Supplementary Material to: Bayesian Optimization of Risk Measures

Sait Cakmak
Georgia Institute of Technology
scakmak3@gatech.edu

Raul Astudillo
Cornell University
ra598@cornell.edu

Peter Frazier
Cornell University
pf98@cornell.edu

Enlu Zhou
Georgia Institute of Technology
enlu.zhou@isye.gatech.edu

Contents

1 Computational complexity

In this section, we analyze the computational complexity of using ρKG to pick the n-th candidate to evaluate. For simplicity, we ignore the multiple restart points used for optimization, and present the analysis for a single restart point. The resulting complexity involves terms for number of LBFGS [1] iterations performed. These terms can be adjusted to account for the cost of multiple restart points.

In this analysis, we assume that evaluating the GP prior mean and kernel functions (and the corresponding derivatives) takes $\mathcal{O}(1)$ time. We break down the computations into several parts and analyze them one by one.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

An iteration starts by fitting the GP hyper-parameters using maximum a posteriori (MAP) estimation with Q_1 iterations of LBFGS algorithm. Each iteration requires calculating the likelihood of the data, which requires computing

$$A_n = \Sigma_0(x_{1:n}, w_{1:n}, x_{1:n}, w_{1:n}) + \text{diag}(\sigma^2(x_1, w_1), \ldots, \sigma^2(x_n, w_n))$$

($\mathcal{O}(n^2)$ kernel evaluations), its Cholesky factor and inverse ($\mathcal{O}(n^3)$), and solving a triangular set of equations ($\mathcal{O}(n^2)$); putting the total cost of fitting the GP at $\mathcal{O}(Q_1 n^3)$.

We can use the Cholesky factor of A_n and the inverse A_n^{-1} computed in the previous step to calculate the posterior mean and variance at the candidate point at a cost of $\mathcal{O}(n^2)$, which are then used to draw K fantasy observations at a cost of $\mathcal{O}(K)$.

Let A_{n+1} be the matrix obtained by adding A_n one row and column corresponding to the candidate (x_{n+1}, w_{n+1}) at a cost of $\mathcal{O}(n)$. Using the pre-computed Cholesky factor of A_n, we can compute the Cholesky factor of A_{n+1} at a cost of $\mathcal{O}(n^2)$. The inverse A_{n+1}^{-1} can also be obtained from A_n^{-1} at a cost of $\mathcal{O}(n^2)$. Note that A_{n+1} is identical for each fantasy model.

For each fantasy model, we need to compute the posterior mean and covariance matrix for the L points $(x, w_{1:L})$, on which we draw the sample paths. Given A_{n+1}^{-1}, we first compute $\Sigma_0(x^i, w_{1:L}, x_{1:n+1}, w_{1:n+1})A_{n+1}^{-1}$ at a cost of $\mathcal{O}(Ln^2)$, then use it to compute the mean vector and the covariance matrix at a total cost of $\mathcal{O}(L^2 n)$. To sample from the multivariate normal distribution, we also need the Cholesky factor of the covariance matrix, which is obtained at a cost of $\mathcal{O}(L^3)$. Repeating this for each fantasy, we obtain all K mean vectors, covariance matrices and the Cholesky factors at a total cost of $\mathcal{O}(KL[n^2 + Ln + L^2])$.

Given the mean vector and the Cholesky factor of the covariance matrix, we can draw a sample of $F(x^i, w_{1:L})$ at a cost of $\mathcal{O}(L^2)$. This results in a total cost of $\mathcal{O}(KML^2)$ to generate all samples. The sorting and other arithmetic operations needed for computing ρ, given the sample paths, cost $\mathcal{O}(L^2)$ per sample-path, thus introducing no additional terms.

Note that each iteration of LBFGS performs a bounded number of line search and a bounded number of function evaluations. Thus, if we use Q_3 iterations of LBFGS to solve the inner optimization problem, this translates to a total computational cost of $\mathcal{O}(Q_3 KL[n^2 + Ln + L^2 + ML])$ for the inner optimization of all K fantasies.

If we then use Q_2 iterations of LBFGS to optimize the acquisition function, we end up with a total cost of $\mathcal{O}(Q_2 Q_3 KL[n^2 + Ln + L^2 + ML])$ for optimization, and a total cost of $\mathcal{O}(Q_1 n^3 + Q_2 Q_3 KL[n^2 + Ln + L^2 + ML])$ to pick the n-th candidate to evaluate, including the cost of fitting the GP model.

2 The SAA problem

In this work, we use the SAA approach [2, 3] to optimize ρKG and ρKG^{apx}, and take advantage of the higher efficiency offered by deterministic optimization algorithms. The SAA approach trades a stochastic optimization problem with a deterministic approximation, which can be efficiently optimized. Suppose that we are interested in the optimization problem $\min_x \mathbb{E}_\omega[h(x, \omega)]$. The corresponding SAA problem is given by

$$\min_x \frac{1}{S} \sum_{i=1}^{S} h(x, \omega_i),$$

where $\omega_{1:S}$ are fixed realizations of ω. It is well known that the (expected) optimal value of SAA lower bounds the original problem, and that it is consistent as $S \to \infty$.

In the case of ρKG, the SAA problem cannot be written out explicitly, due to the sorting operation in the estimation of risk measures. However, the SAA problem and the corresponding gradient can still be obtained following an algorithmic approach, as explained in Algorithm 1. The key to obtaining the SAA problem is using the reparameterization trick [4] to write out the complex multivariate normal random variables as deterministic functions of a standard normal random vector and the mean and covariance functions of the corresponding GP model.

Algorithm 1 SAA of ρKG

Input: The candidate (x, w), the solutions to the inner problems x_*^0 and x_*^i, K, M, L.
Fix the set $\widetilde{W} = w_{1:L}$, as well as the realizations of the random variables Z_0^i, Z_L^{0j} and Z_L^{ij}, for $i = 1:K, j = 1:M$.
Use Z_0^i to obtain the fantasy GP_f^i and its parameters μ_f^i, Σ_f^i as in equation (1).
For sake of brevity, use $\text{GP}_f^0, \mu_f^0, \Sigma_f^0$ to refer to the current GP, μ_n, Σ_n.
for i = 0:K and j=1:M **do**
    Draw a sample of $F(x_*^i, w_{1:L})$ from GP_f^i as $\widehat{F}^{ij}(x_*^i, w_{1:L}) = \mu_f^i(x_*^i, w_{1:L}) + C_f^i Z_L^{ij}$ where C_f^i is the Cholesky factor of $\Sigma_f^i(x_*^i, w_{1:L}, x_*^i, w_{1:L})$.
    Use the sample $\widehat{F}^{ij}(x_*^i, w_{1:L})$ with appropriate operations to get $\mathfrak{r}^{ij}(x_*^i)$, as explained in Section 4 of the paper. The corresponding gradient is then given by $\nabla_{(x,w)}\mathfrak{r}^{ij}(x_*^i)$.
end for
Set $\widehat{\rho\text{KG}}_n(x, w) = \frac{1}{M}\sum_{j=1}^M \mathfrak{r}^{0j}(x_*^0) - \frac{1}{KM}\sum_{i=1}^K\sum_{j=1}^M \mathfrak{r}^{ij}(x_*^i)$.
Set $\nabla_{(x,w)}\widehat{\rho\text{KG}}_n(x, w) = -\frac{1}{KM}\sum_{i=1}^K\sum_{j=1}^M \nabla_{(x,w)}\mathfrak{r}^{ij}(x_*^i)$.
Return: $\widehat{\rho\text{KG}}_n(x, w)$ as the acquisition function value, and $\nabla_{(x,w)}\widehat{\rho\text{KG}}_n(x, w)$ as the corresponding gradient.

Let Z_0^i, Z_L^{ij} be fixed realizations of the standard normal random vectors of dimension 1 and L respectively. The fantasy GP model, GP_f^i, can be obtained as a continuous function of (x, w) [5] as

$$\begin{aligned}\mu_f^i(x', w') &= \mu_n(x', w') + \frac{\Sigma_n(x', w', x, w)}{\sqrt{\Sigma_n(x, w, x, w) + \sigma^2(x, w)}} Z_0^i, \\ \Sigma_f^i(x', w', x'', w'') &= \Sigma_n(x', w', x'', w'') - \frac{\Sigma_n(x', w', x, w)\Sigma_n(x, w, x'', w'')}{\Sigma_n(x, w, x, w) + \sigma^2(x, w)}.\end{aligned} \tag{1}$$

For positive definite matrices, the Cholesky factor is unique, and can be viewed as a differentiable function of the matrix [6]. In light of this fact, we can view the sample

$$\widehat{F}^{ij}(x^i, w_{1:L}) = \mu_f^i(x^i, w_{1:L}) + C_f^i Z_L^{ij}$$

as a deterministic and continuous function of both (x, w) and x^i. For fixed (x, w), x_*^i can then be obtained, again using an SAA approach, by solving

$$x_*^i = \arg\min_{x^i} \frac{1}{M}\sum_{j=1}^M \mathfrak{r}^{ij}(x^i); \tag{2}$$

which is then used to obtain the SAA problem for ρKG:

$$\widehat{\rho\text{KG}}_n(x, w) = \frac{1}{M}\sum_{j=1}^M \mathfrak{r}^{0j}(x_*^0) - \frac{1}{KM}\sum_{i=1}^K\sum_{j=1}^M \mathfrak{r}^{ij}(x_*^i).$$

The only possible gap left in the continuity / differentiability of $\widehat{\rho\text{KG}}$ is the continuity of $\mathfrak{r}^{ij}(x^i)$ as a function of $\widehat{F}^{ij}(x^i, w_{1:L})$. This is conceptually identical to the continuity and almost sure differentiability of maximum of a finite number of continuous functions, where the maximizer only changes in the case of equality. In equation 3 of the paper and the preceding ordering, the ordering changes only in case of equality, which is a probability zero event. Thus, the resulting $\mathfrak{r}^{ij}(x^i)$ is differentiable with probability one.

3 Gradients of $\mathfrak{r}^{ij}$

In this section, we make explicit the definition of the gradient $\nabla_{x^i}\mathfrak{r}^{ij}(x^i)$. In Section 4 of the paper, for a generic sample $\widehat{F}(x, w_{1:L})$, we define

$$\widehat{v}(x) := \widehat{F}(x, w_{(\lceil L\alpha\rceil)}) \quad \text{and} \quad \widehat{c}(x) := \frac{1}{\lceil L(1-\alpha)\rceil}\sum_{j=\lceil L\alpha\rceil}^L \widehat{F}(x, w_{(j)}),$$

as the corresponding MC samples of VaR and CVaR respectively. For $\mathfrak{r}^{ij}$, using the reparameterization trick, we can write the corresponding samples of F as,

$$\widehat{F}^{ij}(x^i, w_{1:L}) = \mu_f^i(x^i, w_{1:L}) + C_f^i Z_L^{ij},$$

and the corresponding $\mathfrak{r}^{ij}(x^i)$ as

$$\widehat{v}^{ij}(x^i) := \widehat{F}^{ij}(x^i, w_{(\lceil L\alpha \rceil)}) \quad \text{and} \quad \widehat{c}^{ij}(x) := \frac{1}{\lceil L(1-\alpha) \rceil} \sum_{j=\lceil L\alpha \rceil}^{L} \widehat{F}^{ij}(x^i, w_{(j)}),$$

where the ordering of $w_{(\cdot)}$ depends on the particular realization of $\widehat{F}^{ij}(\cdot, \cdot)$ as well as the value of x^i. For fixed base samples Z_L^{ij}, this is a deterministic function of μ_f^i, C_f^i and x^i; where μ_f^i, C_f^i are differentiable in x^i, as well as (x, w). Keeping the ordering of $w_{(\cdot)}$, the gradient can then be written as

$$\nabla_{x^i} \widehat{v}^{ij}(x^i) := \nabla_{x^i} \widehat{F}^{ij}(x^i, w_{(\lceil L\alpha \rceil)}) \quad \text{and} \quad \nabla_{x^i} \widehat{c}^{ij}(x) := \frac{1}{\lceil L(1-\alpha) \rceil} \sum_{j=\lceil L\alpha \rceil}^{L} \nabla_{x^i} \widehat{F}^{ij}(x^i, w_{(j)}),$$

where $\nabla_{x^i} \widehat{F}^{ij}(x^i, w_{(\cdot)})$ is given by the corresponding element of

$$\nabla_{x^i} \widehat{F}^{ij}(x^i, w_{1:L}) = \nabla_{x^i} \mu_f^i(x^i, w_{1:L}) + \nabla_{x^i} C_f^i Z_L^{ij}.$$

The treatment of gradient $\nabla_{(x,w)} \mathfrak{r}^{ij}(x^i)$ is identical to the one presented here. Additional discussion on the theoretical properties of these gradients can be found in Section 6 of this supplement.

4 Two time scale optimization

Before starting the discussion on the TTS approach, it is helpful to define the term “raw samples". In many settings as well as in this paper, non-convex functions are optimized using a multi-start gradient-based approach. Multi-start optimization requires selecting a set of restart points, which are typically randomly drawn from the solution space in the absence of domain-specific knowledge. A gradient-based algorithm then performs optimization starting from each of these restart points, which has a cost that is linear in the number of restart points. Raw samples offer an alternative way of selecting these restart points: Consider drawing a large number of points randomly from the solution space; then, instead of performing gradient-based optimization starting from each of these points, consider evaluating the objective function once for each point and then selecting a subset of them, from which to start gradient-based optimization, via a stochastic soft-max approach. We call this larger number of randomly selected points *raw samples* and we refer to the points selected for gradient-based optimization as *restart points*. This approach is better able to select high-quality restart points from which to perform gradient-based optimization, without the need for domain-specific knowledge. This enables us to use a smaller number of restart points for optimization while still providing a high-quality global solution.

Among state-of-the-art BO algorithms, knowledge gradient (KG) algorithms are on the more expensive side due to their underlying nested optimization structure. Although the added computational cost is typically justified by the superior sampling efficiency they offer (see [5, 7] for numerical comparisons), reducing the computational cost would make KG algorithms applicable in much broader settings.

To reduce the computational cost of KG optimization, [7] proposes a one-shot optimization approach, which converts the $d^{\mathcal{X}}$ dimensional nested optimization problem of the KG acquisition function [5],

$$\max_x \frac{1}{K} \sum_{i=1}^{K} \max_{x^i} \mu_f^i(x^i),$$

into a single $(K+1)d^{\mathcal{X}}$ dimensional optimization problem:

$$\max_{x, x^i, i=1,\ldots,K} \frac{1}{K} \sum_{i=1}^{K} \mu_f^i(x^i).$$

The objective function of the one-shot problem is substantially cheaper to evaluate, since it does not require solving an optimization problem. Moreover, this new problem is deterministic and smooth, allowing the use of efficient gradient-based algorithms such as LBFGS. However, the one-shot problem is again non-convex, and its large dimension makes it challenging to solve to global optimality.

Using a gradient-based approach, the one-shot problem is optimized locally from a given set of restart points, resulting in a locally optimal solution. In particular, the choice of some of the x^i can be locally optimal, in contrast with the globally optimal value for x^i found by a nested optimization approach with sufficiently many restarts. If K is large, and the majority of the inner solutions are at their global optimum, the solutions that are not globally optimized do not significantly affect the algorithm's performance. However, getting the majority at their global optimum requires starting with a large number of raw samples.

In the classical BO setting, because evaluations of $\mu_f^i(x^i)$ are cheap, one can afford to evaluate the one-shot acquisition function at a large number of (carefully crafted) raw samples, and select a good set of restart points for optimization. As K increases, both the number of raw samples and the restart points need to increase as well. When the acquisition function evaluations (even without the inner optimization) are significantly more expensive than in the classical setting (e.g., ρKG requires evaluating $\frac{1}{M}\sum_{j=1}^{M} \mathfrak{r}^{ij}(x^i)$), one is restricted to a relatively small number of raw samples, which requires using a small K (≈ 10) to achieve a good performance within a reasonable time.

Consider extending this approach to our setting, where we jointly maximize over candidates (x, w) and $x^i, i = 1, \ldots, K$, and $\mu_f^i(x^i)$ is replaced by the posterior mean of $\rho[F(x, W)]$. Since this posterior mean is more expensive to evaluate than $\mu_f^i(x^i)$, it becomes computationally burdensome to use a large number of raw samples. Using few raw samples, unfortunately, can also cause problems: a good candidate (x, w) can appear poor because many of its x^i are at local optima; while a poor candidate (x, w) can outperform it because more of its x^i happened to reach globally optimal values. In such a scenario, the one-shot approach will recommend the bad candidate with the large acquisition function value, leading to a poor sampling decision. This so-called *instability* became an issue with the one-shot implementation of ρKG in our preliminary testing and led to a search for a more *stable* alternative.

In this paper, we propose *two time scale optimization* (TTS), a novel approach for optimizing KG type of acquisition functions that addresses the issues raised here about the one-shot approach while still providing significant computational savings. The main idea behind the TTS approach is that the objective of the inner optimization problem is a continuous function of the candidate; and the optimal inner solution changes only a small amount between iterations when we update the candidate using a gradient-based algorithm. Thus, the solution to the inner problem obtained with the previous candidate provides a high quality approximation of the acquisition function value and its gradient. TTS optimization leverages this observation to provide computational savings by solving the inner optimization problem once every T iterations, and reusing the obtained solution in between. We provide a detailed description of TTS in Algorithm 2. We note that LBFGS performs multiple line searches within an iteration, and evaluates the solution found at the end of each line search. The description of Algorithm 2 treats each line search as an iteration of LBFGS, and the candidate $(x(t), w(t))$ is updated at the end of each line search. This essentially means that we optimize the inner solution at every T-th evaluation of the ρKG and not necessarily at every T-th iteration of LBFGS.

In the algorithm description, the parameter T is the *TTS-frequency* that determines how often the inner problem is solved. If $T = 1$, the TTS optimization reduces to the standard nested optimization. In numerical testing, we observed that the TTS optimization reduces the cost of optimizing ρKG roughly by a factor of T. In our experiments, we used $T = 10$ without any noticeable change in algorithm performance. We emphasize that TTS optimization can be used with other BO algorithms that involve a nested optimization, including all knowledge gradient type acquisition functions.

We mentioned earlier that the TTS optimization would address the *instability* we faced with the one-shot optimization. Going back to our example with one good and one bad candidate, TTS, by periodically solving the inner problem globally for each candidate, ensures that each candidates acquisition function value is calculated using a global solution, reducing the chances of misidentifying the bad candidate as the better one.

Algorithm 2 Two time scale optimization of ρKG

Input: $Q_2, Q_3, M, K, L, \alpha$, starting candidate $(x(0), w(0))$, and TTS-frequency T.
for $t = 0$ **to** $Q_2 - 1$ **do**
  Generate the fantasy models $\text{GP}_f^i, i = 1, \ldots, K$ using the candidate $(x(t), w(t))$.
  for $i = 1$ **to** K **do**
    if $t \bmod T = 0$ **then**
      Generate a set of random restart points for optimization of i-th inner problem. Include the previous solution $x^i(T)$ if available.
      Use Q_3 iterations of the multi-start LBFGS algorithm to optimize i-th inner problem, and set $x^i(T)$ as the optimizer.
    end if
    Return $x^i(T)$ as the solution to i-th inner problem
  end for
  Set $-\frac{1}{KM}\sum_{i=1}^{K}\sum_{j=1}^{M}\nabla_{(x,w)}\mathfrak{r}^{ij}(x^i(T))$ as the gradient at the current candidate.
  Do an iteration of LBFGS using this gradient to obtain the new candidate $(x(t+1), w(t+1))$.
end for
Return: $(x(Q_2), w(Q_2))$ as the next candidate to evaluate.

We conclude by noting that the TTS implementation presented here is a primitive one, with room for improvement. We would also like to point out that our discussion of one-shot optimization is mostly an observational one, based on certain shortcomings we faced. An in-depth comparison and further development of the two optimization methods is left as a subject for future research.

5 Experiments

5.1 Algorithm parameters

- Common parameters: Each algorithm is optimized using the LBFGS algorithm [1] (except for Covid-19 where we use SLSQP [8] due to the linear constraint on the decision variables), with $10 \times (d^{\mathcal{X}} + d^{\mathcal{W}})$ random restart points which are selected from $500 \times (d^{\mathcal{X}} + d^{\mathcal{W}})$ raw samples. We use low-discrepancy Sobol sequences [9] to generate the raw samples. The restart points are selected using a sof-=max heuristic implemented in the BoTorch package [7], which assigns each raw sample a weight that is an exponential factor of its acquisition function value, then selects the restart points with probability proportional to this weight.
- EI: Does not have any specific parameters.
- MES: A set of $500 \times (d^{\mathcal{X}} + d^{\mathcal{W}})$ randomly drawn points is used to discretize the design space, and the max values are sampled over this discretization using the Gumbel approximation. We draw 10 max value samples over the discretization, and use 128 fantasies to compute the approximate information gain from using the candidate.
- KG: We use the one-shot KG implementation with 64 fantasies.
- UCB: We use parameter $\beta = 0.2$, i.e. the acquisition function is given by $\text{UCB}(x) = \mu_n(x) + \sqrt{0.2\Sigma_n(x,x)}$.
- ρKG and ρKG^{apx}: We use $K = 10$ fantasy models for optimization and $K = 4$ fantasy models to evaluate the raw samples; and generate $M = 10$ sample paths per fantasy model. For the inner optimization problem of ρKG, we use $5 \times d^{\mathcal{X}}$ random restart points selected from $50 \times d^{\mathcal{X}}$ raw samples. We use TTS optimization with $T = 10$.

All fantasy models and sample paths are generated using low-discrepancy Sobol sequences [9]. For our algorithms, we observed a small improvement when increasing the number of fantasy models, however, this comes with a linear increase in run time and memory usage. We used a smaller number of fantasy models to evaluate the raw samples, as this reduces the computational cost without affecting performance. The number of sample paths were chosen rather arbitrarily, since the majority of computational effort is spent pn calculating the Cholesky factor, which is then used for generating all sample paths. We observed very similar performance with $M = 4, 10$ and 40; and decided to use $M = 10$ rather arbitrarily.

5.2 GP model and fitting

We use the "SingleTaskGP" model from the BoTorch package [7] with the given priors on hyper-parameters, which, quoting from the documentation, "... work best when covariates are normalized to the unit cube and outcomes are standardized (zero mean, unit variance)". We use the Matérn 5/2 kernel and fit the hyper-parameters using MAP estimation.

We project, i.e., scale, the function domain $\mathcal{X} \times \mathcal{W}$ to the unit hypercube $[0,1]^{d^{\mathcal{X}}+d^{\mathcal{W}}}$, and all the calculations are done on the unit hypercube. For calculations involving the GP, we use the "Standardize" transformation available in BoTorch, which transforms the outcomes (observations) into zero mean, unit variance observations for better compatibility with the hyper-parameter priors.

5.3 Synthetic test problems

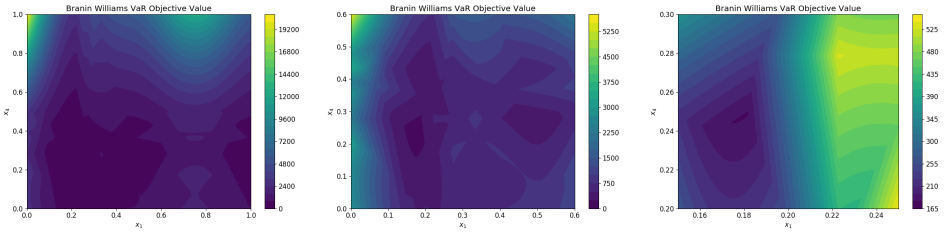

Figure 1: The VaR objective of Branin Williams function. Left to right, full domain and close-ups of the optimal region. It is notable that the objective function is quite flat across most of its domain, making it challenging to identify the optimal solution using a small number of evaluations.

The first two problems we use are synthetic test functions from the BO literature. The first problem is the 4 dimensional Branin-Williams problem formulated in [10]. For $x \in [0,1]^4$, the function is defined as $y(x) = y_b(15x_1 - 5, 15x_2) \times y_b(15x_3 - 5, 15x_4)$, where

$$y_b(u,v) = \left(v - \frac{5.1}{4\pi^2}u^2 + \frac{5}{\pi}u - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(u) + 10.$$

The function is observed with additive Gaussian noise with a standard deviation of 10. The decision variables are x_1 and x_4, and the environmental variables are $w = (x_2, x_3)$. The set $\mathcal{W}$ of size 12 and the corresponding probability distribution are given in Table 1.

The problem is originally formulated for the expectation objective, however, here we consider the minimization of VaR at risk level $\alpha = 0.7$. A plot of the VaR objective of the Branin Willams function is found in Figure 1.

The second problem we consider is $f_6(x_c, x_e)$ function from [11] given by

$$\begin{aligned} f_6(x_c, x_e) = {} & x_{e1}\left(x_{c1}^2 - x_{c2} + x_{c3} - x_{c4} + 2\right) + x_{e2}\left(-x_{c1} + 2x_{c2}^2 - x_{c3}^2 + 2x_{c4} + 1\right) \\ & + x_{e3}\left(2x_{c1} - x_{c2} + 2x_{c3} - x_{c4}^2 + 5\right) + 5x_{c1}^2 + 4x_{c2}^2 + 3x_{c3}^2 + 2x_{c4}^2 - \sum_{i=1}^{2} x_{ei}^2, \end{aligned}$$

where $x_c \in [-5,5]^4$ are the decision variables and $x_e \in [-2,2]^3$ are the environmental variables. The function is evaluated with additive Gaussian noise with a standard deviation of 1. We formulate this problem for minimization of CVaR at risk level α where $w = x_e$ has a continuous uniform distribution over its domain. For the computation of ρKG and ρKG^{apx}, we use a randomly drawn $\widetilde{\mathcal{W}}$ of size 40 at each iteration, and use randomly drawn $\widetilde{\mathcal{W}}$ of size 8 for evaluating CVaR$[F(x,W)]$ in benchmarks.

5.4 Experiment details

We initialize the GP model for the benchmark algorithms using $2d^{\mathcal{X}} + 2$ random samples of $x_j \in \mathcal{X}$, and evaluate (or estimate) $\rho[F(x_j, W)]$ using evaluations of $F(x_j, w), w \in \widetilde{\mathcal{W}}$ where $\widetilde{\mathcal{W}}$ is the set

Table 1: Probability distribution of x_2 and x_3 in Branin Williams problem.

		x_3			
		0.2	0.4	0.6	0.8
x_2	0.25	0.0375	0.0875	0.0875	0.0375
	0.5	0.0750	0.1750	0.1750	0.0750
	0.75	0.0375	0.0875	0.0875	0.0375

corresponding to the benchmark algorithms as described below. For ρKG and ρKG^{apx}, we use equivalent number of evaluations of $F(x, w)$, using (x_j, w_j) randomly drawn from $\mathcal{X} \times \mathcal{W}$. Each replication of the experiments uses a new draw of the samples; and the samples are synchronized across different algorithms.

In what follows, we describe the use of $\mathcal{W}$ in each experiment. Note that all random sets $\widetilde{\mathcal{W}}$ are independently re-drawn at each iteration.

1. Branin-Williams: The set $\mathcal{W}$ and the corresponding probability distribution is given in Table 1. Our algorithms use the full set $\mathcal{W}$ in the inner computations, and all the benchmarks evaluate $\text{VaR}_{0.7}[F(x, W)]$ using the full $\mathcal{W}$, thus using 12 evaluations per iteration.
2. $f_6(x_c, x_e)$: The set $\mathcal{W}$ is the hyper-rectangle $[-2, 2]^3$ with a continuous uniform distribution. At each iteration of our algorithms, we sample a subset $\widetilde{\mathcal{W}}$ of size 40, and use this for the inner computations. The w component of the candidate is not restricted to this subset $\widetilde{\mathcal{W}}$, and can take any value in $\mathcal{W}$. For each evaluation of the benchmarks, we sample a subset $\widetilde{\mathcal{W}}$ of size 8, and calculate $\text{CVaR}_{0.75}[F(x, W)]$ on this set, thus using 8 evaluations per iteration.
3. Portfolio optimization: The set
$$\mathcal{W} = \{w \in \mathbb{R}^2 : w_1 \in [10^{-4}, 10^{-2}], w_2 \in [10^{-4}, 10^{-3}]\}$$
gives the range of bid-ask spread (w_1) and the borrowing cost (w_2), and W follows a uniform distribution on $\mathcal{W}$. For the inner computations of our algorithms, we use a randomly drawn subset $\widetilde{\mathcal{W}}$ of size 40, and use a subset $\widetilde{\mathcal{W}}$ of size 10 for benchmark evaluations of $\text{VaR}_{0.8}[F(x, w)]$, thus using 10 evaluations per iteration.
4. COVID-19: The set $\mathcal{W}$ of initial disease propensity is given by
$$\mathcal{W} = \left\{ w \in \mathbb{R}^3 : \begin{array}{c} w_1 \in \{0.001, 0.0025, 0.004\}, \\ w_2 \in \{0.002, 0.004, 0.006\}, \\ w_3 \in \{0.002, 0.005, 0.008\} \end{array} \right\}$$
with the probability distribution given by
$$\mathbb{P}_W(W = w) = 2^{-8+\mathbb{1}(w_1=0.0025)+\mathbb{1}(w_2=0.004)+\mathbb{1}(w_3=0.005)}.$$
For the inner computation of ρKG^{apx}, we use the full set $\mathcal{W}$. For evaluations of the benchmarks, we draw a random sample $\widetilde{\mathcal{W}}$ of size 10 and use this to calculate $\text{CVaR}_{0.9}[F(x, W)]$, thus using 10 evaluations per iteration.
The output of the Covid-19 simulator is stochastic with a large variation. To enable fair comparison between the algorithms, we fix a set of 10 seeds, and each function evaluation is done using a randomly selected seed from this set. The reported performance is computed as the average performance of using all 10 seeds.

We ran the benchmark algorithms EI, MES, KG, UCB and random for 100 replications in each experiment. For $\rho-$random and ρKG^{apx}, we ran 50 replications in each experiment, and ρKG was run for 30 replications in each experiment.

5.5 Results

In the main body of the paper, we present the results using a moving average window of 3 iterations, to smooth out the oscillations and make the plots easier to read. In Figure 2, we present the non-smoothed plots to complement the moving averages presented in the paper.

The comparatively larger variance of our algorithms is attributed to: (i) our methods reporting a new solution $\sim$10x more frequently (after each evaluation of $F(x, w)$, rather than an evaluation at many w), leading to more points in the plot; (ii) our methods having fewer replications than the benchmark algorithms.

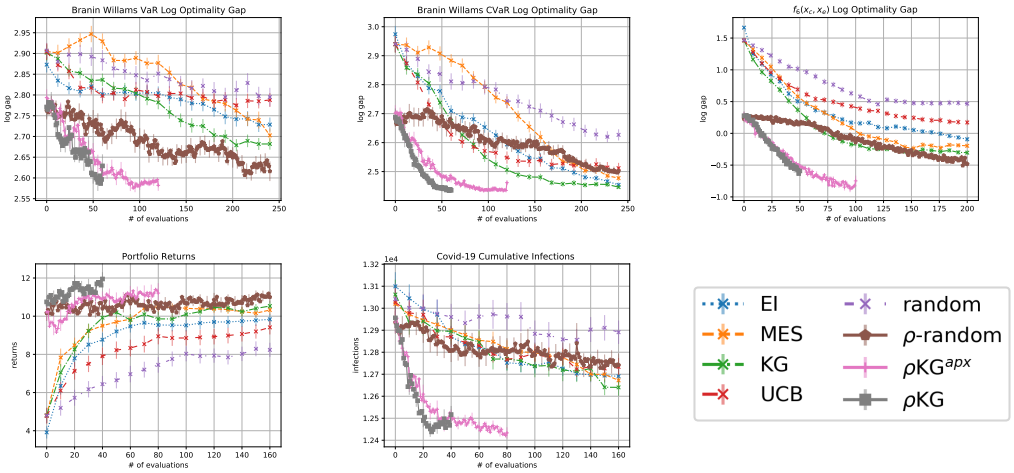

Figure 2: Top: The log optimality gap in Branin Williams with VaR (left) with CVaR (middle), and $f_6(x_c, x_e)$ (right). Bottom: The returns on Portfolio problem (left), the cumulative number of infections in the COVID-19 problem (middle) and the legend (right). The plots are plotted against the number of $F(x, w)$ evaluations.

5.6 Results comparing algorithm run times

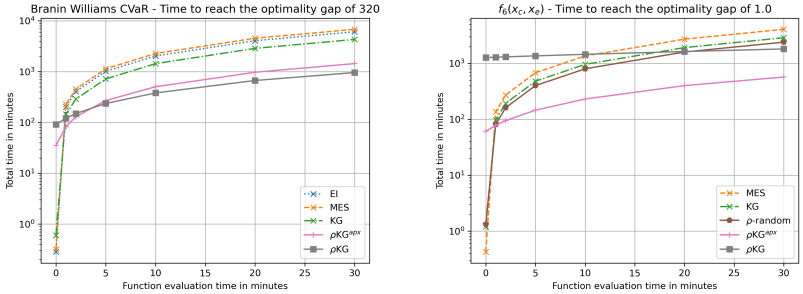

Figure 3: Left: Time to reach an optimality gap of 320 on the Branin Williams problem with CVaR objective. Right: Time to reach an optimality gap of 1 on the $f_6(x_c, x_e)$ function. The plots show the total run time vs the time per a single evaluation of the function $F(x, w)$.

In Figure 3, we present plots that show the total time it takes for the algorithms to reach a given optimality gap threshold, as a function of the cost of a single evaluation of the function $F(x, w)$. The plots make clear the advantage of using our acquisition functions, instead of the benchmarks studied in the paper, even with only moderately expensive function evaluations. The Branin Williams and $f_6(x_c, x_e)$ were chosen since the results for these experiments allow for a meaningful comparison; and that they are representative as, roughly speaking, the least and the most computationally intensive in terms of acquisition function optimization, respectively, of all the numerical examples presented in the paper. It is seen that in $f_6(x_c, x_e)$, the ρKG algorithm is quite expensive to optimize, and is not a good contender unless the function evaluations are expensive enough. However, for both examples, ρKG^{apx} proves to be a better option than all the benchmarks, even when the function evaluations take only a few minutes.

6 Proofs of theoretical results

This section is dedicated to showing that the derivative estimators we propose in this paper are asymptotically unbiased and consistent. Here, we only present the proofs for the case of $d^{\mathcal{W}} = 1$.

The derivative of a GP is again a GP (cf. [12] and section 9.4 of [13]). If we model $F(x)$ as a GP with mean and covariance functions μ, Σ, then F and ∇F follow a multi output GP with mean and covariance functions given by [12]

$$\widetilde{\mu}(x) = (\mu(x), \nabla\mu(x))^\top \text{ and } \widetilde{\Sigma}(x, x') = \begin{pmatrix} \Sigma(x, x') & \nabla\Sigma(x, x') \\ \nabla\Sigma(x', x)^\top & \nabla^2\Sigma(x, x') \end{pmatrix};$$

where $\nabla\Sigma$ and $\nabla^2\Sigma$ are the Jacobian and Hessian of the covariance function. We note that, although this result is found in the literature without any assumptions on the GP, it in fact requires the GP to possess differentiable sample paths. For example, the Brownian motion is a well known example of a GP that has nowhere differentiable sample paths, thus violates this framework. To this end, we refer the reader to [14] for precise conditions on the differentiability of the GP sample paths, and *assume* that these conditions are satisfied.

Remark 1. *Before discussing the assumptions below, we note that the assumptions listed here are more restrictive than needed. The main results we build on require certain conditions on the distribution of the function in a neighborhood of its VaR. However, when the function is drawn from a GP, it is impossible to know where this neighborhood, which would be different for each draw, might be, and thus it is impossible to impose assumptions on such an unknown and ever changing neighborhood. The main purpose of the results presented here is to show that there indeed exists a set of conditions under which our claims hold. However, we are confident that our algorithms are applicable in much broader settings, even when no such assumption can be verified.*

We *assume* that the kernel $\widetilde{\Sigma}$ of the joint GP of the function and its derivative is strictly positive definite; and the random environmental variable W admits a strictly positive and continuous density $p(w)$ over its domain. (Note that this assumption is easily satisfied by adding a small Gaussian noise to W.) Observe that when the kernel is strictly positive definite, the sample paths of the GP are non-zero w.p.1., i.e. the equation $F(x, w) = 0$ has at most countably many solutions, where $F(\cdot, \cdot)$ denotes a sample drawn from the GP.

Lemma 1. *Under conditions outlined above, for any t such that $P_W(F(x, W) \leq t) \in (0, 1)$, the CDF $P_W(F(x, W) \leq t)$ is continuously differentiable w.r.t. both x and t; and the density $\partial_t P_W(F(x, W) \leq t)$ is strictly positive for each x.*

Proof. Since there are at most countably many points where the derivative is zero, we can construct countably many distinct intervals $\mathcal{W}_i, i \in I$ where $F(x, w)$ is monotone (in w) in each $\mathcal{W}_i$, and $\mathcal{W}_i$ are such that $P_W(\cup_{i\in I}\mathcal{W}_i) = 1$ and $P_W(\mathcal{W}_i) > 0, i \in I$. Then,

$$\begin{aligned}
\partial_x P_W(F(x, W) \leq t) &= \partial_x \int_{\{w:F(x,w)\leq t\}} p(w)dw \\
&= \partial_x \sum_{i\in I} P_W(\mathcal{W}_i) \int_{\{w\in\mathcal{W}_i:F(x,w)\leq t\}} \frac{p(w)}{P_W(\mathcal{W}_i)} dw \\
&= \sum_{i\in I} \partial_x P_W(\mathcal{W}_i) \int_{\{w\in\mathcal{W}_i:F(x,w)\leq t\}} p(w)dw \qquad = (*)
\end{aligned}$$

where the interchange of derivative and summation is justified as the sum is bounded. From here on, we ignore the $\mathcal{W}_i$ that do not produce any root to the equation $F(x, w) = t$, as the corresponding derivatives are zero. Since $F(x, w)$ is continuously differentiable and monotone in w in each $\mathcal{W}_i$, by the inverse function theorem [15], there exists a continuously differentiable inverse function $F_i^{-1}(t'; x)$ mapping t' to $w \in \mathcal{W}_i$ for a given x. For the remaining $\mathcal{W}_i$, define the sets I^+ and I^- such that, I^+ is the set of $\mathcal{W}_i$ that has $F_i^{-1}(t; x)$ as the upper bound of the integral and I^- as the ones with the lower bound. For $i \in I^+$, let w_i^- be the lower bound of the integral, and similarly w_i^+

as the upper bound for I^-. Then,

$$\begin{aligned}(*) &= \sum_{i\in I^+} \partial_x \int_{w_i^-}^{F_i^{-1}(t;x)} p(w)dw + \sum_{i\in I^-} \int_{F^{-1}(t;x)}^{w_i^+} p(w)dw \\ &= \sum_{i\in I^+} p(w)\partial_x F_i^{-1}(t;x) - \sum_{i\in I^-} -p(w)\partial_x F_i^{-1}(t;x) \\ &= \sum_{i\in I^+\cup I^-} p(w)\partial_x F_i^{-1}(t;x)\end{aligned}$$

where the result follows by the Leibniz rule. Applying the same steps with ∂_t, we get

$$\partial_t P_W(F(x,W)\le t) = \sum_{i\in I^+\cup I^-} p(w)\partial_t F_i^{-1}(t;x).$$

Since $p(w)$ is continuous and $F_i^{-1}(t;x)$ is continuously differentiable, $P_W(F(x,W)\le t)$ is continuously differentiable w.r.t. both x and t. Since $p(w)$ is strictly positive and the derivative is non-zero (and the CDF is a non-decreasing function), it follows that $\partial_t P_W(F(x,W)\le t)$ is strictly positive, thus completing the proof. □

Proposition 1. *Under the conditions outlined above, for $\alpha\in(0,1)$,*

$$\frac{1}{M}\sum_{j=1}^{M} \nabla_x \widehat{v}^j(x) \xrightarrow{p} \nabla_x \mathbb{E}_{n+1}[\textit{VaR}_\alpha[F(x,W)]]$$

as $M, L \to \infty$, where $\xrightarrow{p}$ denotes convergence in probability. Moreover, the gradient $\nabla_x \mathbb{E}_{n+1}[\textit{VaR}_\alpha[F(x,w)]]$ is continuous.

Proof. Let $F(x,w)(\omega)$ denote a sample path of the GP. As a sample path of the GP, $F(x,w)(\omega)$ is continuously differentiable w.r.t. w w.p.1. By Lemma 1, the CDF $P_W(F(x,W)\le t)$ is continuously differentiable w.r.t. both x and t, and the density $\partial_t P_W(F(x,W)\le t)$ is strictly positive. Then, by Lemma 2 of [16],

$$\lim_{L\to\infty} \mathbb{E}[\nabla_x \widehat{v}(x)] = \nabla_x \text{VaR}_\alpha[F(x,W)]. \tag{3}$$

Since the gradient is also a GP defined in a compact space, $\nabla_x F(x,w)(\omega)$ is bounded w.p.1., so is $\nabla_x \text{VaR}_\alpha[F(x,W)]$. Then, by proposition 1 of [17],

$$\nabla_x \mathbb{E}_{n+1}[\text{VaR}_\alpha[F(x,W)]] = \mathbb{E}_{n+1}[\nabla_x \text{VaR}_\alpha[F(x,W)]]. \tag{4}$$

$F(x,w)$ is bounded by the same argument as $\nabla_x F(x,w)$, thus has finite second moment. Therefore, the main result follows by an application of Chebyshev's inequality. The continuity of $\nabla_x \mathbb{E}_{n+1}[\text{VaR}_\alpha[F(x,W)]]$ follows by Theorem 1 of [16] and the fact that both gradients are continuous as shown in Lemma 1. □

Before going into the main result, we discuss the mapping from the candidate point (x,w) to the sample $\widehat{F}^{ij}(\cdot,\cdot)$, the j-th sample path drawn from the i-th fantasy GP model, GP_f^i. Let's start with the mapping from the candidate to the mean and covariance functions of GP_f^i. Let Z^i be (a fixed realization of) the standard normal random variable used to generate the fantasy. The mean and covariance functions of GP_f^i are given by:

$$\begin{aligned}\mu_f^i(x',w') &= \mu_n(x',w') + \frac{\Sigma_n(x',w',x,w)}{\sqrt{\Sigma_n(x,w,x,w)+\sigma^2(x,w)}} Z^i, \\ \Sigma_f^i(x',w',x'',w'') &= \Sigma_n(x',w',x'',w'') - \frac{\Sigma_n(x',w',x,w)\Sigma_n(x,w,x'',w'')}{\Sigma_n(x,w,x,w)+\sigma^2(x,w)},\end{aligned}$$

where $\sigma^2(x,w)$ is the known observation noise level (see [18] for more details). *Assuming* that the prior mean and covariance functions are continuously differentiable, μ_f^i and Σ_f^i are continuously differentiable w.r.t. the candidate (x,w).

For a given x^i and finite set $\widetilde{W} = \{w_{1:L}\}$, and a (realization of) L - dimensional standard normal random vector Z_L^{ij}, the j-th sample path of i-th fantasy model is generated as

$$\widehat{F}^{ij}(x^i, w_{1:L}) = \mu_f^i(x^i, w_{1:L}) + C(x^i, w_{1:L}) Z_L^{ij}, \tag{5}$$

where $C(x^i, w_{1:L})$ is the Cholesky factor of $L \times L$ covariance matrix $\Sigma_f^i(x^i, w_{1:L}, x^i, w_{1:L})$. Since Σ_f^i is continuously differentiable w.r.t. the candidate, so is $C(x^i, w_{1:L})$, making the sample $\widehat{F}^{ij}(\cdot, \cdot)$ continuously differentiable w.r.t. the candidate (x, w).

The discussion above establishes continuous differentiability of the sample $\widehat{F}^{ij}(\cdot, \cdot)$ w.r.t. the candidate (x, w) only for a finite collection of points $(x^i, w_{1:L})$. However, to be able to use the analysis of Proposition 1, we need continuous differentiability of $\widehat{F}^{ij}(x, w), w \in W$ which, with W being a continuous random variable, is an infinite dimensional random variable. The concept of an infinite dimensional standard Gaussian random variable, or a standard Gaussian random function, is not well defined, and we can no longer use equation (5). In the proposition below, we *assume* that $F(\cdot, \cdot)$ is indeed differentiable w.r.t. the candidate (x, w) and hope that this discussion justifies the assumption. Below, we use $u := (x, w)$ as the candidate to simplify the notation.

Proposition 2. *Under the conditions outlined above, and assuming that the solution to the inner problem x^* is unique w.p.1.,*

$$\frac{1}{K}\sum_{i=1}^{K} -\frac{1}{M}\sum_{j=1}^{M} \nabla_u \widehat{v}^j(x_i^*) \xrightarrow{p} \nabla_u \mathbb{E}_n[-\min_x \mathbb{E}_{n+1}[\textit{VaR}_\alpha[F(x, w)]] \mid u] \tag{6}$$

as $K, L, M \to \infty$

Proof. Following the proof of proposition 1 with ∇_u replacing ∇_x, we get

$$\lim_{L\to\infty} \mathbb{E}_{n+1}[\mathbb{E}[\nabla_u \widehat{v}(x)]] = \nabla_u \mathbb{E}_{n+1}[\text{VaR}_\alpha[F(x, W)]].$$

By the assumption that x^* is unique w.p.1. and the continuity of $\nabla_u E_{n+1}[\text{VaR}_\alpha[F(x, W)]]$ by proposition 1, we can apply the envelope theorem (corollary 4, [19]) to get

$$\lim_{L\to\infty} \mathbb{E}_{n+1}[\mathbb{E}[\nabla_u \widehat{v}(x^*)]] = \nabla_u \mathbb{E}_{n+1}[\text{VaR}_\alpha[F(x^*, W)]].$$

Again by the arguments in the proof of proposition 1, the gradient is continuous and bounded w.p.1., thus applying proposition 1 of [17], we get

$$\lim_{L\to\infty} \mathbb{E}_n[\mathbb{E}_{n+1}[\mathbb{E}[\nabla_u \widehat{v}(x^*)]] \mid u] = \nabla_u \mathbb{E}_n[\mathbb{E}_{n+1}[\text{VaR}_\alpha[F(x^*, W)]] \mid u]. \tag{7}$$

Since the second moment of the estimators are bounded (see the proof of proposition 1), the result follows by Chebyshev's inequality. □

Proposition 3. *The results presented in Propositions 1 & 2 extend to the risk measure CVaR, i.e.*

$$\frac{1}{M}\sum_{j=1}^{M} \nabla_x \widehat{c}^j(x) \xrightarrow{p} \nabla_x \mathbb{E}_{n+1}[\textit{CVaR}_\alpha[F(x, W)]];$$

and

$$\frac{1}{K}\sum_{i=1}^{K} -\frac{1}{M}\sum_{j=1}^{M} \nabla_u \widehat{c}^j(x_i^*) \xrightarrow{p} \nabla_u \mathbb{E}_n[-\min_x \mathbb{E}_{n+1}[\textit{CVaR}_\alpha[F(x, w)]] \mid u] \tag{8}$$

as $K, L, M \to \infty$.

Proof. We have established in the proof of Proposition 1 that $\nabla F(x, w)$ is bounded, and that $\text{VaR}_\alpha[F(x, W)]$ is continuously differentiable. Combining these with the assumption that W is a continuous random variable, we can apply Theorem 3.1 of [20] to get

$$\nabla \text{CVaR}_\alpha[F(x, W)] = \mathbb{E}[\nabla F(x, W) \mid F(x, W) \geq \text{VaR}_\alpha[F(x, W)]].$$

Since $\nabla F(x, W)$ is continuous and bounded, it follows by bounded convergence theorem [21] that $\text{CVaR}_\alpha[F(x, W)]$ is continuously differentiable. The remaining arguments are identical to those of Propositions 1 & 2. □

We conclude this discussion by noting that, while the proposition in the main paper claims both asymptotic unbiasedness and consistency, the propositions proved here only establish the consistency of the estimators, and the asymptotic unbiasedness result is hidden within the proof argument. In the proof of Proposition 1, combining the equations (3) and (4) gives us the asymptotic unbiasedness of the gradient estimators for the inner problem, where the interchange of limit and expectation is justified by the dominated convergence theorem [21]. Similarly, equation (7) shows that the estimators for the candidate optimization of ρKG are asymptotically unbiased.

References

[1] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, "Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization," *ACM Trans. Math. Softw.*, vol. 23, no. 4, p. 550–560, 1997.

[2] A. J. Kleywegt, A. Shapiro, and T. Homem-de Mello, "The sample average approximation method for stochastic discrete optimization," *SIAM J. on Optimization*, vol. 12, no. 2, pp. 479–502, 2002.

[3] S. Kim, R. Pasupathy, and S. G. Henderson, "A guide to sample average approximation," in *Handbook of Simulation Optimization*, pp. 207–243, Springer New York, 2015.

[4] J. Wilson, F. Hutter, and M. Deisenroth, "Maximizing acquisition functions for Bayesian optimization," in *Advances in Neural Information Processing Systems*, pp. 9884–9895, 2018.

[5] J. Wu and P. I. Frazier, "The parallel knowledge gradient method for batch Bayesian optimization," in *Advances in Neural Information Processing Systems*, pp. 3134–3142, 2016.

[6] I. Murray, "Differentiation of the cholesky decomposition," *arXiv: 1602.07527*, 2016.

[7] M. Balandat, B. Karrer, D. R. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy, "BoTorch: Programmable Bayesian Optimization in PyTorch," *arXiv: 1910.06403*, 2019.

[8] D. Kraft, "A software package for sequential quadratic programming," 1988. Tech. Rep. DFVLR-FB 88-28, DLR German Aerospace Center — Institute for Flight Mechanics, Koln, Germany.

[9] A. B. Owen, "Scrambling Sobol' and Niederreiter–Xing points," *Journal of Complexity*, vol. 14, no. 4, pp. 466 – 489, 1998.

[10] B. Williams, T. Santner, and W. Notz, "Sequential design of computer experiments to minimize integrated response functions," *Statistica Sinica*, vol. 10, pp. 1133–1152, October 2000.

[11] J. Marzat, E. Walter, and H. Piet-Lahanier, "Worst-case global optimization of black-box functions through kriging and relaxation," *Journal of Global Optimization*, vol. 55, pp. 707–727, 04 2013.

[12] J. Wu, M. Poloczek, A. G. Wilson, and P. I. Frazier, "Bayesian optimization with gradients," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, p. 5273–5284, 2017.

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

[14] R. J. Adler and J. E. Taylor, *Random Fields and Geometry*. Springer, 2007.

[15] W. Rudin, *Principles of Mathematical Analysis*. McGraw-Hill Inc, 1976.

[16] G. Jiang and M. C. Fu, "Technical note—on estimating quantile sensitivities via infinitesimal perturbation analysis," *Operations Research*, vol. 63, no. 2, pp. 435–441, 2015.

[17] M. Broadie and P. Glasserman, "Estimating security price derivatives using simulation," *Management Science*, vol. 42, no. 2, pp. 269–285, 1996.

[18] J. Wu and P. I. Frazier, "The parallel knowledge gradient method for batch Bayesian optimization," in *Advances Neural Information Processing Systems*, p. 3134–3142, 2016.

[19] P. Milgrom and I. Segal, "Envelope theorems for arbitrary choice sets," *Econometrica*, vol. 70, no. 2, pp. 583–601, 2002.

[20] L. J. Hong and G. Liu, "Simulating sensitivities of conditional value at risk," *Management Science*, vol. 55, no. 2, pp. 281–293, 2009.

[21] R. Durrett, *Probability: Theory and Examples*. Cambridge University Press, 2010.