*Proceedings of the 2010 Winter Simulation Conference*
*B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, eds.*

# CONVERGENCE PROPERTIES OF DIRECT SEARCH METHODS FOR STOCHASTIC OPTIMIZATION

Sujin Kim
Dali Zhang

Department of Industrial and Systems Engineering,
National University of Singapore,
1 Engineering Drive 2, 117576, Singapore

## ABSTRACT

Simulation is widely used to evaluate the performance and optimize the design of a complex system. In the past few decades, a great deal of research has been devoted to solving simulation optimization problems, perhaps owing to their generality. However, although there are many problems of practical interests that can be cast in the framework of simulation optimization, it is often difficult to obtain an understanding of their structure, making them very challenging. Direct search methods are a class of deterministic optimization methods particularly designed for black-box optimization problems. In this paper, we present a class of direct search methods for simulation optimization problems with stochastic noise. The optimization problem is approximated using a sample average approximation scheme. We propose an adaptive sampling scheme to improve the efficiency of direct search methods and prove the consistency of the solutions.

## 1 INTRODUCTION

Consider a class of optimization problems of the form

$$\min_{x\in X} \quad \left\{ F(x) := E\left[f(x,\xi(\omega))\right] = \int_{\Omega} f(x,\xi)P(d\xi(\omega)) \right\}, \tag{1}$$

where $\xi : \omega \to \Xi \subset \mathbb{R}^q$ is a random vector defined on a probability space $(\Omega, \mathscr{F}, P)$, $E[\cdot]$ denotes the mathematical expectation operator, and $x \in X$ is a decision vector with $X$ being a nonempty subset of $\mathbb{R}^p$. Throughout this paper, we assume that $E[f(x,\xi(\omega))]$ is well defined for every $x \in X$. The domain $X$ is assumed to be deterministic and known. We also assume that $E[f(x,\xi)]$ cannot be evaluated analytically by its closed form but has to be estimated by simulation.

In the past few decades, a great deal of research has been devoted to solving these simulation-optimization problems, due to the many practical applications that can be cast in the form of (1). For various introductions and perspectives on the problem see Fu (2002), the optimization-related chapters in Henderson and Nelson (2006), and simulation-optimization tutorials at the Winter Simulation Conference, for example Andradóttir (1998), Kim (2006) and Kim and Henderson (2008).

One standard technique for solving the problem (1) is the sample average approximation (SAA) approach. This method approximates the original simulation optimization problem (1) with a deterministic optimization problem in the following manner. Let $N$ be a deterministic positive integer, and suppose that we generate an independent random sample $\xi_1, .., \xi_N$. For a fixed $x$, define the sample mean over $(f(x,\xi_i) : 1 \le i \le N)$ as

$$\bar{F}^N(x) = \frac{1}{N}\sum_{i=1}^{N} f(x,\xi_i).$$

The SAA problem corresponding to (1) is

$$(\mathscr{P}_N) \quad \min_{x\in X} \bar{F}^N(x), \tag{2}$$

978-1-4244-9864-2/10/$26.00 ©2010 IEEE

i.e., we minimize the *sample* average. Once the sample is fixed, $\bar{F}^N(x)$ becomes deterministic. Consequently, the SAA problem (2) becomes a deterministic optimization problem and one can solve it using any convenient optimization algorithm. For a general introduction to the SAA approach, the readers are referred to Shapiro (2003).

The basic idea of the SAA method yields several possible variations. For instance, instead of solving a single SAA problem, we can solve a sequence of SAA problems $\{\mathscr{P}_{m_k}\}$ with increasing sample size $\{m_k\} \to \infty$ (Chen and Schmeiser 2001, Pasupathy and Schmeiser 2009). At every iteration, the solution from the previous iteration can be used as a starting point and a new SAA problem is resolved. Pasupathy (2010) analyzed the asymptotically optimal sample size $m_k$ and the error-tolerance level $e_k$. However, numerical results show that this optimal convergence often cannot be realized in finite sample simulation (Royset 2010). Bayraksan and Morton (2010) developed a sequential sampling scheme to determine the sample size $m_k$. At iteration $k$, the optimality gap is estimated in order to determine whether to terminate the algorithm, and $m_{k+1}$ is updated based on this estimate if the condition on the optimality gap is not satisfied. Nice statistical inferences on the optimal values and solutions can be obtained in this context, but solving a new SAA problem at each iteration can be time-consuming.

Another variation is using the variable-number sample-path (VNSP) scheme in an iterative optimization algorithm. Instead of solving the SAA problem with a fixed sample size $N$ at every iteration, one may want to use variable sample size $N_k$, where $N_k$ may grow to infinity with the number of iterations $k$. Homem-de-Mello (2003) analyzed the asymptotic property of the objective function under the VNSP scheme and showed the pointwise convergence of the approximate objective function to the true objective function. This work also incorporated VNSP into some basic random search methods and applied the scheme to stochastic optimization problems with a finite feasible set. More recently, Deng and Ferris (2009) applied Powell's unconstrained optimization method by quadratic approximation (UOBYQA) to solve a SAA problem under the VNSP scheme, achieving the convergence of the algorithm to a stationary point. Their numerical results showed that the VNSP scheme has great potential for improving the rate of convergence.

In this paper, we consider a class of direct search methods for simulation optimization problems where the objective function is evaluated through black-box optimization with stochastic noise. Adapting some of the ideas from Deng and Ferris (2009), we propose an adaptive sampling scheme to improve the efficiency of direct search methods and prove the consistency of the solutions. Unlike Deng and Ferris (2009), we do not assume any prior distributions for the algorithm parameters to adaptively update the sample size at each iteration. The numerical results show that the proposed algorithm outperforms SAA methods with fixed sampling, supporting our asymptotic convergence analysis.

In Section 2, we review the direct search method for deterministic black-box optimization. In Section 3, we propose a way of extending this methodology to simulation optimization problems with stochastic noise. Section 4 gives our convergence analysis of the proposed algorithm. Finally, Section 5 presents numerical results comparing the proposed method to other techniques.

## 2 DIRECT SEARCH METHOD

Direct search methods are derivative-free methods that sample the objective function at a finite number of points at each iteration and decide which actions to take next solely on those function values without any derivative information. These methods received a great deal of attention since the great advance in parallel computing made in the 1990s. In particular, the methods have been widely applied to simulation-based optimization problems, in which analytical derivatives cannot be obtained or the objective function is nonsmooth.

In this section, we present a description for a particular class of direct search methods for deterministic continuous optimization problems, named as *generating set search* (GSS). The concept of GSS is introduced by Kolda et al. (2003). It is generic enough to capture many features of direct search methods, including classical pattern search, while still remaining simple enough to discuss with a minimum of notation and without handling a variety of special cases. For a review of direct search methods, the readers are referred to Lewis et al. (2000) and Kolda et al. (2003). For a review of general derivative-free methods for continuous optimization problems, see Conn et al. (2009). Here we present the GSS algorithm for the problem of finding a local minimizer of a real-valued function $f(x)$.

**I. Initialization.** Let $x_0 \in \mathscr{R}^p$ be the initial guess, and $\Delta_0 > \Delta_{\text{tol}} > 0$ be the initial value of the step-length control parameter.

Let $\theta_{\max} < 1$ be an upper bound on the contraction parameter.

Let $\rho : [0,\infty) \to [0,\infty)$ be a nonnegative continuous function. Either $\rho \equiv 0$ or $\rho(t)$ is a positive, continuous, monotonically increasing function such that $\rho(t)/t \to 0$ as $t \downarrow 0$.

Let $\beta_{\max} \geq \beta_{\min} \geq 0$ be upper and lower bounds, respectively, on the lengths of the vectors in any generating set.

**II. Algorithm.** For each iteration $k = 0, 1, 2, \cdots$

- **Step 1**
  (i) (Successful Iteration) If there exists $d_k \in \mathscr{D}_k$ such that $f(x_k + \Delta_k d_k) < f(x_k) - \rho(\Delta_k)$, then do the following:

```
        - Set $x_{k+1} = x_k + \Delta_k d_k$.
        - Set $\Delta_{k+1} = \phi_k \Delta_k$, where $\phi_k \geq 1$.
    (ii) (Unsuccessful Iteration) Otherwise, $f(x_k + \Delta_k d_k) \geq f(x_k) - \rho(\Delta_k)$ for all $d_k \in \mathscr{D}_k$, so do the following:
        - $x_{k+1} = x_k$.
        - $\Delta_{k+1} = \theta_k \Delta_k$ where $0 < \theta_k < \theta_{\max}$.
        - If $\Delta_{k+1} < \Delta_{\text{tol}}$, then terminate.
• Step 2 Let $\mathscr{D}_{k+1} = \mathscr{G}_{k+1} \cup \mathscr{H}_{k+1}$. Here $\mathscr{G}_{k+1}$ is a generating set for $\mathscr{R}^p$ satisfying $\beta_{min} \leq ||d|| \leq \beta_{max}$ for all $d \in \mathscr{G}_{k+1}$ and $\kappa(\mathscr{D}_{k+1}) \geq \kappa_{min}$, and $\mathscr{H}_{k+1}$ is a finite (possibly empty) set of additional search directions such that $\beta_{min} \leq ||d||$ for all $d \in \mathscr{H}_{k+1}$.
```

Figure 1: GSS algorithm by Kolda, Lewis and Torczon (2000).

The algorithm must be equipped with a set of search directions that includes a descent direction. To avoid poor search directions, there must be a descent direction that is not “too orthogonal” to the direction of steepest descent. To ensure that the above conditions are satisfied without using knowledge of explicit gradients, GSS methods use multiple search directions. The set of these directions is known as the *generating set*. A set $\mathscr{G} = \{d^{(1)}, \cdots, d^{(r)}\}$ of $r > p+1$ vectors in $\mathbb{R}^p$ generates $\mathbb{R}^p$ if $\mathscr{G}$ positively spans $\mathbb{R}^p$, that is, for any vector $v \in \mathbb{R}^p$, there exist $\lambda^{(1)}, \cdots, \lambda^{(r)} \geq 0$ such that $v = \sum_{i=1}^r \lambda^{(i)} d^{(i)}$.

A generating set must contain a minimum of $p+1$ vectors. For example, in a two dimensions, a minimal generating set with $p+1 = 3$ vectors is

$$\mathscr{G} = \left\{ \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \begin{bmatrix} -1 \\ 1 \end{bmatrix} \right\}.$$

The coordinate directions also form a generating set with $2p$ vectors. In two dimensions, there are $2p = 4$ vectors in the set

$$\mathscr{G} = \left\{ \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} -1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ -1 \end{bmatrix} \right\}.$$

In GSS, we use the notion of *cosine measure* to quantify the worst-case distance between the steepest descent direction $v = -\nabla f(x)$ and the vector in $\mathscr{G}$ that makes the smallest angle with $v$. This measure is defined to be

$$\kappa(\mathscr{G}) := \min_{v \in \mathbb{R}^n} \max_{d \in \mathscr{G}} \frac{v^T d}{\|v\| \|d\|},$$

where $\mathscr{G}$ is a generating set in $\mathbb{R}^p$. In order to prevent a slow rate of descent, the cosine measure must be bounded below, that is, $\kappa(\mathscr{G}_k) \geq \kappa_{\min}$ for any iteration $k = 1, 2, \cdots$.

The nonnegative function $\rho$ is called the *forcing function*. Choosing $\rho$ to be identically zero imposes a simple decreasing condition on the acceptance of the step. Under a set of regularity conditions (for example, when the forcing function $\rho$ is positive, together with some other relatively mild conditions), GSS algorithm achieves global convergence. The first step of the proof is to show that the sequence of step size parameters $\Delta_k$ tends to zero.

**Theorem 1.** *(Lewis, Torczon, and Trosset 2000) Let $f$ be bounded below. Suppose that $\rho(t)$ is a positive forcing function. Then the sequence of stepsizes produced by GSS satisfies*

$$\liminf_{k \to +\infty} \Delta_k = 0.$$

The following theorem leads to global convergence results and to the validation of stopping criteria for GSS methods.

**Theorem 2.** *(Lewis, Torczon, and Trosset 2000) Let $f : \mathbb{R}^p \to \mathbb{R}$ be continuously differentiable, and suppose $\nabla f$ is Lipschitz continuous with constant $M$. Assume that $\rho(t)$ is continuous, $\rho(t) = o(t)$ as $t \to 0$ and $\rho(t_1) \leq \rho(t_2)$ for $t_1 < t_2$. Then, GSS produces iterations satisfying*

$$\|\nabla f(x_k)\| \leq \kappa(\mathscr{G}_k)^{-1} \left[ M \Delta_k \beta_{\max} + \frac{\rho(\Delta_k)}{\Delta_k \beta_{\min}} \right].$$

By Theorem 2, it can be easily shown that $\liminf_{k \to \infty} \|\nabla f(x_k)\| = 0$ and the sequence of iterates $\{x_k\}$ has a limit point $x^*$ for which $\nabla f(x^*) = 0$.

All the analysis above is based on the assumption that there is no random noise in the simulation output. In the next section, we introduce a version of the GSS algorithm that can handle stochastic noise in the objective function.

## 3 THE GSS METHOD FOR SIMULATION OPTIMIZATION

In the section, we describe the GSS algorithm for solving the simulation optimization problem (1) based on SAA methods. We assume that (a) the underlying expected objective function $F(x)$ has no explicit form and the expectation is computed by taking an average over the observed or sampled objective function $f$; (b) $f$ is a black-box function and every evaluation of the random function $f(x,\xi)$ is obtained by running a simulation with input value $x$. Thus, we do not know the explicit form of either $F$ or $f$, and derivative-based algorithms are not suitable for our problem.

In order to obtain an approximate solution to the true problem (1), we apply the GSS algorithm to the SAA problem (2). However, instead of fixing a sample from the beginning and then iteratively searching for the (local) minimum of the resulting deterministic problem (2), we use an increasing sequence of sample sizes $\{N_k\}$ in order to improve the finite time performance of the algorithm as well as to achieve convergence. At iteration $k$, we can either reuse $N_{k-1}$ samples from the previous iteration and generate $N_k - N_{k-1}$ new samples, or we can generate a new sample that is independent of all the samples generated previously. The detailed GSS algorithm with VNSP scheme is as follows.

**I. Initialization.** Let $x_0 \in \mathscr{R}^p$ be the initial guess, and $\Delta_0 > \Delta_{\text{tol}}$ be the initial value of the step-length control parameter. Let $\rho(t)$ be a positive, continuous, monotonically increasing function such that $\rho(t)/t \to 0$ as $t \downarrow 0$.

**II. Algorithm.** For each iteration $k = 0, 1, 2, \cdots$

- **Step 1**
  Update $N_k$ and generate a sample $\xi_k^1, \xi_k^2, \cdots, \xi_k^{N_k}$. Define

$$\bar{F}_k^{N_k}(x) := \frac{1}{N_k} \sum_{i=1}^{N_k} f\left(x, \xi_k^i\right). \tag{3}$$

- **Step 2**
  (i) (Successful Iteration) If there exists $d_k \in \mathscr{D}_k$ such that $\bar{F}_k^{N_k}(x_k + \Delta_k d_k) < \bar{F}_k^{N_k}(x_k) - \rho(\Delta_k)$, then do the following:
  - Set $x_{k+1} = x_k + \Delta_k d_k$.
  - Set $\Delta_{k+1} = \phi_k \Delta_k$, where $\phi_k \geq 1$.

  (ii) (Unsuccessful Iteration) Otherwise, $\bar{F}_k^{N_k}(x_k + \Delta_k d_k) > \bar{F}_k^{N_k}(x_k) - \rho(\Delta_k)$, for all $d_k \in \mathscr{D}_k$, so do the following:
  - $x_{k+1} = x_k$.
  - $\Delta_{k+1} = \theta_k \Delta_k$ where $0 < \theta_k < \theta_{\max}$.
  - If $\Delta_{k+1} < \Delta_{\text{tol}}$, then **terminate**.
- **Step 3** Same as Step 2 in Figure 1.

Figure 2: Algorithm 1 - GSS algorithm for simulation optimization.

The sample size $N_k$ is crucial to the performance of the algorithm. If $N_k$ is too small, the resulting SAA problem would be inaccurate and may lead to a bad movement of the algorithm. However, when the mesh size $\Delta_k$ is still large and the iterate has not entered a neighborhood of a stationary point, we may not need a highly accurate approximate problem to determine a descent direction, and hence significant computational savings can be achieved by using small $N_k$. We determine the sample size $N_k$ in such a way that the probability that the iterate $x_k$ is worse than $x_{k-1}$ decreases with a certain rate as iteration grows. The sequence of these probabilities eventually converges to zero, meaning that the iterates converge to a true local solution. The variability of the estimated value of the objective function, and the potential improvement in the value of the true objective function at the current iteration, should be taken into account to identify the appropriate value for $N_k$. We do this by introducing the following new algorithm for the true problem:

**At iteration $k$:** $k = 0, 1, 2, \cdots$
Let $\mathscr{S}_k = (x_k, \Delta_k, \mathscr{D}_k)$ be the state of Algorithm 1 at the beginning of iteration $k$ and $x_{k+1}$ be the iterate returned from Algorithm 1 at the end of iteration $k$.

- Case 1: If $x_k \neq x_{k+1}$ and $f(x_{k+1}) < f(x_k) - \rho(\Delta_k)$, then set $x'_{k+1} = x_{k+1}$.
- Case 2: Else if there exists $d'_k \in \mathscr{D}_k$ such that $f(x_k + d'_k \Delta_k) < f(x_k) - \rho(\Delta_k)$, then set $x'_{k+1} = x_k + d'_k \Delta_k$.
- Case 3: Else if there exists $d'_k \in \mathscr{D}_k$ such that $f(x_k + d'_k \Delta_k) = f(x_k) - \rho(\Delta_k)$, then set $x'_{k+1} = x_{k+1}$.

Figure 3: Algorithm 2 - Counterpart GSS algorithm for the true problem.

- Case 4: Else $x'_{k+1} = x_k$.

Algorithm 2 moves to a better solution whenever there exists a descent direction in the given direction set. If the given solution is better than other candidate solutions, Algorithm 2 stays at the same solution. Otherwise, take the solution returned from Algorithm 1. We present the convergence analysis of Algorithm 1 and 2 in the next section.

## 4 THE ANALYSIS OF THE ALGORITHM

From Algorithm 1 and Algorithm 2, we obtain two sequences of iterates $\{x_k\}$ and $\{x'_k\}$, respectively. In this section, we will prove that both sequences converge to a stationary point of the true problem (1) under a set of mild regularity conditions. The key idea for the convergence proof is that increasing the sample size reduces the bias between the SAA problem and the true problem. Then, Algorithm 1 will eventually produce a sequence $\{x_k\}$ whose tail part coincides with that of $\{x'_k\}$.

Before proceeding to the convergence analysis, we introduce assumptions on function $F$ and $f$.

**A1.** $F$ is bounded below on the level set $L(x_0) = \{x \in \mathbb{R}^p : F(x) \leq F(x_0)\}$.
**A2.** $F$ is continuously differentiable.
**A3.** The positive spanning sets $D_k$, $k = 0, 1, \ldots$ in the algorithm are chosen from a finite set $\mathscr{D}$, that is, $|\mathscr{D}| < \infty$ and $\mathscr{D}_k \subset \mathscr{D}$, for all $k$.
**A4.** $\{f(x,\xi) : x \in \mathbb{R}^p\}$ is uniformly integrable and $\sup_{x\in\mathbb{R}^p} Var[f(x,\xi)] = \sigma^2 < \infty$.

Assumptions A1-A3 are sufficient conditions under which the GSS algorithm for the true problem globally converges. Assumption A2 and A3 replace the Lipschitz continuity of $\nabla F$ in Theorem 2. Assumption A4 is necessary so that the SAA problem converges to the true problem and the random noise is regulated. A2 can be satisfied if the sample function $f$ satisfies a set of regularity conditions (see Kim 2006).

Let $E_k^N = \{x_{k+1}^N \neq x_{k+1}^{\prime N}\}$, where $x_{k+1}^N$ and $x_{k+1}^{\prime N}$ are solutions obtained at the end of the $k$th iteration with sample size $N_k = N$ from Algorithm 1 and 2, respectively. Then $P(E_k^N)$ is the probability that Algorithm 1 fails to make a correct decision at iteration $k$ when the objective function values are approximated using sample size $N$. The next theorem shows that this failure probability will eventually converge to zero as the sample size $N$ grows.

**Proposition 3.** *Assume A1-A4. Then $P(E_k^N) \to 0$ as $N \to \infty$, for $k = 1, 2, \ldots$*

*Proof.* Here we provide a sketch of the proof. Let $\tau_k = \inf\{|f(x_k) - f(x_k + d\Delta_k) - \rho(\Delta_k)| > 0 : d \in \mathscr{D}_k\}$. After several steps of computation, we can obtain

$$\begin{aligned} P(E_k^N|\mathscr{S}_k) &\leq P\left((\bar{F}_k^N(x) - f(x)) - (\bar{F}_k^N(y) - f(y)) > \tau_k|\mathscr{S}_k\right) \\ &\leq \frac{1}{\tau_k^2} Var\left(\bar{F}_k^N(x) - \bar{F}_k^N(y)|\mathscr{S}_k\right) \\ &\leq \frac{4\sigma^2}{\tau_k^2 N} \text{ for given } \mathscr{S}_k, \end{aligned} \tag{4}$$

where $x = x_k$ (or $x_k + d\Delta_k$) and $y = x_k + d\Delta_k$ (or $x_k$) for some $d \in \mathscr{D}_k$. Note that $x_k$ is random and so is $\tau_k$. Since there are only finitely many possible values of $\Delta_i$, $i = 0, 1, \ldots, k$ and $\mathscr{D}$ is also finite, the algorithm can visit only a finite number of points until iteration $k$, that is, $|\{x_i(\omega) : i = 0, 1, \ldots, k, \omega \in \Omega\}| < \infty$. Therefore, $\inf\{\tau_k(\omega) : \omega \in \Omega\} > 0$ and $P(E_k^N) \leq C/N$ for some constant $C > 0$ .

□

By Theorem 3, for any given $0 < \alpha_k < 1$, $P(E_k^N) \leq \alpha_k$ with large enough $N$. To prove the convergence of Algorithm 1, we need to assume the following property on the sequence $\{\alpha_k\}$ of failure probabilities.

**A5.** $\{\alpha_k\}$ is a sequence of real numbers such that $0 < \alpha_k < 1$, for all $k = 1, 2, \ldots$ and $\sum_{k=1}^{\infty} \alpha_k < \infty$.

Let $\{N_k\}$ be the sequence of sample sizes that satisfies $P(E_k^{N_k}) \leq \alpha_k$, $k = 1, 2, \ldots$ For notational convenience, we let $x_k = x_k^{N_k}$ and $x'_k = x_k^{\prime N_k}$. Then, by the Borel-Cantelli Lemma, the events $E_k^{N_k}, k = 1, 2, \ldots$ occur only finitely many times, meaning that there exists a large $K > 0$ such that for all $k \geq K$, $x_k = x'_k$.

**Theorem 4.** *Assume A1-A5. Then,* $\liminf_{k\to\infty}\|\nabla f(x_k)\| = 0$ *w.p.1 and the sequence of iterates* $\{x_k\}$ *has a limit point* $x^*$ *for which* $\nabla f(x^*) = 0$.

*Proof.* The result is an immediate consequence of Theorem 2. Since $x_k = x'_k, k \geq K$ for a large $K$, it suffices to show that a subsequence of $\{x'_k : k \geq K\}$ converges to a stationary point. At iteration $k$, if there exists $d'_k \in \mathscr{D}_k$ such that $f(x'_k + d'_k\Delta_k) < f(x'_k) - \rho(\Delta_k)$, or $f(x'_k + d\Delta_k) > f(x'_k) - \rho(\Delta_k)$ for all $d \in \mathscr{D}_k$, then $x'_{k+1}$ is the same with the iterate produced by the original GSS algorithm in Figure 1. If $f(x'_k + d\Delta_k) \geq f(x'_k) - \rho(\Delta_k)$ for all $d \in \mathscr{D}_k$ and there exists $d'_k \in \mathscr{D}_k$ such that $f(x'_k + d'_k\Delta_k) = f(x'_k) - \rho(\Delta_k)$, then Algorithm 2 either stays in $x'_k$ like the the original GSS algorithm or moves to $x'_k + d'_k\Delta_k$. Since the iterate $x'_k + d'_k\Delta_k$ makes a sufficient improvement in the objective function value, moving to this point does not affect the convergence result in Theorem 1 and 2. □

Equation (4) in the proof of Theorem 3 provides a guideline on how to select an appropriate form for $N_k$ to guarantee the asymptotic convergence of Algorithm 1. The term $1/\tau_k^2$ is a measure of the difficulty of determining a descent direction, and $\sigma^2/N$ represents the degree of random noise in the approximate objective function. In order to converge to the true solution, the stochastic error should converge to zero fast enough so that the algorithm can detect a descent direction with significantly high probability. Let $\alpha_k = k^{-\alpha}$ for some $\alpha > 1$. Then, Equation (4) suggests $N_k = O(k^\alpha/\tau_k^2)$.

Another way to obtain the upper bound of $P(E_k^N|\mathscr{S}_k)$ is by using large deviation theory. Suppose that $(f(x_k,\xi_k^i), f(x_k + d\Delta_k,\xi_k^i)), i = 1,2,\ldots,N_k$ are conditionally independent for given $\mathscr{S}_k$. Then, Chernoff's bound yields

$$P\left((\bar{F}_k^{N_k}(x) - f(x)) - (\bar{F}_k^{N_k}(y) - f(y)\right) > \tau_k) \leq e^{(-N_k I(\tau_k))} \text{ for given } \mathscr{S}_k. \tag{5}$$

Here $I(\cdot)$ is the so-called *rate function* corresponding to the distribution of $f(x,\xi) - f(y,\xi)$, which is defined by

$$I(z) := \sup_{t\in\mathbb{R}}[tz - \log M(t)]$$

and $M(t) := \mathbb{E}\left[e^{t(f(x,\xi)-f(y,\xi))}\right]$ is the moment generating function of $f(x,\xi) - f(y,\xi)$. Since the variance if $f(x,\xi) - f(y,\xi)$ is finite, we can obtain (see Shapiro and Homem-de-Mello 2000)

$$I(\tau_k) \geq \frac{\tau_k}{3Var(f(x,\xi) - f(y,\xi))} \geq \frac{\tau_k}{12\sigma^2} \text{ for given } \mathscr{S}_k,$$

where $\tau_k$ is in a neighborhood of zero. This result suggests that $N_k$ has the form $O(\alpha_k \ln(k)/\tau_k^2)$. If $L(x_0)$ is compact, $f$ is Lipschitz continuous on $L(x_0)$ and we obtain $\tau_k = O(\Delta_k)$. By replacing $\tau_k$ by $\Delta_k$ we can obtain $N_k = O(k^\alpha/\Delta_k^2)$ and $N_k = O(\ln(k)/\Delta_k^2)$ from Equation (4) and (5), respectively. In our numerical experiments, we tested the efficiency of these two forms of $N_k$.

## 5 NUMERICAL EXPERIMENTS

We applied GSS methods with several sample size updating schemes to an extended two-dimensional Rosenbrock function. This function is a popular performance test problem for optimization algorithms, because its global minimum is inside a long, narrow valley with a parabolic shape. We added random noise to the first decision variable $x^{(1)}$, changing the problem to the following:

$$f(x,\xi) = 100\cdot\left(x^{(2)} - \xi(x^{(1)})^2\right)^2 + \left(\xi x^{(1)} - 1\right)^2,$$

where $x = (x^{(1)}, x^{(2)})$. We assume the random variable $\xi$ is independent of $x$ and normally distributed with mean 1 and variance $0.1^2$. The closed form for $F(x) = E[f(x,\xi)]$ can be easily computed, and the optimal solution is $x^* = (0.4162, 0.1750)$. The optimal value is $f(x^*) = 0.4632$.

In our numerical experiments, we applied the coordinate direct search algorithm with several types of VNSP methods to the test problem and compared them. In particular, we used the following forms of $N_k$:

- VNSP1: $c\cdot k^\alpha$
- VNSP2: $c\cdot k^\alpha/\Delta_k^2$
- VNSP3: $c\cdot \ln(k)/\Delta_k^2$

The real numbers $c$ and $\alpha$ are positive, and $\alpha > 1$. We propose the last two forms based on our convergence analysis in Section 4. The first form only depends on the iteration number $k$, and we test this method to investigate the effect

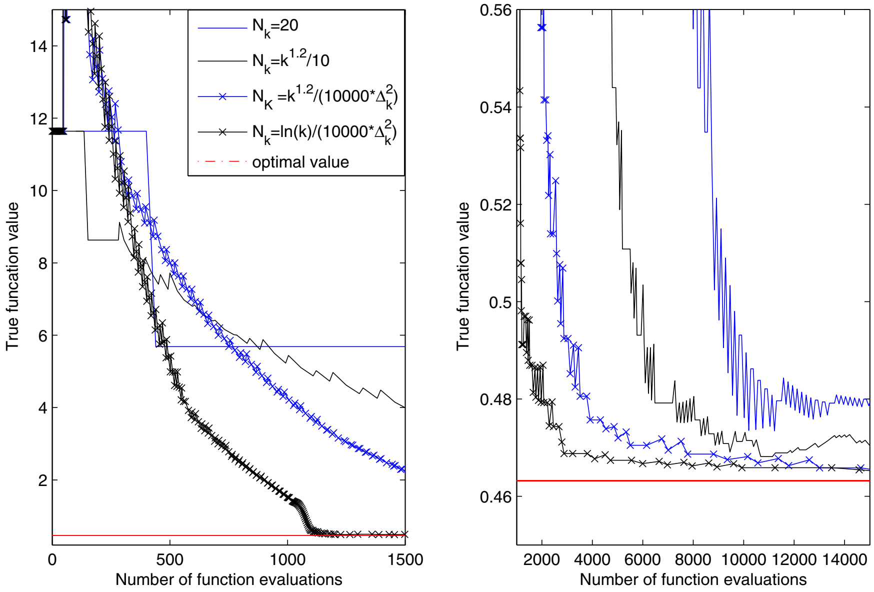

Figure 4: Number of function evaluations vs. true function value.

of the term $1/\Delta_k$. The numerical results are somewhat sensitive to the parameters $\alpha$ and $c$, and we conducted a pilot run to tune these parameters.

In Figure (4), we plot a single-run performance of accumulated function evaluations against the true function value at the current iterate. We observe that all the VNSP methods performed better than the constant sample size scheme both at the early iteration of the algorithm (left plot) and the long term (right plot). At the early iteration, the VNSP methods use a smaller sample size and can execute more iterations for a given number of function evaluations. However, as the iterate approaches the optimal solution, a larger sample size should be used to obtain a more accurate approximation and to make the correct decision on the direction in which to move. All VNSP methods use a larger sample size as the iteration counter grows and seems to converge to the true optimal solution. On the other hand, the fixed sample size scheme converges to the solution to the problem $(\mathscr{P}_N)$, which may not be the solution to the true problem.

Although VNSP1 outperforms the fixed sample size scheme, it does not converge to the true optimal solution as fast as the other two VNSP schemes. To investigate this issue, we show in Figure (5) how the sample size $N_k$ and the mesh size $\Delta_k$ change with the iteration counter $k$. We see that VNSP2 and VNSP3 take a smaller sample size than VNSP1 for some time during the early iterations, and then increase exponentially fast as the mesh size shrinks. Figure (4) shows that VNSP2 and VNSP3 steadily move to better solutions, whereas the solution path of VNSP1 oscillates much more than the other two VNSP methods at later iterations. When the mesh size is still large, the stochastic error tends to be relatively small compared to the optimization error, and the algorithm can detect the descent direction reasonably well with a small sample size approximation. However, when the mesh size is small, the algorithm requires a much larger sample size to determine the descent direction. These empirical results demonstrate that our proposed algorithm increases the sample size fast enough so that the stochastic error term is dominated by the optimization error.

In practice, often the computational budget is limited and users usually look for an acceptably good solution, instead of the optimal. To study the performance and robustness of the different methods, we ran 100 simulations of each method. We present the mean and standard error of the true objective function value obtained with 2000 and $10,000$ function evaluations. The statistical results are summarized in Table 1. The results show that our proposed two VNSP methods perform more robustly than the other two methods, particularly in the early stage, which is very advantageous for finite-time performance.

## 6 CONCLUSIONS

We proposed a stochastic version of direct search algorithms that uses a variable number sampling scheme in the context of the sample average approximation method. We proposed sample size updating schemes that allow the algorithm to achieve global convergence. Numerical results show that the proposed methods perform better than a fixed sample size

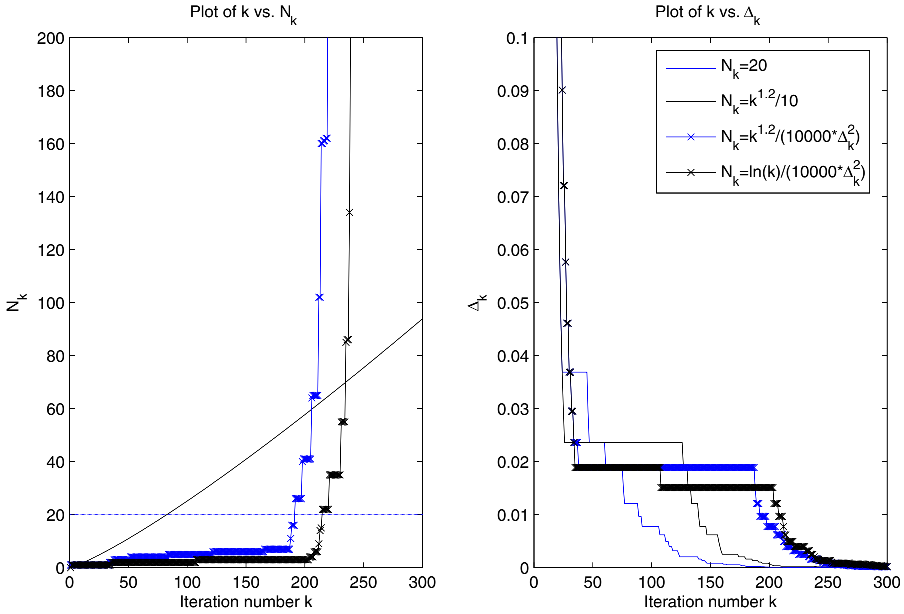

Figure 5: Plots of number of function evaluations $N_k$ and mesh size $\Delta_k$.

Table 1: Statistical summary

| $N_k$ | 2,000 function evaluations | | 10,000 function evaluations | |
|---|---|---|---|---|
| | Mean of $F$ | Standard error | Mean of $F$ | Standard error |
| 20 | 5.40 | 0.08 | 0.65 | 0.03 |
| $k^{1.2}/10$ | 2.18 | 0.12 | 0.53 | 0.02 |
| $k^{1.2}/(10000\Delta_k^2)$ | 1.77 | 0.17 | 0.52 | 0.02 |
| $\ln(k)/(10000\Delta_k^2)$ | 1.37 | 0.14 | 0.50 | 0.01 |

method and other VNSP methods in the literature, both in the short and long term. There are some issues which require more consideration and should be taken into account as further research. In the numerical test, $N_k$ involves two tunable parameters $\alpha$ and $c$. Selecting good values for these parameters is a nontrivial problem. We are currently investigating how such values can be determined, as well as a nonparametric form of the sample size $N_k$. Another direction for further research is to consider other types of direct search methods, in addition to coordinate search. We believe that the proposed method shows potential as an analog of the direct search method in deterministic optimization, and is a promising subject for further study.

## ACKNOWLEDGMENTS

The authors thank Amr El-Bakry and Jin-Hwa Song at ExxonMobil Research and Engineering for very helpful comments. We also thank Ilya O. Ryzhov for helpful discussions on our paper. This research was partially supported by ExxonMobil Research and Engineering Company Grant R-266-000-052-597 and NUS Research Grant R-266-000-049-133.

## REFERENCES

Andradóttir, S. 1998. A review of simulation optimization techniques. In *Proceedings of the 1998 Winter Simulation Conference*, ed. D. Medeiros, E. Watson, J. S. Carson, and M. S. Manivannan, 151–158. Piscataway, NJ: IEEE.

Bayraksan, G., and D. P. Morton. 2010. A sequential sampling procedure for stochastic programming. *Operations Research*. To appear.

Chen, H., and B. W. Schmeiser. 2001. Stochastic root-finding via retrospective approximation. *IIE Transactions*.

Conn, A., K. Scheinberg, and L. N. Vincent. 2009. *Introduction to derivative-free optimization*. USA: MPS-SIAM.

Deng, G., and M. Ferris. 2009. Variable-number sample-path optimization. *Mathematical Programming B*.

Fu, M. C. 2002. Optimization for simulation: theory vs. practice. *INFORMS Journal on Computing* 14:192–215.

Henderson, S. G., and B. L. Nelson. (Eds.) 2006. *Simulation*. Handbooks in Operations Research and Management Science. Amsterdam: Elsevier.

Homem-de-Mello, T. 2003. Variable-sample methods for stochastic optimization. *ACM Transactions on Modeling and Computer Simulation* (2).

Kim, S. 2006. Gradient-based simulation optimization. In *Proceedings of the 2006 Winter Simulation Conference*, ed. L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, 159–167. Piscataway NJ: IEEE.

Kim, S., and S. G. Henderson. 2008. The mathematics of continuous-variable simulation optimization. In *Proceedings of the 2008 Winter Simulation Conference*, ed. S. J. Mason, R. R. Hill, L. Moench, and O. Rose, 159–167. Piscataway NJ: IEEE.

Kolda, T. G., R. M. Lewis, and V. Torczon. 2003. Optimization by direct search: New persepectives on some classical and mordern methods. *SIAM Review* 45:385–482.

Lewis, R. M., V. Torczon, and M. W. Trosset. 2000. Direct search methods: Then and now. *Journal of Computational and Applied Mathematics* 124:191–207.

Pasupathy, R. 2010. On choosing parameters in retrospective-approximation algorithms for stochastic root-finding and simulation optimization. *Operations Research*. To appear.

Pasupathy, R., and B. W. Schmeiser. 2009. Retrospective-approximation algorithms for the multidimensional stochastic root-finding problem. *ACM Transactions on Modeling and Computer Simulation*.

Royset, J. O. 2010. On sample size control in sample average approximations for solving smooth stochastic programs. manuscript.

Shapiro, A. 2003. Monte Carlo sampling methods. In *Stochastic Programming*, ed. A. Ruszczynski and A. Shapiro, Handbooks in Operations Research and Management Science. Elsevier.

Shapiro, A., and T. Homem-de-Mello. 2000. On the rate of convergence of optimal solutions of monte carlo approximations of stochastic programs. *SIAM Journal on Optimization* 11 (1): 70–86.

## AUTHOR BIOGRAPHIES

**SUJIN KIM** is an assistant professor in the Department of Industrial and Systems Engineering at the National University of Singapore. She received her Ph.D. degree in Operations Research from Cornell University in 2006. Before she joined the National University of Singapore, she was a visiting assistant professor in the Department of Industrial Engineering at Purdue University. Her research concerns simulation methodology and stochastic simulation-based optimization, with applications in electric power and health service systems. Her e-mail address is <iseks@nus.edu.sg>.

**DALI ZHANG** is a post-doc research fellow at Department Industrial and Systems Engineering, National University of Singapore. His research interest is the area of Monte Carlo sampling methods, random search algorithm and stochastic game theory, and the applications in electricity markets, risk analysis and logistics. His email address is <isezd@nus.edu.sg>.