlin, Heidelberg: Springer.

Frazier, P. 2018. "A Tutorial on Bayesian Optimization". https://arxiv.org/abs/1807.02811, accessed 28th April 2020.

Frazier, P., W. Powell, and S. Dayanik. 2009. "The Knowledge-Gradient Policy for Correlated Normal Beliefs". *INFORMS Journal on Computing* 21(4):517–656.

Fröhlich, L. P., E. D. Klenske, J. Vinogradska, C. Daniel, and M. N. Zeilinger. 2020. "Noisy-Input Entropy Search for Efficient Robust Bayesian Optimization". *ArXiv* abs/2002.02820.

Hennig, P., and C. J. Schuler. 2012. "Entropy Search for Information-Efficient Global Optimization". *Journal of Machine Learning Research* 13(1):1809–1837.

Jones, D. R., M. Schonlau, and W. J. Welch. 1998. "Efficient Global Optimization of Expensive Black-Box Functions". *Journal of Global Optimization* 13(4):45–492.

Marzat, J., E. Walter, and H. Piet-Lahanier. 2013. "Worst-Case Global Optimization of Black-Box Functions through Kriging and Relaxation". *Journal of Global Optimization* 55(4):707–727.

McKay, M., R. J. Beckman, and W. J. Conover. 1979. "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code". *Technometrics* 21(2):239–245.

Paenke, I., J. Branke, and Y. Jin. 2006. "Efficient Search for Robust Solutions by Means of Evolutionary Algorithms and Fitness Approximation". *IEEE Transactions on Evolutionary Computation* 10(4):405–420.

Pearce, M., and J. Branke. 2017. "Bayesian Simulation Optimisation with Input Uncertainty". In *Proceedings of the 2017 Winter Simulation Conference*, edited by W. K. Chan, A. D'Ambrogiom, and G. Zacharewicz, 2268–2278. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Rasmussen, C. E., and C. K. I. Williams. 2006. *Gaussian Processes for Machine Learning*. The MIT Press.

Sanders, N. D., R. M. Everson, J. E. Fieldsend, and A. A. M. Rahat. 2019. "A Bayesian Approach for the Robust Optimisation of Expensive-To-Evaluate Functions". *IEEE Transactions on Evolutionary Computation*. submitted.

Scott, W., P. Frazier, and W. Powell. 2011. "The Correlated Knowledge Gradient for Simulation Optimization of Continuous Parameters using Gaussian Process Regression". *SIAM Journal on Optimization* 21(3):996–1026.

Toscano-Palmerin, S., and P. Frazier. 2018. "Bayesian Optimization with Expensive Integrands". https://arxiv.org/pdf/1803.08661.pdf, accessed 28th April 2020.

ur Rehman, S., and M. Langelaar. 2017. "Expected Improvement Based Infill Sampling for Global Robust Optimization of Constrained Problems". *Optimization and Engineering* 18(3):723–753.

ur Rehman, S., M. Langelaar, and F. Keulen. 2014. "Efficient Kriging-based Robust Optimization of Unconstrained Problems". *Journal of Computational Science* 5(6):872–881.

AUTHOR BIOGRAPHIES

HOAI PHUONG LE is a PhD student at the Mathematics for Real-World Systems Centre for Doctoral Training (MathSys CDT), University of Warwick, UK. She graduated with a BSc in Mathematics at Rostov State University, Russian Federation in 2008 and an MSc in Mathematics at the University of Warwick in 2019. Her e-mail address is Hoai-Phuong.Le@warwick.ac.uk.

JUERGEN BRANKE is Professor of Operational Research and Systems of Warwick Business School, University of Warwick, UK. He is Editor of ACM Transactions on Evolutionary Learning and Optimization, Area Editor for the Journal of Heuristics and the Journal on Multi Criteria Decision Analysis, and Associate Editor for IEEE Transaction on Evolutionary Computation, and the Evolutionary Computation Journal. His research interests include metaheuristics and Bayesian optimization, multiobjective optimization and decision making, optimization in the presence of uncertainty, and simulation-based optimization. He has published over 180 peer-reviewed papers in international journals and conferences. His e-mail address is Juergen.Branke@wbs.ac.uk.