Stochastic Optimization by Simulation: Numerical Experiments with the M/M/1 Queue in Steady-State
Author(s): Pierre L'Ecuyer, Nataly Giroux, Peter W. Glynn
Source: *Management Science*, Vol. 40, No. 10 (Oct., 1994), pp. 1245-1261
Published by: INFORMS
Stable URL: http://www.jstor.org/stable/2661620
Accessed: 21/07/2010 03:04

Your use of the JSTOR archive indicates your acceptance of JSTOR's Terms and Conditions of Use, available at http://www.jstor.org/page/info/about/policies/terms.jsp. JSTOR's Terms and Conditions of Use provides, in part, that unless you have obtained prior permission, you may not download an entire issue of a journal or multiple copies of articles, and you may use content in the JSTOR archive only for your personal, non-commercial use.

Please contact the publisher regarding any further use of this work. Publisher contact information may be obtained at http://www.jstor.org/action/showPublisher?publisherCode=informs.

Each copy of any part of a JSTOR transmission must contain the same copyright notice that appears on the screen or printed page of such transmission.

JSTOR is a not-for-profit service that helps scholars, researchers, and students discover, use, and build upon a wide range of content in a trusted digital archive. We use information technology and tools to increase productivity and facilitate new forms of scholarship. For more information about JSTOR, please contact support@jstor.org.

INFORMS is collaborating with JSTOR to digitize, preserve and extend access to *Management Science.*

http://www.jstor.org

Stochastic Optimization by Simulation: Numerical Experiments with the $M/M/1$ Queue in Steady-state

Pierre L'Ecuyer • Nataly Giroux • Peter W. Glynn
Département d'I.R.O., Université de Montréal, C.P. 6128, Montréal, H3C 3J7, Canada
Bell Northern Research, Dept. 6J33, Fitzgerald Building, P.O. Box 3511, Ottawa, K1Y 4H7, Canada
Operations Research Department, Stanford University, Stanford, CA 94305, USA

This paper gives numerical illustrations of the behavior of stochastic approximation, combined with different derivative estimation techniques, to optimize a steady-state system. It is a companion paper to L'Ecuyer and Glynn (1993), which gives convergence proofs for most of the variants experimented here. The numerical experiments are made with a simple $M/M/1$ queue, which while simple, serves to illustrate the basic convergence properties and possible pitfalls of the various techniques.
(*Discrete Event Systems; Stochastic Approximation; Gradient Estimation; Optimization; Steady-state*)

1. Introduction

More traditional optimization methods, like mathematical programming and optimal control methods, are very efficient in some contexts, but for large classes of complex (realistic) stochastic models, they are no longer practical. For such models, simulation is often the only viable tool. Developing efficient ways of optimizing stochastic discrete event systems by simulation is not easy but is extremely important in practice. Current approaches include, among others, ranking and selection procedures (for finite parameter spaces), response surface methodology, gradient-based stochastic approximation, and the stochastic counterpart method (the latter methods are for continuous parameter spaces). See Fu (1994) for a recent survey. Recent advances in gradient estimation methodology have increased interest in stochastic approximation (SA) algorithms for simulation optimization. Different variants of SA, combined with a variety of derivative estimation techniques (DETs), have been proposed and studied. See, e.g., Andradóttir (1990, 1991a, b), Chong and Ramadge (1990, 1992a, b, 1993), Dupuis and Simha (1991), Fu (1990), Gaivoronski (1992), Glynn (1986, 1987, 1989), Pflug (1990), and Suri and Leung (1989). Convergence proofs have been given for many of them. There were also some numerical results in a few cases, but no extensive numerical investigation involving all (or most) of those methods. This paper reports the results of such a numerical investigation. It is a companion paper to L'Ecuyer and Glynn (1993), which contains most of the theory.

Suri and Leung (1989) have performed preliminary numerical experiments with an $M/M/1$ queue. The objective was to find the value of the average service time θ that would minimize a given function of the average sojourn time per customer, in steady-state. That problem is easy to solve analytically, and they wanted to use it as a "benchmark" to compare two SA-DET combinations, one of them based on infinitesimal perturbation analysis (IPA) and the other one on finite differences (FD). These two methods were presented as heuristics, and they observed empirically that the one based on IPA converged much faster. We show in this paper that their second method, based on FD, ac-

0025-1909/94/4010/1245$01.25
Copyright © 1994, The Institute of Management Sciences

tually converges to the wrong value. In L'Ecuyer and Glynn (1994b), we prove convergence to the optimum for their SA-IPA combination, as well as for other variants involving FD, FD with common random numbers (FDC), IPA, and the likelihood ratio (LR) method. For most of the DETs, in order for SA to converge towards the optimum, the simulation length must increase from iteration to iteration to make the bias of the derivative estimator go toward zero. One exception is IPA. Some might think that in that case, keeping a (small) fixed simulation length for all iterations should be better than having longer and longer simulations, because for a given computer budget, the former allows more iterations to be performed. But our experiments show that it is not necessarily the case. This was first observed and illustrated graphically in L'Ecuyer, Giroux, and Glynn (1989), then in Chong and Ramadge (1990, 1993). The proof of convergence in Chong and Ramadge (1993) gives some theoretical support to that observation. Indeed, if the variance of the derivative estimator decreases linearly with the simulation length, as is the case for the example examined here, it appears that the simulation length per iteration should not matter much. For a given computer budget, short or long, fixed or increasing simulation lengths yield comparable results. Of course, this does not hold universally. If the simulation lengths per iteration are so long that they allow very few SA iterations, performance deteriorates.

In §2, we consider an $M/M/1$ example similar to the one studied by Suri and Leung (1989). We feel that most of the important questions that would arise in more general models are well illustrated by this simple example. Section 3 recalls some variants of SA. Section 4 describes many derivative estimators and discusses implementation issues. Our experimental setup is established in §5. For each SA-DET variant, we look at the empirical mean-square error of the value of θ produced by the optimization algorithm, after a fixed number of simulated customers. Section 6 reports our numerical investigations. In the conclusion, we summarize our results and mention prospects for further research.

2. An $M/M/1$ Example

Consider an $M/M/1$ queue with arrival rate $\lambda = 1$ and mean service time

$$\theta \in \bar{\Theta} = [\bar{\theta}_{\min}, \bar{\theta}_{\max}] \subset (0, 1).$$

So, the service time has distribution $B_\theta(\zeta) = 1 - e^{-\zeta/\theta}$, with density $b_\theta(\zeta) = (1/\theta)e^{-\zeta/\theta}$. Let $w(\theta)$ be the average sojourn time in the system per customer, in steady-state, at parameter level θ. The objective function is defined by

$$\alpha(\theta) = w(\theta) + C_1/\theta, \tag{1}$$

where C_1 is a positive constant. We want to minimize $\alpha(\theta)$ over $\Theta = [\theta_{\min}, \theta_{\max}]$, a strict subinterval of $\bar{\Theta}$. The optimal value θ^* can be computed easily in this case. Indeed, if $l(\theta)$ and $u(\theta)$ denote respectively the expected number of customers and expected total sojourn time for all the customers in a busy cycle, one has

$$\begin{aligned} l(\theta) &= 1/(1-\theta), \\ u(\theta) &= \theta/(1-\theta)^2, \\ w(\theta) &= u(\theta)/l(\theta) = \theta/(1-\theta), \\ \alpha'(\theta) &= 1/(1-\theta)^2 - C_1/\theta^2, \\ \alpha''(\theta) &= 2/(1-\theta)^3 + 2C_1/\theta^3, \\ \theta^* &= \sqrt{C_1}/(1+\sqrt{C_1}) \end{aligned}$$

(if this value is not in Θ, the optimum is at the nearest boundary point). We will compare our empirical results to this theoretical value using the empirical mean-square error. In Appendix II, we verify that this example satisfies the assumptions of L'Ecuyer and Glynn (1994b).

3. SA and Some of Its Variants

The classical SA algorithm has the form

$$\theta_{n+1} := \pi_\Theta(\theta_n - \gamma_n Y_n), \tag{2}$$

where θ_n is the parameter value at the beginning of iteration n, Y_n is an estimate of $\alpha'(\theta_n)$ obtained at iteration n, $\{\gamma_n, n \geq 1\}$ is a deterministic sequence of *gains* decreasing to 0, and π_Θ is a projection operator on the set Θ. Typically, one takes $\gamma_n = \gamma_0 n^{-1}$ for some constant $\gamma_0 > 0$. Conditions under which θ_n converges almost surely (a.s.) to the optimizer are given in many places, including Kushner and Clark (1978) and L'Ecuyer and Glynn (1994b). For $n = 1, 2, 3, \ldots$, let E_n denote the expectation conditional on $(\theta_1, \ldots, \theta_n, Y_1, \ldots, Y_{n-1})$. If $E_n[Y_n] = \alpha'(\theta_n)$ and $E_n[(Y_n - \alpha'(\theta_n))^2] \leq K$ for all n for some finite constant K, α has a bounded third derivative, $\alpha''(\theta^*) > 0$, $\alpha'(\theta^*) = 0$, and $\theta_n \xrightarrow{\text{a.s.}} \theta^*$, then the (asymp-

totically) "optimal" sequence is $\gamma_n = \gamma_0^* n^{-1}$, with $\gamma_0^* = (\alpha''(\theta^*))^{-1}$, yielding the canonical convergence rate, in the sense that $n^{-1/2}(\theta_n - \theta^*)$ converges in distribution to a centered normal with minimal variance (see Chung 1954, Fabian 1968, Goldstein 1988, Major and Revesz 1973). We put the word optimal in quotes because in fact, this is optimal only if we assume that all the Y_n's have equivalent computational costs. More generally, for $\gamma_n = \gamma_0 n^{-\gamma}$, under the same set of assumptions, $n^{-\beta}(\theta_n - \theta^*)$ converges to a centered normal in the cases covered by the following definition of β:

$$\beta = \begin{cases} \gamma/2 & \text{if } \tfrac{1}{2} < \gamma < 1; \\ \tfrac{1}{2} & \text{if } \gamma = 1 \text{ and } \gamma_0 > \gamma_0^*/2; \\ \gamma_0/\gamma_0^* & \text{if } \gamma = 1 \text{ and } \gamma_0 < \gamma_0^*/2. \end{cases} \quad (3)$$

For further details, more general SA algorithms, multidimensional parameters, and more general results, see also Benveniste, Métivier, and Priouret (1987), Fabian (1968), Goldstein (1988), Kushner and Clark (1978), Kushner and Yang (1993), Polyak and Tsypkin (1980), Polyak (1990), and Yin (1992).

Unfortunately, the conditions under which the above "convergence rate" results have been proved do not hold for the problem considered here, for most of our DET variants. Indeed, typically, each Y_n is a *biased* derivative estimator and, when Y_n is based on a simulation of length t_n which increases with n, the variance and computational cost of Y_n vary with n. The convergence rates and optimal sequence γ_n might then be quite different. Finding the optimal sequence and convergence rate for each SA-DET combination would be a demanding task that goes beyond the scope of this paper and will be the subject of further research. Nevertheless, our numerical exploration will show that for some DET's, the above convergence rate results appear to hold for our problem. They also hold for some regenerative DET variants, for which the above conditions are satisfied. For instance, as explained in L'Ecuyer and Glynn (1994b) (see also equations (12) and (17)), unbiased estimators of $l(\theta_n)\alpha'(\theta_n)$ or $l^2(\theta_n)\alpha'(\theta_n)$ might be available and can be used in (2) instead of an estimator of $\alpha'(\theta_n)$ to find a root of $\alpha'(\theta)$. For such estimators, however, γ_0^* must be replaced by

$$\tilde{\gamma}_0^* = [l(\theta^*)\alpha''(\theta^*)]^{-1} = (1 - \theta^*)\gamma_0^* \quad \text{and}$$

$$\tilde{\tilde{\gamma}}_0^* = [l^2(\theta^*)\alpha''(\theta^*)]^{-1} = (1 - \theta^*)^2\gamma_0^*,$$

respectively.

Choosing the right sequence of gains γ_n turns out to be rather important in practice. For example, if γ_0 is too large, θ_n will bounce around too much while if γ_0 is too small, θ_n will move too slowly towards the optimum (see (3)). Unfortunately, in practical applications, one often has little idea of the right γ_0. This is why people have introduced various "adaptive" approaches, whose aim is to speed up convergence by (roughly speaking) reajusting dynamically the sequence of gains. Some variants also rescale the derivative estimators, which is formally different, but practically similar. These methods are often very helpful. But unfortunately, some of them do not always work well and might even slow down the algorithm, as will be illustrated by our experiments. We will now describe a few of those adaptive approaches.

Kesten (1952) has proposed a rule under which instead of diminishing γ_n at each iteration, one diminishes it only when the sign of the gradient estimate (for one parameter) is different from the one of the previous iteration (i.e., when the change on the parameter changes direction). The heuristic idea is that if the parameter keeps moving in the same direction, it should be because we are still far away from the optimum, and so we let it move faster. That heuristic might help in situations where we start really far away from the optimum, and where the change on the parameter at each iteration tends to be very small.

Andradóttir (1990, 1991a) has proposed a variant of SA whose aim is to insure convergence even if Θ is unbounded, or to reduce the "bouncing around" behavior when the function $\alpha(\theta)$ is too steep. At each iteration of SA, it uses two independent derivative estimates, say Y_n^1 and Y_n^2, based on any DET like FD, FDC, LR, or IPA, and computes the "modified" derivative estimator

$$Y_n := \frac{Y_n^1}{\max(\epsilon, |Y_n^2|)} + \frac{Y_n^2}{\max(\epsilon, |Y_n^1|)}, \quad (4)$$

where $\epsilon > 0$ is a predetermined constant (a parameter of the algorithm). That Y_n is then used in SA as usual (see equation (2)). Assuming that Y_n^1 and Y_n^2 are both

unbiased derivative estimators, and under a few additional conditions, Andradóttir proves the convergence of her algorithm to the optimizer. Since each Y_n requires two independent estimates, SA will have less iterations available for a given computer budget with this method than with the regular one. The motivation for this method is to reduce the step size when the function is too steep. Its behavior will depend on the choice of ϵ. If ϵ is near zero, the derivative estimates are more or less "normalized." That is, if the two independent estimators are not too noisy, Y_n should be near ± 2. On the other hand, if ϵ is large, the algorithm becomes equivalent to the regular one by rescaling the sequence $\{\gamma_n, n \geq 0\}$ appropriately (multiply γ_n by $\epsilon/2$), except that an average of two estimators is taken instead of just taking one estimator at each SA iteration. Further, in the case of a steady-state model as we have here, if we simulate for a fixed number of customers to obtain Y_n^1, and then *continue* the simulation for a fixed number of customers to obtain Y_n^2, then Y_n^1 and Y_n^2 typically will be *correlated*, introducing a bias in (4).

Azadivar and Talavage (1980) had previously proposed a somewhat related (heuristic) normalization scheme, based on only one derivative estimator. They implemented their method in a package called SAMOPT. More specifically, they obtain at each iteration a FD estimator Y_n^1 and replace it by its sign, that is $Y_n := Y_n^1/|Y_n^1|$. Of course, the same can be done with FDC, LR, or IPA. One difficulty with that estimator is that it could remain too noisy near the optimizer. For example, if Y_n^1 has low variance and $E[Y_n^1] \approx 0$ near the optimum, then Y_n^1 should be near zero, which is fine if we use it directly in (2). Replacing it by its sign is really not a good idea in this case. In their SAMOPT algorithm, Azadivar and Talavage also implemented some heuristics, with specially tuned parameters, to define the sequences γ_n and c_n adaptively. These heuristics seem to work well for the examples given in their paper, but we are skeptical concerning their general robustness.

Perron (1992) suggested the following heuristic: start with a very large γ_0 and, each time the parameter value wants to bounce from one boundary of Θ to the opposite boundary in one iteration, divide γ_0 by 2 and reset the parameter value to the midway point between the boundaries. This rule can be easily adapted to the multidimensional case if the admissible region Θ is a rectangular box and if each component of θ has its own γ_n: just apply it to each component individually.

Wardi (1988) proposed a SA variant which takes bigger stepsizes by taking $\gamma_n = \gamma_0 n^{-\gamma}$ for some $\gamma < 1$, and t_n increasing with n. Under some assumptions, he showed convergence in *zero upper density* to the optimizer. Dupuis and Simha (1991) went further; they advocated using a constant stepsize, namely $\gamma_n = \gamma_0$ for all n, with an increasing t_n. They proved a.s. convergence under some conditions, but did not obtain convergence rates or numerical results.

Some adaptive approaches attempt estimating $\alpha''(\theta^*)$ along with the estimation of θ^* (Fabian 1968, Venter 1967). A major drawback of those adaptive approaches is high computational costs, especially in the multidimensional case.

Recently, Polyak (1990) has introduced the interesting idea of taking the *average* of the values of θ_n over all the iterations, instead of just taking the value of θ_n from the last iteration, as an estimator of the optimizer. Roughly speaking, he showed under some conditions that for $\frac{1}{2} < \gamma < 1$, the average converges to θ^* at the optimal rate for whatever γ_0. Kushner and Yang (1993) and Yin (1992) gave different proofs, requiring less restrictive assumptions. They also suggested taking the average over a moving window, whose size may increase linearly with n. More specifically, the estimator of θ^* has the form

$$\bar{\theta}_{n,m} = \frac{1}{m} \sum_{i=n-m+1}^{n} \theta_i . \tag{5}$$

Again, the required assumptions are not verified by our $M/M/1$ example for most of the SA-DET combinations. Therefore, the averaging approach should be viewed here as an heuristic.

4. Derivative Estimation and Implementation Issues

At iteration n of SA, to obtain a derivative estimator Y_n, we simulate the system for one or more "subrun(s)" of finite duration t_n, starting from some initial state s_n. When the queue is not empty at the end of an iteration, we must be careful to generate the new service time only at the beginning of the next iteration, i.e., *after* modifying the parameter. For some of the DET variants,

t_n is a deterministic truncated horizon, representing the number of customers in the subrun. Other variants exploit the regenerative structure (the system regenerates whenever a customer arrives into an empty system), and for those, t_n represents (here) the number of regenerative cycles in the subrun at iteration n. In our implementations, we insisted on using exactly the same simulation program for all of the DET variants. The simulation model and the variants were in fact implemented in two different "modules," the latter being model independent. We now summarize the DET's described in L'Ecuyer and Glynn (1994b) and discuss their implementation.

Let W_i, ζ_i, and $W_i^* = W_i + \zeta_i$ denote the waiting time, service time, and sojourn time of customer i. The initial state of the system is the waiting time of the first customer, $W_1 = s$, where $s \in \bar{S} = [0, c]$ for some fixed constant c. For $t \geq 1$ let

$$h_t(\theta, s, \omega) = \sum_{i=1}^{t} W_i^*, \tag{6}$$

where the sample point ω can be viewed as representing the sequence of i.i.d. $U(0, 1)$ variates underlying the simulation. For some DET variants, the initial state s_n at the beginning of iteration n is $s_n = 0$ (empty system) for all the subruns, while for other variants, the final state of each simulation subrun is taken as the initial state for the next one (projecting on $\bar{S}$ whenever necessary). Since we are interested in steady-state behavior, taking a terminal state of the previous iteration appears intuitively better.

4.1. Finite Differences

For the finite-difference (FD) estimators, one takes a positive sequence $\{c_n, n \geq 1\}$ converging to 0. At iteration n, simulate from some initial state $s_n^- \in \bar{S}$ at parameter value $\theta_n^- = \max(\theta_n - c_n, \bar{\theta}_{\min})$ for t_n customers. Simulate also (independently) from state $s_n^+ \in \bar{S}$ at parameter value $\theta_n^+ = \min(\theta_n + c_n, \bar{\theta}_{\max})$ for t_n customers. Let ω_n^- and ω_n^+ denote the respective sample points. The (centered) FD estimator is then

$$Y_n = C'(\theta_n) + \frac{h_{t_n}(\theta_n^+, s_n^+, \omega_n^+) - h_{t_n}(\theta_n^-, s_n^-, \omega_n^-)}{(\theta_n^+ - \theta_n^-)t_n}. \tag{7}$$

The FDC estimator is the same, except that one takes $\omega_n^- = \omega_n^+$ (common random numbers across the subruns at each iteration), starts the two subruns from the same state: $s_n^- = s_n^+ = s_n$, and synchronizes.

For FD, one can choose the initial states of the subruns as follows. Start the first subrun of iteration n from state $s_n \in \bar{S}$. Then, take the terminal state of any given subrun as the initial state of the next one. (Project on $\bar{S}$ whenever necessary.) For s_{n+1}, take the terminal state of the last subrun of iteration n. Still, the two subruns of a given iteration can be ordered in two different ways. More generally, if θ has dimension d, one can permute the $2d$ subruns of a given iteration in any given way, and select the terminal state of any subrun for s_{n+1}. It is not clear what the best way of doing this is, if any. Another approach is to take the same initial state for each subrun: $s_n^- = s_n^+ = s_n$, but this is more costly to implement (we shall discuss that in a moment) and there are still different possibilities for the selection of s_{n+1}. What we did in our experiments is to take, as initial state s_{n+1}, the final state of the subrun at iteration n which had been performed with parameter value the closest to the parameter value θ_{n+1} used at iteration $n + 1$. In general, if θ is a d-dimensional vector, the same heuristic can be applied for each component of θ to choose the new parameter value among the $2d$ terminal states of the previous iteration. We also made experiments for which we took $s_n = 0$ for all n (all subruns starting from an empty system).

For FDC, one can take $s_n = s_0 \in \bar{S}$ for all n, for some fixed s_0 (e.g., $s_0 = 0$), or s_{n+1} can be one of the two terminal states of iteration n (projecting on $\bar{S}$ if necessary). Implementing this method for complex simulations is not without pain. Saving the simulation state means saving the states of the random number generators, the event list, all the objects in the model, etc. In practice, many objects in the model are pointers to data structures that can be created, modified or destroyed dynamically, and whose types have been defined by the programmer. When saving the state of the system, one cannot only save the pointer values, but must make an explicit "backup" copy of all these structures. When restoring the system to a given state, these must be recopied again. This is different than saving and restoring the state of the *program*, because some variables associated with the SA and FD or FDC algorithms (e.g., the index of the current subrun for FD(C)) should *not* be saved and restored. Usually, the simulation package

cannot do that and specific code must be written. In fact, it would be very difficult to implement "state saving" facilities in a general simulation package, because typically the package has no way of knowing with certainty the structures of all the dynamic objects created by the user. All this implies overhead not only for the computer, but also for the programmer. Another source of programming overhead in FDC comes from the need to insure synchronization of the random numbers across the subruns.

Another FD approach is to estimate $l^2(\theta_n)\alpha'(\theta_n)$, instead of $\alpha'(\theta_n)$, using finite differences with a regenerative approach. This is adapted from Glynn (1986), without the *arctan* transformation. At iteration n, simulate for $2t_n$ independent regenerative cycles, using parameter value θ_n for the odd numbered cycles and $\theta_n^+ = \min(\theta_n + c_n, \bar{\theta}_{\max})$ for the even numbered cycles. Let τ_j be the number of customers during the jth cycle and h_j be the total sojourn time for those τ_j customers. The (forward) regenerative FD estimator is then

$$Y_n = \frac{1}{t_n}\sum_{j=1}^{t_n}\left(\frac{h_{2j}\tau_{2j-1} - h_{2j-1}\tau_{2j}}{\theta_n^+ - \theta_n} + \tau_{2j}\tau_{2j-1}C'(\theta_n)\right). \quad (8)$$

4.2. Perturbation Analysis

The IPA estimator is just $C'(\theta)$ plus the sample derivative of $h_t(\theta, s, \omega)/t$ for a fixed ω, namely, at iteration n,

$$Y_n = C'(\theta_n) + h_t'(\theta_n, s_n, \omega)/t_n. \quad (9)$$

The sample derivative is

$$h_t'(\theta, s, \omega) = \sum_{i=1}^{t}\sum_{j=v_i}^{i}\frac{\partial B_\theta^{-1}(U_j)}{\partial\theta} = \sum_{i=1}^{t}\sum_{j=v_i}^{i} -\theta\ln(1-U_j), \quad (10)$$

where v_i is the first customer in the busy period to which customer i belongs. One can either impose $v_i \geq 1$, or allow zero or negative values of v_i. The former means that the inside sum in (10), called the IPA accumulator, is reset to zero between iterations. The initial state s_n can also be either 0 for all n (always restart from an empty system), or be taken from the previous iteration. See L'Ecuyer and Glynn (1994b) for more details.

One regenerative version of IPA goes as follows. For a given regenerative cycle, let τ be the number of customers in the cycle, and $h_\tau'(\theta, 0, \omega)$ be the sample derivative for that cycle. At iteration n, take t_n regenerative cycles, let τ_j and h_j' denote the respective values of τ and $h_\tau'(\theta, 0, \omega)$ for the jth cycle, and let $T_n = \sum_{j=1}^{t_n}\tau_j$. One has the regenerative IPA estimator

$$Y_n = C'(\theta_n) + \frac{\sum_{j=1}^{t_n} h_j'}{\sum_{j=1}^{t_n}\tau_j} = C'(\theta_n) + \frac{1}{T_n}\sum_{i=1}^{T_n}\sum_{j=v_i}^{i} -\theta\ln(1-U_j). \quad (11)$$

This estimator is biased for $\alpha'(\theta_n)$. On the other hand, the alternative regenerative IPA estimator

$$Y_n = \frac{1}{t_n}\sum_{j=1}^{t_n}(h_j' + \tau_j C'(\theta_n)), \quad (12)$$

suggested by Fu (1990), is unbiased for $l(\theta_n)\alpha'(\theta_n)$, even for $t_n = 1$.

When c_n is very small, FDC becomes (in principle) essentially the same as IPA. But beware of implementation details; they can make a big difference. For example, it is shown in L'Ecuyer and Glynn (1994b) that SA with IPA, with a fixed number of customers per iteration, converges weakly to the optimizer, provided that the IPA accumulators are kept (no reset) between iterations. In Appendix I, we show that SA with FDC, with a fixed number of customers per iteration, converges to a *different* value than the optimizer θ^*. Our numerical results also illustrate that. The intuitive explanation is that if the parameter converges, its change eventually becomes negligible and keeping the IPA accumulator across iterations yields a derivative estimator whose bias eventually becomes negligible even with constant (and small) t_n. With FDC, on the other hand, there is no similar transfer of information between iterations, so that with constant t_n, the bias of the derivative estimator does not vanish. Note that exactly the same problem occurs with IPA if the IPA accumulator is reset to zero between iterations. The latter case really corresponds to the limit of FDC as $c_n \to 0$.

4.3. Likelihood Ratio

For a simulation of t customers, with initial state $X_1 = s$, one has the associated *score function*

$$S_t(\theta, s, \omega) = \sum_{i=1}^{t}\frac{\partial}{\partial\theta}\ln b_\theta(\zeta_i) = \sum_{i=1}^{t}\frac{\zeta_i - \theta}{\theta^2}.$$

The (truncated horizon) LR derivative estimator at iteration n is then (L'Ecuyer and Glynn 1993):

$$Y_n = C'(\theta_n) + h_{t_n}(\theta_n, s_n, \omega_n)S_{t_n}(\theta_n, s_n, \omega_n)/t_n. \quad (13)$$

A slightly different estimator, which uses the score function as a control variate, is

$$Y_n = C'(\theta_n) + [h_{t_n}(\theta_n, s_n, \omega_n) - w(\theta^*)]S_{t_n}(\theta_n, s_n, \omega_n)/t_n. \quad (14)$$

It has the same expectation as (13), but reduces the variance from $O(t_n)$ to $O(1)$ at $\theta = \theta^*$ (see L'Ecuyer and Glynn 1994a). Of course, in practice, $w(\theta^*)$ is unknown, but it can be estimated. Another variant of (13) is the *triangular* LR estimator

$$Y_n = \sum_{i=1}^{t_n}\left(W_i^* \sum_{j=1}^{i}\frac{\zeta_i - \theta}{\theta^2}\right), \quad (15)$$

in which, roughly speaking, each customer has its own score function.

One problem with LR estimators is (typically) their large variances. Sometimes this can be countered by exploiting the regenerative structure. Suppose we simulate the system for t_n regenerative cycles at iteration n, using parameter value θ_n. Let τ_j be the number of departures during the jth cycle, h_j the total system time for those τ_j customers who left during that cycle, and S_j the score function associated with that cycle. A (biased) regenerative LR estimator of $\alpha'(\theta_n)$ is then (see Glynn 1987):

$$Y_n = C'(\theta_n) + \frac{\sum_{j=1}^{t_n}\tau_j \sum_{j=1}^{t_n} h_j S_j - \sum_{j=1}^{t_n} h_j \sum_{j=1}^{t_n}\tau_j S_j}{(\sum_{j=1}^{t_n}\tau_j)^2}. \quad (16)$$

On the other hand, instead of trying to estimate $\alpha'(\theta_n)$, one can rather estimate $l^2(\theta_n)\alpha'(\theta_n)$, as suggested by Glynn (1986). From $2t_n$ regenerative cycles, one obtains the unbiased estimator:

$$Y_n = \frac{1}{t_n}\sum_{j=1}^{t_n}\left(\frac{1}{2}[h_{2j}S_{2j}\tau_{2j-1} + h_{2j-1}S_{2j-1}\tau_{2j} - h_{2j-1}S_{2j}\tau_{2j} - h_{2j}S_{2j-1}\tau_{2j-1}] + \tau_{2j}\tau_{2j-1}C'(\theta_n)\right). \quad (17)$$

5. The Experimental Setup

In these experiments, we have tried many SA-GET combinations, or *variants*. For each variant, we made N simulation runs, each yielding an estimation of θ^*. The N initial parameter values were randomly chosen, uniformly in Θ, and the initial state was $s = 0$ (an empty system). Across the variants, we used common random numbers and the same set of initial parameter values. This means that the different entries of Table 1 are strongly correlated. Each run was stopped after a (fixed) total of $\bar{T}$ ends of service. Hence, all variants were subjected to approximately the same sequence of random numbers and, if we neglect the differences in overhead for the GETs, used about the same CPU time. (The overhead was quite low in general, except for very small values of t_n, like $t_n = 1$.)

For each variant, we computed the empirical mean $\bar{\theta}$, standard deviation s_d, and standard error s_e of the N retained parameter values. If y_i denotes the retained parameter value for run i (i.e., the value of θ_n after the last iteration, for that run), the above quantities are defined by

$$\bar{\theta} = \frac{1}{N}\sum_{i=1}^{N} y_i; \quad s_d^2 = \frac{1}{N-1}\sum_{i=1}^{N}(y_i - \bar{\theta})^2; \quad s_e^2 = \frac{1}{N}\sum_{i=1}^{N}(y_i - \theta^*)^2. \quad (18)$$

We also computed a confidence interval I_θ on the expectation of $\bar{\theta}$, assuming that $\sqrt{N}(\bar{\theta} - E(\bar{\theta}))/s_d$ follows a Student distribution with $N - 1$ degrees of freedom. This assumed limit distribution is reasonable, because the limit theory for SA states that the estimators y_i typically are asymptotically normally distributed (Benveniste, Metivier, and Priouret 1987, Fabian 1968, Kushner and Clark 1978).

6. Numerical Results

6.1. The Acronyms Used in the Tables

In the tables giving the numerical results, the acronyms FD, FDC, IPA, and LR refer to SA combined with the corresponding DET. An R appended to the acronym means that a regenerative approach was used. For example, LRR refers to the regenerative version of LR given in (16), while IPAR refers to the regenerative

version of IPA given in (11). FDR86 and LRR86 refer to the regenerative methods (8) and (17). IPARFU refers to the regenerative IPA method of Fu (1990), given in (12). TLR means the "triangular" version of LR given by (15). CLR means the "control variate" version of LR given in (14), while CTLR is the corresponding "control variate" version of TLR.

The symbol -0 means that the state was reset to $s = 0$ at the beginning of each iteration. The symbol -Z following IPA means that the IPA accumulator was reset to 0 between iterations. The symbol -S following FD means that instead of always simulating first at $\theta_n - c_n$ and then at $\theta_n + c_n$, we adopted the following heuristic rule: if the parameter has just decreased, simulate first on the right (at $\theta_n + c_n$), otherwise simulate first on the left. The rationale is that if the parameter has just decreased, the current state has been reached by simulating at a parameter value larger than θ_n, and should thus be a statistically "better" initial state for a simulation at $\theta_n + c_n$ than at $\theta_n - c_n$ (and symmetrically if the parameter has just increased).

The symbol -K following the acronym signifies that Kesten's rule was used. The symbol -P means that Perron's rule was used to reduce γ_n quickly at the beginning. The symbol -A means that Andradóttir's method was used. The symbol -Av means that we used the averaging method of Polyak (1990), Kushner and Yang (1993), and Yin (1992). For the averaging, we selected a constant T_0 and as soon as the number of simulated customers reached T_0, we started averaging the θ_n's. In other words, we fixed T_0 and the window size m in (5) was the number of iterations performed with the last $\bar{T} - T_0$ customers.

6.2. Results for the First Example

Table 1 summarizes the results of an experiment we made with $\bar{T} = 10^6$, $N = 10$, $\Theta = [0.01, 0.95]$ and $C_1 = 1$. The optimal solution is $\theta^* = 0.5$. We computed 99% confidence intervals I_θ as described in §5, and the entries for which I_θ does not contain θ^* are marked by the symbol "◁" in the tables. This could be the symptom of a parameter that converges to the wrong value, but not necessarily so. In most cases, the convergence intervals do not contain the optimizer because over a finite simulation length, the retained parameter value at the end of the SA procedure is a biased estimator of the optimizer, and the squared bias converges to zero slowly compared to the variance.

For all the methods whose results are given in the table, with a few exceptions, the algorithm has been proved to converge to the optimizer (L'Ecuyer and Glynn 1994b). The exceptions are IPAR and FDC with constant t_n (which converge to the wrong value), FD-S, SAMOPT, and the methods which use Kesten's or Perron's rules. For most of the methods, however, the convergence rate is unknown (although this is the subject of ongoing research). Exceptions are the regenerative methods IPARFU and LRR86 with constant t_n, for which SA converges at the canonical rate when γ_0 is large enough. Indeed, since (12) and (17) are unbiased estimators of $l(\theta_n)\alpha'(\theta_n)$ and $l^2(\theta_n)\alpha'(\theta_n)$, respectively, and their variance is bounded uniformly over Θ, it follows from the discussion of §3 that $n^{-1/2}(\theta_n - \theta^*)$ converges to a centered normal for those two methods when $\gamma = 1$ and $\gamma_0 > (1 - \theta^*)\gamma_0^*/2$ (for IPARFU) or $\gamma_0 > (1 - \theta^*)^2\gamma_0^*/2$ (for LRR86). For the other cases, one can get some empirical idea of the convergence rates by looking at Table 2.

6.3. Infinitesimal Perturbation Analysis

The DET which performs the best for this example is clearly IPA. One of the smallest MSE, among the variants that we have tried, was obtained by the SA-IPA combination with a fixed number of customers per iteration, with $\gamma_n = 0.03n^{-1}$, and with the IPA accumulator *not* reset to zero between iterations. The regenerative SA-IPA method proposed by Fu, as well as SA-IPA with averaging, also yielded similar MSE values. However, it appears very clearly that the performance of most variants depends heavily on the choice of the constant γ_0, which scales the sequence of gains $\{\gamma_n, n \geq 1\}$. Recall that in the case where all iterations of SA use *unbiased* i.i.d. gradient estimators, the (asymptotically) optimal sequence is $\gamma_n = \gamma_0^*/n$ (equation (3)). Here, for most of the DET variants, we do not have i.i.d. unbiased estimators. But for IPA with a fixed number of customers per iteration, it seems that we have a good approximation of it: as n increases, Y_n should approach some steady-state distribution, and its expectation should converge to $w'(\theta^*)$ (the bias should become negligible), because the parameter θ converges to

Table 1 **Experimental Results for $\bar{T} = 10^6$, $N = 10$, and $C_1 = 1$ ($\theta^* = \frac{1}{2}$).**

Algorithm	t_n	γ_n	s_d	s_e	
IPA	10	$100n^{-1}$	0.02050	0.01953	
IPA	10	n^{-1}	0.00227	0.00216	
IPA	10	$0.1n^{-1}$	0.00091	0.00087	
IPA	10	$0.05n^{-1}$	0.00068	0.00066	
IPA	10	$0.04n^{-1}$	0.00063	0.00062	
IPA	10	$0.03n^{-1}$	0.00057	0.00057	
IPA	10	$0.02n^{-1}$	0.00061	0.00068	
IPA	10	$0.01n^{-1}$	0.00548	0.00530	
IPA	10	$0.001n^{-1}$	0.10614	0.11974	
IPA	1	$0.03n^{-1}$	0.00057	0.00057	
IPA	100	$0.03n^{-1}$	0.00056	0.00056	
IPA	n	n^{-1}	0.00195	0.00185	
IPA	n	$0.1n^{-1}$	0.00068	0.00066	
IPA	n	$0.07n^{-1}$	0.00060	0.00060	
IPA	n	$0.05n^{-1}$	0.00054	0.00055	
IPA	n	$0.04n^{-1}$	0.00061	0.00070	
IPA	n	$0.03n^{-1}$	0.00128	0.00170	◁
IPA	n	$0.02n^{-1}$	0.00464	0.00608	◁
IPA	$100 + n$	$0.07n^{-1}$	0.00063	0.00062	
IPA	$100 + n$	$0.05n^{-1}$	0.00055	0.00055	
IPA	$100 + n$	$0.03n^{-1}$	0.00053	0.00056	
IPA	n	$0.03n^{-1/2}$	0.00202	0.00192	
IPA	n	$0.01n^{-1/2}$	0.00132	0.00125	
IPA	n	$0.005n^{-1/2}$	0.00088	0.00085	
IPA	n	$0.003n^{-1/2}$	0.00066	0.00063	
IPA	n	$0.001n^{-1/2}$	0.01453	0.01583	
IPA	n	0.01	0.00272	0.00356	
IPA	n	0.001	0.00217	0.00206	
IPA	n	0.0005	0.00172	0.00163	
IPA	n	0.0003	0.00138	0.00131	
IPA	n	0.0002	0.00110	0.00105	
IPA	n	0.0001	0.00144	0.00165	
IPA	n	0.00005	0.01401	0.01653	
IPA	$n^{1/2}$	0.01	0.01800	0.01833	
IPA	$n^{1/2}$	0.001	0.00462	0.00477	
IPA	$n^{1/2}$	0.0001	0.00222	0.00211	
IPA-Z	n	n^{-1}	0.00192	0.00189	
IPA-Z	n	$0.2n^{-1}$	0.00087	0.00100	
IPA-Z	n	$0.1n^{-1}$	0.00065	0.00093	◁
IPA-Z	n	$0.07n^{-1}$	0.00059	0.00103	◁
IPA-Z	n	$0.03n^{-1}$	0.00180	0.00528	◁
IPA-Z	n	$0.1n^{-3/4}$	0.00160	0.00160	
IPA-Z	n	$0.05n^{-3/4}$	0.00111	0.00117	
IPA-Z	n	$0.01n^{-3/4}$	0.00081	0.00171	◁
IPA-Z	n	$0.01n^{-1/2}$	0.00125	0.00130	
IPA-Z	10	n^{-1}	0.00169	0.07365	◁
IPA-Z	10	$0.03n^{-1}$	0.00165	0.07461	◁
IPA-0	n	$0.1n^{-1}$	0.00065	0.00103	◁
IPA-K	10	$0.01n^{-1}$	0.00057	0.00057	
IPA-K	10	$0.001n^{-1}$	0.03972	0.03811	
IPA-K	n	$0.03n^{-1}$	0.00058	0.00058	
IPA-K	n	$0.01n^{-1}$	0.00300	0.00312	
IPA-KP	10	$100n^{-1}$	0.00225	0.00215	
IPA-KP	10	n^{-1}	0.00175	0.00167	
IPA-KP	10	$0.03n^{-1}$	0.00085	0.00082	
IPA-KP	10	$0.01n^{-1}$	0.00057	0.00057	
IPA-KP	n	$100n^{-1}$	0.00175	0.00166	
IPA-KP	n	n^{-1}	0.00135	0.00129	
IPA-KP	n	$0.03n^{-1}$	0.00058	0.00058	
IPA-KP	n	$0.01n^{-1}$	0.00300	0.00312	
IPAR	5	$0.03n^{-1}$	0.00079	0.06188	◁
IPAR	n	n^{-1}	0.00203	0.00200	
IPAR	n	$0.1n^{-1}$	0.00067	0.00101	◁
IPAR	n	$0.03n^{-1}$	0.00129	0.00424	◁
IPAR	n	0.01	0.00334	0.00344	
IPAR	n	0.001	0.00191	0.00186	
IPAR	n	0.0001	0.00590	0.00614	
IPARFU	1	$0.1n^{-1}$	0.00135	0.00129	
IPARFU	1	$0.03n^{-1}$	0.00072	0.00070	
IPARFU	1	$0.015n^{-1}$	0.00056	0.00056	
IPARFU	1	$0.01n^{-1}$	0.00059	0.00060	
IPARFU	10	$0.03n^{-1}$	0.00072	0.00070	
IPARFU	10	$0.01n^{-1}$	0.00060	0.00060	
IPARFU	10	$0.005n^{-1}$	0.00539	0.00547	
IPARFU	10	$0.001n^{-1}$	0.07176	0.09161	
IPARFU	n	$0.01n^{-1}$	0.00378	0.00379	
IPARFU	n	0.001	0.00244	0.00232	
IPARFU	n	0.0001	0.00086	0.00083	

	ϵ	t_n	γ_n	s_d	s_e	
IPA-A	0.01	n	$0.03n^{-1}$	0.03067	0.03075	
IPA-A	0.1	n	$0.03n^{-1}$	0.00125	0.00121	
IPA-A	1	n	$0.03n^{-1}$	0.00055	0.00054	
IPA-A	10	n	$0.03n^{-1}$	0.04802	0.04703	
IPA-A	100	n	$0.03n^{-1}$	0.14211	0.16153	
IPA-A	33	n	n^{-1}	0.00058	0.00058	
IPA-A	1	n	n^{-1}	0.00256	0.00244	
IPA-A	1	n	$100n^{-1}$	0.15651	0.17652	
IPA-A	100	n	$100n^{-1}$	0.00249	0.00238	
IPA-A	0.1	10	$0.03n^{-1}$	0.00111	0.01145	◁
IPA-A	1	10	$0.03n^{-1}$	0.00080	0.01444	◁
IPA-A	10	10	$0.03n^{-1}$	0.02878	0.02737	

	$T - T_0$ (window)	t_n	γ_n	s_d	s_e	
IPA-Av	1000000	10	$100n^{-1}$	0.00070	0.00846	◁
IPA-Av	1000000	10	n^{-1}	0.00058	0.00061	
IPA-Av	100000	10	n^{-1}	0.00220	0.00209	

Table 1 *Continued*

Algorithm	$T - T_0$ (window)	t_n	γ_n	s_d	s_e	
IPA-Av	10000	10	n^{-1}	0.00296	0.00281	
IPA-Av	1000	10	n^{-1}	0.00231	0.00220	
IPA-Av	100000	10	$0.1n^{-1}$	0.00074	0.00072	
IPA-Av	100000	10	$0.05n^{-1}$	0.00060	0.00060	
IPA-Av	1000000	10	$0.03n^{-1}$	0.00055	0.00060	
IPA-Av	100000	10	$0.03n^{-1}$	0.00052	0.00053	
IPA-Av	10000	10	$0.03n^{-1}$	0.00058	0.00058	
IPA-Av	100000	10	$0.01n^{-1}$	0.00562	0.00544	
IPA-Av	100000	10	$0.001n^{-1}$	0.10635	0.11998	
IPA-Av	1000000	10	$n^{-5/6}$	0.00059	0.00075	
IPA-Av	1000000	10	$n^{-3/4}$	0.00058	0.00112	◁
IPA-Av	100000	10	$n^{-3/4}$	0.00259	0.00246	
IPA-Av	10000	10	$n^{-3/4}$	0.00526	0.00499	
IPA-Av	1000	10	$n^{-3/4}$	0.00753	0.00773	
IPA-Av	100000	10	$0.03n^{-3/4}$	0.00167	0.00159	
IPA-Av	100000	10	$0.01n^{-3/4}$	0.00102	0.00099	
IPA-Av	100000	10	$0.005n^{-3/4}$	0.00068	0.00067	
IPA-Av	1000000	10	$n^{-1/2}$	0.00068	0.01009	◁
IPA-Av	100000	10	$0.01n^{-1/2}$	0.00271	0.00257	
IPA-Av	100000	10	$0.001n^{-1/2}$	0.00131	0.00126	
IPA-Av	100000	10	$0.0005n^{-1/2}$	0.00093	0.00090	
IPA-Av	100000	10	$0.0002n^{-1/2}$	0.00257	0.00299	
IPA-KP-Av	100000	10	$100n^{-1}$	0.00189	0.00179	
IPARFU-Av	1000000	1	$100n^{-1}$	0.00083	0.02437	◁
IPARFU-Av	1000000	1	n^{-1}	0.00058	0.00059	
IPARFU-Av	1000000	1	$0.1n^{-1}$	0.00051	0.00053	
IPARFU-Av	100000	1	$0.1n^{-1}$	0.00111	0.00107	
IPARFU-Av	1000000	1	$0.015n^{-1}$	0.00052	0.00053	
IPARFU-Av	100000	1	$0.015n^{-1}$	0.00052	0.00052	
IPARFU-Av	1000000	1	$0.001n^{-1}$	0.07402	0.08674	
IPARFU-Av	1000000	1	$0.001n^{-1/2}$	0.00052	0.00050	

	t_n	γ_n	c_n	s_d	s_e	
FD	n	n^{-1}	$0.1n^{-1/6}$	0.00979	0.00967	
FD	n	$0.1n^{-1}$	$0.1n^{-1/6}$	0.00150	0.00251	◁
FD	n	$0.03n^{-1}$	$0.1n^{-1/6}$	0.00807	0.01708	◁
FD	$100 + n$	n^{-1}	$0.1n^{-1/6}$	0.01075	0.01044	
FD	$100 + n$	$0.03n^{-1}$	$0.1n^{-1/6}$	0.00160	0.00216	◁
FD	n	$0.1n^{-3/4}$	$0.1n^{-1/6}$	0.00676	0.00643	
FD	n	$0.03n^{-3/4}$	$0.1n^{-1/6}$	0.00200	0.00252	
FD	n	$0.01n^{-3/4}$	$0.1n^{-1/6}$	0.00285	0.00559	◁
FD	n	$0.03n^{-1/2}$	$0.1n^{-1/6}$	0.00946	0.00930	
FD	n	$0.01n^{-1/2}$	$0.1n^{-1/6}$	0.00406	0.00395	
FD-S	n	n^{-1}	$0.1n^{-1/6}$	0.00760	0.00732	
FD-S	n	$0.1n^{-1}$	$0.1n^{-1/4}$	0.00467	0.00443	
FD-S	n	$0.1n^{-1}$	$0.1n^{-1/6}$	0.00352	0.00334	
FD-S	n	$0.03n^{-1}$	$0.1n^{-1/2}$	0.02581	0.02480	
FD-S	n	$0.03n^{-1}$	$0.1n^{-1/4}$	0.00583	0.00751	◁
FD-S	n	$0.03n^{-1}$	$0.1n^{-1/6}$	0.00440	0.00750	◁

Algorithm	t_n	γ_n	c_n	s_d	s_e	
FD-S	n	$0.1n^{-3/4}$	$0.1n^{-1/6}$	0.00682	0.00647	
FD-S	$100 + n$	$0.1n^{-3/4}$	$0.1n^{-1/6}$	0.00584	0.00554	
FDR86	1	$0.01n^{-1}$	$0.1n^{-1/2}$	0.08867	0.14053	◁
FDR86	1	$0.01n^{-1}$	$0.1n^{-1/4}$	0.02266	0.02512	
FDR86	1	$0.1n^{-1}$	$0.1n^{-1/6}$	0.01633	0.01549	
FDR86	1	$0.03n^{-1}$	$0.1n^{-1/6}$	0.00819	0.01019	◁
FDR86	1	$0.01n^{-1}$	$0.1n^{-1/6}$	0.00609	0.00585	
FDR86	1	$0.005n^{-1}$	$0.1n^{-1/6}$	0.00802	0.00761	
FDR86	5	$0.03n^{-1}$	$0.1n^{-1/4}$	0.03666	0.03682	
FDR86	5	$0.03n^{-1}$	$0.1n^{-1/6}$	0.00511	0.00485	
FDR86	5	$0.01n^{-1}$	$0.1n^{-1/4}$	0.01683	0.01597	
FDR86	5	$0.01n^{-1}$	$0.1n^{-1/6}$	0.00309	0.00311	
FDR86	5	$0.005n^{-1}$	$0.1n^{-1/6}$	0.00468	0.00480	
FRD86	n	$0.01n^{-1}$	$0.1n^{-1.6}$	0.00227	0.00216	
SAMOPT	n			0.08045	0.17934	◁
FDC	5	n^{-1}	$0.1n^{-1}$	0.00115	0.15253	◁
FDC	5	$0.03n^{-1}$	$0.1n^{-1}$	0.00928	0.15485	◁
FDC	100	$0.03n^{-1}$	$0.1n^{-1}$	0.00104	0.00546	◁
FDC	$100 + n$	$0.03n^{-1}$	$0.1n^{-1}$	0.00112	0.00192	◁
FDC	n	n^{-1}	$0.1n^{-1}$	0.00178	0.00176	
FDC	n	n^{-1}	$0.1n^{-1/5}$	0.00193	0.00184	
FDC	n	$0.2n^{-1}$	$0.1n^{-1}$	0.00122	0.00126	
FDC	n	$0.1n^{-1}$	$0.1n^{-1/5}$	0.00109	0.00105	
FDC	n	$0.1n^{-1}$	$0.1n^{-1/2}$	0.00104	0.00114	
FDC	n	$0.1n^{-1}$	$0.1n^{-1}$	0.00102	0.00114	
FDC	n	$0.1n^{-1}$	$0.01n^{-1}$	0.00102	0.00114	
FDC	n	$0.1n^{-1}$	$0.001n^{-1}$	0.00102	0.00114	
FDC	n	$0.1n^{-1}$	$0.001n^{-2}$	0.00102	0.00114	
FDC	n	$0.07n^{-1}$	$0.1n^{-1}$	0.00094	0.00125	◁
FDC	n	$0.03n^{-1}$	$0.1n^{-1}$	0.00276	0.00686	◁
FDC	n	$0.1n^{-1/2}$	$0.1n^{-1}$	0.00167	0.00159	
FDC	$n^{1/2}$	$0.1n^{-1}$	$0.1n^{-1}$	0.00115	0.00722	◁
FDC	n	$0.03n^{-1/2}$	$0.1n^{-1}$	0.00175	0.00173	
FDC	$n^{1/2}$	$0.03n^{-1}$	$0.1n^{-1}$	0.00297	0.01880	◁
FDC-0	n	$0.1n^{-1}$	$0.1n^{-1}$	0.00106	0.00129	
FDC-0	n	$0.03n^{-1}$	$0.1n^{-1}$	0.00342	0.00781	◁
FDC-K	n	$0.03n^{-1}$	$0.1n^{-1}$	0.00093	0.00139	◁
FDC-K	n	$0.01n^{-1}$	$0.1n^{-1}$	0.00240	0.00651	◁
FDC-KP	n	$100n^{-1}$	$0.1n^{-1}$	0.00179	0.00180	
FDC-KP	n	n^{-1}	$0.1n^{-1}$	0.00165	0.00171	
FDC-KP	n	$0.03n^{-1}$	$0.1n^{-1}$	0.00093	0.00139	◁
FDC-KP	n	$0.01n^{-1}$	$0.1n^{-1}$	0.00240	0.00651	◁

	t_n	γ_n	s_d	s_e	
LR	$n^{1/3}$	$0.1n^{-1}$	0.00431	0.02015	◁
LR	$n^{1/3}$	$0.03n^{-1}$	0.00884	0.04538	◁
LR	$n^{1/2}$	n^{-1}	0.03038	0.02899	
LR	$n^{1/2}$	$0.1n^{-1}$	0.01001	0.01017	
LR	$n^{1/2}$	$0.03n^{-1}$	0.00640	0.01597	◁

Table 1 *Continued*

Algorithm	t_n	γ_n	s_d	s_e		Algorithm	t_n	γ_n	s_d	s_e	
LR	$n^{2/3}$	$0.1n^{-1}$	0.02401	0.02280		LRR	$n^{2/3}$	$0.03n^{-1}$	0.00385	0.02631	◁
LR	$n^{2/3}$	$0.03n^{-1}$	0.01365	0.01501		LRR	n	n^{-1}	0.00449	0.00456	
LR	n	$0.1n^{-1}$	0.09962	0.09491		LRR	n	$0.2n^{-1}$	0.00286	0.00313	
TLR	$n^{1/2}$	$0.1n^{-1}$	0.00535	0.00632		LRR	n	$0.1n^{-1}$	0.00240	0.00318	
TLR	$n^{1/2}$	$0.03n^{-1}$	0.00454	0.01526	◁	LRR	n	$0.03n^{-1}$	0.00396	0.01449	◁
CLR	$n^{1/2}$	n^{-1}	0.00772	0.00749		CTLRR	$n^{2/3}$	$0.03n^{-1}$	0.00244	0.02239	◁
CLR	$n^{1/2}$	$0.1n^{-1}$	0.00383	0.00582	◁	CTLRR	n	$0.2n^{-1}$	0.00220	0.00268	
CLR	$n^{1/2}$	$0.03n^{-1}$	0.00452	0.01564	◁	CTLRR	n	$0.1n^{-1}$	0.00186	0.00283	◁
CLR-0	$n^{1/2}$	$0.1n^{-1}$	0.00990	0.01073		CTLRR	n	$0.03n^{-1}$	0.00256	0.01251	◁
CLR-0	$n^{1/2}$	$0.03n^{-1}$	0.00760	0.02194	◁	LRR86	1	n^{-1}	0.01161	0.01108	
CTLR	$n^{1/2}$	n^{-1}	0.00533	0.00615		LRR86	1	$0.1n^{-1}$	0.00556	0.00531	
CTLR	$n^{1/2}$	$0.1n^{-1}$	0.00222	0.00575	◁	LRR86	1	$0.03n^{-1}$	0.00383	0.00368	
CTLR	$n^{1/2}$	$0.03n^{-1}$	0.00310	0.01518	◁	LRR86	1	$0.01n^{-1}$	0.00308	0.00293	
CTLR	$n^{2/3}$	n^{-1}	0.00706	0.00688		LRR86	1	$0.005n^{-1}$	0.00372	0.00376	
CTLR	$n^{2/3}$	$0.2n^{-1}$	0.00395	0.00387		LRR86	5	$0.03n^{-1}$	0.00396	0.00376	
CTLR	$n^{2/3}$	$0.1n^{-1}$	0.00271	0.00317		LRR86	5	$0.01n^{-1}$	0.00340	0.00323	
CTLR	$n^{2/3}$	$0.03n^{-1}$	0.00273	0.00907	◁	LRR86	5	$0.005n^{-1}$	0.00107	0.00123	
CTLR	n	$0.1n^{-1}$	0.00503	0.00524							

For the values marked with ◁, the 99% confidence interval does not contain θ^*.

θ^*. The proof of Proposition 6 of L'Ecuyer and Glynn (1994b) is in fact based on this reasoning. Therefore, $\gamma_0 = \gamma_0^* = \frac{1}{32} = .03125 \approx .03$ should be expected to yield the best asymptotic performance, and this explains our numerical results. With a larger γ_0, there is too much noise, while with a smaller γ_0, convergence is slower because the SA steps are too small. According to (3), for $\gamma_0 < .0156$, we should not even expect to obtain the canonical convergence rate. When t_n is not constant, the above reasoning is not necessarily true. For example, with $t_n = n$, $\gamma_0 = .03$ no longer gives the best performance here; $\gamma_0 = .05$ is much better.

The idea of taking $\gamma_n = \gamma_0 n^{-\gamma}$ for some $\gamma < 1$, with increasing t_n, as discussed in Wardi (1988) and Dupuis and Simha (1991) (for $\gamma = 0$), does not bring any improvement here. The best γ_0 is smaller for smaller γ, but even with the best γ_0, the results are not quite as good as when using the standard sequence ($\gamma = 1$).

The performance of IPA also deteriorates when the IPA accumulator is reset to zero between iterations (IPA-Z or IPA-0). This resetting introduces a bias, which forces one to increase t_n with n, otherwise θ_n converges to the wrong value. For example, with $t_n = 10$, θ_n converges to somewhere around 0.575 instead of 0.5, and this is why s_e is much larger than s_d. For IPA-Z with $t_n = n$, θ_n converges to θ^* (see Proposition 5 of L'Ecuyer and Glynn 1994b), but the numerical results suggest that the best value of γ_0 here is much larger than 0.03. When γ_0 is too small, θ_n still converges to the optimum, but very slowly, and (apparently) in such a way that the variance of the noise converges significantly faster than the squared bias. As a result, the confidence interval I_θ, based on the N final values of θ_n, is very likely not to cover θ^*. This is what happens, for instance, for IPA-Z with $\gamma_0 \leq 0.1$.

So, the results are very sensitive to the choice of γ_0. In §3, we have described a few "adaptive" approaches designed for the (usual) case where the optimal γ_0 is unknown. We now look at how well they perform here. Kesten's rule helps somewhat when γ_0 has been chosen slightly too small, but it does not prevent disaster when γ_0 is much too small. Combining the heuristics of Perron and Kesten appears effective; it gives reasonable results even with a much too large initial γ_0. It is not clear whether this observation can be extrapolated to more general systems, but if so, a suggested practical strategy could be to start with a large value of γ_0 and use both Perron and Kesten rules.

Andradóttir's algorithm does not help here. With the "optimal" choice of ϵ and $t_n = n$, its performance is the same as the standard algorithm, but for other values of ϵ, it is much worse. Note that the optimal ϵ depends on γ_0. For example, with $t_n = n$, $\epsilon = 1$ with $\gamma_0 = .03$ behaves pretty much the same as $\epsilon = 33$ with $\gamma_0 = 1.0$. With constant t_n (like $t_n = 10$), it fails completely. The reason is that here, Y_n^1 and Y_n^2 are *correlated*, and therefore the combined estimator (4) is biased. That bias goes to zero if t_n increases with n, but not if t_n is held constant.

With IPAR, the number of ends of service during the t_n regenerative cycles is now random, and the derivative estimator is biased because it is a ratio with that number in the denominator. So, with t_n held constant, SA with IPAR converges quickly, but to the wrong value. However, the bias goes to zero as t_n goes to infinity, and as proved in the companion paper, SA-IPAR converges towards the optimum with $t_n = n$. For small γ_0, though, s_e is much larger than s_d, which indicates that the squared bias converges more slowly than the variance.

The IPARFU variant works pretty well, even with $t_n = 1$. Here, each Y_n is an unbiased estimator of $\alpha'(\theta_n)l(\theta_n)$. These estimators are also approximately i.i.d. when θ_n converges to θ^*. However, the optimal γ_0 here is $\tilde{\gamma}_0^* = \frac{1}{64} \approx 0.0156$. The numerical results agree with that.

The averaging method gives no significant improvement over standard SA with a well chosen sequence γ_n, but good improvement when γ_0 is larger than the optimum and the window is wide enough. If γ_0 is much too large (e.g., $\gamma_0 = 100$), averaging still reduces the variance but there is a highly significant bias, except if we combine averaging with Kesten and Perron's rules, in which case we obtain very good results. Convergence is not speeded up significantly by averaging when γ_0 is much too small (e.g., $\gamma_0 = 0.001$). If we take $\gamma_n = \gamma_0 n^{-\gamma}$ for $\gamma < 1$ (instead of $\gamma = 1$), with averaging, we still obtain fair results, but not quite as good as with $\gamma = 1$. Further, the best γ_0 is smaller for smaller γ, and convergence is still very slow when γ_0 is too small or much too large. This is the same kind of behavior that we have observed for IPA with $\gamma < 1$, without averaging.

All of this suggests taking γ_0 on the "large" side when its optimal value is unknown, and averaging with a wide window, perhaps combined with the heuristics of Kesten and Perron.

6.4. Finite Differences

In general, FD without the common random numbers gives a rather large MSE. We see, however, that if the sequences γ_n and c_n are chosen in the best possible way, the performance could still be acceptable. FD-S is sometimes better than FD, but not always. FDR86 resembles LRR86 to some extent: at each iteration, it computes an estimator of $\alpha'(\theta_n)l^2(\theta_n)$. So, the optimal γ_0 in this case should be

$$\tilde{\gamma}_0^* = (1 - \theta^*)^2 / \alpha''(\theta^*) = \tfrac{1}{128} \approx 0.0078.$$

This agrees with our numerical results. SAMOPT (Azadivar and Talavage 1980) is clearly not competitive for this example. The parameters implemented in the SAMOPT package might have been well tuned for some classes of problems, but do not seem to always work well.

For both FD and FDC, when t_n is fixed to a constant, convergence is to the wrong value, as for IPA-Z, and as discussed in the Appendix. The results of FDC with $t_n = 5$ illustrate that: convergence is quick, but the large value of s_e indicates that the limit is not θ^*. Even for $t_n = 100$, the bias is still quite apparent. In general, FDC behaves pretty much the same as IPA-Z. This is to be expected, since IPA-Z is the limiting case of FDC as $c_n \to 0$. The only significant difference is that the number of customers per iteration required by FDC is twice that of IPA-Z. Therefore, for a fixed "budget" of $\bar{T}$ customers, fewer SA iterations will be performed with FDC, and this explains its slightly larger MSE. For FDC and IPA-Z, we see that the best value of γ_0 is larger than 0.03. Convergence is slow when γ_0 is too large; while when γ_0 is too small, not only is the convergence excruciatingly slow, but trusting the confidence intervals is also misleading. Using FDC-KP with an initially large γ_0 looks like a good heuristic.

6.5. Likelihood Ratio Derivative Estimators

The LR methods in general have trouble due to their large associated variance. They give the worst numerical results here. For the nonregenerative variants, when t_n grows more slowly, the variance is usually smaller; but the bias then goes down much too slowly compared to the variance, and we get the same problem as for IPA-Z and FDC: we cannot trust the confidence intervals. This is what happens, for instance, with $t_n = n^{1/3}$.

Among the truncated-horizon (nonregenerative) variants, CTLR is a significant improvement over LR but falls far behind IPA and FDC with good parameter choices.

The regenerative LR variants perform better. The best are CTLRR with $t_n = n$ and LRR86 with $t_n = 5$. For the latter, the optimal γ_0 is around $\tilde{\tilde{\gamma}}_0^* \approx 0.008$. With $t_n = n^{2/3}$, for both LRR and CTLRR, the bias goes down too slowly and I_θ does not contain θ^*. Nevertheless, the MSE of all the LR variants given in this table converges (slowly) to zero, as proved in L'Ecuyer and Glynn (1994b).

6.6. Shorter and Longer Runs

We made other experiments with $\bar{T} = 10^4$, 10^5, and 10^7, for some of the variants, to see how s_e evolves with the computer budget $\bar{T}$. We took $N = \min(10, 10^7/\bar{T})$. The results are given in Table 2. The fact that FDC with $t_n = 5$ converges to the wrong value is obvious from this table: s_e clearly fails to converge to zero. For all other variants, the results indicate that the s_e converges to zero, in accordance with the theory. Further, for many of the variants, the confidence intervals appear to become increasingly reliable as $\bar{T}$ increases.

6.7. Other Traffic Intensities

We also made other sets of experiments with $C_1 = \frac{1}{25}$ (for which $\theta^* = \frac{1}{6}$) and $C_1 = 25$ (for which $\theta^* = \frac{5}{6}$). The results appear in Tables 3 and 4. For $C_1 = \frac{1}{25}$, the traffic intensity for θ near θ^* is low, and we get a much lower variance than for $C_1 = 1$. The opposite is true for $C_1 = 25$. The relative "rankings" of the algorithms are about the same. One exception is LRR86, which becomes much less competitive in higher traffic.

Table 2 Values of s_e for Different Values of T, for $C_1 = 1$

				s_e							
$\bar{T}$	t_n	γ_n	c_n	10^4		10^5		10^6		10^7	
IPA	10	$0.03n^{-1}$		0.00753		0.00210		0.00056		0.00022	
IPA	n	$0.05n^{-1}$		0.00788		0.00222		0.00055		0.00019	
IPA	n	$0.02n^{-1}$		0.02844		0.01288		0.00608		0.00283	
IPA	n	0.001		0.00899		0.00378		0.00205		0.00167	
IPA-Z	n	$0.1n^{-1}$		0.00989	◁	0.00261	◁	0.00093	◁	0.00025	
IPA-Z	n	$0.03n^{-1}$		0.04381	◁	0.01592	◁	0.00527	◁	0.00175	◁
IPA-KP	n	$100n^{-1}$		0.01623		0.00440		0.00166		0.00075	
IPA-KP	10	$100n^{-1}$		0.02343		0.00761		0.00214		0.00116	
IPA-KP	10	$0.01n^{-1}$		0.00891		0.00286		0.00056		0.00022	
IPAR	n	$0.03n^{-1}$		0.02666	◁	0.00963	◁	0.00424	◁	0.00145	◁
IPARFU	1	$0.015n^{-1}$		0.00756		0.00211		0.00055		0.00022	
IPA-Av	10	n^{-1}		0.01117	◁	0.00239		0.00061		0.00073	
IPA-Av	10	$0.03n^{-1}$		0.01164		0.00343		0.00059		0.00020	
IPARFU-Av	1	$0.1n^{-1}$		0.00820		0.00235		0.00052		0.00046	
FD	n	$0.1n^{-1}$	$0.1n^{-1/6}$	0.03850	◁	0.00975	◁	0.00250	◁	0.00116	
FD	n	$0.03n^{-1}$	$0.1n^{-1/6}$	0.12724	◁	0.04582	◁	0.01708	◁	0.00557	◁
FD-S	n	$0.1n^{-1}$	$0.1n^{-1/6}$	0.02565	◁	0.00964	◁	0.00334		0.00122	
FDC	5	n^{-1}	$0.1n^{-1}$	0.15417	◁	0.15249	◁	0.15252	◁	0.15286	◁
FDC	n	$0.1n^{-1}$	$0.1n^{-1}$	0.01398	◁	0.00393	◁	0.00114		0.00053	
FDC-KP	n	$100n^{-1}$	$0.1n^{-1}$	0.02742	◁	0.00863		0.00180		0.00076	
CTLR	$n^{1/2}$	$0.1n^{-1}$		0.04102	◁	0.01401	◁	0.00575	◁	0.00248	◁
CTLR	$n^{2/3}$	$0.1n^{-1}$		0.02612	◁	0.00826	◁	0.00316		0.00157	
LRR	n	$0.1n^{-1}$		0.03777	◁	0.00954	◁	0.00318		0.00070	
CTLRR	n	$0.1n^{-1}$		0.02983	◁	0.00850	◁	0.00282	◁	0.00068	
LRR86	1	$0.01n^{-1}$		0.02974		0.00776		0.00292		0.00081	
LRR86	5	$0.008n^{-1}$		0.04347	◁	0.00751		0.00162		0.00059	

Note that the optimal γ_0 here is no longer $\frac{1}{32}$. For $C_1 = \frac{1}{25}$, one has $\gamma_0^* = \frac{125}{2592} \approx 0.0482$, $\tilde{\gamma}_0^* \approx 0.0402$, and $\tilde{\tilde{\gamma}}_0^* \approx 0.0335$; while for $C_1 = 25$, one has $\gamma_0^* = \frac{5}{2592} \approx 0.0019$, $\tilde{\gamma}_0^* \approx 0.00032$, and $\tilde{\tilde{\gamma}}_0^* \approx 0.000054$.

7. Conclusion

Using a simple $M/M/1$ queuing example, we have illustrated the numerical behavior of different variants of SA, combined with various derivative estimation methods, to optimize a steady-state stochastic system with respect to a continuous parameter. We observed that the results are quite sensitive to the choice of the sequence of gains in the SA algorithm. That kind of sensitivity had also been pointed out previously by many authors in different contexts. See Benveniste, Metivier, and Priouret (1987), Goldstein (1988), Kushner and Clark (1978), Polyak and Tsypkin (1980), and the references cited there. We have experimented with different variants of SA designed to improve convergence when the optimal sequence of gains is unknown. Some of them did improve convergence significantly in some situations, but others did not. The best results were obtained when IPA was used as a derivative estimation method, with the IPA accumulators *not* reset

Table 3 **Experimental Results for $\bar{T} = 10^6$, $N = 10$, and $C_1 = 1/25$ ($\theta^* = 1/6$)**

Algorithm	t_n	γ_n	c_n	s_d	s_e	
IPA	10	$0.1n^{-1}$		0.00021	0.00020	
IPA	10	$0.05n^{-1}$		0.00014	0.00013	
IPA	n	$0.1n^{-1}$		0.00016	0.00016	
IPA	n	$0.05n^{-1}$		0.00502	0.00503	
IPA-Z	n	$0.1n^{-1}$		0.00020	0.00020	
IPA-Z	n	$0.05n^{-1}$		0.00656	0.00729	
IPA-KP	n	$100n^{-1}$		0.00040	0.00038	
IPA-KP	10	$100n^{-1}$		0.00043	0.00041	
IPA-KP	10	$0.01n^{-1}$		0.00078	0.00112	◁
IPAR	n	$0.05n^{-1}$		0.00496	0.00569	
IPARFU	1	$0.04n^{-1}$		0.00015	0.00014	
IPA-Av	10	$100n^{-1}$		0.00023	0.00566	◁
IPA-Av	10	n^{-1}		0.00014	0.00016	
IPA-Av	10	$0.05n^{-1}$		0.00030	0.00034	
IPARFU-Av	1	$0.1n^{-1}$		0.00021	0.00020	
FD	n	$0.1n^{-1}$	$0.1n^{-1/6}$	0.00087	0.00093	
FD	n	$0.05n^{-1}$	$0.1n^{-1/6}$	0.01865	0.01981	
FD-S	n	$0.1n^{-1}$	$0.1n^{-1/6}$	0.00083	0.00081	
FDR86	5	$0.03n^{-1}$	$0.1n^{-1/6}$	0.00073	0.00078	
FDC	n	$0.1n^{-1}$	$0.1n^{-1/6}$	0.00016	0.00016	
FDC	n	$0.1n^{-1}$	$0.1n^{-1}$	0.00017	0.00019	
FDC-KP	n	$100n^{-1}$	$0.1n^{-1}$	0.00029	0.00039	
CTLR	$n^{1/2}$	$0.1n^{-1}$		0.01381	0.01465	
CTLR	$n^{2/3}$	$0.1n^{-1}$		0.02808	0.03317	
LRR	n	$0.1n^{-1}$		0.00041	0.00061	◁
CTLRR	n	$0.1n^{-1}$		0.00043	0.00062	◁
LRR86	1	$0.03n^{-1}$		0.00035	0.00034	
LRR86	5	$0.03n^{-1}$		0.00034	0.00032	

Table 4 **Experimental Results for $\bar{T} = 10^6$, $N = 10$, and $C_1 = 25$ ($\theta^* = 5/6$)**

Algorithm	t_n	γ_n	c_n	s_d	s_e	
IPA	10	$0.01n^{-1}$		0.00319	0.00318	
IPA	10	$0.002n^{-1}$		0.00122	0.00125	
IPA	n	$0.01n^{-1}$		0.00205	0.00206	
IPA	n	$0.002n^{-1}$		0.00410	0.00413	
IPA-Z	n	$0.01n^{-1}$		0.00224	0.00387	◁
IPA-Z	n	$0.002n^{-1}$		0.00118	0.01649	◁
IPA-KP	n	$100n^{-1}$		0.00610	0.00707	
IPA-KP	10	$100n^{-1}$		0.01957	0.02083	
IPA-KP	10	$0.001n^{-1}$		0.00323	0.00322	
IPAR	n	$0.01n^{-1}$		0.00210	0.00276	
IPARFU	1	$0.001n^{-1}$		0.00228	0.00228	
IPARFU	1	$0.0003n^{-1}$		0.00278	0.00421	◁
IPA-Av	10	$100n^{-1}$		0.00114	0.02395	◁
IPA-Av	10	n^{-1}		0.00140	0.00137	
IPA-Av	10	$0.002n^{-1}$		0.00134	0.00138	
IPARFU-Av	1	$0.01n^{-1}$		0.00651	0.00621	
FD	n	$0.01n^{-1}$	$0.1n^{-1/6}$	0.00442	0.01458	◁
FD	n	$0.002n^{-1}$	$0.1n^{-1/6}$	0.00975	0.06232	◁
FD-S	n	$0.01n^{-1}$	$0.1n^{-1/6}$	0.00520	0.00631	
FDC	n	$0.01n^{-1}$	$0.1n^{-1/6}$	0.00231	0.00563	
FDC	n	$0.01n^{-1}$	$0.1n^{-1}$	0.00234	0.00626	
FDC-KP	n	$100n^{-1}$	$0.1n^{-1}$	0.00860	0.00876	
CTLR	$n^{1/2}$	$0.01n^{-1}$		0.00517	0.06826	◁
CTLR	$n^{2/3}$	$0.05n^{-1}$		0.01169	0.01783	◁
CTLR	$n^{2/3}$	$0.01n^{-1}$		0.00593	0.02031	◁
LRR	n	$0.05n^{-1}$		0.01235	0.01252	
LRR	n	$0.01n^{-1}$		0.00595	0.00842	◁
CTLRR	n	$0.01n^{-1}$		0.00394	0.00766	◁
LRR86	1	$0.001n^{-1}$		0.02803	0.02699	
LRR86	1	$0.005n^{-1}$		0.07296	0.09760	
LRR86	1	$0.0002n^{-1}$		0.02141	0.03336	◁
LRR86	1	$0.0005n^{-1}$		0.11193	0.12495	◁
LRR86	1	$0.00005n^{-1}$		0.09838	0.23290	◁

to zero between SA iterations, and also with the regenerative IPA approach of Fu (1990). FDC and other IPA variants followed closely, while the LR method performed well only in its regenerative versions and when the regenerative cycles were very short. Our results also indicate that one must be *very careful* about confidence intervals in these kinds of experiments, even if they are asymptotically valid, because the bias sometimes converges rather slowly.

Our experiments dealt with an example where the decision parameter θ was one-dimensional. A multidimensional case certainly involves more intensive computations and perhaps further difficulties. For most complex systems encountered in practice, the regenerative variants would become impractical; then IPA, if it applies, could be used with a growing truncated horizon. When IPA does not apply directly, try SPA or some other PA variant (Ho and Cao 1991, L'Ecuyer 1991). If no PA-like estimator is available, FDC is likely to be the best choice, unless one can use a regenerative approach with short regenerative cycles.

As always, since our experiments were done on a specific example, one should be careful in making any generalizations. The primary goal of this example is not really to compare performance, but to illustrate convergence properties and possible dangers. We also recall that in many cases IPA and/or LR do not apply (L'Ecuyer 1990). Numerical results for other kinds of examples are given in Giroux (1989), which has been the starting point of this paper.

The performance of the derivative estimators and of the optimization algorithms could be further improved by incorporating variance reduction techniques. For instance, for our $M/M/1$ example, one can simulate at a parameter value different than the one at which the derivative must be estimated. This is *importance sampling* (Bratley, Fox, and Schrage 1987). Asmussen and Rubinstein (1991) and Rubinstein and Shapiro (1993) argue that a queueing system should in general be simulated at heavier traffic than the one at which we want the estimation. Exploring the impact of such variance reduction techniques on the performance of stochastic optimization methods is the subject of ongoing research. Another optimization approach, different from SA, is the so-called *stochastic counterpart* method, developed in Rubinstein and Shapiro (1993). Comparisons between the latter approach and SA should be reported in forthcoming papers.[1]

[1] The work of the first author was supported by NSERC-Canada grant no. OGP0110050 and FCAR-Québec grant no. EQ2831. The third author's work was supported by the IBM Corporation under SUR-SST Contract 12480042 and by the U.S. Army Research Office under Contract DAAL-03-91-G-0101. Part of the work was done while the first author was visiting the Operations Research Department at Stanford University. The authors also wish to thank Y. T. Leung, G. Perron, and R. Suri for helpful suggestions.

Appendix I

In this appendix, we look at what could happen with the combination of SA with FDC when the number t_n of customers per iteration is kept constant. We will examine the simplest case, namely $t_n = 1$ for each n, and prove that the algorithm converges to the wrong value. It will be clear from the proof that with $t_n = t$ for any larger constant t, the algorithm will also display a similar kind of biased behavior, although the bias should be expected to decrease with t.

PROPOSITION 1. *Let Assumptions A–C of L'Ecuyer and Glynn* (1994b) *hold. Suppose that* $C(\theta) + \theta$ *has its minimum at* $\theta = \tilde{\theta}^0$. *Let* $\theta^0 = \pi_\Theta(\tilde{\theta}^0)$. *Then, with* $t_n = 1$, *SA with FDC converges a.s. to* θ^0.

PROOF. When we estimate the average cost using $t_n = 1$, we actually look at the time spent in the system by *one* customer, i.e., the customer being served in that subrun. This time can be expressed as $h_1(\theta, s, \omega) = s + \theta B^{-1}(\omega)$, where s is the (waiting) time already spent in the system by that customer and ω is viewed as the $U(0, 1)$ variate used to generate its service time. We then have

$$Y_n - C'(\theta) = \frac{h_1(\theta_n^+, s_n, \omega) - h_1(\theta_n^-, s_n, \omega)}{\theta_n^+ - \theta_n^-}$$

$$= \frac{\theta_n^+ B^{-1}(\omega) - \theta_n^- B^{-1}(\omega)}{\theta_n^+ - \theta_n^-} = B^{-1}(\omega),$$

which has finite variance, from Assumption A. Also, $E_n[Y_n] = 1 + C'(\theta)$. If we redefine for the moment $w(\theta) = \theta + C(\theta)$ and apply Proposition 1 of L'Ecuyer and Glynn (1994b), the conclusion follows. □

As an illustration, take an $M/M/1$ queue with arrival rate $\lambda = 1$, mean service time $\theta \in \Theta = [\theta_{\min}, \theta_{\max}]$ for $\theta_{\max} < 1$, and $C(\theta) = 1/\theta$. Here, $C(\theta) + \theta$ has its minimum at $\tilde{\theta}^0 = 1$. Therefore, θ_n converges to $\theta_{\max}$ with probability one. The problem here is that with a different θ, the time spent in the queue by the customers already there at the beginning of the iteration would have been different and the method does not take that into account. This flaw also exists for any fixed $t_n = t$. The difference $|\theta^0 - \theta^*|$ should decrease with t. In our numerical results, for t as large as 100, the effect is still significant.

Appendix II

We verify that the $M/M/1$ example of §2 satisfies Assumptions A–C and the assumptions of Proposition 7 of L'Ecuyer and Glynn

(1994b). Using the notation of the latter paper, one has $B_\theta(\zeta) = 1 - e^{-\zeta/\theta}$, $B_\theta^{-1}(u) = -\theta \ln(1-u)$, $b_\theta(\zeta) = (1/\theta)e^{-\zeta/\theta}$, and $(\partial/\partial\theta) \ln b_\theta(\zeta) = (\zeta - \theta)/\theta^2$. For A (i), take $\tilde{B} = B_{\tilde{\theta}_{\max}}$. For A (ii), one has $Z_i = -\ln \times (1 - U_i)$ and one can take $\Gamma(u) = -\ln(1-u)$. Since the exponential distribution has finite moments of all orders, both ζ (under $\tilde{B}$) and $\Gamma(u)$ have finite moments of all orders. The Laplace transform $\int_0^\infty e^{s\zeta} e^{-\zeta} d\zeta$ of the exponential density is finite for $|s| < 1$, which gives B (i). For the exponential density, B (ii) and (iii) clearly hold. For B (iv), for any given $K > 1$ and $\theta_0 \in \bar{\Theta}$, take $\epsilon_0 = \theta_0(K-1)/(K+1)$ and $\bar{\theta} = \theta_0 + \epsilon_0$. Then,

$$\sup_{|\theta-\theta_0|<\epsilon_0} \left(\frac{b_\theta(\zeta)}{b_{\theta_0+\epsilon_0}(\zeta)} \right)$$

$$\leq \frac{\theta_0 + \epsilon_0}{\theta} \exp\left[-\zeta \left(\frac{1}{\theta} - \frac{1}{\theta_0 + \epsilon_0} \right) \right]$$

$$\leq \frac{\theta_0 + \epsilon_0}{\theta} \leq \frac{\theta_0 + \epsilon_0}{\theta_0 - \epsilon_0} = K, \quad \text{and}$$

$$E_{\theta_0+\epsilon_0}\left[\sup_{|\theta-\theta_0|<\epsilon_0} \left(\frac{\partial}{\partial\theta} \ln b_\theta(\zeta) \right)^8 \right]$$

$$= E_{\theta_0+\epsilon_0}\left[\sup_{|\theta-\theta_0|<\epsilon_0} \left(\frac{\zeta - \theta}{\theta^2} \right)^8 \right]$$

$$= \frac{E_{\theta_0+\epsilon_0}[(\theta_0 + \epsilon_0 - \zeta)^8 + (\theta_0 - \epsilon_0 - \zeta)^8]}{(\theta_0 - \epsilon_0)^{16}} < \infty,$$

since the exponential distribution has finite moments of all orders. Finally, $C(x) = C_1/x$ is convex, so Assumption C is also satisfied. The assumptions on the interarrival and service time distributions made in Proposition 7 of L'Ecuyer and Glynn (1994b) also hold. For the latter, Z_i can be expressed as $Z_i = \varphi(\theta, \zeta_i) = \zeta_i/\theta$.

References

Andradóttir, S., "A New Algorithm for Stochastic Approximation," *Proceedings of the* 1990 *Winter Simulation Conference*, IEEE Press (1990), 364–366.

——, "A Stochastic Approximation Algorithm with Bounded Iterates," Department of Industrial Engineering, University of Wisconsin, Madison, 1991a.

——, "Optimization of the Steady-State Behavior of Discrete Event Systems," Department of Industrial Engineering, University of Wisconsin, Madison, 1991b.

Asmussen, S. and R. Y. Rubinstein, "The Performance of Likelihood Ratio Estimators Using the Score Function," Technical Report, Chalmers University of Technology, Göteborg, Sweden, 1991.

Azadivar, F. and J. Talavage, "Optimization of Stochastic Simulation Models," *Mathematics and Computers in Simulation*, XXII (1980), 231–241.

Benveniste, A., M. Métivier, and P. Priouret, *Algorithmes Adaptatifs et Approximations Stochastiques*, Masson, Paris, 1987.

Bratley, P., B. L. Fox, and L. E. Schrage, *A Guide to Simulation*, Springer-Verlag, New York, Second Edition, 1987.

Chong, E. K. P. and P. J. Ramadge, "On a Stochastic Optimization Algorithm Using IPA Which Updates After Every Customer," *Proceedings of the* 28*th Allerton Conference on Communication, Control, and Computing* (1990), 658–667.

—— and ——, "Convergence of Recursive Optimization Algorithms using Infinitesimal Perturbation Analysis Estimates," *Discrete Event Dynamic Systems: Theory and Applications*, 1 (1992a), 339–372.

—— and ——, "Stochastic Optimization of Regenerative Systems Using Infinitesimal Perturbation Analysis," submitted for publication, 1992b.

—— and ——, "Optimization of Queues Using an IPA based Stochastic Algorithm with General Update Times," *SIAM J. Control and Optimization* (1993), 698–732.

Chung, K. L., "On a Stochastic Approximation Method," *Annals of Mathematical Statistics*, 25 (1954), 463–483.

Dupuis, P. and R. Simha, "On Sampling Controlled Stochastic Approximation," *IEEE Transactions on Automatic Control*, 36, 8 (1991), 915–924.

Fabian, V., "On Asymptotic Normality in Stochastic Approximation," *Annals of Mathematical Statistics*, 39 (1968), 1327–1332.

Fu, M. C., "Convergence of a Stochastic Approximation Algorithm for the *GI/G/1* Queue Using Infinitesimal Perturbation Analysis," *J. Optimization Theory and Applications*, 65, 1 (1990), 149–160.

——, "Optimization by Simulation: A Review," *Annals of Operations Research*, to appear, 1994.

Gaivoronski, A., "Optimization of Stochastic Discrete Event Dynamic Systems: A Survey of Some Recent Results," *Lecture Notes in Economics and Mathematical Systems*, 374 (1992), Proceedings of a workshop on Simulation and Optimization, Laxenburg, Austria, August 1990, G. Pflug and U. Dieter (Eds.), 24–44.

Giroux, N., "Optimisation Stochastique de Type Monte Carlo," Mémoire de maîtrise, Dépt. d'Informatique, Université Laval, January 1989.

Glasserman, P., *Gradient Estimation via Perturbation Analysis*, Kluwer Academic, 1991.

Glynn, P. W., "Stochastic Approximation for Monte-Carlo Optimization," *Proceedings of the Winter Simulation Conference* 1986, IEEE Press (1986), 356–364.

——, "Likelihood Ratio Gradient Estimation: an Overview," *Proceedings of the Winter Simulation Conference* 1987, IEEE Press (1987), 366–375.

——, "Optimization of Stochastic Systems Via Simulation," *Proceedings of the Winter Simulation Conference* 1989, IEEE Press (1989), 90–105.

Goldstein, L., "On the Choice of Step Size in the Robbins-Monro Procedure," *Statistics and Probability Letters*, 6 (1988), 299–303.

Heidelberger, P., X.-R. Cao, M. A. Zazanis, and R. Suri, "Convergence Properties of Infinitesimal Perturbation Analysis Estimates," *Management Sci.*, 34, 11 (1989), 1281–1302.

Ho, Y.-C. and X.-R. Cao, *Discrete Event Dynamic Systems and Perturbation Analysis*, Kluwer Academic, 1991.

Kesten, H., "Accelerated Stochastic Approximation," *Annals of Mathematical Statistics*, 23 (1952), 463–467.

Kushner, H. J. and D. S. Clark, *Stochastic Approximation Methods for Constrained and Unconstrained Systems*, Springer-Verlag, Applied Mathematics Sciences, Vol. 26, 1978.

—— and J. Yang, "Stochastic Approximation with the Averaging of the Iterates: Optimal Asymptotic Rate of Convergence for General Processes," *SIAM J. Control and Optimization*, 31 (1993), 1045–1062.

L'Ecuyer, P., "A Unified View of the IPA, SF, and LR Gradient Estimation Techniques," *Management Sci.*, 36, 11 (1990), 1364–1383.

——, "An Overview of Derivative Estimation," *Proceedings of the* 1991 *Winter Simulation Conference*, IEEE Press (1991), 207–217.

——, N. Giroux, and P. W. Glynn, "Stochastic Optimization by Simulation: Some Experiments with a Simple Steady-State Queue," Technical Report, Operations Research Department, Stanford University, 1989.

—— and P. W. Glynn, "A Control Variate Scheme for Likelihood Ratio Gradient Estimation," in preparation, 1994a.

—— and ——, "Stochastic Optimization by Simulation: Convergence Proofs for GI/G/1 Queues in Steady-State," to appear in *Management Sci.*, 1994b.

Major, P. and P. Revesz, "A Limit Theorem for the Robbins Monro Approximation," *Z. Wahrsch. Verw. Geb.*, 27 (1973), 79–86.

Perron, G., "Estimation de gradient et optimisation par simulation," Mémoire de maîtrise, Dépt. d'Informatique, Université Laval, January 1992.

Pflug, G. Ch., "On-line Optimization of Simulated Markovian Processes," *Math. Oper. Res.*, 15, 3 (1990), 381–395.

Polyak, B. T., "New Method of Stochastic Approximation Type," *Automation and Remote Control*, 51 (1990), 937–946.

—— and Y. Z. Tsypkin, "Optimal Pseudogradient Adaptation Algorithms," *Avtomatika i Telemekhanika*, 41, 8 (1980), 74–84. English translation in *Automatic and Remote Control*, 41 (1980), 1101–1110.

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

Suri, R. and Y. T. Leung, "Single Run Optimization of Discrete Event Simulations—An Empirical Study Using the $M/M/1$ Queue," *IIE Transactions*, 21, 1 (1989), 35–49.

—— and M. Zazanis, "Perturbation Analysis Gives Strongly Consistent Estimates for the $M/G/1$ Queue," *Management Sci.*, 34, 1 (1988), 39–64.

Venter, J. H., "An Extension of the Robbins-Monro Procedure," *Annals of Mathematical Statistics*, 38 (1967), 181–190.

Wardi, Y., "Simulation-Based Stochastic Algorithm for Optimizing $GI/G/1$ Queues," Manuscript, Department of Industrial Engineering, Ben Gurion University, 1988.

Wolff, R., *Stochastic Modeling and the Theory of Queues*, Prentice-Hall, 1989.

Yin, G., "Stochastic Approximation via Averaging: The Polyak's Approach Revisited," *Lecture Notes in Economics and Mathematical Systems*, 374 (1992), Proceedings of a workshop on Simulation and Optimization, Laxenburg, Austria, August 1990, G. Pflug and U. Dieter (Eds.), 119–134.

Accepted by James R. Wilson, former Departmental Editor; received July 13, 1992. *This paper has been with the authors* $2\frac{1}{2}$ *months for* 1 *revision.*