Proceedings of the 2004 Winter Simulation Conference
R. G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A. Peters, eds.

RETROSPECTIVE APPROXIMATION ALGORITHMS FOR THE MULTIDIMENSIONAL STOCHASTIC ROOT-FINDING PROBLEM

Raghu Pasupathy
Bruce W. Schmeiser

School of Industrial Engineering
Purdue University
West Lafayette, IN 47907, U.S.A.

ABSTRACT

The stochastic root-finding problem (SRFP) is that of solving a system of q equations in q unknowns using only an oracle that provides estimates of the function values. This paper presents a family of algorithms to solve the multidimensional ($q \geq 1$) SRFP, generalizing Chen and Schmeiser's one-dimensional retrospective approximation (RA) family of algorithms. The fundamental idea used in the algorithms is to generate and solve, with increasing accuracy, a sequence of approximations to the SRFP. We focus on a specific member of the family, called the Bounding RA algorithm, which finds a sequence of polytopes that progressively decrease in size while approaching the solution. The algorithm converges almost surely and exhibits good practical performance with no user tuning of parameters, but no convergence proofs or numerical results are included here.

1 INTRODUCTION AND PROBLEM STATEMENT

Consider the well-known problem of solving a $q \times q$ system of linear, known and deterministic equations:

$$\begin{aligned} g_1(x) = a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,q}x_q &= \gamma_1 \\ g_2(x) = a_{2,1}x_1 + a_{2,2}x_2 + \cdots + a_{2,q}x_q &= \gamma_2 \\ &\vdots \\ g_q(x) = a_{q,1}x_1 + a_{q,2}x_2 + \cdots + a_{q,q}x_q &= \gamma_q \end{aligned} \tag{1}$$

where $x = (x_1, x_2, \ldots, x_q)^{\mathrm{T}}$. The equations in System (1) are *known* because the coefficients, and therefore each function g_j, $j \in \{1, 2, \ldots, q\}$, are given to us beforehand. The equations are *deterministic* since there is no uncertainty in the given coefficients. The solution to System (1) is $x^* = A^{-1}\gamma$, where $\gamma = (\gamma_1, \gamma_2, \ldots, \gamma_q)^{\mathrm{T}}$ and A^{-1} is the inverse of the matrix of coefficients $a_{i,j}$, $i, j \in \{1, 2, \ldots, q\}$.

There is no such simple solution to the *stochastic root-finding problem* (SRFP), a generalization of (1) where the equations are possibly non-linear, the form of the equations is *unknown* in the sense that function values at particular x values are available only upon request (i.e., the functions are known only through an oracle), and the system is *stochastic* in the sense that the oracle gives only estimates of the function values.

We are motivated by stochastic simulation where γ_i is the ith performance target, x_i is the ith model parameter, and the oracle is a stochastic simulation model parameterized by x_i. The simulation model, for each value of x, produces a noisy estimate $Y(x)$ of the underlying performance function $g(x)$. The objective is to find the model parameters x^* at which the performance function g attains the performance targets γ.

Chen and Schmeiser (1994a) state the problem more formally as

Given: (a) a constant vector $\gamma \in \Re^q$ and (b) an oracle that returns, for any $x \in \Re^q$, a q-dimensional consistent estimate $\overline{Y}_m(x)$ of a continuous function $g(x) = (g_1(x), g_2(x), \ldots, g_q(x))^{\mathrm{T}}$.

Find: the unique root $x^* \in \Re^q$ satisfying $g(x) = \gamma$, using only the oracle.

SRFPs occur naturally in many physical settings. For example, consider a company that sells q products. Product $i \in \{1, 2, \ldots, q\}$ is periodically re-ordered to a specific threshold x_i. Say we want to determine the thresholds $x_i, i \in \{1, 2, \ldots, q\}$, such that the stock-out probability of the ith product is γ_i. The performance measures are the function $g(x) = (\mathrm{P}(D_1 > x_1), \mathrm{P}(D_2 > x_2), \ldots, \mathrm{P}(D_q > x_q))$, the stock-out probabilities for a given set of thresholds $x = (x_1, x_2, \ldots, x_q)$, where D_i denotes the ith product's periodic demand. Then the problem is

$$\text{find } x \text{ such that } g(x) = \gamma. \tag{2}$$

Problem (2) is a multidimensional stochastic root-finding problem when the function g is not known but a consistent estimator of g is available via a simulation that reflects

the logic and probability assumptions about the inventory system.

This paper presents a family of algorithms for solving multidimensional SRFPs. The objective is to develop numerical algorithms that perform well across a wide range of applications without having to set algorithm parameters. Evaluation criteria include (1) numerical stability, (2) robustness, (3) convergence, (4) computational efficiency and (5) the ability to report solution accuracy. See Chen (1994) for more on (1), (2), (3) and (4).

2 COMPLICATING ISSUES

The SRFP is one stochastic generalization of the problem of solving a non-linear deterministic system of equations. Not surprisingly, this generalization brings with it some complications (Pasupathy and Schmeiser 2003) in addition to the known issues that arise in the context of solving a non-linear system of equations (Ortega and Rheinboldt 1970). We briefly discuss two of these complications in Sections 2.1 and 2.2. These complications are general in that they affect any SRFP algorithm.

2.1 Discontinuous Sample Paths

Let $\overline{Y}_m(x)$, the given consistent estimator of the unknown function $g : \Re^q \to \Re^q$, be defined on some probability space $(\Omega, \mathfrak{F}, P)$. (We write $\overline{y}_m(x, \omega)$ to denote the value of $\overline{Y}_m(x)$ at $\omega \in \Omega$ when the sample size is m.) Each $\omega \in \Omega$ then implicitly generates a *sample-path* function $\overline{y}_m(x, \omega) : \Re^q \to \Re^q$ and a corresponding deterministic *sample-path* problem:

$$\text{find } x^*(\omega) \text{ such that } \overline{y}_m(x^*(\omega), \omega) = \gamma. \tag{3}$$

Problem (3) is an approximation to the original problem, but with an important difference: The underlying root-finding function $\overline{y}_m$ in (3) is not guaranteed to be continuous. In fact, $\overline{y}_m$ is often a step-function when $g(x)$ is a probability and $\overline{Y}_m(x)$ is the associated relative frequency. The implications are two-fold:

1. the solution set of (3) can be empty;
2. the local derivative estimates obtained using the sample path can be meaningless.

2.2 Bias Induced by Finite Sample Size

SRFPs can be solved only approximately by averaging the solutions obtained from fixed sample size realizations of the approximate problem (Atlason, Epelman, and Henderson 2002; Healy and Schruben 1991). The reason is that such solutions can be biased. Consider, for example, the single-dimensional SRFP with $g(x) = x^2$, $\gamma = 2$, and the estimator $\overline{y}_m(x, \omega) = x^2 + \epsilon_m(\omega)$. With a sample size m, the generated approximate problem is to

$$\text{find } x^*(\omega) \text{ such that } x^2 + \epsilon_m(\omega) = 2. \tag{4}$$

The solution of (4) is thus

$$x^*(\omega) = \sqrt{2 - \epsilon_m(\omega)}. \tag{5}$$

We see from (5) that even if $\mathrm{E}[\overline{Y}_m(x)] = g(x)$,

$$\mathrm{E}[x^*(\omega)] \neq \sqrt{2}. \tag{6}$$

Worse, some realizations provide imaginary roots.

3 LITERATURE REVIEW

The current literature on solving SRFPs can be classified into methods based on Classical Stochastic Approximation (CSA) and Retrospective Approximation. In this section, we briefly discuss some important works, salient features and drawbacks of both approaches.

3.1 CSA and Variants

CSA is the original stochastic root-finding algorithm developed by Robbins and Munro (1951). The algorithm as originally proposed has the simple iterative structure

$$X_{k+1} = X_k - a_k(\overline{Y}_k - \gamma),$$

where $k = 0, 1, \ldots, X_0$ is an initial guess of the root x^*, $\overline{Y}_k = \sum_{j=1}^{m} Y_j(X_k)/m$, $\{Y_1(x), \ldots, Y_m(x)\}$ is a random sample from the distribution of $Y(x)$, and $\{a_k\}_{k=0}^{\infty}$ is a predetermined sequence of positive constants satisfying $\sum_{k=0}^{\infty} a_k = \infty$ and $\sum_{k=0}^{\infty} a_k^2 < \infty$. Owing to its simple structure, CSA converges to x^* in mean square under fairly general conditions. Much of the current literature on SRFPs are variants of CSA. For example, Kesten (1958), Venter (1967), Fabian (1968), Wasan (1969), Kushner and Clark (1978) and more recently Andradóttir (1990, 1991) and Spall (1999) focus either on relaxing convergence conditions, increasing convergence rates, and/or on randomizing the step-size sequence while achieving optimal asymptotic efficiency. CSA and its variants usually have a simple structure and extend naturally to multiple dimensions. They, however, have three important drawbacks.

1. The algorithms are generally not *black-box* algorithms in the sense that they are not robust to algorithm parameters. Since little guidance exists on the choice of algorithm parameters, the identification of *good* parameter values for any given problem is often difficult.

2. CSA and its variants usually use $-(\overline{Y}_k - \gamma)$ as the movement direction. Here, $\overline{Y}_k$ is the estimate of the underlying root-finding function at the design point X_k. This is akin to using the *negative gradient* direction in the context of function minimization. While the direction $-(\overline{Y}_k - \gamma)$ is convenient since it does not rely on estimating derivatives, it has been observed that, even for very simple problems, using the direction $-(\overline{Y}_k - \gamma)$, like the negative gradient, can lead to poor practical performance.
3. CSA and many of its variants have used a fixed sample size and provide no guidance on what this sample size should be.

Spall (2000) addresses 2. above by incorporating derivative estimates in the CSA structure. Called Adaptive Stochastic Approximation (ASA), the method has attractive asymptotic properties but exhibits poor practical performance, especially when the initial solution is far from the root. Spall (2000) suggests obtaining an initial solution by running an algorithm such as CSA prior to running ASA. It is unclear, however, as to how long CSA should be run prior to running ASA. Moreover, since CSA also exhibits poor practical performance when the initial solution is far from the root, the benefits associated with running CSA as a preprocessor are not apparent. Spall (2000) does not address 1. or 3. above.

3.2 Retrospective Approximation

Retrospective Approximation (RA) methods form the other broad category of solution techniques for solving SRFPs. RA techniques were first suggested by Healy and Schruben (1991) and later used by Rubinstein and Shapiro (1993), Gürkan, Ozge, and Robinson (1994), Plambeck et al. (1996), Shapiro and Homem-de-Mello (1997) and by Homem-de-Mello, Shapiro, and Spearman (1999) in the context of stochastic optimization. Chen and Schmeiser (1994b, 2001) propose a family of retrospective approximation (RA) algorithms for solving single-dimensional SRFPs. Fundamental to their approach is the concept of sample-path approximation to the function g. At any point x, this is simply $\overline{Y}_m(x)$. Let $\underline{\omega} = \{\omega_1, \omega_2, \ldots, \omega_m\}$ represent the vector of random numbers used to obtain $\overline{y}_m(x; \underline{\omega})$. Then for any fixed $\underline{\omega}$, the function $\overline{y}_m(x; \underline{\omega})$ yields a sample-path approximation to the function g. RA algorithms solve with *increasing accuracy* the sequence of sample-path approximations

$$\overline{y}_{m_k}(x^*(\underline{\omega}_k); \underline{\omega}_k) = \gamma, \tag{7}$$

for $k = 1, 2, \ldots$. The approximations are generated independently and with increasing sample size. At every *retrospective iteration* k, RA algorithms use a deterministic root-finding procedure to solve (7) to a specified accuracy ϵ_k,$\{\epsilon_k\} \to 0$. Individual members of the RA family are produced upon specifying the various algorithm parameters and the deterministic root-finding procedure used to solve (7).

The Bounding RA algorithm is a member of the RA family that works well in practice. Like all members of the RA family, the Bounding RA algorithm, during the kth iteration, uses a deterministic root-finding procedure to solve (7) to a specified accuracy ϵ_k. In one dimension, this is accomplished by *bounding*, i.e. finding two points $x_l, x_u \in \Re$ such that $|x_u - x_l| \leq \epsilon_k$ and whose images straddle the target γ. One-dimensional Bounding RA converges almost surely and exhibits good practical performance (Chen 1994). Three aspects help Bounding RA work well in practice.

1. Monotonicity: The Bounding RA algorithm works on g functions that are monotone increasing in $\Re$, an assumption that is useful for the deterministic root-finding procedure that solves (7), because $\overline{y}_{m_k}(x; \underline{\omega}_k)$ approximates g and thus g's monotonicity provides a strong direction clue.
2. Termination Criterion: The kth iteration in the RA family is terminated upon solving the kth approximate problem (7) to within ϵ_k. More precisely, the kth iteration stops when the deterministic root-finding procedure finds two points $x_l, x_u \in \Re$ such that $|x_u - x_l| \leq \epsilon_k$ and the line segment joining $\overline{y}_{m_k}(x_l; \underline{\omega}_k)$ and $\overline{y}_{m_k}(x_u; \underline{\omega}_k)$ contains the target γ. As we discuss in Section 4.2, this is a simple but powerful idea for root finding on discontinuous functions.
3. Deterministic Root-Finding Logic: The logic for the deterministic root-finding procedure in the single-dimensional Bounding RA algorithm is simple, again due to g's monotonicity assumption. First, sample at a specific x_1 (usually the solution from the previous iteration) and if $\overline{y}_{m_k}(x_1; \underline{\omega}_k) < \gamma$ keep stepping to the right with progressively increasing step sizes until a point x_2 is found such that $\overline{y}_{m_k}(x_2; \underline{\omega}_k) \geq \gamma$. Then use bisection search to find x_l, x_u that satisfy the termination criterion 3. Similar logic holds if $\overline{y}_{m_k}(x_1; \underline{\omega}_k) > \gamma$, but by stepping to the left.

4 THE MULTIDIMENSIONAL BOUNDING RA ALGORITHM

The objective of this paper is to generalize a member of Chen and Schmeiser's RA family, the Bounding RA algorithm, to multiple dimensions. For extending single-dimensional Bounding RA, its various components need to be generalized to multiple dimensions. Not all components generalize easily though. We specifically focus on generalizing the

three key aspects discussed in Section 3.2. We thus ask: (i) what is the analogue of monotonicity in multiple dimensions? (ii) what is the termination criterion in multiple dimensions? (iii) what is the deterministic root-finding logic used to solve (7) in multiple dimensions? Sections 4.1 – 4.3 provide answers to these questions.

4.1 Monotonicity in Multiple Dimensions

We argue that the appropriate generalization of monotonicity in multiple dimensions is as defined by Ortega and Rheinboldt (1970): A function $g : \Re^q \to \Re^q$ is *strictly monotone* if $(x_1 - x_2) \cdot (g(x_1) - g(x_2)) > 0,\ \forall x_1, x_2 \in \Re^q, x_1 \neq x_2$. We chose this definition for three reasons: It is familiar in one dimension, it is useful algorithmically, and it encompasses a large class of functions. We discuss each below.

First, the definition automatically reduces to the class of monotone increasing functions when $q = 1$. Moreover, for any direction $d \in \Re^q$ and any point $x_0 \in \Re^q$, the one-dimensional function $g_{d,x_0}(t) = g(x_0 + td) \cdot d$ (obtained by projecting g onto d) is monotone increasing.

Second, the definition is useful in that direction clues are available. Because g is strictly monotone, $(x - x^*) \cdot (g(x) - \gamma) > 0$ for every $x \neq x^*$ and we know that the direction $g(x) - \gamma$ makes an angle less than $\pi/2$ with the direction $x - x^*$. Therefore, from any point x, the direction $-(g(x) - \gamma)$ "roughly" points in the direction of the root x^*. When $q = 1$, this direction clue is especially useful since there are only two possible directions (right or left) and the vector $-(g(x) - \gamma)$ always points in the correct direction.

Third, the class is large. If g is a *gradient function*, i.e. $\exists G : \Re^q \to \Re$ such that $G' = g$, then G is strictly convex if and only if g is strictly monotone (Hiriart-Urruty and Lemarechal 1993). Therefore, the class of strictly monotone functions in $\Re^q$ is larger than the class consisting of the derivatives of strictly convex differentiable functions, a possible extension of monotonicity that we have rejected.

4.2 Termination Criterion

We now answer the question, "what is the termination criterion for the deterministic root-finding procedure used in the multidimensional Bounding RA algorithm?". Recall the corresponding termination criterion $\mathfrak{S}_1$ in one dimension:
Criterion $\mathfrak{S}_1$: during the kth iteration, the Bounding RA algorithm finds two points $x_l, x_u \in \Re$ such that $|x_u - x_l| \leq \epsilon_k$ and the line segment joining $\overline{y}_{m_k}(x_l; \underline{\omega}_k)$ and $\overline{y}_{m_k}(x_u; \underline{\omega}_k)$ contains γ.

4.2.1 Multidimensional Termination Criterion

Before we generalize Criterion $\mathfrak{S}_1$ to multiple dimensions, we collect four definitions.

Definition 1 *The polytope P_k is the closed convex hull of $j < \infty$ points $x_{k,1}, x_{k,2}, \ldots, x_{k,j} \in \Re^q$:*

$$P_k \equiv \{x : x = \sum_{i=1}^{j} \alpha_i x_{k,i}, \sum_{i=1}^{j} \alpha_i = 1, \alpha_i \geq 0\}.$$

A polytope is thus a bounded closed convex polyhedron that is completely characterized by the points $x_{k,1}, x_{k,2}, \ldots, x_{k,j}$.

Definition 2 *The image $\overline{y}_{m_k}(P_k)$, $\overline{y}_{m_k} : \Re^q \to \Re^q$ of the polytope P_k is the closed convex hull of the points $\overline{y}_{m_k}(x_{k,1}), \overline{y}_{m_k}(x_{k,2}), \ldots, \overline{y}_{m_k}(x_{k,j})$, where $x_{k,1}, x_{k,2}, \ldots, x_{k,j} \in \Re^q$ are the points whose convex hull is P_k.*

Definition 3 *The distance $s(A, B)$ between two sets $A, B \subset \Re^q$ is*

$$s(A, B) = \inf_{x_1 \in A, x_2 \in B} \|x_1 - x_2\|_2,$$

the smallest Euclidean distance between the two sets A and B.

Definition 4 *The size $v(P_k)$ of the polytope $P_k \subset \Re^q$ is*

$$v(P_k) \quad = \quad \sup\{\|x_1 - x_2\|_2, x_1, x_2 \in P_k\},$$

the largest Euclidean distance between any two points in the polytope P_k.

To generalize Criterion $\mathfrak{S}_1$ to multiple dimensions, we propose stopping the iteration when the deterministic root-finding procedure finds a *small-enough* polytope whose image polytope is *close-enough* to the target γ.

Formally, the termination criterion in multiple dimensions is given by Criterion $\mathfrak{S}_{\mathfrak{M}}$.
Criterion $\mathfrak{S}_{\mathfrak{M}}$: during the kth iteration, the Bounding RA algorithm finds a polytope P_k of size at most ϵ_k such that the distance between the image polytope $\overline{y}_{m_k}(P_k; \underline{\omega}_k)$ and γ does not exceed η_k.

The termination criterion $\mathfrak{S}_{\mathfrak{M}}$ reduces to Criterion $\mathfrak{S}_1$ for $q = 1$ by setting $\eta_k = 0$.

4.2.2 Rationale

The key idea in Criterion $\mathfrak{S}_{\mathfrak{M}}$ is to find a small-enough polytope whose image *bounds* or *comes close to bounding* the target γ. Could we have used points rather than polytopes? In other words, what advantages does $\mathfrak{S}_{\mathfrak{M}}$ have compared to an alternative termination criterion such as $\mathfrak{S}'_{\mathfrak{M}}$?
Criterion $\mathfrak{S}'_{\mathfrak{M}}$: during the kth iteration, the Bounding RA

algorithm finds a point X_k such that the distance between its image $\overline{y}_{m_k}(X_k;\underline{\omega}_k)$ and the target γ does not exceed η_k. This question is especially relevant since many deterministic root-finding algorithms have a termination criterion such as $\mathfrak{S}'_{\mathfrak{M}}$.

The problem with Criterion $\mathfrak{S}'_{\mathfrak{M}}$ is that it may never be satisfied when the function $\overline{y}_{m_k}$ is discontinuous in x. We provide one- (Figure 1) and two-dimensional (Figure 2) examples to illustrate this point, but we discuss only Figure 1. Say $\overline{y}_{m_k}$ is the step function shown in Figure 1 and say $\eta_k = 0.1$ in Criterion $\mathfrak{S}'_{\mathfrak{M}}$. Then the deterministic root-finding procedure can never terminate because there exists no point x whose image $\overline{y}_{m_k}(x;\underline{\omega}_k)$ is within $\eta_k = 0.1$ units from the target $\gamma = 0.6$. Therefore, with $\eta_k = 0.1$, the Criterion $\mathfrak{S}'_{\mathfrak{M}}$ is unattainable. In addition, since we don't know before hand as to how close we could get to the target γ during a particular iteration, choosing an attainable η_k value is not guaranteed.

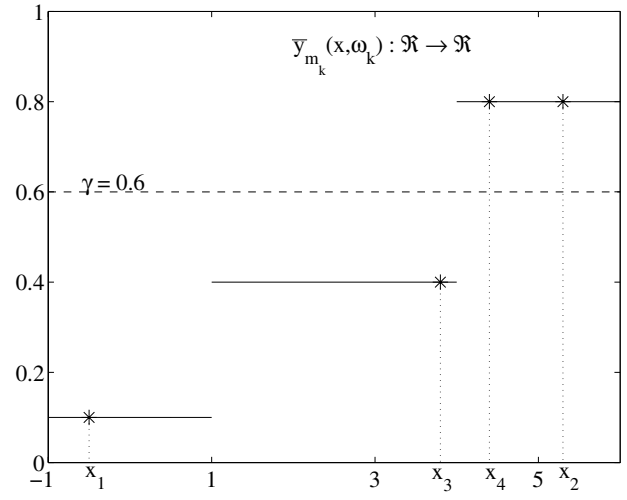

Figure 1: Estimator of g in One Dimension

The above problem does not arise with Criterion $\mathfrak{S}_{\mathfrak{M}}$. Since Criterion $\mathfrak{S}_{\mathfrak{M}}$ seeks a polytope (polytopes in one dimension are line segments obtained by connecting two points), the identification of points x_3 and x_4 terminates the kth iteration since the distance between the image polytope (the line segment obtained by joining $\overline{y}_{m_k}(x_3;\underline{\omega}_k) = 0.4$ and $\overline{y}_{m_k}(x_4;\underline{\omega}_k) = 0.8$) and the target $\gamma = 0.6$ is zero.

4.3 Deterministic Root-Finding Logic

Having generalized to multiple dimensions the notions of monotonicity and the termination criterion, we now present the logic for solving the deterministic root-finding problem that arises from the kth sample size. In other words, we answer the question, "how does the Bounding RA algorithm attain the termination criterion $\mathfrak{S}_{\mathfrak{M}}$ during each iteration?" We first state one more definition.

Definition 5 *The point $x_2 \in \Re^q$ is γ-feasible at $x_1 \in \Re^q$ under $\overline{y}_{m_k} : \Re^q \to \Re^q$ if*

$$\overline{y}_{m_k}(x_1;\underline{\omega}_k)\cdot d - \gamma\cdot d < 0 \text{ and } \overline{y}_{m_k}(x_2;\underline{\omega}_k)\cdot d - \gamma\cdot d < 0.$$

where $d = x_2 - x_1$.

In other words, x_2 is γ-feasible at x_1 under $\overline{y}_{m_k} : \Re^q \to \Re^q$ if the projections of $\overline{y}_{m_k}(x_1)$ and $\overline{y}_{m_k}(x_2)$ onto $x_2 - x_1$ do not straddle the projection of γ onto $x_2 - x_1$. In the single-dimensional case ($q = 1$), if x_2 is greater (lesser) than x_1, x_2 being *γ-feasible* at x_1 under $\overline{y}_{m_k}$ implies that both $\overline{y}_{m_k}(x_1;\underline{\omega}_k)$ and $\overline{y}_{m_k}(x_2;\underline{\omega}_k)$ are lesser (greater) than the target γ.

The point $x_2 \in \Re^q$ is *γ-infeasible* at $x_1 \in \Re^q$ under $\overline{y}_{m_k}$ if it is not *γ-feasible*. The phrase "under $\overline{y}_{m_k}$" in the above definitions is used only when necessary for clarity.

The rudimentary version of the deterministic root-finding logic proceeds as follows. Starting with the point $Z = \overline{X}_{k-1}$, where $\overline{X}_{k-1}$ is obtained during the previous iteration, points $V_1, V_2, \ldots$ are sampled on the ϵ_k sphere centered on Z until a point V_n is obtained so that V_n is *γ-feasible* at Z. The point V_n now becomes the new Z, i.e. Z is reset to the point V_n, and the sampling procedure restarts on the sphere of radius ϵ_k centered on the new Z.

The deterministic root-finding logic, at all times, maintains a polytope P_k formed from the current Z and each of the *γ-infeasible* points $V_1, V_2, \ldots$ sampled on the ϵ_k sphere centered on Z. The termination criterion $\mathfrak{S}_{\mathfrak{M}}$ is checked after each new *γ-infeasible* point, V_i, sampled. The deterministic root-finding logic is stated in more detail in Step 3 of the Bounding RA algorithm in Section 5.

5 BOUNDING RA ALGORITHMS

Algorithm Parameters

x_0:	Initial solution, $x_0 \in \Re^q$.
$\widehat{\sigma}_1$:	Initial $q \times q$ covariance parameter matrix.
m_1:	Initial sample size, $m_1 \in \{1, 2, \ldots\}$.
c_1:	Sample-size multiplier, $c_1 > 1$.
c_2:	Step-size multiplier, $c_2 \geq 1$.
$\{\epsilon_k\}$:	Sequence of domain-space tolerances, $\epsilon_k > 0$.
$\{\eta_k\}$:	Sequence of range-space tolerances, $\eta_k \geq 0$.

Algorithm Logic

Given: default algorithm-parameter values.
Find: the root x^* using

0. Initialize $k = 0$, $\overline{x}_0 = x_0$.
1. Set $k \leftarrow k + 1$, $x_l = \overline{x}_{k-1}$.
2. Independently generate $\underline{\omega}_k = \{\omega_1, \omega_2, \ldots, \omega_{m_k}\}$.

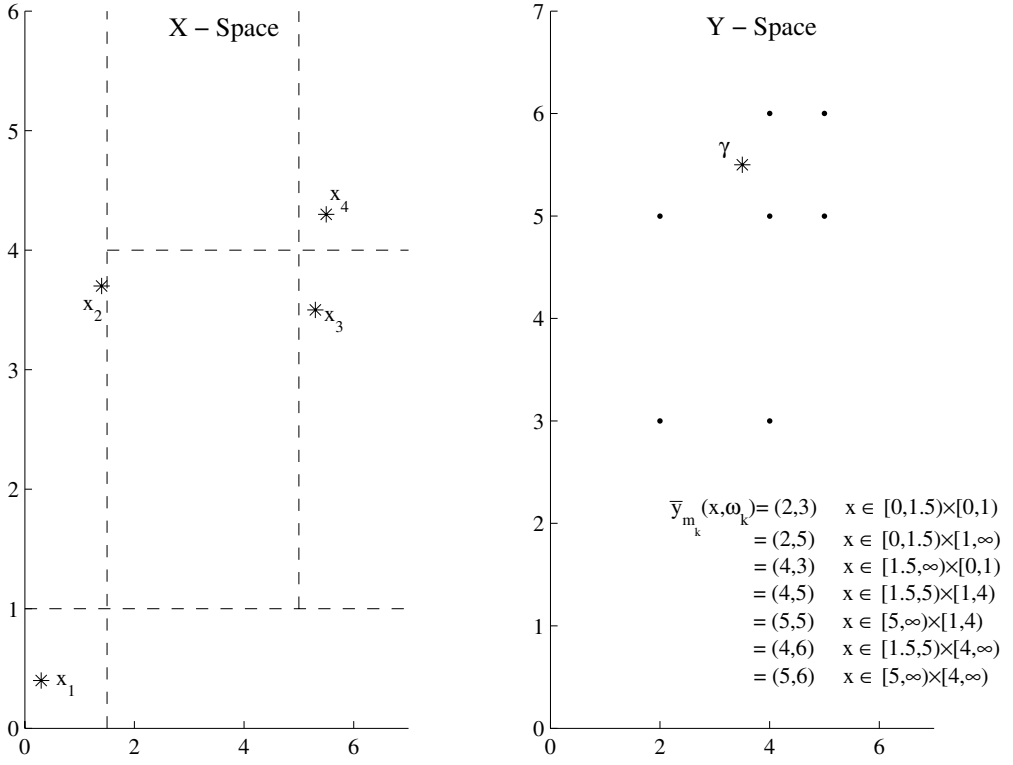

Figure 2: Estimator of g in Two Dimensions

3. Find a polytope P_k such that $v(P_k) \leq \epsilon_k$ and $s(\{\gamma\}, \overline{y}_{m_k}(P_k, \underline{\omega}_k)) \leq \eta_k$:

   3.1 Find points $x_l, x_u \in \Re^q$ such that $\|x_l - x_u\|_2 \leq \epsilon_k$ and the direction $x_u - x_l$ is *γ-infeasible*.

      3.1.1 Initialize $c_s = 1, x_u = x_l$ and $d = \gamma - \overline{y}_{m_k}(x_l, \underline{\omega}_k)$.

      3.1.2 While $\overline{y}_{m_k}(x_l, \underline{\omega}_k) \cdot d - \gamma \cdot d < 0$ and $\overline{y}_{m_k}(x_u, \underline{\omega}_k) - \gamma \cdot d < 0$, repeat Steps 3.1.2 (a)–(f).

         (a) Set $x_l = x_u$.

         (b) Generate a unit direction vector d such that $\overline{y}_{m_k}(x_l, \underline{\omega}_k) \cdot d - \gamma \cdot d < 0$.

         (c) Compute step-size $\delta = c_s d^T \widehat{\sigma_k} d$, where $\widehat{\sigma_k}$ is from Step 6 of the $(k-1)$th iteration.

         (d) Set $c_s = c_2 c_s$.

         (e) Set $x_u = x_l + d\delta$.

         (f) Simulate to obtain $\overline{y}_{m_k}(x_u, \underline{\omega}_k)$.

      3.1.3 While $\|x_l - x_u\|_2 > \epsilon_k$ repeat Steps 3.1.3 (a)–(c).

         (a) Set $x = (x_l + x_u)/2$.

         (b) Simulate to obtain $\overline{y}_{m_k}(x, \underline{\omega}_k)$.

         (c) If $\overline{y}_{m_k}(x, \underline{\omega}_i) \cdot d - \gamma \cdot d < 0$, then set $x_l = x$. Otherwise set $x_u = x$.

   3.2 Set $P_k \equiv \{x_l, x_u\}$.

   3.3 From among x_l, x_u, select the point x whose image is closest to γ:
   If $\|\overline{y}_{m_k}(x_l, \underline{\omega}_k) - \gamma\|_2 \leq \|\overline{y}_{m_k}(x_u, \underline{\omega}_k) - \gamma\|_2$, set $x = x_l$. Otherwise, set $x = x_u$.

   3.4 While $s(\{\gamma\}, \overline{y}_{m_k}(P_k, \underline{\omega}_k)) > \eta_k$, repeat Steps 3.4 (a)–(c).

      (a) Given P_k, generate the next unit direction vector d.

      (b) Simulate to obtain $\overline{y}_{m_k}(x + d\epsilon_k, \underline{\omega}_k)$.

      (c) Update P_k or move the search region:
      If

$$\overline{y}_{m_k}(x, \underline{\omega}_k) \cdot d - \gamma \cdot d \geq 0 \text{ or}$$
$$\overline{y}_{m_k}(x + d\epsilon_k, \underline{\omega}_k) \cdot d - \gamma \cdot d \geq 0,$$

      set $P_k = \{P_k, x + d\epsilon_k\}$; otherwise, set $x_l = x + d\epsilon_k$ and go to Step 3.1.1.

4. Compute the retrospective solution through linear interpolation:

$$x_k = \sum_{j=1}^{n_k} \alpha_j v_j$$

where $v_j \in \Re^q$, $j \in \{1, 2, \ldots, n_k\}$, form P_k; $\alpha_j \geq 0$, $\sum_{j=1}^{n_k} \alpha_j = 1$, $j \in \{1, 2, \ldots, n_k\}$; and $\gamma_P = \sum_{j=1}^{n_k} \alpha_j \overline{y}_{m_k}(v_j, \underline{\omega}_k)$, where γ_P is the projection of γ onto $\overline{y}_{m_k}(P_k, \underline{\omega}_k)$.

5. Compute the root estimate through weighted averaging of retrospective solutions:

$$\overline{x}_k = \sum_{j=1}^{k} m_j x_j / \sum_{j=1}^{k} m_j.$$

6. Compute $\widehat{\mathrm{Var}}(\overline{x}_k)$.
7. If $\widehat{\mathrm{Var}}(\overline{x}_k)$ is small enough, return $\overline{x}_k$. Otherwise, set $m_{k+1} = c_1 m_k$ and go to Step 1.

The stopping rule in Step 7. is purposely imprecise to allow the user to stop whenever desired.

6 CONVERGENCE

The convergence proof of the Bounding RA algorithm involves two steps. First, it can be shown that if each iteration of the algorithm terminates, i.e. if a *small-enough* polytope is found during each iteration whose image polytope is *close-enough* to the target γ, then the sequence of solutions obtained from the algorithm converges to the true root almost surely. Second, it can be shown that, under certain conditions, each iteration of the Bounding RA algorithm terminates. Convergence proofs are not included in this paper.

7 IMPLEMENTATION ISSUES

Algorithm convergence alone is insufficient to guarantee good practical performance. An example is CSA, which converges in mean square, but its practical performance is not robust with respect to algorithm parameters. CSA often requires experimentation with algorithm parameter settings before good practical performance can be achieved. The objective here is to develop *black-box* algorithms to solve SRFPs, i.e. algorithms that work well in practice with no user-tuning of algorithm parameters.

Empirical investigation using the multidimensional Bounding RA algorithm shows that three parameter sequences affect its performance: the sequence $\{\epsilon_k\}$ of error tolerances in the X-space, the sequence $\{\eta_k\}$ of error tolerances in the Y-space and the sequence $\{m_k\}$ of sample sizes used across iterations. Recall from criterion $\mathfrak{S}_\mathfrak{m}$ of Section 4 that the Bounding RA algorithm, during the kth iteration, finds a polytope P_k of size at most ϵ_k such that the distance between the image polytope $\overline{y}_{m_k}(P_k; \underline{\omega}_k)$ and the target γ does not exceed η_k. Thus, while the parameter m_k is a measure of the quality of the kth approximate problem, the parameters ϵ_k and η_k together decide how accurately the kth approximate problem is solved. These parameters are intimately linked in the sense that *more approximate* (small m_k values) problems should be solved with less accuracy (large ϵ_k, η_k values) while *less approximate* (large m_k values) problems should be solved with greater accuracy (small ϵ_k, η_k values).

To guide choice of the sequence $\{\epsilon_k\}$, we assume that the retrospective solution X_k to the kth approximate problem has the variance $\sigma^2/\sqrt{m_k}$ where σ^2 is some underlying matrix of variance constants. If $X_1, X_2, \ldots, X_k$ are the retrospective solutions obtained after the first k iterations, an estimator $\widehat{\sigma^2}$ of σ^2 is

$$\widehat{\sigma^2} = \frac{1}{k-1} \sum_{j=1}^{k} m_j (X_j - \overline{X}_k)(X_j - \overline{X}_k)^{\mathrm{T}},$$

where X_j, $j = 1, 2, \ldots, k$ is a column vector and $\overline{X}_k = \sum_{j=1}^{k} m_j X_j / \sum_{j=1}^{k} m_j$. This motivates choosing $\epsilon_{k+1} = h(\widehat{\sigma})$, where $\widehat{\sigma}$ is the $q \times q$ matrix whose (i, j)th element is the square-root of the (i, j)th element of $\widehat{\sigma^2}$, and $h(\widehat{\sigma})$ is some function of $\widehat{\sigma}$. In practice, two good choices of $h(\widehat{\sigma})$ are $\det(\widehat{\sigma})$ and $d^{\mathrm{T}} \widehat{\sigma} d$ where d is the unit vector along the initial search direction for the $(k+1)$th iteration.

The parameter η_k can be chosen as the size of the image polytope $\overline{y}_{m_k}(P_k, \underline{\omega}_k)$. This choice ensures that the distance between the image polytope $\overline{y}_{m_k}(P_k, \underline{\omega}_k)$ and the target γ is less than the image polytope's size. Finally, for the sample-size sequence $\{m_k\}$, setting $m_1 = 1$ and increasing sample size by a fixed percentage (e.g. 10 percent) each iteration seems to work well in practice.

8 CONCLUDING REMARKS AND FUTURE RESEARCH

The RA family of algorithms presents a promising framework for solving SRFPs. The RA structure is central to the simultaneous goals of proving convergence and achieving good practical performance. RA algorithms use, by definition, increasing sample sizes and decreasing error tolerances. The key idea for computational efficiency in RA algorithms is that previous retrospective iterations, based upon small sample sizes, provide information for efficient numerical solution in future retrospective iterations. Computing in early iterations is inexpensive because sample sizes are small; computing in later iterations is inexpensive because the approximate location of the retrospective root is known.

The multidimensional Bounding RA is a member of the RA family of algorithms. Numerical results on solving a class of one- and two-dimensional problems using the multidimensional Bounding RA algorithm suggests that the algorithm is robust with respect to its parameters. In other words, the algorithm exhibits good practical performance and requires no user tuning.

Four issues are currently under investigation. First, we have been able to prove convergence of the Bounding RA algorithm assuming that the sequences $\{\epsilon_k\}$, $\{\eta_k\}$ and $\{m_k\}$ are deterministic and satisfy $\{\epsilon_k\} \to 0$, $\{\eta_k\} \to 0$, $\{m_k\} \to \infty$. We would like to extend the convergence proofs to include stochastic sequences such as those suggested in Section 7. Second, how should the sample size sequence $\{m_k\}$ be chosen if one is to ensure some sort of optimal algorithm performance? Analytical results on sample size increase in the RA context would be useful even to a wider audience. Third, performance of the multidimensional Bounding RA algorithm in dimensions higher than $q = 2$ is yet to be studied. Last, Bounding RA algorithms have been developed for unconstrained problems. Extensions to include problems defined on constrained spaces would be useful.

REFERENCES

Andradóttir, S. 1990. Stochastic Optimization with Applications to Discrete Event Systems, Doctoral Dissertation, Department of Operations Research, Stanford University, Stanford, California.

Andradóttir, S. 1991. A projected stochastic approximation algorithm. In *Proceedings of the 1991 Winter Simulation Conference*, ed. B. L. Nelson, W. D. Kelton, and G. M. Clark, 954–957. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Atlason, J., M. Epelman and S. G. Henderson. 2002. Call center staffing with simulation and cutting plane methods. *Annals of Operations Research* 127: 333–358.

Chen, H. 1994. Stochastic Root Finding in System Design, Doctoral Dissertation, School of Industrial Engineering, Purdue University, West Lafayette, Indiana.

Chen, H. and B. W. Schmeiser. 1994a. Stochastic root finding: problem definition, examples, and algorithms. In *Proceedings of the Third Industrial Engineering Research Conference*, ed. L. Burke and J. Jackman, 605–610. Norcross, Georgia: Institute of Industrial Engineers.

Chen, H. and B. W. Schmeiser. 1994b. Retrospective approximation algorithms for stochastic root finding. In *Proceedings of the 1994 Winter Simulation Conference*, ed. J. D. Tew, S. Manivannan, D. A. Sadowski, and A. F. Seila, 255–261. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Chen, H. and B. W. Schmeiser. 2001. Stochastic root finding via retrospective approximation. *IIE Transactions* 33: 259–275.

Fabian, V. 1968. On asymptotic normality in stochastic approximation. *Annals of Mathematical Statistics* 39: 1327–1332.

Gürkan, G., A. Y. Ozge and S. Robinson. 1994. Sample-path optimization in simulation. In *Proceedings of the 1991 Winter Simulation Conference*, ed. J. D. Tew, S. Manivannan, D. A. Sadowski, and A. F. Seila, 247–254. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Healy, K. and L. W. Schruben. 1991. Retrospective simulation response optimization. In *Proceedings of the 1991 Winter Simulation Conference*, ed. B. L. Nelson, W. D. Kelton, and G. M. Clark, 954–957. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Hiriart-Urruty, J. -B., and C. Lemarechal. 1993. *Convex Analysis and Minimization Algorithms I: Fundamentals (Grundlehren Der Mathematischen Wissenschaften, No 305)*. New York: Springer-Verlag.

Homem-de-Mello, T., A. Shapiro and M. L. Spearman. 1999. Finding optimal release times using simulation based optimization. *Management Science* 45: 86–102.

Kesten, H. 1958. Accelerated stochastic approximation. *Annals of Mathematical Statistics* 29: 41–59.

Kushner, H. and D. Clark. 1978. *Stochastic Approximation Methods for Constrained and Unconstrained Systems*. New York: Springer-Verlag.

Ortega, J. M. and W. C. Rheinboldt. 1970. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press.

Pasupathy, R. and B. W. Schmeiser. 2003. Some issues in multivariate stochastic root finding. In *Proceedings of the 2003 Winter Simulation Conference*, ed. S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, 574–577. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Plambeck, E. L., B. -R. Fu, S. M. Robinson and R. Suri. 1996. Sample-path optimization of convex stochastic performance functions. *Mathematical Programming* 75: 137–176.

Robbins, H. and S. Munro. 1951. A stochastic approximation method. *Annals of Mathematical Statistics* 22: 400–407.

Rubinstein, R. Y. and A. Shapiro. 1993. *Discrete Event Systems: Sensitivity Analysis and Stochastic Optimization by the Score Function Method*. New York: John Wiley & Sons.

Shapiro, A. and T. Homem-de-Mello. 1997. A simulation-based approach to two-stage stochastic programming with recourse. *Mathematical Programming* 81: 301–325.

Spall, J. C. 1999. Stochastic optimization and the simultaneous perturbation method. In *Proceedings of the 1999 Winter Simulation Conference*, ed. P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, 101–109. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Spall, J. C. 2000. Adaptive stochastic approximation by the simultaneous perturbation method. *IEEE Transactions on Automatic Control* 45 (10): 1839–1853.

Venter, H. J. 1967. An extension of the Robbins-Monro procedure. *Annals of Mathematical Statistics* 38: 181–190.

Wasan, M. T. 1969. *Stochastic Approximation*. Cambridge University Press.

AUTHOR BIOGRAPHIES

RAGHU PASUPATHY is a Ph.D. candidate in the School of Industrial Engineering at Purdue University. His dissertation has focused on designing efficient algorithms for the solution of general stochastic root-finding problems. His broad research interests are in the area of stochastic operations research. His e-mail address is <pasupath@purdue.edu>.

BRUCE W. SCHMEISER is a professor in the School of Industrial Engineering at Purdue University. His interests lie in applied operations research, with emphasis in stochastic models, especially the probabilistic and statistical aspects of stochastic simulation. He is an active participant in the Winter Simulation Conference, including being Program Chair in 1983 and chairing the Board of Directors from 1988–1990. His e-mail address is <bruce@purdue.edu>.