Parallel Bayesian Global Optimization of Expensive Functions

Jialei Wang*[1], Scott C. Clark†[2], Eric Liu‡[2], and Peter I. Frazier§[1]

[1]School of Operations Research and Information Engineering, Cornell University
[2]Yelp, Inc., 140 New Montgomery, San Francisco, CA

April 2, 2015

Abstract

We consider parallel global optimization of derivative-free expensive-to-evaluate functions, and proposes an efficient method based on stochastic approximation for implementing a conceptual Bayesian optimization algorithm proposed by [10]. To accomplish this, we use infinitessimal perturbation analysis (IPA) to construct a stochastic gradient estimator and show that this estimator is unbiased.

1 Introduction

We consider derivative-free global optimization of expensive functions, in which (1) our objective function is time-consuming to evaluate, limiting the number of function evaluations we can perform; (2) evaluating the objective function provides only the value of the objective, and not the gradient or Hessian; (3) we seek a global, rather than a local, optimum. Such problems typically arise when the objective function is evaluated by running a complex computer code, but also arises when the objective function can only be evaluated by performing a laboratory experiment, or building a prototype system to be evaluated in the real world. In this paper we assume our function evaluations are deterministic, i.e., free from noise.

Bayesian Global Optimization (BGO) methods constitute one class of methods attempting to solve such problems. These methods were initially proposed in [19], with early work being pursued in [26, 25], and more recent work including improved algorithms (give a bunch of cites), convergence analysis [1, 2, 35], and allowing noisy function evaluations [3, 37, 8, 15].

The most well-known BGO method is Efficient Global Optimization (EGO) [32, 17], which uses the notion of expected improvement. Expected improvement quantifies the benefit gained through one additional evaluation of the objective function, given the set of previously evaluated points. It can be viewed as the value of information [14] obtained from a single function evaluation. To decide where to evaluate next, EGO searches over the set of possible evaluation points, to find the point for which the expected improvement is largest. If the *implementation decision* (the

*jw865@cornell.edu
†sclark@yelp.com
‡eliu@yelp.com
§pf98@cornell.edu

solution that will be implemented in practice after the optimization is complete) is restricted to be a previously evaluated point and evaluations are free from noise, then EGO is a one-step Bayes optimal algorithm.

A number of other BGO methods perform similar value of information calculations. These include probability of improvement [20], upper confidence bounds [27], Thompson sampling [34], knowledge-gradient [6], etc.

Almost all BGO methods, including EGO, are sequential, in that they perform one function evaluation at a time, requiring the results from all previously suggested function evaluations before deciding on the next point to evaluate. This prevents them from taking full advantage of parallel computing architectures, which in principle would allow an algorithm to perform multiple simultaneous function evaluations, reducing the elapsed time required to find an approximate optimium.

An exception is [10], which proposed a generalization of expected improvement appropriate for optimization in parallel settings, called the *q-EI*. This generalization is consistent with the decision-theoretic motivation for expected improvement, and quantifies the expected utility that will result from the evaluation of a *set* of points. Finding the set of points to evaluate next that jointly maximize the q-EI results in a one-step optimal algorithm for global optimization, which can take advantage of the ability to evaluate points in parallel.

However, actually finding the set of points that maximizes the q-EI is itself a very challenging optimization problem. Stymied by this difficulty, [10], as well as the later works [9, 4, 11, 4, 16], propose heuristic methods that are *motivated* by the one-step optimal algorithm of evaluating the set of points that jointly maximize the q-EI, but that do not actually achieve this gold standard.

In addition to these parallel BGO algorithms motivated by maximization of the q-EI, [7, 38] proposed a BGO algorithm that could evaluate two points in parallel. This algorithm, however, is limited to evaluating pairs, and does not extend to a higher level of parallelism.

In this work, we provide a method that makes this gold-standard one-step optimal algorithm implementable. To accomplish this we use infinitessimal perturbation analysis (IPA) [13] to construct a stochastic gradient estimator of the gradient of the q-EI surface, and show that this estimator is unbiased. Our method uses this estimator within a stochastic gradient ascent algorithm, which converges to a stationary point of the q-EI surface [21]. We use multiple restarts to identify multiple stationary points, and then use ranking and selection to identify the best stationary point found. As the number of restarts and the number of iterations of stochastic gradient ascent within each start both grow large, the one-step optimal set of points to evaluate is recovered.

In our numerical experiments, we compare this implementation of the one-step optimal method to previously proposed heuristics, and show that there is substantial benefit to a full implementation of the one-step optimal algorithm. While it is more expensive to compute the set of points to evaluate next, it results in a substantial savings in the number of evaluations required to find a point with a desired quality. When function evaluations are expensive, this results in a substantial reduction in overall time to reach an approximately optimal solution.

We also compare the one-step optimal parallel method to the fully sequentially algorithm EGO algorithm, which is one-step optimal when parallel resources are unavailable, and show that using a one-step optimal parallel method results in a substantial speedup in a wide range of problems.

Finally, we compare our implementation of the maximization of the q-EI using stochastic gradient descent, to an implementation using exact evaluations of the q-EI with a method recently proposed in [4], combined with a standard derivative-free solver. We find that for small values of q ($q < 4$), using exact function evaluations results in faster solve times, but the time required by this alternate method increases rapidly with q, causing it to underperform our proposed stochastic gradient method when q is large ($q > 4$). This is because exact evaluation of the q-EI requires q^2

evaluation of the $q-1$ dimensional multivariate normal cdf, which is computationally expensive for q large, in contrast with our unbiased estimator of the gradient, which can be computed quickly even for large values of q.

Our method can be implemented in both synchronous environments, in which function evaluations are performed in batches and finish at the same time, and asynchronous ones, in which a function evaluation may finish before others are done. High performance computing environments are typically asynchronous, but synchronous environments also occur. We show that our proposed method provides an advantage over previously proposed heuristics in both asynchronous and synchronous settings, but that this advantage is particularly large in synchronous settings.

We begin in Section 2 by precisely describing the mathematical setting in which Bayesian global optimization is performed, and then defining the q-EI and the one-step optimal algorithm. In Section 3 we construct our stochastic gradient, and show that it is an unbiased estimator of the gradient of the q-EI surface under certain mild regularity conditions. In Section 4.2 we combine this estimator together with stochastic gradient ascent to define a one-step optimal method for parallel Bayesian global optimization. Finally, in Section 5 we present numerical experiments: we compare our proposed method against previously proposed heuristics from the literature; we demonstrate that our proposed method provides a speedup over single-threaded EGO; and we show that our proposed method is more efficient than optimizing exact evaluations of the q-EI when q is large.

2 Problem Formulation

In this section, we describe a decision-theoretic approach to Bayesian global optimization in parallel computing environments, previously proposed by [10]. This approach was considered to be purely conceptual in [10], as it contains a difficult-to-solve optimization sub-problem. In this section, we present this optimization sub-problem, and in later sections we show it can be solved efficiently.

2.1 Bayesian Global Optimization

In Bayesian global optimization, one considers optimization of a function f with domain $\mathbb{A} \subseteq \mathbb{R}^d$. Our overarching goal is to find an approximate solution to

$$\min_{\boldsymbol{x} \in \mathbb{A}} f(\boldsymbol{x}).$$

We suppose that evaluating f is expensive or time-consuming, and that these evaluations provide only the value of f at the evaluated point, and not its gradient or Hessian. Such situations occur most typically when f is the output of a complex deterministic simulator. We assume that the function definining the domain $\mathbb{A}$ is easy-to-evaluate, and that projections from $\mathbb{R}^d$ into the nearest point in $\mathbb{A}$ can be performed quickly.

Rather than focusing on asymptotic performance as the number of function evaluations grows large, we wish to find an algorithm that performs well, on average, given a limited budget of function evaluations. To formalize this, we model our prior beliefs on the function f with a Bayesian prior distribution, and we suppose that f was drawn at random by nature from this prior distribution, before any evaluations were performed. We then seek to develop an optimization algorithm that will perform well, on average, when applied to a function drawn at random from this prior.

2.2 Gaussian process priors

For our Bayesian prior distribution on f, we adopt a Gaussian process prior [29], with mean function $m(\boldsymbol{x}) : \mathbb{A} \to \mathbb{R}$ and covariance kernal $k(\boldsymbol{x}, \boldsymbol{x}') : \mathbb{A} \times \mathbb{A} \to \mathbb{R}$, and write the Gaussian process as

$$f(\boldsymbol{x}) \sim \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')).$$

For any specified collection of points $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q)$, prior distribution of f on $\boldsymbol{X}$ is

$$f(\boldsymbol{X}) = [f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_q)]^T \sim \mathcal{N}(\boldsymbol{\mu}^{(0)}, \boldsymbol{\Sigma}^{(0)}), \tag{1}$$

where $\boldsymbol{\mu}_i^{(0)} = m(\boldsymbol{X}_i)$ and $\boldsymbol{\Sigma}_{ij}^{(0)} = k(\boldsymbol{X}_i, \boldsymbol{X}_j), i, j \in \{1, \ldots, q\}$.

Our proposed method for choosing the points to evaluate next additionally require that $\boldsymbol{\mu}^{(0)}$ and $\boldsymbol{\Sigma}^{(0)}$ satisfy some mild regularity assumptions discussed below (beyond just the requirement that $\boldsymbol{\Sigma}^{(0)}$ be positive semi-definite), but otherwise adds no additional requirements on $\boldsymbol{\mu}^{(0)}$ and $\boldsymbol{\Sigma}^{(0)}$. When Bayesian global optimization is used in practice, $\boldsymbol{\mu}^{(0)}$ and $\boldsymbol{\Sigma}^{(0)}$ are typically chosen using an empirical Bayes approach, in which (1) a parameterized functional form for $\boldsymbol{\mu}^{(0)}$ and $\boldsymbol{\Sigma}^{(0)}$ is assumed; (2) a first stage of data is collected in which f is evaluated at points chosen according to a Latin hypercube or uniform design; and (3) maximum likelihood estimates for the parameters specifying $\boldsymbol{\mu}^{(0)}$ and $\boldsymbol{\Sigma}^{(0)}$ are obtained. In some implementations, these estimates are updated iteratively as more evaluations of f are obtained. We adopt this method in our numerical experiments below in section 5, and describe it in more detail there. However, the specific contribution of this paper, a new method for solving an optimization sub-problem arising in the choice of design points, works with any choice of mean function $\boldsymbol{\mu}^{(0)}$ and covariance matrix $\boldsymbol{\Sigma}^{(0)}$, as long as they satisfy mild regularity conditions discussed below.

In addition to the prior distribution (1), we may also have some previously observed function values, say $y^{(i)} = f(\boldsymbol{x}^{(i)})$, for $i = 1, \ldots, n$. These might have been obtained through the previously mentioned first stage of sampling, or running the second stage sampling method we are about to describe, or from some additional runs of the expensive objective function f performed by another party outside of the control of our algorithm. If no additional function values are available, we set $n = 0$. We use boldface in our notation $\boldsymbol{x}^{(i)}$ to indicate that $\boldsymbol{x}^{(i)} \in \mathbb{R}^d$ is a vector. We define notation $\boldsymbol{x}^{(1:n)} = (\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)})$ and $y^{(1:n)} = (y^{(1)}, \ldots, y^{(n)})$.

We then combine these previously observed function values with our prior to obtain a posterior distribution on f. This posterior distribution is still a multivariate normal

$$\begin{aligned} f \mid \boldsymbol{X}, \boldsymbol{x}^{(1:n)}, y^{(1:n)} \sim \mathcal{N}(&K(\boldsymbol{X}, \boldsymbol{x}^{(1:n)}) K(\boldsymbol{x}^{(1:n)}, \boldsymbol{x}^{(1:n)})^{-1} y^{(1:n)}, \\ &K(\boldsymbol{X}, \boldsymbol{X}) - K(\boldsymbol{X}, \boldsymbol{x}^{(1:n)}) K(\boldsymbol{x}^{(1:n)}, \boldsymbol{x}^{(1:n)})^{-1} K(\boldsymbol{x}^{(1:n)}, \boldsymbol{X})), \end{aligned} \tag{2}$$

where $K(\cdot, \cdot)$ is covariance matrix which is typically determined by a specified kernel function [29, Section 2.2].

2.3 Multi-points expected improvement

In a parallel computing environment, we wish to use this posterior distribution to choose the next set of points to evaluate next. [10] previously proposed making this choice using a decision-theoretic approach, in which we consider the utility that evaluating a particular candidate set of points would provide, in terms of their ability to reveal points with better objective function values than were previously known.

Let q be the number of function evaluations that we may perform in parallel, and let $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q)$ be a candidate set of points that we are considering evaluating next. Let $f_n^\star =$

$\min_{m \le n} f(\boldsymbol{x}^{(m)})$ indicate the value of the best point evaluated, before begininning these q new function evaluations. The value of the best point evaluated after all q function evaluations are complete will be $\min\left(f_n^\star, \min_{i=1,\ldots,q} f(\boldsymbol{x}_i)\right)$. The difference between these two values (the values of the best point evaluated, before and after these q new function evaluations) is called the *improvement*, and is equal to $\left(f_n^\star - \min_{i=1,\ldots,q} f(\boldsymbol{x}_i)\right)^+$, where $a^+ = \max(a, 0)$ for $a \in \mathbb{R}$.

We then value a joint set of evaluations at these candidate points $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q)$ as the expected value of this improvement, and we refer to this quantity as the *multi-points expected improvement* [10]. This multi-points expected improvement can be written as,

$$\text{q-EI}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q) = \mathbb{E}_n\left[\left(f_n^\star - \min_{i=1,\ldots,q} f(\boldsymbol{x}_i)\right)^+\right], \tag{3}$$

where $\mathbb{E}_n[\cdot] := \mathbb{E}\left[\cdot | \boldsymbol{x}^{(1:n)}, y^{(1:n)}\right]$ is the expectation taken with respect to the posterior distribution, and given the proposed set of points to evaluate next.

[10] then proposes that we should choose to next evaluate the set of points that maximizes the multi-points expected improvement,

$$\underset{(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_q)\subset\mathbb{A}}{\operatorname{argmax}} \ \text{q-EI}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q). \tag{4}$$

In the special case $q = 1$, which occurs when we are operating without parallelism, the multi-points expected improvement reduces to the expected improvement, as considered by [25, 17], and can be evaluated in closed-form, in terms of the normal pdf and cdf. The algorithm that chooses the next point to evaluate according to (4) is the EGO algorithm of [17]. [10] offered analytical calculation of EI when $q = 2$, but in the same paper Ginsbourger commented that the general case of q-EI has complex expressions depending on q-dimensional gaussian cumulative distribution functions, and computation of q-EI when q is large would have to reply on numerical multivariate integral approximation techniques, which is generally intractable and makes solving (4) difficult. [9] writes "directly optimizing the q-EI becomes extremely expensive as q and d(the dimension of inputs) grow".

In this paper, our main contribution is to present a more efficient method for solving (4). We proceed as follows. First, in Section 3, we construct an unbiased estimator of the gradient of the multi-points expected improvement with respect to $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q)$. Then, in Section 4.2, we show how this stochastic estimator of the gradient can be used in a multistart stochastic gradient algorithm to solve (4). Then, in Section 5, we demonstrate in numerical experiments that (1) the resulting algorithm for parallel Bayesian global optimization provides a significant speedup over the single-threaded expected improvement method EGO; (2) and is faster than both previously proposed heuristic schemes based on multi-points expected improvement, and faster than directly optimizing exact evaluations of (3) as computed via numerical integration.

3 Gradient Estimator

We use stochastic gradient ascent to solve (4), and a natural question is how we obtain the gradient, which is discussed in this section.

Consider random vector $\boldsymbol{Y} = (f(\boldsymbol{x_1}), \ldots, f(\boldsymbol{x_q})) \in \mathbb{R}^q$, generated from the multivariate normal given by (2). Fix $\boldsymbol{x}^{(1:n)}, y^{(1:n)}$, given $\boldsymbol{X} = (\boldsymbol{x_1}, \ldots, \boldsymbol{x_q})$, its posterior distribution is

$$\boldsymbol{Y} | \boldsymbol{X}, \boldsymbol{x}^{(1:n)}, y^{(1:n)} \sim \mathcal{N}\left(\boldsymbol{\mu}(\boldsymbol{X}), \Sigma(\boldsymbol{X})\right) \tag{5}$$

where $\boldsymbol{\mu}(\boldsymbol{X})$ and $\boldsymbol{\Sigma}(\boldsymbol{X})$ can be obtained from (2).

Let $L(\boldsymbol{X})$ be the Cholesky decomposition of $\boldsymbol{\Sigma}(\boldsymbol{X})$ and $\boldsymbol{Z}$ be q-dimensional standard normal random vector, then (5) becomes

$$\begin{aligned} \boldsymbol{Y} &= \boldsymbol{\mu}(\boldsymbol{X}) + \boldsymbol{L}(\boldsymbol{X})\boldsymbol{Z} \\ &= \boldsymbol{\mu}\left(\boldsymbol{x_1}, \ldots, \boldsymbol{x_q}\right) + \boldsymbol{L}\left(\boldsymbol{x_1}, \ldots, \boldsymbol{x_q}\right)\boldsymbol{Z} \end{aligned} \tag{6}$$

Substitute (6) into (3), and we have

$$\text{q-EI}\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q\right) = \mathbb{E}\left[\left(f_n^* - \min_{i=1,\ldots,q} \boldsymbol{e}_i\left[\boldsymbol{\mu}\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q\right) + \boldsymbol{L}\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q\right)\boldsymbol{Z}\right]\right)^+\right] \tag{7}$$

To make (7) more compact, let

$$m_i\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q\right) = \begin{cases} f_n^* - \mu_i\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q\right) & \text{if } i > 0\ , \\ 0 & \text{if } i = 0\ , \end{cases}$$

$$C_{ij}\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q\right) = \begin{cases} -L_{ij} & \text{if } i > 0\ , \\ 0 & \text{if } i = 0\ , \end{cases}$$

and (7) becomes

$$\text{q-EI}\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q\right) = \mathbb{E}\left[\max_{i=0,\ldots,q} \boldsymbol{e}_i\left[\boldsymbol{m}\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q\right) + \boldsymbol{C}\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q\right)\boldsymbol{Z}\right]\right]. \tag{8}$$

3.1 Constructing the Gradient Estimator

We construct the gradient estimator of $\nabla\text{q-EI}\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q\right)$. Let

$$f\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z}\right) = \max_{i=0,\ldots,q} \boldsymbol{e}_i\left[\boldsymbol{m}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q) + \boldsymbol{C}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q)\boldsymbol{Z}\right], \tag{9}$$

then under certain conditions, specified in theorem 1,

$$\nabla\text{q-EI}\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q\right) = \nabla\mathbb{E}f\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z}\right) = \mathbb{E}\boldsymbol{g}\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z}\right),$$

where

$$\boldsymbol{g}\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z}\right) = \begin{cases} \nabla f\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z}\right) & \text{if } \nabla f\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z}\right) \text{exists}, \\ 0 & \text{if does not exist}, \end{cases}$$

is the gradient estimator and exists almost surely if theorem 1 holds, and can be computed using results from [33] on differentiation of the Cholesky decomposition. A sufficient condition for almost sure existence of the gradient is differentiability of mean vector and Cholesky factor of the covariance matrix (see Lemma 2). In practice, we often know that the covariance matrix differentiable, and by following the results of [33], we know that mth-order of differentiability of the covariance matrix implies mth-order differentiability of its Cholesky factor.

3.2 Unbiasedness of the Estimator

The following theorem shows unbiasedness of the estimator constructed in section 3.1, which is typically required in proofs of convergence for stochastic approximation algorithms [22].

Theorem 1. *Under the definition of* (9)*, if* $\boldsymbol{m}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q)$ *and* $\boldsymbol{C}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q)$ *are three times continuously differentiable in a neighborhood of* $\boldsymbol{x_1}, \ldots, \boldsymbol{x_q}$ *and* $\boldsymbol{C}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q)$ *has no duplicate rows, then* $\nabla f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z})$ *exists almost surely and*

$$\nabla \mathbb{E} f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z}) = \mathbb{E} \nabla f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z}).$$

Before we prove Theorem 1, we first show the following two lemmas:

Lemma 1. *If* $h(x)$ *is twice continuously differentiable over* $[-\epsilon, \epsilon]$*, then for a sequence* $(\delta_\ell) \subseteq [-\epsilon, \epsilon]$*,*

$$\sup_\ell |\frac{h(\delta_\ell) - h(0)}{\delta_\ell}| < \infty.$$

Proof. By Taylor's theorem,

$$h(\delta_\ell) = h(0) + h'(0)\delta_\ell + \frac{h''(r_\ell)}{2}\delta_\ell^2,$$

where $|r_\ell| \in [0, |\delta_\ell|]$. Then

$$\begin{aligned}
\sup_\ell |\frac{h(\delta_\ell) - h(0)}{\delta_\ell}| &= \sup_\ell |h'(0) + \frac{h''(r_\ell)}{2}\delta_\ell|, \\
&\leq |h'(0)| + \sup_\ell |\frac{h''(r_\ell)}{2}\delta_\ell| \quad \text{(by triangular inequality)}.
\end{aligned}$$

Because $\delta_\ell \in [-\epsilon, \epsilon], r_\ell \in [-\epsilon, \epsilon]$, and $h''(\cdot)$ is continuous over $[-\epsilon, \epsilon]$,

$$\begin{aligned}
\sup_\ell |\frac{h''(r_\ell)}{2}\delta_\ell| &\leq \sup_\ell \epsilon |\frac{h''(r_\ell)}{2}|, \\
&< \infty.
\end{aligned}$$

□

Lemma 2. *If* $\boldsymbol{m}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q)$ *and* $\boldsymbol{C}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q)$ *are differentiable in a neighborhood of* $\boldsymbol{x_1}, \ldots, \boldsymbol{x_q}$*, and there are no duplicated rows in* $\boldsymbol{C}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q)$*, then* $\nabla f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z})$ *exists almost surely.*

Proof. Let

$$f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z}) = e_{I^*}[\boldsymbol{m}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q) + \boldsymbol{C}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q)\boldsymbol{Z}],$$

where $I^* \in \operatorname{argmax}_{i=0,\ldots,q} \boldsymbol{e}_i[\boldsymbol{m}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q) + \boldsymbol{C}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q)\boldsymbol{Z}] := \mathcal{S}$ and e_i is the unit vector. To simplify notations, let

$$\begin{aligned}
\boldsymbol{m} &:= \boldsymbol{m}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q), \\
\boldsymbol{C} &:= \boldsymbol{C}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q), \\
\nabla f &:= \nabla f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z}).
\end{aligned}$$

We claim that if $\boldsymbol{e}_I(\frac{\partial \boldsymbol{m}}{\partial x_{ik}} + \frac{\partial \boldsymbol{C}}{\partial x_{ik}} Z)$ are equal $\forall I \in \mathcal{S}$, and $\forall i, k$ (k iterate over dimension), then ∇f exists.

$$\begin{aligned}
\mathbb{P}(\nabla f \text{ does not exist}) &\leq \mathbb{P}(|\mathcal{S}| \geq 2), \\
&\leq \sum_{i \neq j} \mathbb{P}(\boldsymbol{e}_i[\boldsymbol{m} + \boldsymbol{C}\boldsymbol{Z}] = \boldsymbol{e}_j[\boldsymbol{m} + \boldsymbol{C}\boldsymbol{Z}]), \\
&= \sum_{i \neq j} \mathbb{P}((\boldsymbol{C}_i - \boldsymbol{C}_j)^T \boldsymbol{Z} = m_j - m_i).
\end{aligned}$$

Since $\boldsymbol{C}_i \neq \boldsymbol{C}_j$, $\{\boldsymbol{Z} : (\boldsymbol{C}_i - \boldsymbol{C}_j)^T \boldsymbol{Z} = m_j - m_i\}$ is subspace of $\mathbb{R}^q$ with dimension smaller than q, thus

$$\mathbb{P}((\boldsymbol{C}_i - \boldsymbol{C}_j)^T \boldsymbol{Z} = m_j - m_i) = 0 \quad \forall i \neq j.$$

Hence

$$\mathbb{P}(\nabla f \text{ does not exist}) \leq 0 = 0.$$

□

Now we proceed to proving Theorem 1.

Proof. First we define some notation for ease. Fix $(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_q)$, and let

$$\begin{aligned} f(\delta, \boldsymbol{Z}) &:= f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_j + \delta \boldsymbol{e}_k, \ldots, \boldsymbol{x}_q, \boldsymbol{Z}), \\ \boldsymbol{m}(\delta) &:= \boldsymbol{m}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_j + \delta \boldsymbol{e}_k, \ldots, \boldsymbol{x}_q), \\ \boldsymbol{C}(\delta) &:= \boldsymbol{C}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_j + \delta \boldsymbol{e}_k, \ldots, \boldsymbol{x}_q), \\ I^*_{(\delta, \boldsymbol{Z})} &:= \min \left(\underset{i=0,\ldots,q}{\operatorname{argmax}} \boldsymbol{e}_i \left[\boldsymbol{m}(\delta) + \boldsymbol{C}(\delta) \boldsymbol{Z} \right] \right). \end{aligned}$$

Then

$$f(\delta, \boldsymbol{Z}) = \max_{i=0,\ldots,q} \boldsymbol{e}_i \left[\boldsymbol{m}(\delta) + \boldsymbol{C}(\delta) \boldsymbol{Z} \right] = \boldsymbol{e}_{I^*_{(\delta, \boldsymbol{Z})}} \left[\boldsymbol{m}(\delta) + \boldsymbol{C}(\delta) \boldsymbol{Z} \right],$$

where e_i is the unit vector in direction i. Define

$$\begin{aligned} \triangle(\delta, \boldsymbol{Z}) :&= f(\delta, \boldsymbol{Z}) - f(0, \boldsymbol{Z}) \\ &= \boldsymbol{e}_{I^*_{(\delta, \boldsymbol{Z})}} \left[\boldsymbol{m}(\delta) + \boldsymbol{C}(\delta) \boldsymbol{Z} \right] - \boldsymbol{e}_{I^*_{(0, \boldsymbol{Z})}} \left[\boldsymbol{m}(0) + \boldsymbol{C}(0) \boldsymbol{Z} \right]. \end{aligned}$$

Let $\epsilon > 0$. Consider a sequence $(\delta_\ell) \subseteq [-\epsilon, \epsilon]$ and $\delta_\ell \searrow 0$ as $\ell \to \infty$. We want to show $\lim_{\ell \to \infty} \frac{\triangle(\delta_\ell, \boldsymbol{Z})}{\delta_\ell}$ exists almost surely, and

$$\lim_{\ell \to \infty} \mathbb{E} \left[\frac{\triangle(\delta_\ell, \boldsymbol{Z})}{\delta_\ell} \right] = \mathbb{E} \left[\lim_{\ell \to \infty} \frac{\triangle(\delta_\ell, \boldsymbol{Z})}{\delta_\ell} \right].$$

As a first step we show that $\sup_\ell \left| \frac{\triangle(\delta_\ell, \boldsymbol{Z})}{\delta_\ell} \right|$ is bounded. For any δ in the sequence (δ_ℓ), we consider 2 cases,

- Case 1: If $\boldsymbol{e}_{I^*_{(\delta, \boldsymbol{Z})}} \left[\boldsymbol{m}(\delta) + \boldsymbol{C}(\delta) \boldsymbol{Z} \right] \geq \boldsymbol{e}_{I^*_{(0, \boldsymbol{Z})}} \left[\boldsymbol{m}(0) + \boldsymbol{C}(0) \boldsymbol{Z} \right]$, then

$$\begin{aligned} |\triangle(\delta, \boldsymbol{Z})| &= \boldsymbol{e}_{I^*_{(\delta, \boldsymbol{Z})}} \left[\boldsymbol{m}(\delta) + \boldsymbol{C}(\delta) \boldsymbol{Z} \right] - \boldsymbol{e}_{I^*_{(0, \boldsymbol{Z})}} \left[\boldsymbol{m}(0) + \boldsymbol{C}(0) \boldsymbol{Z} \right], \\ &\leq \boldsymbol{e}_{I^*_{(\delta, \boldsymbol{Z})}} \left[\boldsymbol{m}(\delta) + \boldsymbol{C}(\delta) \boldsymbol{Z} \right] - \boldsymbol{e}_{I^*_{(\delta, \boldsymbol{Z})}} \left[\boldsymbol{m}(0) + \boldsymbol{C}(0) \boldsymbol{Z} \right], \\ &= \boldsymbol{e}_{I^*_{(\delta, \boldsymbol{Z})}} \left[\boldsymbol{m}(\delta) - \boldsymbol{m}(0) + \boldsymbol{C}(\delta) \boldsymbol{Z} - C(0) \boldsymbol{Z} \right], \\ &\leq \left| \boldsymbol{e}_{I^*_{(\delta, \boldsymbol{Z})}} \left[\boldsymbol{m}(\delta) - \boldsymbol{m}(0) + \boldsymbol{C}(\delta) \boldsymbol{Z} - C(0) \boldsymbol{Z} \right] \right|. \end{aligned}$$

- Case 2: If $\boldsymbol{e}_{I^*_{(\delta, \boldsymbol{Z})}} \left[\boldsymbol{m}(\delta) + \boldsymbol{C}(\delta) \boldsymbol{Z} \right] \leq \boldsymbol{e}_{I^*_{(0, \boldsymbol{Z})}} \left[\boldsymbol{m}(0) + \boldsymbol{C}(0) \boldsymbol{Z} \right]$, then

$$|\triangle(\delta, \boldsymbol{Z})| \leq \left| \boldsymbol{e}_{I^*_{(0, \boldsymbol{Z})}} \left[\boldsymbol{m}(\delta) - \boldsymbol{m}(0) + \boldsymbol{C}(\delta) \boldsymbol{Z} - C(0) \boldsymbol{Z} \right] \right|$$

  by a similar argument.

In general, we have shown that,

$$|\triangle(\delta, \boldsymbol{Z})| \leq \sum_{i=1}^{q} |\boldsymbol{e}_i \left[\boldsymbol{m}(\delta) - \boldsymbol{m}(0) + \boldsymbol{C}(\delta)\boldsymbol{Z} - C(0)\boldsymbol{Z}\right]|,$$

so

$$\begin{aligned}\sup_{\ell} \left|\frac{\triangle(\delta_\ell, \boldsymbol{Z})}{\delta_\ell}\right| &\leq \sum_{i=1}^{q} \sup_{\ell} \left|\frac{\boldsymbol{e}_i \left[\boldsymbol{m}(\delta_\ell) - \boldsymbol{m}(0)\right]}{\delta_\ell} + \frac{\boldsymbol{e}_i \left[\left(\boldsymbol{C}(\delta_\ell) - \boldsymbol{C}(0)\right)\boldsymbol{Z}\right]}{\delta_\ell}\right|, \\ &\leq \sum_{i=1}^{q} \sup_{\ell} \left|\frac{\left[m_i(\delta_\ell) - m_i(0)\right]}{\delta_\ell}\right| + \sup_{\ell} \left|\frac{C_i(\delta_\ell) - C_i(0)}{\delta_\ell}\boldsymbol{Z}\right|,\end{aligned}$$

where $m_i(\delta) = \boldsymbol{e}_i \boldsymbol{m}(\delta)$ is a scalar and $C_i(\delta) = \boldsymbol{e}_i \boldsymbol{C}(\delta)$ is a row vector.

We know $m_i(\cdot)$ and $C_i(\cdot)$ are three times continuously differentiable over $[\inf_l \delta_l, \sup_l \delta_l] \subseteq [-\epsilon, \epsilon]$. Let

$$\begin{aligned}v_i &= \sup_{\ell} \left|\frac{\partial m_i}{\partial \delta}\Big|_{\delta=\delta_\ell}\right|, \\ V_{ij} &= \sup_{\ell} \left|\frac{\partial C_{ij}}{\partial \delta}\Big|_{\delta=\delta_\ell}\right|.\end{aligned}$$

Since $(\delta_l) \subseteq [-\epsilon, \epsilon]$ and $\frac{\partial m_i}{\partial \delta}$, $\frac{\partial C_{ij}}{\partial \delta}$ are twice continuously differentiable over $[-\epsilon, \epsilon]$, by Lemma 1 we have $v_i < \infty$, $V_{ij} < \infty$ $\forall i, j$.

Then

$$\sup_{\ell} \left|\frac{\triangle(\delta_\ell, \boldsymbol{Z})}{\delta_\ell}\right| \leq \sum_{i=1}^{q} \left(v_i + \sum_{j=1}^{d} V_{ij} |Z_j|\right) =: M(\boldsymbol{Z}),$$

and

$$\mathbb{E}\left[M(\boldsymbol{Z})\right] = \sum_{i=1}^{q} v_i + \sum_{i=1}^{q}\sum_{i=1}^{d} V_{ij}\mathbb{E}|Z_j| < \infty.$$

Thus, $M(\boldsymbol{Z})$ is integrable. Also $\lim_{\ell\to\infty} \frac{\triangle(\delta_\ell, Z)}{\delta_\ell} = \frac{\partial f(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_q,\boldsymbol{Z})}{\partial x_{ik}}$ exists almost surely by Lemma 2. Then by the Dominated Convergence Theorem [[30]],

$$\lim_{\ell\to\infty} \mathbb{E}\left[\frac{\triangle(\delta_\ell, \boldsymbol{Z})}{\delta_\ell}\right] = \mathbb{E}\left[\lim_{\ell\to\infty} \frac{\triangle(\delta_\ell, \boldsymbol{Z})}{\delta_\ell}\right]. \tag{10}$$

Since $\delta_\ell \searrow 0$ as $\ell \to \infty$, (10) becomes

$$\frac{\partial \mathbb{E} f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z})}{\partial x_{ik}} = \mathbb{E}\frac{\partial f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z})}{\partial x_{ik}}, \tag{11}$$

where $x_{ik} = \boldsymbol{e}_k \boldsymbol{x}_i$, and (11) applies to any i, k. Thus $\nabla f\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z}\right)$ exists almost surely, and

$$\nabla \mathbb{E} f\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z}\right) = \mathbb{E}\nabla f\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q, \boldsymbol{Z}\right).$$

□

The proof used Infinitesimal Perturbation Analysis (IPA), and is similar to the proof of a theorem of unbiased stochastic derivatives (see [12, p.14] or [23]). However, the result shown in both Glasserman and L'Ecuyer required countability of the set of non-differentiability. In our case, this coutability requirement is not met. Instead, the set of non-differentiability is uncountable, but with measure zero. Our proof adapts to this novel situation.

4 Optimization of q-EI

If theorem 1 holds, we can simply use sample average to estimate gradient of q-EI. Let $\boldsymbol{G}_n$ be the estimate of gradient of q-EI at $\boldsymbol{X_n} = (\boldsymbol{x}_{n1}, \ldots, \boldsymbol{x}_{nq})$, then

$$\boldsymbol{G}_n = \frac{1}{M} \sum_{m=1}^{M} \boldsymbol{g}(\boldsymbol{X}_n, \boldsymbol{Z}_m), \tag{12}$$

where M is the number of samples to generate for the estimation and the expression of $\boldsymbol{g}(\boldsymbol{X}_n, \boldsymbol{Z}_m)$ is provided in section 3.1. The stochastic gradient ascent algorithm is to begin with some $\boldsymbol{X}_0$, and use a predetermined sequence $\{\epsilon_n : n = 0, 1, \ldots\}$ to generate a sequence $\{\boldsymbol{X}_n : n = 1, 2, \ldots\}$ using

$$\boldsymbol{X}_{n+1} = \boldsymbol{X}_n + \epsilon_n \boldsymbol{G}_n, \tag{13}$$

and hope the sequence converges to some stationary point $\boldsymbol{X}^*$. In 4.1 we show that under certain conditions, the algorithm converges to a stationary point almost surely. In 4.2 we propose an algorithm that starts from multiple points in the search space and runs stochastic gradient ascent for each starting point, and we also provide the pseudo code.

4.1 Convergence Analysis

Let's consider (13) operating on a compact space $H = \{\boldsymbol{X} : a_i(\boldsymbol{X}) \leq 0, i = 1, \ldots, p\} \subseteq \mathbb{R}^{d \times q}$, where $a_i(\cdot)$ can be any real-valued constraint function. Then (13) becomes

$$\boldsymbol{X}_{n+1} = \prod_{H} \left[\boldsymbol{X}_n + \epsilon_n \boldsymbol{G}_n \right], \tag{14}$$

where $\prod_H(\boldsymbol{X})$ denotes the closest point in H to $\boldsymbol{X}$, and if the closest point is not unique, select a closest point such that the function $\prod_H(\cdot)$ is measurable. In the following theorem, we show that under certain condition, (14) converges to a stationary point almost surely.

Theorem 2. *If the following assumptions hold,*

1. *$a_i(\cdot), i = 1, \ldots, p$ are twice continuously differentiable.*
2. *$\epsilon_n \to 0$ for $n \geq 0$ and $\epsilon_n = 0$ for $n < 0$; $\sum_{n=1}^{\infty} \epsilon_n = \infty$ and $\sum_{n=0}^{\infty} \epsilon_n^2 < \infty$.*
3. *$\forall \boldsymbol{X} \in H$, under the definiton of (9), $\boldsymbol{m}(\boldsymbol{X})$ and $\boldsymbol{C}(\boldsymbol{X})$ are three times continuously differentiable and $\boldsymbol{C}(\boldsymbol{X})$ does not have duplicate rows.*

Then the sequence $\{\boldsymbol{X}_n : n = 0, 1, \ldots\}$ generated by algorithm (14) converges to a stationary point almost surely.

Proof. We use convergence analysis result from [22, theorem 5.2.3] to prove our theorem. First let me state [22, theorem 5.2.3] using our notation and setting: if the following assumptions hold for algorithm (14),

1. $\epsilon_n \to 0$ for $n \geq 0$ and $\epsilon_n = 0$ for $n < 0$; $\sum_{n=1}^{\infty} \epsilon_n = \infty$
2. $\sup_n \mathbb{E}|\boldsymbol{G}_n|^2 < \infty$

3. There are functions $h_n(\cdot)$ of $\boldsymbol{X}$, which are continuous uniformly in n, a continuous function $\overline{h}(\cdot)$ and random variables β_n such that

$$\mathbb{E}_n \boldsymbol{G}_n = h_n(\boldsymbol{X}_n) + \beta_n,$$

and for each $\boldsymbol{X} \in H$,

$$\lim_n | \sum_{i=n}^{m(t+n+t)} \epsilon_i [h_i(\boldsymbol{X}) - \overline{h}(\boldsymbol{X})]| \to 0$$

for each $t > 0$, and $\beta_n \to 0$ with probability one.

4. $\sum_i \epsilon_i^2 < \infty$.

5. There exists a twice continuously differentiable real-valued $f(\cdot)$, and $\overline{h}(\cdot) = -f_{\boldsymbol{X}}(\cdot)$.

Then $\{\boldsymbol{X}_n\}$ converges to a stationary point almost surely. We show these conditions are all satisfied one by one and therefore this convergence analysis applies.

1. Condition 1 is statisfied by the assumption in theorem 2, and in section 4.2, construction of this sequence will be discussed.

2. Without loss of generality, let $M = 1$, then $\boldsymbol{G}_n = \boldsymbol{g}(\boldsymbol{X}_n, \boldsymbol{Z})$.

$$\begin{aligned}
\mathbb{E}|\boldsymbol{G}_n|^2 &= \mathbb{E} \sum_{k=1}^{dq} G_{nk}^2 \\
&= \sum_{k=1}^{dq} \mathbb{E} \left(\frac{\partial f(\boldsymbol{X}_n, \boldsymbol{Z})}{\partial X_{nk}} \right)^2 \\
&= \sum_{k=1}^{dq} \mathbb{E} \left[\left(\frac{\partial \boldsymbol{m}(\boldsymbol{X}_n)}{\partial X_{nk}} + \frac{\partial \boldsymbol{C}(\boldsymbol{X_n})}{\partial X_{nk}} \boldsymbol{Z} \right) \boldsymbol{e}_{I_{\boldsymbol{Z}}^*} \right]^2 \\
&\leq \sum_{k=1}^{dq} \mathbb{E} \sum_{i=1}^{q} \left[\left(\frac{\partial \boldsymbol{m}(\boldsymbol{X}_n)}{\partial X_{nk}} + \frac{\partial \boldsymbol{C}(\boldsymbol{X_n})}{\partial X_{nk}} \boldsymbol{Z} \right) \boldsymbol{e}_i \right]^2 \\
&= \sum_{k=1}^{dq} \sum_{i=1}^{q} \mathbb{E} \left[\left(\frac{\partial \boldsymbol{m}(\boldsymbol{X}_n)}{\partial X_{nk}} + \frac{\partial \boldsymbol{C}(\boldsymbol{X_n})}{\partial X_{nk}} \boldsymbol{Z} \right) \boldsymbol{e}_i \right]^2
\end{aligned}$$

where $I_{\boldsymbol{Z}}^* = \underset{i=1,\ldots,q}{\arg\max} \left(\boldsymbol{m}(\boldsymbol{X_n}) + \boldsymbol{C}(\boldsymbol{X}_n)\boldsymbol{Z} \right) \boldsymbol{e}_i$. Since $\boldsymbol{m}(\boldsymbol{X})$ and $\boldsymbol{C}(\boldsymbol{X})$ are continously differentiable for $\forall \boldsymbol{X} \in H$, then $\sup_n \frac{\partial \boldsymbol{m}(\boldsymbol{X}_n)}{\partial X_{nk}} < \infty$ and $\frac{\partial \boldsymbol{C}(\boldsymbol{X_n})}{\partial X_{nk}} < \infty$. It is easy to see that $\sup_n \mathbb{E} \left[\left(\frac{\partial \boldsymbol{m}(\boldsymbol{X}_n)}{\partial X_{nk}} + \frac{\partial \boldsymbol{C}(\boldsymbol{X_n})}{\partial X_{nk}} \boldsymbol{Z} \right) \boldsymbol{e}_i \right]^2 < \infty$, thus $\sup_n \mathbb{E}|\boldsymbol{G}_n|^2 < \infty$. Therefore, condtion 2 is satisfied.

3. Since evaluation of $\boldsymbol{G}_n$ in (12) does not depend on previous points in the sequence $\{\boldsymbol{X}_n\}$, $\mathbb{E}_n \boldsymbol{G}_n = \mathbb{E}\boldsymbol{G}_n = \mathbb{E}\boldsymbol{g}(\boldsymbol{X}_n, \boldsymbol{Z})$. From the assumptions, we know Theorem 1 holds, then define a function $\overline{\boldsymbol{g}}(\cdot)$ on H, such that $\overline{\boldsymbol{g}}(\boldsymbol{X}) = \mathbb{E}\boldsymbol{g}(\boldsymbol{X}, \boldsymbol{Z}) = \nabla \mathbb{E} f(\boldsymbol{X}, \boldsymbol{Z})$. We want to show $\overline{\boldsymbol{g}}(\boldsymbol{X})$ is continuous on H.

Without loss of generality, we only look at purturbation on kth dimension of $\boldsymbol{X}$. Fix $\boldsymbol{X}$, let purturbation be δ, and only look at jth component of $\overline{\boldsymbol{g}}(\cdot)$. Let's first define some notations, let

$$\begin{aligned}
\frac{\partial \boldsymbol{m}(\delta)}{\partial X_j} &:= \frac{\partial \boldsymbol{m}(\boldsymbol{X}+\delta \boldsymbol{e}_k)}{\partial X_j}, \\
\frac{\partial \boldsymbol{C}(\delta)}{\partial X_j} &:= \frac{\partial \boldsymbol{C}(\boldsymbol{X}+\delta \boldsymbol{e}_k)}{\partial X_j}, \\
I^*_{(\delta,\boldsymbol{Z})} &:= \min\left(\underset{i=0,\ldots,q}{\operatorname{argmax}} \boldsymbol{e}_i \left[\boldsymbol{m}(\delta)+\boldsymbol{C}(\delta)\boldsymbol{Z}\right]\right).
\end{aligned}$$

then

$$\begin{aligned}
\Delta(\delta) &= \left[\overline{\boldsymbol{g}}(\boldsymbol{X}+\delta \boldsymbol{e}_k)-\overline{\boldsymbol{g}}(\boldsymbol{X})\right]\boldsymbol{e}_j \\
&= \mathbb{E}\left[\nabla f(\boldsymbol{X}+\delta \boldsymbol{e}_k,\boldsymbol{Z})-\nabla f(\boldsymbol{X},\boldsymbol{Z})\right]\boldsymbol{e}_j \\
&= \mathbb{E}\left[\left[\frac{\partial \boldsymbol{m}(\delta)}{\partial X_j}+\frac{\partial \boldsymbol{C}(\delta)}{\partial X_j}\boldsymbol{Z}\right]\boldsymbol{e}_{I^*_{(\delta,\boldsymbol{Z})}}-\left[\frac{\partial \boldsymbol{m}(0)}{\partial X_j}+\frac{\partial \boldsymbol{C}(0)}{\partial X_j}\boldsymbol{Z}\right]\boldsymbol{e}_{I^*_{(0,\boldsymbol{Z})}}\right]
\end{aligned} \tag{15}$$

Using similar argument as in the proof of Theorem 1, we can show

$$\begin{aligned}
&\left|\left[\frac{\partial \boldsymbol{m}(\delta)}{\partial X_j}+\frac{\partial \boldsymbol{C}(\delta)}{\partial X_j}\boldsymbol{Z}\right]\boldsymbol{e}_{I^*_{(\delta,\boldsymbol{Z})}}-\left[\frac{\partial \boldsymbol{m}(0)}{\partial X_j}+\frac{\partial \boldsymbol{C}(0)}{\partial X_j}\boldsymbol{Z}\right]\boldsymbol{e}_{I^*_{(0,\boldsymbol{Z})}}\right| \\
&\leq \left|\left[\frac{\partial \boldsymbol{m}(\delta)}{\partial X_j}-\frac{\partial \boldsymbol{m}(0)}{\partial X_j}+\frac{\partial \boldsymbol{C}(\delta)}{\partial X_j}\boldsymbol{Z}-\frac{\partial \boldsymbol{C}(0)}{\partial X_j}\boldsymbol{Z}\right]\boldsymbol{e}_{I*_{(0,\boldsymbol{Z})}}\right| \\
&\leq \sum_{i=1}^{q}\left|\left[\boldsymbol{\delta}^m+\boldsymbol{\delta}^C\boldsymbol{Z}\right]\boldsymbol{e}_i\right|
\end{aligned} \tag{16}$$

where $\boldsymbol{\delta}^m = \frac{\partial \boldsymbol{m}(\delta)}{\partial X_j}-\frac{\partial \boldsymbol{m}(0)}{\partial X_j}$ and $\boldsymbol{\delta}^C = \frac{\partial \boldsymbol{C}(\delta)}{\partial X_j}-\frac{\partial \boldsymbol{C}(0)}{\partial X_j}$. Then

$$\Delta(\delta) \leq \sum_{i=1}^{q}\mathbb{E}\left|\boldsymbol{\delta}^m\boldsymbol{e}_i\right|+\mathbb{E}\left|\boldsymbol{\delta}^C\boldsymbol{Z}\boldsymbol{e}_i\right|. \tag{17}$$

Because $\frac{\partial \boldsymbol{m}(\cdot)}{\partial X_j}$ and $\frac{\partial \boldsymbol{C}(\cdot)}{\partial X_j}$ are continuous, then $\forall|\delta|<\xi$, $|\boldsymbol{\delta}^m|<\boldsymbol{\epsilon}^m$, and $\left|\boldsymbol{\delta}^C\right|<\boldsymbol{\epsilon}^C$.

$$\begin{aligned}
\mathbb{E}\left|\boldsymbol{\delta}^C\boldsymbol{Z}\boldsymbol{e}_i\right| &= \mathbb{E}\left|\sum_{j=1}^{q}\boldsymbol{\delta}^C_{ij}Z_j\right| \\
&\leq \sum_{j=1}^{q}\mathbb{E}\left|\boldsymbol{\delta}^C_{ij}Z_j\right| \\
&\leq \sum_{j=1}^{q}\mathbb{E}\left|\boldsymbol{\delta}^C_{ij}\right||Z_j| \\
&< \sum_{j=1}^{q}\boldsymbol{\epsilon}^C\mathbb{E}\left|Z_j\right|
\end{aligned} \tag{18}$$

Thus

$$\begin{aligned}\Delta(\delta) &< \sum_{i=1}^{q} \boldsymbol{\epsilon}_i^m + \sum_{i=1}^{q}\sum_{j=1}^{q} \boldsymbol{\epsilon}_{ij}^C \mathbb{E}|Z_j| \\ &= \sum_{i=1}^{q} \boldsymbol{\epsilon}_i^m + \sqrt{\frac{2}{\pi}} \sum_{i=1}^{q}\sum_{j=1}^{q} \boldsymbol{\epsilon}_{ij}^C\end{aligned} \tag{19}$$

Thus $\overline{\boldsymbol{g}}(\boldsymbol{X})$ is continuous on H. Let $h_n(\cdot) \equiv \overline{h}(\cdot) \equiv \overline{\boldsymbol{g}}(\cdot)$, and $\beta_n = 0$, then condition 3 is satisfied.

4. Condition 4 is satisfied by the assumption in Theorem 2.

5. From the proof of condition 3, we know $\overline{h}(\cdot)$ is $\overline{\boldsymbol{g}}(\cdot)$, and thus $f(\cdot)$ is just $-$q-EI. Since $\overline{\boldsymbol{g}}(\cdot)$ is continuously differentialble, $f(\cdot)$ is twice continuously differentiable. Therefore, condition 5 is satisfied.

In conclusion, all conditions are satisfied and therefore $\{\boldsymbol{X}_n\}$ converges to a stationary point almost surely. □

4.2 Multistart Stochastic Gradient Ascent

We use multistart stochastic gradient ascent to find solution to (4). For each start within this multistart framework, we draw an initial point from a Latin hypercube design, and then iteratively update this point using the stochastic gradient and a Polyak-Ruppert stepsize [31, 28], until a convergence criterion is met. We then select the best among the points found by each start.

This algorithm is summarized below.

Algorithm 1 Multistart Stochastic Gradient Ascent

Require: number of initial solutions R; Polyak-Ruppert stepsize constants a and γ; maximum number of steps for gradient ascent T; thresholds $\epsilon > 0$; number of Monte Carlo samples used for estimating the gradient M; number of Monte Carlo samples used for estimating q-EI N.

1: Draw R initial solutions from a Latin hypercube design in $\mathbb{A}^q$, $\boldsymbol{X}_i = (\boldsymbol{x}_{i1}, \ldots, \boldsymbol{x}_{iq})$ for $i = 1, \ldots, R$.
2: **for** $r = 1$ to R **do**
3: **for** $t = 1$ to T **do**
4: Calculate $\boldsymbol{g}^{(t)} = \frac{1}{M}\sum_{m=1}^{M} \boldsymbol{g}(\boldsymbol{x}_{r1}, \ldots, \boldsymbol{x}_{rq}, \boldsymbol{Z}^{(m)})$ where $\boldsymbol{Z}^{(m)}$ are iid standard normal random vectors.
5: Update solution using stochastic gradient ascent $\boldsymbol{X}_r^{(t+1)} = \boldsymbol{X}_r^{(t)} + \frac{a}{t^\gamma}\boldsymbol{g}^{(t)}$.
6: **if** $|\boldsymbol{X}_r^{(t+1)} - \boldsymbol{X}_r^{(t)}| < \epsilon$ **then**
7: Go to *Step 10*
8: **end if**
9: **end for**
10: Average the solutions of $\boldsymbol{X}_r$ to obtain $\overline{\boldsymbol{X}_r} = \frac{1}{T}\sum_{i=1}^{t} \boldsymbol{X}_r$.
11: Estimate q-EI($\overline{\boldsymbol{X}_r}$) using Monte Carlo with N iid samples. Store the estimate as $\widehat{\text{q-EI}}_r$.
12: **end for**
13: **return** $\overline{\boldsymbol{X}_I}$ where $I = \text{argmax}_{i=1,\ldots,r} \widehat{\text{q-EI}}_r$.

Additionally, in our implementation of this algorithm, we supply optional fallback logic. This fallback logic takes two additional parameters: a strictly positive real number ϵ', and an integer

L. If $\max_{i=1,\dots,r} \widehat{\text{q-EI}}_r \leq \epsilon'$, so that multistart stochastic gradient ascent failed to find a point with estimated expected improvement better than ϵ', then we generate L additional solutions from a Latin Hypercube on $\mathbb{A}^q$, estimate the expected improvement at each of these using the same Monte Carlo approach as in Step 11, and select the one with the largest estimated expected improvement.

4.3 Asynchronous Parallel Optimization

(4) is synchronous parallel optimization, i.e., generating a new batch of q points until all previous points to sample have been evaluated. However, there is situation when we need to generate a new batch while there are p points still sampling and we do not have their values yet. This is very common in scientific experiments (biology for example) or expensive computer simulations, where parallel experiments or simulations do not necessarily finish at the same time, and people want the resources freed up from finished experiments to take new jobs. We can extend (4) to asynchronous parallel optimization: suppose $\boldsymbol{x}'_1, \dots, \boldsymbol{x}'_p$ are the points still under evaluation, to generate a new batch $\boldsymbol{x}_1, \dots, \boldsymbol{x}_q$, we want to maximize q-EI$(\boldsymbol{x}_1, \dots, \boldsymbol{x}_q, \boldsymbol{x}'_1, \dots, \boldsymbol{x}'_p)$ while holding $\boldsymbol{x}'_1, \dots, \boldsymbol{x}'_p$ fixed:

$$\underset{(\boldsymbol{x}_1,\dots,\boldsymbol{x}_q)\subset\mathbb{A}}{\operatorname{argmax}} \text{ q-EI}(\boldsymbol{x}_1, \dots, \boldsymbol{x}_q, \boldsymbol{x}'_1, \dots, \boldsymbol{x}'_p). \tag{20}$$

To solve (20), we still estimate ∇q-EI$(\boldsymbol{x}_1, \dots, \boldsymbol{x}_q, \boldsymbol{x}'_1, \dots, \boldsymbol{x}'_p)$, but set its components corresponding to $\boldsymbol{x}'_1, \dots, \boldsymbol{x}'_p$ to zero, then proceed the same way as stated in Algorithm 1.

5 Numerical results

In this section, we present a number of numerical experiments demonstrating the usefulness of our proposed method.

In Section 5.1 we show that our proposed method provides a nearly linear speedup over single-threaded EGO. In Section 5.2 we compare our proposed method against previously proposed heuristics from the literature. In Section 5.3 we show that our proposed method is more efficient than optimizing exact evaluations of the q-EI when q is large.

Although (3) is defined by assuming function evaluation is noise free, in numerical experiments, the covariance matrix $K(\cdot,\cdot)$ in (2) could be ill conditioned. To resolve this problem, we manually impose a small noise $\sim \mathcal{N}(0, \sigma^2)$ where $\sigma^2 = 10^{-4}$ and use the noisy version of Gaussian Process model, which is almost indentical to (2), except that $K(\boldsymbol{x}^{(1:n)}, \boldsymbol{x}^{(1:n)})$ is replaced by $K(\boldsymbol{x}^{(1:n)}, \boldsymbol{x}^{(1:n)}) + \sigma^2 I_n$ where I_n is identity matrix [29, Section 2.2]. [36] performed numerical experiments on random functions generated from sample paths of a Gaussian Process to compare noisy version of Expected Improvement (EIm [36], AEI [15]), noise free version of EI [17] and their own proposed algorithm LAGO [36], which is different from the class of EI algorithms, and their results showed that noise free EI did not lose much performance against noisy version of EI. Thus we think despite that adding a small noise in our numerical experiments violates the noise free assumption in our algorithm, the performance will not be afffected much.

5.1 Parallel Speedup vs. EGO

We show the effect of parallel speedup versus sequential EGO by running numerical experiments on two standard test functions, Branin and Hartmann $H_{6,4}$. Branin has domain in 2-dimensional space and global minimum 0.397887; Hartmann $H_{6,4}$ has domain in 6-dimensional space and global minimum -3.32237. Before running EI algorithm, we randomly sample $n = 10d$ points in the domain, where d is dimension of domain, and estimate hyperparameters for the metamodel. Then

we run q-EI algorithm with $q = 1, 2, 4$, where $q = 1$ is simply EGO. During the experiment we estimate hyperparameters every 10 function evaluations, to keep our metamodel close to the actual function form. We run the numerical experiments 100 times repeatly, and obtain the average performance of q-EI algorithm for different q.

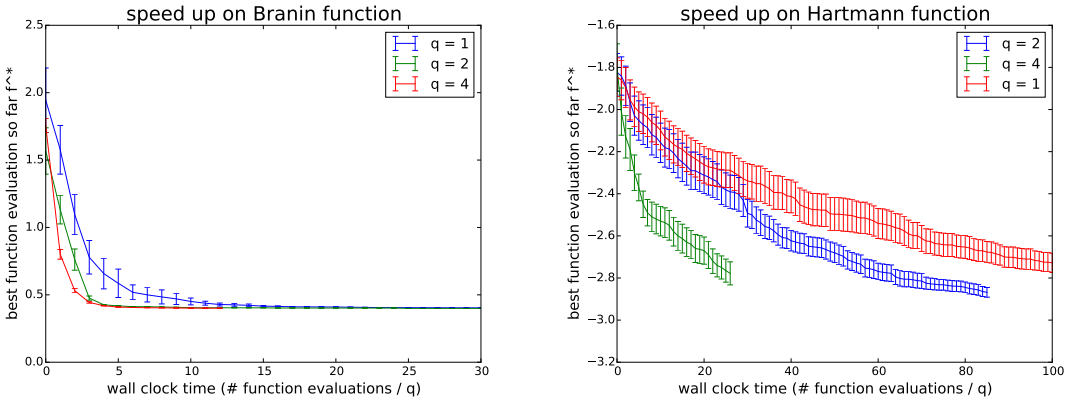

Figure 1: Average performance of q-EI algorithm for q = 1, 2, 4 on two test functions with 95% Confidence Interval (left is on Branin function, and right is on Hartmann $H_{6,4}$ function)

From the plots in figure 1, q-EI algorithm converges to global minimum of Brainin function very fast, and as q increases, convergence is faster in terms of wall clock time, which shows speedup of our parallel algorithm; q-EI did not find global minimum within 100 function evaluations, because Hartmann $H_{6,4}$ has 6 dimensions and is a harder problem to solve, but the algorithms have a clear trend of finding the minimum, and the parallel speedup also exists.

5.2 Comparison vs. previously proposed parallel methods

Constant Liar is an approximation parallel EI algorithm proposed by [4], and CL-MIX was considered the best variate among the group of methods in that paper (also available in R package 'DiceOptim'). We compare our method with CL-MIX on two standard test functions, Branin and Hartmann $H_{6,4}$. The experiment setup is similar to 5.1: we start the algorithms with $n = 10d$ initial randomly sampled points, and stop until certain number of function evaluations were reached. We repeat 100 times to obtain average performance. We choose to show the plot with $q = 4$, but the behavior is similar for other q s.

Figure 2 shows that q-EI and CL-MIX have almost identical performance on Branin function, this is because Brainin is a fairly easy problem and both algorithm converges to global optimum quickly. When dimensionality goes up and objective function is more complicated, for example, on Hartmann $H_{6,4}$ function, q-EI performance significantly better than CL-MIX. Since the approach of our algorithm is to estimate gradient of q-EI and solve (4) directly instead of using heuristic like CL-MIX did, we are confident that our method is superior over CL-MIX on average case. However, CL-MIX is generally much faster, since it only needs to evaluate 1-EI analytically within its algorithm, while our algorithm has to use Monte Carlo simulation to estimate gradient of q-EI when $q > 1$.

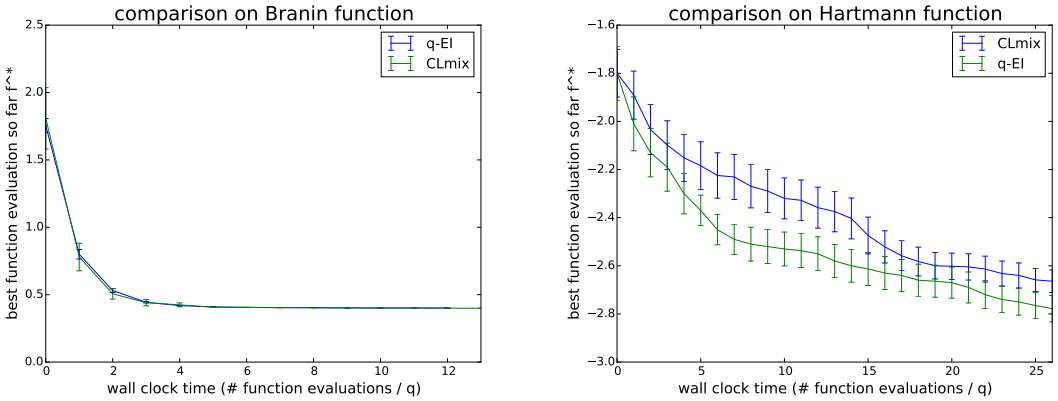

Figure 2: Average performance of q-EI vs. CL-MIX for q = 4 on two test functions with 95% Confidence Interval(left is on Branin function, and right is on Hartmann $H_{6,4}$ function)

5.3 Comparison vs. exact evaluations of q-EI

As Chevalier and Ginsbourger proposed a method to evaluate q-EI exactly [5], the method suffers from quickly slowing down as q increases. We use the Monte-Carlo approach to estimate q-EI and its stochastic gradient, with the help of GPU parallel programming, thanks to the parallelizable nature of Monte-Carlo method. We show CPU time vs. q to evaluate the q-EI using both methods, and demonstrate that exact q-EI method slows down quickly with q, while our method stays fast and remains accurate. We used 10^8 Monte-Carlo iterations for each evaluation of q-EI on GPU, to maintain average relative difference vs. exact evaluation of q-EI below 10^{-5}, and repeat the experiments 100 times using different starting points to obtain average performance.

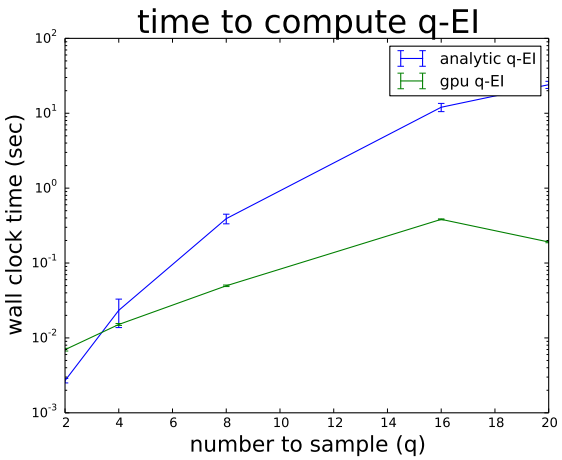

Figure 3: Average time to compute q-EI vs. q with 95% Confidence Interval

Although [5] did not attempt to solve (4) using their exact evaluation of q-EI, we are interested to see how Derivative Free Optimization solver along with exact evaluation of q-EI performs compare against stochastic gradient ascent with estimation of gradient of q-EI in our algorithm. We implemented exact evaluation of q-EI, and used L-BFGS [24] solver available in SciPy [18] to solve (4). To show the result, we first compare the best EI found by either optimization algorithm vs. number of iterations used in the algorithm. We fixed $q = 4$ (q is arbitrarily chosen and can

be other values), randomly sampled $10d$ points on either Branin or Hartmann $H_{6,4}$, where d is dimension of the domain, and fit the Gaussian Process model, then let both stochastic gradient ascent and L-BFGS start from the same random chosen point in the domain, and find the best EI given specified number of iterations. We repeated it 100 times and show the averaged performance on both Branin and Hartmann $H_{6,4}$ figure 4.

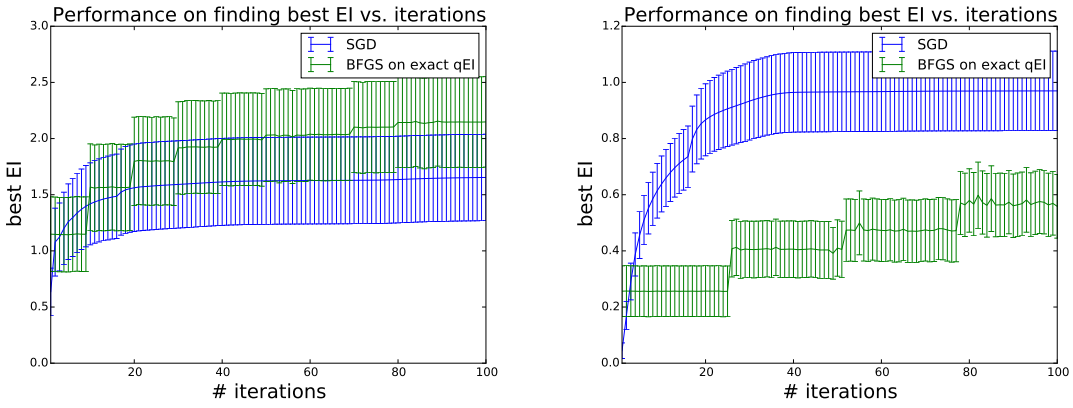

Figure 4: Average performance of stochastic gradient ascent vs. L-BFGS with 95% Confidence Interval (left is on Branin function, right is on Hartmann $H_{6,4}$ function

Stochastic gradient ascent and L-BFGS have similar performance on Branin function, but on Hartmann $H_{6,4}$, stochastic gradient ascent is better, because the additional informatin of the gradient helps it converges faster.

Using a similar experiment setup, we fixed number of iteration = 100 and compared best EI found vs. q, and show the result in figure 5. From the result we can infer that as dimension of the search space increases and q increases, stochastic gradient ascent wins.

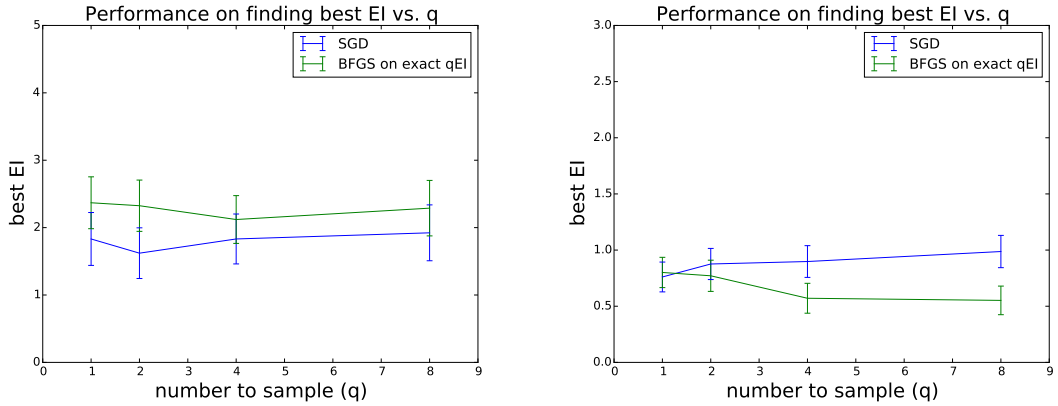

Figure 5: Average performance of stochastic gradient ascent vs. L-BFGS with 95% Confidence Interval (left is on Branin function, right is on Hartmann $H_{6,4}$ function

6 Conclusion

We proposed an efficient method based on stochastic approximation for implementing a conceptual parallel Bayesian global optimization algorithm proposed by [10]. To accomplish this, we used infinitessimal perturbation analysis (IPA) to construct a stochastic gradient estimator and showed that this estimator is unbiased. Through numerical experiments, our method performs better than the best available approximation method, and is much faster than other potential solution proposed so far.

References

[1] J. M. Calvin. Average performance of a class of adaptive algorithms for global optimization. *The Annals of Applied Probability*, 7(3):711–730, 1997.

[2] J. M. Calvin and A. Zilinskas. One-dimensional Global Optimization Based on Statistical Models. *Nonconvex Optimization and its Applications*, 59:49–64, 2002.

[3] J. M. Calvin and A. Zilinskas. One-Dimensional global optimization for observations with noise. *Computers & Mathematics with Applications*, 50(1-2):157–169, 2005.

[4] C. Chevalier and D. Ginsbourger. Fast Computation of the Multi-points Expected Improvement with Applications in Batch Selection. In *Lecture Notes in Computer Science*, pages 59–69. 2013.

[5] C. Chevalier and D. Ginsbourger. Fast computation of the multi-points expected improvement with applications in batch selection. In *Learning and Intelligent Optimization*, pages 59–69. Springer, 2013.

[6] P. Frazier, W. Powell, and S. Dayanik. The knowledge-gradient policy for correlated normal beliefs. *INFORMS journal on Computing*, 21(4):599–613, 2009.

[7] P. Frazier, J. Xie, and S. Chick. Value of information methods for pairwise sampling with correlations. In *Proceedings of the 2011 Winter Simulation Conference*, 2011.

[8] P. I. Frazier, W. B. Powell, and S. Dayanik. The Knowledge Gradient Policy for Correlated Normal Beliefs. *INFORMS Journal on Computing*, 21(4):599–613, 2009.

[9] D. Ginsbourger. Two advances in Gaussian Process-based prediction and optimization for computer experiments. In *MASCOT09 Meeting*, pages 1–2, 2009.

[10] D. Ginsbourger, R. Le Riche, and L. Carraro. A Multi-points Criterion for Deterministic Parallel Global Optimization based on Kriging. In *Intl. Conf. on Nonconvex Programming, NCP07*, page ..., Rouen, France, Dec. 2007.

[11] D. Ginsbourger, R. Le Riche, and L. Carraro. Kriging is well-suited to parallelize optimization. In *Computational Intelligence in Expensive Optimization Problems*, volume 2, pages 131–162. Springer, 2010.

[12] P. Glasserman. *Gradient Estimation via Perturbation Analysis*. Kluwer Academic Publishers, New York, 1991.

[13] Y. Ho. Performance evaluation and perturbation analysis of discrete event dynamic systems. *Automatic Control, IEEE Transactions on*, 32(7):563–572, 1987.

[14] R. Howard. Information Value Theory. *Systems Science and Cybernetics, IEEE Transactions on*, 2(1):22–26, 1966.

[15] D. Huang, T. T. Allen, W. I. Notz, and N. Zeng. Global Optimization of Stochastic Black-Box Systems via Sequential Kriging Meta-Models. *Journal of Global Optimization*, 34(3):441–466, 2006.

[16] J. Janusevskis, R. L. Riche, and D. Ginsbourger. Expected Improvements for the Asynchronous Parallel Global Optimization of Expensive Functions : Potentials and Challenges. In *Lecture Notes in Computer Science*, pages 413–418. 2012.

[17] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization*, 13(4):455–492, 1998.

[18] E. Jones, T. Oliphant, P. Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. [Online; accessed 2014-12-01].

[19] H. J. Kushner. A new method of locating the maximum of an arbitrary multi- peak curve in the presence of noise. *Journal of Basic Engineering*, 86:97–106, 1964.

[20] H. J. Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Fluids Engineering*, 86(1):97–106, 1964.

[21] H. J. Kushner and G. G. Yin. *Stochastic Approximation Algorithms and Applications*. Springer-Verlag, New York, 1997.

[22] H. J. Kushner and G. G. Yin. *Stochastic Approximation Algorithms and Applications*. Springer-Verlag, New York, 1997.

[23] P. L'Ecuyer. On the interchange of derivative and expectation for likelihood ratio derivative estimators. *Management Science*, pages 738–748, 1995.

[24] D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

[25] J. Mockus. *Bayesian approach to global optimization: theory and applications*. Kluwer Academic, Dordrecht, 1989.

[26] J. Mockus, V. Tiesis, and A. Zilinskas. The application of Bayesian methods for seeking the extremum. In L. C. W. Dixon and G. P. Szego, editors, *Towards Global Optimisation*, volume 2, pages 117–129. Elsevier Science Ltd., North Holland, Amsterdam, 1978.

[27] D. Owen, K. Craswell, and D. Hanson. Nonparametric upper confidence bounds for pr {Y¡ X} and confidence limits for pr {Y¡ X} when x and y are normal. *Journal of the American Statistical Association*, 59(307):906–924, 1964.

[28] B. T. Polyak. New stochastic approximation type procedures. *Automat. i Telemekh*, 7(98-107):2, 1990.

[29] C. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006.

[30] H. Royden. *Real analysis*. Macmillan, 1988.

[31] D. Ruppert. Efficient estimations from a slowly convergent robbins-monro process. Technical report, Cornell University Operations Research and Industrial Engineering, 1988.

[32] M. Schonlau. *Computer experiments and global optimization*. PhD thesis, University of Waterloo, 1997.

[33] S. Smith. Differentiation of the Cholesky algorithm. *Journal of Computational and Graphical Statistics*, pages 134–147, 1995.

[34] J. Thompson. Adaptive sampling. 1996.

[35] E. Vazquez and J. Bect. Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. *Journal of Statistical Planning and Inference*, 140(11):3088–3095, 2010.

[36] E. Vazquez, J. Villemonteix, M. Sidorkiewicz, and É. Walter. Global optimization based on noisy evaluations: an empirical study of two statistical approaches. In *Journal of Physics: Conference Series*, volume 135, page 012100. IOP Publishing, 2008.

[37] J. Villemonteix, E. Vazquez, and E. Walter. An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization*, 44(4):509–534, 2009.

[38] J. Xie, P. Frazier, and S. Chick. Bayesian optimization via simulation with pairwise sampling and correlated prior beliefs. in review, 2013.