IIE Transactions (1997) **29**, 233–243

Optimization of discrete event systems via simultaneous perturbation stochastic approximation

MICHAEL C. FU[1] and STACY D. HILL[2]

[1] *College of Business and Management, Institute for Systems Research, University of Maryland, College Park, MD 20742, USA*
[2] *The Johns Hopkins University, Applied Physics Laboratory, Laurel, MD 20723, USA*

Received August 1995 and accepted September 1996

We investigate the use of simultaneous perturbation stochastic approximation for the optimization of discrete-event systems via simulation. Application of stochastic approximation to simulation optimization is basically a gradient-based method, so much recent research has focused on obtaining direct gradients. However, such procedures are still not as universally applicable as finite-difference methods. On the other hand, traditional finite-difference-based stochastic approximation schemes require a large number of simulation replications when the number of parameters of interest is large, whereas the simultaneous perturbation method is a finite-difference-like method that requires only *two* simulations per gradient estimate, regardless of the number of parameters of interest. This can result in substantial computational savings for large-dimensional systems. We report simulation experiments conducted on a variety of discrete-event systems: a single-server queue, a queueing network, and a bus transit network. For the single-server queue, we also compare our work with algorithms based on finite differences and perturbation analysis.

1. Introduction

We consider the problem of optimizing a stochastic discrete event system under the overriding assumption that the system cannot be adequately modeled using analytical means, e.g., optimizing the operations of a complex manufacturing system. For such systems, simulation is often used to estimate performance. Under suitable conditions, the resulting optimization problem reduces to finding the zero of the objective function gradient, so that gradient-based techniques based on stochastic approximation can be applied. Such optimization techniques attempt to mimic steepest-descent algorithms from the deterministic domain of non-linear programming, with two major complications: only *noisy* estimates of system performance are available, and *gradients* are not automatically available. Techniques such as perturbation analysis (PA) or the likelihood ratio method (cf. [1]) provide efficient means for estimating gradients based on simulation sample paths, but these techniques are not universally applicable, in which case gradient estimates based on finite differences of system performance estimates are employed. However, the computational requirements of finite-difference methods grow linearly with the dimension of the controllable parameter vector, making it burdensome for high-dimensional problems.

This paper considers a technique called simultaneous perturbation stochastic approximation (SPSA) that requires minimal assumptions on the system of interest [2]. Like finite-difference-based stochastic approximation procedures, SPSA uses only estimates of the objective function itself, so it does not require detailed knowledge of system dynamics and input distributions; hence, it is applicable to any system that can be simulated. Moreover, SPSA requires only *two* sample estimates to calculate a gradient estimate, regardless of the dimension of the parameter vector, and therefore requires significantly fewer simulations than finite differences for estimating gradients in high dimensions.

To be more specific, let $\theta = ((\theta)_1, \ldots, (\theta)_p) \in \Theta \subset \mathscr{R}^p$ denote a p-dimensional vector of controllable (adjustable) parameters and ω the stochastic effects (e.g., the random numbers in a simulation), where Θ is a compact set. Let $L(\theta, \omega)$ denote the sample path performance measure of interest, with expectation $E[L(\theta, \omega)]$, and define the objective function

$$J(\theta) = E[L(\theta, \omega)] + C(\theta), \tag{1}$$

where C is an analytically available function, usually representing some cost on the parameter. The problem is to find

$$\theta^* \triangleq \arg\min\{J(\theta) : \theta \in \Theta\}.$$

For example, in one problem that we consider – the minimization of mean system time in a queueing network – the ith component of θ is the mean service time at the ith station. The stochastic approximation (SA) algorithm for solving $\nabla J = 0$, where '∇' indicates the gradient operator, is given by the following iterative scheme that is basically the stochastic version of a steepest descent algorithm:

$$\theta_{n+1} = \Pi_{\Theta}(\theta_n - a_n \hat{g}_n), \tag{2}$$

where $\theta_n = ((\theta_n)_1, \ldots, (\theta_n)_p)$ represents the nth iterate, $\hat{g}_n$ represents an estimate of the gradient ∇J at θ_n, $\{a_n\}$ is a positive sequence of numbers converging to 0, and Π_{Θ} denotes a projection on Θ.

Aside from the earlier finite-difference-based work of Azadivar and Talavage [3], the application of SA to the optimization of stochastic discrete-event systems has been a fairly recent development, and has been coupled closely with the emergence of an entire research area devoted to simulation-based direct gradient estimation techniques (see [1]). It is well known that these direct gradient estimates yield the best convergence rate (cf. [4] and references therein), and this is borne out in simulation experiments as well. For example, following the earlier empirical work of Suri and Zazanis [5] and Suri and Leung [6], L'Ecuyer *et al.* [7] compared a large number of different SA algorithms on a single-server queue problem (with a scalar parameter), and found that the method utilizing an infinitesimal perturbation analysis gradient estimator (cf. [8]) outperformed all the others. Other related work includes Fu [9], Andradóttir [10], and Wardi [11]. However, for many problems direct gradient estimates may not be readily available, in which case gradient estimates based on (noisy) measurements of the performance measure itself are the only recourse. When the dimension, p, of the parameter is large, traditional finite-difference methods can become prohibitively expensive, whereas the method of simultaneous perturbations allows the gradient to be estimated with two simulations using only estimates of the performance measure itself. In sum, the SPSA method exhibits three desirable characteristics:

(1) **generality** – its advantage over direct estimation methods such as PA and the likelihood ratio method is its applicability to any system that can be simulated;
(2) **efficiency** – its advantage over usual finite-difference methods is its practicality for high-dimensional problems;
(3) **ease of use** – it is as easy to apply as finite-difference methods.

SPSA has been applied successfully to nonlinear control problems using neural networks [12]. In this paper, we investigate the application of SPSA to simulation optimization of stochastic discrete-event systems by conducting simulation experiments on three different optimization problems: a single-server queueing problem, a queueing network problem, and a transportation problem. Preliminary experimental results were reported in Hill and Fu [13,14]. In the first example we compare the performance of SPSA with SA algorithms based on finite-difference estimates and on infinitesimal perturbation analysis estimates. The purpose is not to make definitive conclusions as to superiority, but simply to demonstrate reasonably comparable performance on the smallest multi-dimensional problem possible (two dimensions). The second example was chosen to test SPSA on a higher-dimensional problem, and the third example attempted to test it on a more 'practical' problem, that of reducing transfer waiting times in a transit network. Difficulties experienced in naive application of the method are discussed. The rest of the paper is organized as follows. In Section 2 we present the simultaneous perturbation gradient estimator, contrasting it with the usual finite difference estimator. In Section 3 we describe the three problem settings. Discussion of convergence for the SPSA algorithm applied to the examples is provided in Section 4, with a proof outlined for the single-server queue. The results of simulation experiments are provided in Section 5. Section 6 contains a summary and conclusions.

2. Simultaneous perturbations

In this section, we present and contrast finite difference estimators and simultaneous perturbation estimators for a performance measure gradient $\nabla E[L]$. Let e_i denote the unit vector in the ith direction of $\mathscr{R}^p$, t_n the simulation length of the nth iteration, s_n the starting state for the nth iteration, $L_t(\theta, \omega, s)$ the observed (sample) system performance at θ on sample path ω for duration t starting from state s, $(\hat{g}_n)_i$ the ith component of $\hat{g}_n$ (an estimate of $\nabla E[L]$ at θ_n), and $\{c_n\}$ a positive sequence converging to 0.

The usual symmetric difference (SD) estimator for $\nabla E[L]$ is given by

$$(\hat{g}_n)_i = \frac{L_{t_n}(\theta_n + c_n e_i, \omega_n^{i+}, s_n^{i+}) - L_{t_n}(\theta_n - c_n e_i, \omega_n^{i-}, s_n^{i-})}{2c_n}, \tag{3}$$

where ω_n^{i+} and ω_n^{i-} denote the pair of sample paths used for the ith component of the nth iterate of the algorithm with respective starting states s_n^{i+} and s_n^{i-}. If the method of common random numbers is employed, then $\omega_n^{i-} = \omega_n^{i+} = \omega_n$ (cf. [7]).

Let $\{\Delta_n\}$ be an i.i.d. vector sequence of perturbations of i.i.d. components $\{(\Delta_n)_i, i = 1, \ldots, p\}$ symmetrically distributed about 0 with $E|(\Delta_n)_i^{-2}|$ uniformly bounded (see [2] for a definition). Then the simultaneous perturbation (SP) estimator is given by

$$(\hat{g}_n)_i = (L_n^+ - L_n^-)/[2c_n(\Delta_n)_i], \tag{4}$$

$$\text{where} \quad L_n^+ = L_{t_n}(\theta_n + c_n\Delta_n, \omega_n^+, s_n^+), \tag{5}$$

$$L_n^- = L_{t_n}(\theta_n - c_n\Delta_n, \omega_n^-, s_n^-), \tag{6}$$

ω_n^+ and ω_n^- denote the pair of sample paths used for the nth iterate of the algorithm with respective starting states s_n^+ and s_n^-, and L_n^+ and L_n^- are performance estimates at the parameter value θ_n *simultaneously* perturbed in all directions. Again, if the method of common random numbers is employed, then $\omega_n^+ = \omega_n^- = \omega_n$.

The key point to note is that each estimate $L_t(\theta, \omega, s)$ is computationally expensive relative to the generation of Δ. We see that the SD estimator (3) requires a different pair of estimates in the *numerator* for each parameter to estimate the gradient, thus requiring $2p$ simulations, whereas the SP estimator (4) uses the *same* pair of estimates in the numerator for all parameters, and instead the *denominator* changes. Thus, SP requires only two discrete-event simulations at each SA iteration to form a gradient estimate.

Remark. In both cases there is a practical and technical issue when $\theta_n \pm c_n\Delta_n$ or $\theta_n \pm c_n e_i$ lies outside the constraint set. Usually this is handled by taking the projection (nearest point) on Θ.

3. Problem settings

We consider three optimization problems, all basically minimizing a waiting time performance measure.

3.1. *A single-server queue*

Consider a single-server queue with Poisson arrivals and service times from a uniform distribution (an $M/U/1$ queue). The goal is to minimize the mean steady-state time in system T with costs for improving service times. Specifically, we wish to determine the values of the two parameters in the uniform service time distribution $U((\theta)_1 - (\theta)_2, (\theta)_1 + (\theta)_2)$ to minimize the objective function

$$J(\theta) = E[T] - C_1 \cdot (\theta)_1 - C_2 \cdot (\theta)_2, \quad \theta \in \Theta, \qquad (7)$$

where C_1 and C_2 represent 'costs' for reducing the service time mean and 'variability,' respectively, $\Theta = \{\theta : 0 < \theta_{\min} \leq (\theta)_2 \leq (\theta)_1 \leq \theta_{\max} < 1/\lambda\}$ is a constraint set that ensures stability of the system, and λ is the arrival rate. This example was considered in [15], and the objective function (7) fits the general form given by (1). The gradient estimate is of the form

$$\hat{g}_n = \hat{\nabla} E_{\theta_n}[T] - \begin{pmatrix} C_1 \\ C_2 \end{pmatrix},$$

where $\hat{\nabla} E_{\theta_n}[T]$ represents the SP estimate of $\nabla E[T]$ at θ_n.

3.2. *Queueing network optimization problem*

Consider an open queueing network with N stations and general customer routes. Again, the goal is to minimize mean steady-state time in the system T, this time subject to a constraint on the allocation of mean service times throughout the system (similar to an assembly line balancing problem):

$$\min_{\theta \in \Theta} E[T], \quad \text{subject to} \quad \sum_{i=1}^{N} (\theta)_i = K,$$

where $(\theta)_i$ is the mean service time at the ith station, K is the system total for mean processing times over all the stations, $\Theta = \{\theta : 0 < (\theta_{\min})_i \leq (\theta)_i \leq (\theta_{\max})_i < 1/\lambda_i\}$ is the stability constraint set, and $\lambda_i, i = 1, \ldots, N$, is the total arrival rate at station i. Moving the constraint into the objective function, the gradient estimate is of the form

$$\hat{g}_n = \hat{\nabla} E_{\theta_n}[T] - \frac{1}{N} \sum_{i=1}^{N} (\hat{\nabla} E_{\theta_n}[T])_i,$$

where again $\hat{\nabla} E_{\theta_n}[T]$ represents the SP estimate of $\nabla E[T]$ at θ_n.

3.3. *Transportation network*

Consider a transit network with bus lines traveling in four directions on a grid: east, west, north, and south. Transfers occur, for instance, from a west-bound line to a north-bound line, and multiple transfers are possible. As summarized in [16], there are two basic approaches to this problem: timed transfer and transfer optimization. The former focuses on coordinating the transfer points, and is more applicable for networks where transfers constitute a relatively smaller proportion of overall traffic, e.g., intercity trains and planes. This approach would not be appropriate for a large transit network, such as is found in a large urban bus network, where there are many decentralized transfers. In this case, transfer optimization is usually employed, where the decision is to specify the departure times of the first bus on a line, called the *offset times*.

In transfer optimization, the following are assumed to be given: the network routes; the headways, defined as the times between adjacent buses on the same line (assumed to be constant and equal); the transfer points; the passenger traffic; and transfers. Once the headways are given, the offset times determine the timetable, or schedule. Let p be the number of transit lines, $\theta = ((\theta)_1, \ldots, (\theta)_p)$ the vector of offset times for the transit network, and $\Theta = \Theta_1 \times \cdots \times \Theta_p$ the constraint set, where Θ_i is the set of allowable offset times for transit line i, $i = 1, \ldots, p$. The goal is to minimize the total expected waiting time in the network. Bookbinder and Désilets [16] formulate this problem as a mathematical program, under the assumption that the sets $\Theta_i, i = 1, \ldots, p$, are discrete and finite. The formulation is equivalent to the well-known quadratic assignment problem in facilities layout planning, and hence is NP-complete.

The key assumption in the mathematical programming formulations is that the expected waiting times be analytically available. Incorporation of stochastic effects in the bus travel times and passenger arrivals may preclude this (although empirically based approximate expressions are often used in practice), in which case simulation must be employed. Thus, the optimization problem is

$$\min_{\theta\in\Theta} E[\overline{W}_N(\theta)], \tag{8}$$

where $\overline{W}_n$ is the mean waiting time over n boardings, N is the number of boardings in a day, and now $\Theta_i = [0, K_i]$ is a continuous interval, K_i being the maximum allowable offset time on transit line i. Two crucial, but unverified, assumptions in applying SA are that the objective function in (8) is sufficiently smooth and that the optimum (at least local) is found at a zero gradient point. As we shall discuss in the experimental results section, this need not always be the case in practice, causing difficulty in applying an SA algorithm.

4. Convergence of the SPSA algorithm

The focus of our work is not on theoretical convergence issues. However, some of the issues that arise in the theory are also of practical consideration in applications. In this section we discuss the key issues, and then sketch a proof for one of our examples, the single-server queue. The basic convergence requirements place conditions on the following (cf. [17]):

(1) objective function $J(\theta)$ (differentiable and either convex or unimodal);
(2) step-size sequence $\{a_n\}$;
(3) bias and variance of gradient estimate $\hat{g}_n$.

Let

$$b_n = E[\hat{g}_n|\theta_n, s_n] - \nabla J(\theta_n), \tag{9}$$

$$e_n = \hat{g}_n - E[\hat{g}_n|\theta_n, s_n], \tag{10}$$

i.e., b_n and e_n are the bias and the noise, respectively, in the gradient estimate. In the case of SPSA, the bias and variance requirement translates into conditions on the following:

- noise sequence $\{e_n\}$;
- difference sequence $\{c_n\}$ for the gradient estimate;
- simultaneous perturbation sequence $\{\Delta_n\}$.

Traditional finite-difference methods have similar conditions, except for the obvious absence of the last sequence.

Denote the gradient of $E[L(\theta_n, \omega)]$ by $g_n = \nabla E[L]$, where $E[L]$ is assumed continuously differentiable on Θ. Then for the SP gradient estimator $\hat{g}_n$ of g_n, we have the following lemmas concerning the corresponding bias and noise terms (see also [2]):

Lemma 1. *Assume $\{(\Delta_n)_i\}$ are all mutually independent with zero mean, bounded second moments, and $E|(\Delta_n)_i^{-1}|$ uniformly bounded on Θ. Then $b_n \to 0$ w.p. 1.*

Proof. Expressing the expectation of the SP gradient estimate (4) as a conditional expectation on Δ_n, and using Taylor's theorem to expand $E[L_n^+]$ and $E[L_n^-]$ as defined by (5) and (6), respectively, we write

$$\begin{aligned}
&E[(\hat{g}_n)_i] \\
&= E\left[\frac{L_n^+ - L_n^-}{2c_n(\Delta_n)_i}\right] = E\left[E\left[\frac{L_n^+ - L_n^-}{2c_n(\Delta_n)_i}\middle|\Delta_n\right]\right] \\
&= E\left[\frac{E[L_n^+|\Delta_n] - E[L_n^-|\Delta_n]}{2c_n(\Delta_n)_i}\right] \\
&= E\left[\frac{E[L(\theta_n + c_n\Delta_n, \omega)] - E[L(\theta_n - c_n\Delta_n, \omega)]}{2c_n(\Delta_n)_i}\right] \\
&= E\left[\frac{\begin{array}{r}(E[L(\theta_n, \omega)] + g_n^{\mathrm{T}}\Delta_n c_n + O(\Delta_n^2 c_n^2)) \\ -\,(E[L(\theta_n, \omega)] - g_n^{\mathrm{T}}\Delta_n c_n + O(\Delta_n^2 c_n^2))\end{array}}{2c_n(\Delta_n)_i}\right] \\
&= E\left[\frac{g_n^{\mathrm{T}}\Delta_n}{(\Delta_n)_i} + \frac{O(\Delta_n^2 c_n)}{(\Delta_n)_i}\right] \\
&= E\left[\frac{\sum_{j=1}^{p}(g_n)_j(\Delta_n)_j}{(\Delta_n)_i}\right] + O\left(c_n E\left[\frac{\sum_{j=1}^{p}[(\Delta_n)_j]^2}{(\Delta_n)_i}\right]\right) \\
&= (g_n)_i + O(c_n),
\end{aligned}$$

where the superscript T denotes the transpose operator. The last step follows from the conditions that $\{(\Delta_n)_j, j \neq i\}$ have zero mean and are independent of $(\Delta_n)_i$, $E|(\Delta_n)_i^{-1}|$ is uniformly bounded, and $\{(\Delta_n)_j\}$ have bounded second moments. Applying the definition of b_n given by (9), the result then follows since $c_n \to 0$.

Remarks. Our result differs slightly from Lemma 1 in [2]. We require weaker conditions, because we are considering a constrained optimization problem on a compact set Θ and because we do not achieve $O(c_n^2)$ bias. As a result, the existence of third derivatives is not needed, and the perturbations need not be symmetrical (which would give $E[(\Delta_n)_j^3] = 0$ in the proof).

Lemma 2. *Let $L(\theta_n, \omega)$ have uniformly (in Θ) bounded second moments, and $E[|(\Delta_n)_i|^{-2}]$ be uniformly bounded on Θ. Then $E[e_n^{\mathrm{T}} e_n]$ is $O(c_n^{-2})$.*

Proof. Since $L(\theta_n, \omega)$ has uniformly bounded second moments, similar arguments used in the proof of Lemma 1 can be used to establish that the conditional variance of $(\hat{g}_n)_i$ is $O(c_n^{-2}(\Delta_n)_i^{-2})$. Applying the definition of e_n given by (10), the result follows from the assumption that $E[|(\Delta_n)_i|^{-2}]$ is uniformly bounded.

One form of a general convergence result is the following:

Proposition 1 [7]. *For the SA algorithm* (2), *let* $\sum_{n=1}^{\infty} a_n = \infty$. *If*

(i) $J(\theta)$ *is differentiable for each* $\theta \in \Theta$, *and either convex or unimodal*;
(ii) $b_n \to 0$ *w.p.* 1; *and*
(iii) $\sum_{n=1}^{\infty} E[e_n^{\mathrm{T}} e_n] a_n^2 < \infty$ *w.p.* 1;

then $\theta_n \to \theta^*$ *w.p.* 1.

In practice, it may be difficult to verify the conditions on the objective function, since simulation is applied to those systems for which analytical properties are not readily available. In our transportation application, there are potential discontinuities in the objective function due to the nature of the underlying system. Even for the simple single-server queue example, convergence proofs require a significant amount of analysis of the system [18].

There are also issues specific to the application to stochastic discrete-event systems (e.g., [7, 9, 10, 18–23]. Optimization problems for these systems fall into two general classes:

- finite-horizon problems;
- steady-state (infinite-horizon) problems.

The two queueing examples are steady-state problems, whereas the transportation application is a finite-horizon problem (the horizon being a day). Steady-state problems are more difficult, because they introduce two sources of bias:

(1) finite-difference estimate for a gradient;
(2) finite-horizon estimate for a steady-state performance measure.

Finite-horizon problems are easier to handle, since the second source of bias is absent. In fact, to handle the second source of bias in steady-state problems often requires that the observation length t_n increase with n without bound [18]. An alternative is to use regenerative-based estimators that take the observation length as some multiple of regenerative cycles, e.g., busy periods in a queue. These types of estimator also allow for simpler convergence proofs [9]. We now sketch such a proof for the single-server queue example, largely following L'Ecuyer and Glynn [18].

Let

$$N(\theta, \omega) = \text{the number of customersin a busy period,} \tag{11}$$

$$H(\theta, \omega) = \sum_{j=1}^{N} T_j = \text{the sum of customer system times in a busy period,} \tag{12}$$

where T_j is the system time of the jth customer in a busy period. We drop the explicit dependence of N on θ and ω for notational brevity. Note that there is no dependence of N and H on an initial state, since all busy periods start empty. For the stable single-server queue, we have the well-known regenerative result (cf. [18]):

$$E[T] = E[H]/E[N].$$

Taking the gradient,

$$\nabla E[T] = \frac{E[N]\nabla E[H] - E[H]\nabla E[N]}{(E[N])^2},$$

we consider the problem of finding the zero of

$$(E[N])^2 \nabla J = E[N]\nabla E[H] - E[H]\nabla E[N] - (E[N])^2 \begin{pmatrix} C_1 \\ C_2 \end{pmatrix},$$

which is equivalent to finding the zero gradient of (7). The reason for this transformation is that we can then avoid the question of the bias in estimating steady-state performance over a finite horizon. A new estimator taken over two separate busy periods, one with $\theta_n^+ = \theta_n + c_n \Delta_n$ and the other with $\theta_n^- = \theta_n - c_n \Delta_n$, would be of the form

$$\begin{aligned}(\hat{g}_n)_i &= \frac{H_n^+ - H_n^-}{2c_n(\Delta_n)_i} N_n^- - H_n^- \frac{N_n^+ - N_n^-}{2c_n(\Delta_n)_i} - N_n^+ N_n^- C_i \\ &= \frac{H_n^+ N_n^- - H_n^- N_n^+}{2c_n(\Delta_n)_i} - N_n^+ N_n^- C_i,\end{aligned}$$

where $H_n^+ = H(\theta_n^+, \omega_n^+), H_n^- = H(\theta_n^-, \omega_n^-), N_n^+ = N(\theta_n^+, \omega_n^+)$, and $N_n^- = N(\theta_n^-, \omega_n^-)$.

Lemma 3. *$H_n^{\pm}$ and $N_n^{\pm}$ have uniformly* (in Θ) *bounded second moments.*

Proof. This follows from Proposition 8 in [18], as the $U((\theta)_1 - (\theta)_2, (\theta)_1 + (\theta)_2)$ distribution satisfies Assumptions A(i) and B there, being uniformly bounded and having uniformly bounded second moments.

Proposition 2. Suppose

$$\sum_{n=1}^{\infty} a_n = \infty, \quad \sum_{n=1}^{\infty} \left(\frac{a_n}{c_n}\right)^2 < \infty, \quad \textit{and } E\left[|(\Delta_n)_i|^{-2}\right]$$

is uniformly bounded on Θ. *Then for the SPSA algorithm applied to the M/U/*1 *queue problem*, $\theta_n \to \theta^*$ *w.p.* 1.

Proof Sketch. Following closely the proof of Proposition 4 in [18], we consider each of the three conditions in Proposition 1. Condition 1 is proved as Propositions 15 and 19 there. For Condition 2, Proposition 15 there establishes that $E[N]$ and $E[H]$ are continuously differentiable, uniformly on Θ, so the result follows from Lemma 1. Finally, Lemmas 2 and 3 and the assumption that $E[|(\Delta_n)_i|^{-2}]$ is uniformly bounded imply that $E[e_n^{\mathrm{T}} e_n]$ is $O(c_n^2)$. The last result and the assumption of square summability of the ratio a_n/c_n imply Condition 3.

Remark. Although the proof was carried out for our uniform distribution example, similar arguments hold for any distribution satisfying Assumptions A(i) and B of [18].

5. Experimental results

We now return to the examples of Section 3, for which we performed numerical experiments using the SPSA algorithm. Implementation of SPSA requires three sequences: the step-size multiplier sequence $\{a_n\}$; the difference sequence $\{c_n\}$ for the gradient estimate; and the vector of simultaneous perturbations sequence $\{\Delta_n\}$. The first two must be positive sequences converging to zero at the appropriate rate. In our experiment we took $a_n = a/n^{\alpha}$ and $c_n = c/n^{\beta}$, where α, β, a, and c are constants to be selected, subject to $\alpha \leq 1$ and $\alpha - \beta > 0.5$, to satisfy the conditions of Proposition 2. For the $\{\Delta_n\}$ sequence, we took i.i.d. symmetric Bernoulli random variables, i.e., $P(\Delta_n = 1) = P(\Delta_n = -1) = 0.5$, in all of our simulation experiments, following [2].

5.1. *Single-server queue*

For the single-server queue example, we compared the SPSA implementation with SA implementations utilizing finite-difference (both one-sided and symmetric) gradient estimates and PA estimates that require only a single simulation per estimate. We considered six sets of values of C_1 and C_2. Table 1 gives the resulting optimal values and the corresponding values of the objective function and the two partial second derivatives, from which the value of a is determined, as described in the next paragraph. As noted in [15], for some values of C_1 and C_2, the theoretical optimal solution could lie arbitrarily close to the boundary of the constraint set. The minimum occurs at a zero gradient point if $C_1 > 6C_2^2 + 3C_2 + 1$, a condition satisfied in all six of the cases we considered. In this case, the minimum occurs at

$$\theta^* = \left(1 - \frac{1}{\sqrt{\kappa}}, \frac{3C_2}{\sqrt{\kappa}}\right)\frac{1}{\lambda}, \quad \kappa = 2C_1 - 3C_2^2 - 1. \tag{13}$$

Further implementation values are as follows: $\lambda = 1$; $c = 0.001$; starting point of $\theta_1 = (0.5, 0.3)$; 100 customer completions per SA iteration; 1000 iterations per replication (total budget of 100 000 customers/replication); a equal to geometric mean of second derivatives (approximated to one significant figure); 40 independent replications. In general, of course, the parameter a could not be calculated 'optimally' in advance, since the objective function is unknown.

Since analytical results are available, performance of the algorithms was measured by the value of $J(\theta_n)$, lower being better. The results are summarized in Table 2. The headings SDSA, FDSA, and PASA refer to stochastic approximation algorithms based on symmetric differences, one-sided (forward) finite differences, and perturbation analysis, respectively; the starting objective function value and the minimum objective function value are included in the left-hand column as benchmarks. The table gives the mean of the estimated minimum $\pm$ standard error (SE), based on 40 independent replications, for the algorithm after 500 customers simulated and after 1000 customers simulated. In terms of simulation budget, the $n = 1000$ case of SPSA corresponds approximately to the $n = 500$ case of SDSA, since each iteration of SDSA requires twice ($p = 2$) as many simulations as each iteration of SPSA. In this limited set of cases, SPSA performs comparably to the other techniques. Compared with the finite differences, it does so with half as many computations.

5.2. *Queueing network*

We considered two cases: a symmetric case with deterministic service times, and an asymmetric case with exponential service times. We chose these two cases because analytical optima can be determined to evaluate the performance of the algorithm. We considered a network of five stations ($N = 5$), took $1/\lambda = 8$ as our mean interarrival time, and constrained the total of the mean service times to be $K = 20$.

In the balanced case, there were N possible customer process routes that were cyclic and complete (i.e., $1 \longrightarrow 2 \longrightarrow \cdots \longrightarrow N, 2 \longrightarrow 3 \longrightarrow \cdots \longrightarrow N \longrightarrow 1, \cdots$), and each route was equally likely to be chosen (w.p. $1/N$). By symmetry, the optimal solution is the balanced solution $\theta^* = (K/N, \ldots, K/N)$. We considered two sets of starting values for the parameters: $\theta_1 = (1,7,2,5,5)$, and $\theta_1 = (4,4,4,4,4)$, the latter actually being the optimum. Note that since service times are deterministic, the only randomness in the system comes from the interarrival times.

Table 1. Optimization cases for $M/U/1$ queue ($\lambda = 1$)

Case	C_1	C_2	θ^*		$J(\theta^*)$	$(\nabla^2 J(\theta^*))_1$	$(\nabla^2 J(\theta^*))_2$	a
1	1.28125	0.00125	0.2	0.003	−0.03125	1.953	0.4167	1.0
2	1.28969	0.075	0.2	0.180	−0.03969	1.974	0.4167	1.0
3	2.5	0.002	0.5	0.003	−0.5000	8.000	0.6667	0.4
4	2.6536	0.32	0.5	0.480	−0.6536	8.614	0.6667	0.4
5	13.0	0.005	0.8	0.003	−8.000	125	1.6667	0.1
6	15.535	1.3	0.8	0.780	−10.535	150	1.6667	0.1

Table 2. SA results: $J(\theta_n)$ as a function of the number of iterations, n, mean ± SE

Case	*SPSA(n)*		*SDSA(n)*		*FDSA(n)*		*PASA(n)*	
$J(\theta_1)/J(\theta^*)$	*500*	*1000*	*500*	*1000*	*500*	*1000*	*500*	*1000*
+0.139	−0.0299	−0.0294	−0.0309	−0.0311	−0.0309	−0.0311	−0.0309	−0.0311
−0.03125	±0.0003	±0.0004	±0.0002	±0.0001	±0.0002	±0.0001	±0.0002	±0.0001
+0.112655	−0.0394	−0.0396	−0.0390	−0.0393	−0.0391	−0.0394	−0.0392	−0.0395
−0.03969	±0.0001	±0.0000	±0.0019	±0.0012	±0.0014	±0.0007	±0.0013	±0.0007
−0.4706	−0.4902	−0.4905	−0.4983	−0.4987	−0.4983	−0.4988	−0.4982	−0.4987
−0.5000	±0.0056	±0.0049	±0.0015	±0.0011	±0.0013	±0.0009	±0.0015	±0.0011
−0.6428	−0.6522	−0.6527	−0.6526	−0.6528	−0.6524	−0.6526	−0.6524	−0.6526
−0.6536	±0.0013	±0.0008	±0.0011	±0.0008	±0.0018	±0.0014	±0.0019	±0.0014
−5.7215	−7.840	−7.824	−7.911	−7.903	−7.911	−7.903	−7.911	−7.903
−8.000	±0.101	±0.108	±0.046	±0.040	±0.047	±0.040	±0.046	±0.040
−7.3775	−10.346	−10.329	−10.380	−10.359	−10.379	−10.358	−10.379	−10.358
−10.535	±0.089	±0.084	±0.083	±0.078	±0.084	±0.078	±0.084	±0.078

For the asymmetric case, there were two possible customer process routes, each occurring with probability 0.5: $1 \longrightarrow 2 \longrightarrow 3 \longrightarrow 4 \longrightarrow 5$, or $2 \longrightarrow 5 \longrightarrow 3$. Since the service times are exponential, the analytical performance can be found by solving the appropriate traffic equations, and applying the usual steady-state $M/M/1$ queue formulas. Via Mathematica [24], the optimal assignment for this set of parameters turns out to be $(\theta)_1 = (\theta)_4 = 40/7$, and $(\theta)_2 = (\theta)_3 = (\theta)_5 = 20/7$.

In addition, we took $a = 0.08$ and $c = 1$. Iteration updates were done after every 20, 100, or 500 customer departures from the system, with the number of iterations fixed so that the total number of customer departures totaled 20 000. The technique of common random numbers was not employed, and common states were not enforced for the beginning of the positive and negative perturbation runs. The following projection algorithm was implemented when the update given by (2) took θ outside of the feasible set Θ: project $x\%(0 < x < 100)$ of the way – as measured in the parameter direction most violated – to the boundary. A value of $x = 90$ was used in our experiments. In addition, to avoid stability problems, the upper limit of the feasible region Θ was diminished by a factor of 0.98.

Tables 3 and 4 summarize the results of a simulation study for the network examples, based on 10 replications for each example. It appears that taking a large number of customers per observation with a fewer number of resulting iterations works better for these network cases than the converse.

We next considered a larger 10-station asymmetric case with three classes of customers and $K = 40$:

$$1 \longrightarrow 2 \longrightarrow 3 \longrightarrow 4 \longrightarrow 5 \longrightarrow \cdots \longrightarrow 10 \text{ w.p. } 0.2;$$
$$2 \longrightarrow 5 \longrightarrow 3 \text{ w.p. } 0.5;$$
$$3 \longrightarrow 1 \longrightarrow 8 \longrightarrow 10 \text{ w.p. } 0.3.$$

Iterations were taken over a fixed customer observation horizon of 500 customers. As before, technically this horizon should increase with the number of iterations in

Table 3. Optimization results for network example with deterministic service times: $\theta^* = (4, 4, 4, 4, 4)$

	No. of customers/ iteration	*θ_n at total number of customers simulated: mean ±SE*									
θ_1		10 000					20 000				
(4,4,4,4,4)	20	3.93	3.67	4.27	4.19	3.95	3.75	4.04	3.95	4.36	3.90
		±0.35	±0.70	±0.66	±0.51	±0.42	±0.42	±0.46	±0.88	±1.31	±0.53
	100	4.09	3.94	4.07	3.89	4.01	4.09	3.80	4.07	3.96	4.07
		±0.39	±0.34	±0.42	±0.37	±0.30	±0.29	±0.29	±0.22	±0.22	±0.20
	500	4.00	4.01	4.03	4.04	3.92	3.96	3.92	4.14	4.06	3.93
		±0.38	±0.50	±0.45	±0.39	±0.50	±0.40	±0.46	±0.36	±0.30	±0.24
(1,7,2,5,5)	20	1.78	6.35	3.22	4.44	4.21	1.61	6.07	3.15	4.16	5.02
		±0.93	±1.01	±1.15	±1.04	±0.70	±0.67	±0.97	±1.11	±1.41	±1.24
	100	2.61	4.85	2.97	4.73	4.84	2.78	5.07	2.79	4.70	4.66
		±1.35	±0.82	±1.29	±0.46	±0.54	±1.53	± 0.57	±1.52	±1.02	±0.81
	500	2.62	4.93	3.47	4.68	4.31	2.09	5.18	3.14	4.65	4.94
		±1.53	±0.73	±1.72	±0.71	±1.08	±1.19	±0.26	±1.40	±0.65	±0.54

Table 4. Optimization results for network example with exponential service times: $\theta^* = (40/7, 20/7, 20/7, 40/7, 20/7)$

θ_1	*No. of customers/ iteration*	θ_n *at total number of customers simulated: mean* $\pm SE$									
		10 000					20 000				
(4,4,4,4,4)	20	4.07	4.34	3.57	5.12	2.89	5.47	3.73	3.28	4.33	3.20
		±1.20	±1.82	±1.90	±1.42	±1.28	±1.48	±0.83	±1.51	±1.23	±1.01
	100	4.43	3.52	3.37	5.63	3.05	5.11	3.00	3.29	5.43	3.16
		±1.19	±0.83	±0.74	±1.21	±0.53	±1.03	±0.63	±0.50	±1.47	±0.79
	500	5.40	3.32	3.03	5.11	3.14	4.99	3.38	3.32	4.68	3.62
		±1.27	±0.83	±0.90	±1.64	±0.57	±0.86	±0.58	±0.47	±1.11	±0.38

order to achieve convergence. Via Mathematica, the optimal assignment for this set of parameters turns out to be: $(\theta)_3 = 1.33971, (\theta)_2 = (\theta)_5 = 1.91388, (\theta)_1 = (\theta)_8 = (\theta)_{10} = 2.67943, (\theta)_4 = (\theta)_6 = (\theta)_7 = (\theta)_9 = 6.69856$, with an objective function value of 48.046. The results shown in Table 5 include the objective function evaluated at the beginning and the end of each replication, indicating the improvement.

5.3. *Transportation application*

We considered a four-line transit network model that consists of transit lines traveling in four directions on a grid: east, west, north, and south. These four lines are represented by the two bi-directional routes in Fig. 1: an east–west route and a north–south route. There are just three stops on each line: an origin point, a potential transfer point that we will call the *center* point, and a destination point. Bus travel between stops is represented by eight different traffic 'links' labeled in Fig. 1 on the four bi-directional segments of two links each. Henceforth we will refer to the transit vehicles as 'buses' and the transit lines as 'bus lines.' We built a simulation program of the transit network using the SIMAN simulation language, and then attached the SPSA optimization shell to it.

We considered first a very simple experiment where only $N = 4$ customers rode in the entire day, one at each of the origin points, with routes chosen randomly. We chose a fixed headway of $K_i = 10$, uniformly distributed bus travel times, and considered various arrival time distributions, starting with deterministic arrivals: all four customers arriving exactly at time 10. However, the optimization scheme failed in this case. A little thought reveals that the objective function is insufficiently smooth. Uniform distributions also fail to meet the objective function smoothness requirement, so we used triangular distributions that are in some sense the minimally smoothest distributions, as the objective function is then continuously differentiable, with discontinuities in the second derivative. All arrival times were at time TR(9,10,11), with bus travel times also TR(9,10,11), where $\mathrm{TR}(k_1, k_2, k_3)$ indicates a triangular distribution with minimum value k_1, maximum value k_3, and mode k_2.

Example 1. We conducted simulation experiments using three different step sizes for each of the two sequences: $a = 1.0, 3.0, 10.0$ and $c = 1.0, 0.5, 0.1$; two different start-

Table 5. Optimization results for network example with 10 stations and exponential service times: $\theta^* = (2.67943, 1.91388, 1.33971, 6.69856, 1.91388, 6.69856, 6.69856, 2.67943, 6.69856, 2.67943)$, $J(\theta^*) = 48.046$, $\theta_1 = (4, 4, 4, 4, 4, 4, 4, 4, 4, 4)$, $J(\theta_1) = 54.0855$

Rep.	θ_n *after 1000 iterations of 500 customers each: mean*										$J(\theta_n)$
1	4.78	1.76	0.27	7.96	2.67	7.60	4.49	2.30	4.26	3.90	49.7
2	1.46	1.92	0.05	7.98	1.11	7.72	4.85	3.76	7.98	3.18	49.1
3	2.56	1.10	0.03	7.89	1.03	7.12	5.37	4.56	7.97	2.37	49.2
4	3.03	1.65	0.11	7.82	1.38	7.93	6.48	0.24	7.91	3.46	49.2
5	3.93	1.59	0.14	4.04	1.44	7.99	7.90	2.55	7.98	2.43	49.1
6	4.02	1.99	0.10	7.89	1.44	6.99	5.35	0.67	7.83	3.73	49.3
7	3.21	1.59	0.15	7.82	1.32	6.78	5.94	3.14	7.83	2.22	48.6
8	2.77	0.84	0.15	6.60	1.35	7.79	7.14	1.40	7.97	3.99	49.0
9	1.87	0.19	0.22	7.89	1.53	7.99	6.72	3.57	7.85	2.17	49.1
10	2.48	1.57	0.24	7.98	0.36	7.92	6.42	4.02	7.80	1.20	49.2
Mean	3.01	1.42	0.15	7.39	1.36	7.58	6.07	2.62	7.54	2.87	49.15
SE	0.32	0.17	0.03	0.39	0.18	0.14	0.34	0.46	0.37	0.29	0.03

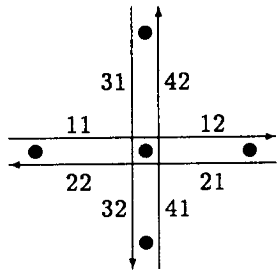

Fig. 1. Schematic of four-line traffic network: link 'ij' is the jth link on the ith line.

ing points for the offset times: θ_1 = (9,9,9,9), (9,7,13,11); two different values for the exponent of the finite-difference step size: $\alpha = 1.0$, 0.751 (the second value is one recommended by Spall [2] in conjunction with the β value following); and a single value for $\beta = 0.25$ and $\Theta = (0, 19)$. The results after 500 SPSA iterations are given in Tables 6 and 7, where the estimated values of the objective function are given with half-widths of 95% confidence intervals. At $\theta_1 = (9,9,9,9)$, the 95% confidence interval for the mean waiting time is 6.73 ± 0.32. At $\theta_1 = (9,7,13,11)$, the average wait is approximately 5.62 ± 0.05. Based on the analysis of the last section (which was not complete, as it did not consider the transfers), the optimum was guessed to be at $\theta^* = (10.9, 10.9, 10.9, 10.9)$, which yielded the 95% confidence interval for the mean waiting time of 2.19 ± 0.29. The best results for the parameter values occur at $a = c = 1.0$, with $\alpha = 0.751$, yielding estimates of the average wait that have lower means than the estimate for the optimum; however, the confidence intervals overlap, so the experiments were not statistically conclusive.

Example 2. This differs from Example 1 only in the number of customers simulated, which was increased from 4 to 80 (20 on each line). We fixed $a = 1.0$ for all the runs, and just considered the first starting point $\theta_1 = (9,9,9,9)$; otherwise, we again used the same system parameter values as in the previous example. Table 8 gives two sample runs of the algorithm after 500 SPSA iterations, where the estimated values of the objective function are given with 95% confidence half-widths. At $\theta_1 = (9,9,9,9)$, the 95% confidence interval for the mean waiting time is 5.80 ± 0.18. Since only 500 iterations of the algorithm have been carried out, convergence has not yet been achieved, and so randomness in the outcomes is to be expected. The important observation is that both random samples show substantial improvement from the initial values. The final values for the offsets in the two best results in Table 8 ($\alpha = 1.0$, $c = 1.0$, and $c = 0.5$) were θ_{500} = (11.60,10.86,11.56,10.94) and θ_{500} = (11.34,10.64, 11.52,10.78), which are quite close to the guessed optimal of $\theta^* = (10.9, 10.9, 10.9, 10.9)$. In fact, in these two cases, the estimated performance was better than the guessed optimal.

Example 3. Here, we employed the suggestions of Spall [2] in choosing the settings of the parameters a, c, α, β, but otherwise, everything is the same as in the previous example. We took $\alpha = 0.602, \beta = 0.101$, and $a = c = 1$. Table 9 gives the results for 10 different seed sets, all after 500 iterations. Again, substantial improvement is achieved.

6. Conclusions

We have applied the technique of SPSA to the optimization of discrete-event systems via simulation. The

Table 6. Transportation example 1 results: 95% confidence intervals for $E[\overline{W}]$ at θ_{500}; $E[\overline{W}] = 6.73 \pm 0.32$ at θ_1 = (9, 9, 9, 9); $E[\overline{W}] = 2.19 \pm 0.29$ at θ^* = (10.9, 10.9, 10.9, 10.9); $\beta = 0.25$

		a		
α	c	*1.0*	*3.0*	*10.0*
1.0	1.0	3.01 ± 0.19	3.03 ± 0.16	4.21 ± 0.06
	0.5	2.10 ± 0.20	2.17 ± 0.20	4.91 ± 0.07
	0.1	2.32 ± 0.16	2.73 ± 0.24	3.87 ± 0.07
0.751	1.0	2.04 ± 0.15	4.10 ± 0.19	5.45 ± 0.05
	0.5	2.43 ± 0.07	2.98 ± 0.13	5.36 ± 0.04
	0.1	3.41 ± 0.33	4.48 ± 0.07	5.78 ± 0.05

Table 7. Transportation example 1 results: 95% confidence intervals for $E[\overline{W}]$ at θ_{500}; θ_1 = (9, 7, 13, 11); $E[\overline{W}] = 5.62 \pm 0.05$; θ^* = (10.9, 10.9, 10.9, 10.9); $E[\overline{W}] = 2.19 \pm 0.29$; $\beta = 0.25$

		a		
α	c	*1.0*	*3.0*	*10.0*
1.0	1.0	3.04 ± 0.21	2.97 ± 0.04	4.75 ± 0.07
	0.5	4.04 ± 0.20	3.49 ± 0.05	4.96 ± 0.05
	0.1	3.25 ± 0.16	5.23 ± 0.33	3.65 ± 0.05
0.751	1.0	2.10 ± 0.18	3.31 ± 0.21	4.25 ± 0.15
	0.5	3.42 ± 0.16	3.96 ± 0.05	3.82 ± 0.11
	0.1	3.84 ± 0.08	3.96 ± 0.05	4.44 ± 0.04

Table 8. Transportation example 2 results: 95% confidence intervals for $E[\overline{W}]$ at θ_{500}; $E[\overline{W}] = 5.80 \pm 0.18$ at θ_1 = (9, 9, 9, 9); $E[\overline{W}] = 2.46 \pm 0.16$ at θ^* = (10.9, 10.9, 10.9, 10.9); $\beta = 0.25$; $a = 1.0$

		c		
Seed set	α	*1.0*	*0.5*	*0.1*
1	1.0	2.99 ± 0.12	3.03 ± 0.10	4.34 ± 0.05
	0.751	3.53 ± 0.07	3.45 ± 0.07	5.07 ± 0.13
2	1.0	2.03 ± 0.10	2.07 ± 0.13	4.01 ± 0.05
	0.751	4.10 ± 0.07	3.25 ± 0.07	3.75 ± 0.13

Table 9. Transportation example 3 results: 95% confidence intervals for $E[\overline{W}]$ at θ_{500}; $E[\overline{W}] = 5.76 \pm 0.12$ at $\theta_1 = (9, 9, 9, 9)$; $E[\overline{W}] = 2.45 \pm 0.10$ at $\theta^* = (11, 11, 11, 11)$; $\alpha = 0.602$; $\beta = 0.101$; $a = c = 1.0$

Rep.	θ_{500}				$E[\overline{W}]$
1	11.83	11.01	11.89	10.99	2.06 ± 0.07
2	11.95	10.89	11.74	10.96	2.08 ± 0.08
3	11.21	10.83	11.65	7.25	3.03 ± 0.07
4	11.05	8.29	10.97	6.63	3.86 ± 0.06
5	12.02	10.80	11.62	11.00	2.17 ± 0.08
6	1.66	11.00	12.06	11.00	2.07 ± 0.07
7	11.24	10.88	11.70	6.50	3.04 ± 0.07
8	11.92	10.98	12.02	11.04	2.11 ± 0.07
9	12.06	11.03	12.05	10.99	2.12 ± 0.06
10	10.93	6.35	11.11	6.57	3.94 ± 0.07
Mean					2.65

technique offers significant computational savings over traditional finite-difference methods and is applicable to any system that can be simulated. The primary purpose of this paper was to introduce this technique to the discrete-event simulation community. We considered three different discrete-event systems and conducted experiments to investigate the viability of the technique. The simulation results, though not comprehensive, were very promising and should help encourage further experimental and theoretical work on the application of SPSA to the optimization of discrete-event systems.

Acknowledgements

This work was supported in part by a JHU/APL IRAD Grant. We thank the Simulation Department Co-Editor David Goldsman and the Associate Editor for their close reading of the manuscript. M.C.F. is supported in part by the National Science Foundation under grant no. NSF EEC 94-02384.

References

[1] Ho, Y. C. and Cao, X. R. (1991) *Discrete Event Dynamic Systems and Perturbation Analysis*, Kluwer Academic, Boston.

[2] Spall, J. C. (1992) Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, **37**, 332–341.

[3] Azadivar, F. and Talavage, J. J. (1980) Optimization of stochastic simulation models. *Mathematics and Computers in Simulation*, **22**, 231–241.

[4] Fu, M. C. (1994) Optimization via simulation: a review. *Annals of Operations Research*, **53**, 199–248.

[5] Suri, R. and Zazanis, M.A. (1988) Perturbation analysis gives strongly consistent sensitivity estimates for the M/G/1 queue. *Management Science*, **34**, 39–64.

[6] Suri, R. and Leung, Y. T. (1991) Single run optimization of discrete event simulations – an empirical study using the M/M/1 queue. *IIE Transactions*, **21**, 35–49.

[7] L'Ecuyer, P., Giroux, N. and Glynn, P. W. (1994) Stochastic optimization by simulation: numerical experiments with a simple queue in steady-state. *Management Science*, **40**, 1245–1261.

[8] Glasserman, P. (1991) *Gradient Estimation Via Perturbation Analysis*, Kluwer Academic, Boston.

[9] Fu, M. C. (1990) Convergence of a stochastic approximation algorithm for the GI/G/1 queue using infinitesimal perturbation analysis. *Journal of Optimization Theory and Applications*. **65**, 149–160.

[10] Andradóttir, S. (1990) A new algorithm for stochastic approximation, in *Proceedings of the 1990 Winter Simulation Conference*, Balci, O., Sadowski, R.P. and Nance, R.E. (ed.) IEEE, Piscataway, NJ, pp. 364–366.

[11] Wardi, Y. (1990) Stochastic algorithms with Armijo stepsizes for minimization of functions. *Journal of Optimization Theory and Applications*, **64**, 399–418.

[12] Spall, J. C. and Cristion, J. A. (1994) Nonlinear adaptive control using neural networks: estimation with a smoothed simultaneous perturbation gradient approximation. *Statistica Sinica*, **4**, 1–27.

[13] Hill, S. D. and Fu, M. C. (1994) Optimizing discrete event systems with the simultaneous perturbation stochastic approximation algorithm, in *Proceedings of the 33rd IEEE Conference on Decision and Control*, IEEE, Piscataway, NJ, pp. 2631–2632.

[14] Hill, S. D. and Fu, M. C. (1994) Simulation optimization via simultaneous perturbation stochastic approximation, in *Proceedings of the 1994 Winter Simulation Conference*, Tew, J.D., Manivannan, S., Sadowski, D.A. and Seila, A.F. (ed.) IEEE, Piscataway, NJ, pp. 1461–1464.

[15] Fu, M. C. and Ho, Y. C. (1988) Using perturbation analysis for gradient estimation, averaging, and updating in a stochastic approximation algorithm, in *Proceedings of the 1988 Winter Simulation Conference*, Abrams, M.A., Haigh, P.L. and Comfort, J.C. (ed.), IEEE, Piscataway, NJ, pp. 509–517.

[16] Bookbinder, J. H. and Désilets, A. (1992) Transfer optimization in a transit network. *Transportation Science*, **26**, 106–118.

[17] Kushner, H. J. and Clark, D. C. (1978) *Stochastic Approximation Methods for Constrained and Unconstrained Systems*, Springer-Verlag, New York.

[18] L'Ecuyer, P. and Glynn, P. W. (1994) Stochastic optimization by simulation: convergence proofs for the GI/G/1 queue in steady-state. *Management Science*, **40**, 1562–1578.

[19] Glynn, P. W. (1986) Stochastic approximation for Monte Carlo optimization, in *Proceedings of the 1986 Winter Simulation Conference*, Wilson, J.R., Henriksen, J.O. and Roberts, S.D. (ed.) IEEE, Piscataway, NJ, pp. 356–364.

[20] Glynn, P. W. (1989) Optimization of stochastic systems via simulation, in *Proceedings of the 1989 Winter Simulation Conference*, MacNair, E.A., Musselman, K.J. and Heidelberger, P. (ed.) IEEE, Piscataway, NJ, pp. 90–105.

[21] Chong, E. K. P. and Ramadge, P. J. (1992) Convergence of recursive optimization algorithms using infinitesimal perturbation analysis. *Discrete Event Dynamic Systems: Theory and Applications*, **1**, 339–372.

[22] Chong, E. K. P. and Ramadge, P. J. (1993) Optimization of queues using an IPA based stochastic algorithm with general update times. *SIAM Journal on Control and Optimization*, **31**, 698–732.

[23] Chong, E. K. P. and Ramadge, P. J. (1994) Stochastic optimization of regenerative systems using infinitesimal perturbation analysis. *IEEE Transactions on Automatic Control*, **39**, 1400–1410.

[24] Wolfram, S. (1991) *Mathematica: A System for Doing Mathematics by Computer*, Addison-Wesley.

Biographies

Michael C. Fu is an Associate Proffessor of Management Science & Statistics in the College of Business and Management, with a joint appointment in the Institute for Systems Research, at the University of Maryland at College Park. He received a bachelor's degree in mathematics and bachelor's and master's degrees in electrical engineering from MIT in 1985, and M.S. and Ph.D. degrees in applied mathematics from Harvard University in 1986 and 1989, respectively. His research interests include simulation optimization and sensitivity analysis, particularly with applications towards manufacturing systems, inventory control, and the pricing of financial derivatives. He teaches courses in applied probability, stochastic processes, discrete-event simulation, and operations management, and in 1995 was awarded the Allen J. Krowe Award for Teaching Excellence. He served on the Program Committee for the Spring 1996 INFORMS National Meeting and is on the Editorial Boards of *IIE Transactions* and *INFORMS Journal on Computing*.

Stacy D. Hill is a Systems Engineer on the Senior Professional Staff of the Johns Hopkins University Applied Physics Laboratory. He received the B.S. and M.S. degrees from Howard University in 1975 and 1977, respectively, and the D.Sc. degree in control systems engineering and applied mathematics from Washington University in 1983. He has been project and technical lead in developing, testing, and applying statistical estimation techniques and software, and has authored and co-authored journal articles on estimation theory and techniques, including an article in the cross-disciplinary book *Bayesian Analysis of Time Series and Dynamic Models*. His present research interests include optimization and simulation of discrete event systems, with application to transportation problems.