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

STOCHASTIC APPROXIMATION WITH SIMULATED ANNEALING AS AN APPROACH TO GLOBAL DISCRETE-EVENT SIMULATION OPTIMIZATION

Matthew H. Jones
K. Preston White, Jr.

Department of Systems and Information Engineering
University of Virginia
P.O. Box 400747
Charlottesville, VA 22904-4747, U.S.A.

ABSTRACT

This paper explores an approach to global, stochastic, simulation optimization which combines stochastic approximation (SA) with simulated annealing (SAN). SA directs a search of the response surface efficiently, using a conservative number of simulation replications to approximate the local gradient of a probabilistic loss function. SAN adds a random component to the SA search, needed to escape local optima and forestall premature termination. Using a limited set of simple test problems, we compare the performance of SA/SAN with the commercial package OptQuest. Results demonstrate that SA/SAN can outperform OptQuest when properly tuned. The practical difficulty lies in specifying an appropriate set of SA/SAN gain coefficients for a given application. Further results demonstrate that a multi-start approach greatly improves the coverage and robustness of SA/SAN, while also providing insights useful in directing iterative improvement of the gain coefficients before each new start. This preliminary study is sufficiently encouraging to invite further research on SA/SAN.

1 INTRODUCTION

A principle motivation for many simulation studies is to compare the performance and efficacy of alterative system designs (Law and Kelton 2000). When alternatives can be characterized by the values assumed by a discrete set of variables under the designers' control, and when the basis for evaluation can be quantified in terms of simulation outputs, the comparison problem can be cast as a simulation optimization. The objective is to discover an optimal set of control variables which minimizes a loss function of the simulation response.

Methods for simulation optimization have been studied extensively over the past two decades (Fu 2002), as have methods for the closely related problem of ranking and selection of limited set of discrete design alternatives (Swisher 1999; Goldsman and Nelson 2001; Nelson *et al.* 2001). For the purposes of this paper, it suffices to note that the principle technical difficulties in simulation optimization inhere in (1) the probabilistic nature of the simulation response and the consequent need to estimate the value of the loss function by efficient sampling; (2) the lack of a closed-form loss function and the consequent need to approximate gradient information using neighboring estimates of the loss-function; and (3) the potential existence of local optimal and the consequent need to forestall premature termination at a suboptimal solution.

Stochastic approximation (SA) is an adaptation of deterministic hill-climbing (steepest-decent) for optimization problems in the presence of noise (Spall 1998, 2003). Specifically, SA directs a search of the response surface using estimates of a noisy loss function to approximate the local gradient, in a way that minimizes the number of probes of the response surface. SA has many properties advantageous for simulation optimization, in which noise takes the form of estimation error in the simulation output and each probe corresponds to a replication of the simulation.

Because SA is a local gradient-search method, it lacks the ability to escape local optima. Simulated annealing (SAN) provides this ability by injecting a Monte Carlo randomness term in the SA recursion. This provides a means to escape the neighborhood of a local optimum through probabilistic acceptance of inferior intermediate solutions. While SAN was developed for deterministic problems, here we combine SAN with SA to create a joint algorithm, exploiting the strengths of each algorithm with respect to simulation optimization.

In the following section, the simulation optimization problem is stated formally. The SA, SAN, SA/SAN, and OptQuest approaches to simulation optimization are briefly outlined in Sections 3-6. Section 7 describes the two test cases studied and compares the results for several SA/SAN variants versus those for OptQuest. Conclusions and directions for future research are provided in the final section.

2 PROBLEM FORMALISM

The simulation optimization problem can be cast the following continuous search problem (Spall, 2003):

$$\min_{\theta \in \Theta} L(\boldsymbol{\theta}) = E[f(\boldsymbol{\theta}, V)] \tag{1}$$

where $\boldsymbol{\theta} = [\theta_1, \theta_2, \ldots, \theta_p]^T$ is an p-dimensional *control vector*, constrained to the vector space $\boldsymbol{\Theta}$ of feasible controls; $L(\boldsymbol{\theta})$ is a scalar *loss function*, providing an aggregate measure of system performance; and $f(\boldsymbol{\theta},V)$ is a random variable with distribution dependent on $\boldsymbol{\theta}$, denoting realizations of the loss function estimated from a simulation run with associated variability V. The sample mean from N replications of the simulation for a given value of the control $\boldsymbol{\theta}_i$:

$$\overline{L}(\boldsymbol{\theta}_i) = \frac{\sum_{n=1}^{N} f_n(\boldsymbol{\theta}_i, V)}{N} \tag{2}$$

is and unbiased estimator of the value of the loss function for this control value.

The efficiency of any approach to this optimization problem is strongly linked to the total number of *probes* (realizations of the loss function) executed during the search, since each new probe requires a single replication of the simulation. In general, probes are needed both to direct the search through a sequence of intermediary control values, as well as to improve the precision of the estimated loss function for any given control value. In contrast to traditional mathematical programming problems, in which computing the value of the loss function typically is trivial, a single probe for a large and complex simulation might take several hours or even days to evaluate. For this reason, modern global optimization systems, such as OptQuest (Lagune 1997a, Lagune 1997b, Glover e*t al.* 2000) and GROPE (Elder 1993) invest considerable effort and ingenuity in carefully managing the number of probes employed.

3 STOCHASTIC APPROXIMATION

SA derives from standard, deterministic, steepest-decent optimization methods. The basic recursion is:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - a\mathbf{g}_k(\boldsymbol{\theta}_k) \tag{3}$$

where $\boldsymbol{\theta}_k$ is the value the control vector at stage k;

$$\mathbf{g}(\boldsymbol{\theta}_k) = \partial L / \partial \boldsymbol{\theta}_k = [\partial L / \theta_1, \partial L / \theta_2, .., \partial L / \theta_p]_k^T \tag{4}$$

is the corresponding value of gradient of the loss function; and a is a (scalar) gain coefficient. The gain coefficient determines the step size to be taken at each iteration and therefore the rate and stability of convergence to a local optimum.

3.1 Robbins-Monro Recursion

In SA, at any stage the gradient of the loss function in equation (4) must be estimated, since the loss function itself is the expectation of a random variable. Obviously, this can be accomplished by averaging a sample of the gradient observed multiple times the same control $\boldsymbol{\theta}_k$, as in equation (2). The precision of the local estimate will increase probabilistically as the number observations increases.

Robbins and Monro (1951) recognized that this obvious strategy is wasteful of probes. The estimated gradient at any stage is an intermediate calculation in finding the optimum control $\boldsymbol{\theta}^*$ and a precise local estimate is not and end in itself. Rather than computing averages at each iteration, therefore, it is more efficient to use a single observation at each control and instead rely on averaging estimates locally across iterations as the search progresses. This is of particular benefit given the computational effort in evaluating the gradient using simulation.

In SA, therefore, equation (3) is replaced by the Robbins-Monro recursion:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - a_k \hat{g}_k(\boldsymbol{\theta}_k) \tag{5}$$

Two key differences are noted. First is just the recognition that the gradient must be estimated. Second is the recognition that the gain coefficient a_k must be a function of k, chosen to ensure convergence of the algorithm by properly performing across-iteration averaging.

A typical form for gain coefficient at iteration k is:

$$a_k = \frac{a}{(k+1+A)^{\alpha}} \tag{6}$$

where a, A, and α are positive constants (Spall 2003). The intended effect is to decrease the gain coefficient to zero as the search progresses. Initially, large steps speed the rate of convergence and protect against premature termination. As the search progresses, however, and the algorithm approaches an optimum, increasingly smaller steps improve the precision of across-iteration averaging. (Note that for a_k=0, "across-iteration averaging" is identically averaging over the same value of $\boldsymbol{\theta}_k$.)

3.2 Gradient Estimation

The Robbins-Monro algorithm assumes that instances of the loss-function gradient are directly measurable via equation (4). However, in many applications, including simulation optimization, the form of the gradient is unknown and must be approximated using only instances of the loss function itself. Two so-called "gradient free" approaches—Finite-Difference Stochastic Approximation (FDSA) and Simultaneous Perturbation Stochastic Approximation (SPSA)—are described in the following subsections.

3.2.1 The Finite-Difference SA

FDSA varies each of the control parameters and assesses the (discrete) change in the loss function. A one-sided measurement involves perturbing each variable by just positive or negative quantities, while a two-sided measurement involves perturbation by both positive and negative quantities. The two-sided method employs the following approximate, *p*-dimensional, estimated gradient vector:

$$\hat{\mathbf{g}}_k(\boldsymbol{\theta}_k) = \begin{bmatrix} \dfrac{L(\boldsymbol{\theta}_k + c_k\boldsymbol{\varepsilon}_1) - L(\boldsymbol{\theta}_k - c_k\boldsymbol{\varepsilon}_1)}{2c_k} \\ \vdots \\ \dfrac{L(\boldsymbol{\theta}_k + c_k\boldsymbol{\varepsilon}_p) - L(\boldsymbol{\theta}_k - c_k\boldsymbol{\varepsilon}_p)}{2c_k} \end{bmatrix} \tag{7}$$

The constant c_k is the step size, or perturbation gain coefficient, from the current values for $\boldsymbol{\theta}_k$ and follows a decaying sequence in k similar to that of the gains a_k in equation (6). Each of the set of p-dimensional unit vectors $\boldsymbol{\varepsilon}_i$, i=1,…p, has all zero elements except for a one in the i^{th} component. To calculate the two-sided gradient therefore requires $2p$ evaluations of the loss function.

3.2.2 Simultaneous Perturbation SA

SPSA randomly perturbs a subset of the p elements of $\boldsymbol{\theta}_k$, employing the following p-dimensional estimated gradient vector:

$$\hat{\mathbf{g}}_k(\boldsymbol{\theta}_k) = \begin{bmatrix} \dfrac{L(\boldsymbol{\theta}_k + c_k\boldsymbol{\Delta}_k) - L(\boldsymbol{\theta}_k - c_k\boldsymbol{\Delta}_k)}{2c_k\Delta_{k1}} \\ \vdots \\ \dfrac{L(\boldsymbol{\theta}_k + c_k\boldsymbol{\Delta}_k) - L(\boldsymbol{\theta}_k - c_k\boldsymbol{\Delta}_k)}{2c_k\Delta_{kp}} \end{bmatrix} \tag{8}$$

The constant c_k is defined as in the FDSA. In contrast to the FDSA, however, SPSA employs a single perturbation p-vector $\boldsymbol{\Delta}_k = [\Delta_{k1}, \Delta_{k2}, ..., \Delta_{kp}]^T$. Each of the elements of $\boldsymbol{\Delta}_k$ is a Bernoulli random variable assuming a value of either zero or one.

SPSA therefore reduces the number of evaluations of the loss function from $2p$ to just two, independently of the number of control variables. Although this local gradient estimate is less accurate, given the across-iteration averaging of the Robbins-Monro algorithm, the accuracy of the two procedures is nearly the same in the region of convergence. Given that the loss function is evaluated using multiple simulation replications, the savings in computation times can be substantial and SPSA is preferred.

4 THE DETERMINISTIC SAN ALGORITHM

The deterministic SAN algorithm comprises the following five steps:

Step 1. Set the initial temperature T, and control $\boldsymbol{\theta}_0 \in \boldsymbol{\Theta}$, and increment counter $k \leftarrow 0$. Evaluate the loss function $L(\boldsymbol{\theta}_0)$ at the initial control.

Step 2. Determine a new value for $\boldsymbol{\theta}_{new}$ in the neighborhood of the value $\boldsymbol{\theta}_k$ according to a predefined rule and evaluate $L(\boldsymbol{\theta}_{new})$.

Step 3. Calculate $\lambda = L(\boldsymbol{\theta}_{new}) - L(\boldsymbol{\theta}_k)$. If $\lambda < 0$, accept the new value of $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_{new}$; otherwise, generate a uniform random variable $U \sim$UNIF(0,1) and accept the new value of $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_{new}$ if only if $U \le e^{-\lambda/T}$ (the *Metropolis Criterion* (Gelfand and Mitter 1993)).

Step 4. Increment the counter $k \leftarrow k+1$. Repeat Steps 2 and 3 until the control stabilizes or until the predefined number of replications for a given temperature is exceeded.

Step 5. Test for convergence using a predefined criterion (typically based on time, probe budget, and tolerance) and stop if this criterion is satisfied. Otherwise, decrease T according to a predefined cooling schedule and return to Step 2.

The Metropolis Criterion embodies the essence of SAN. At Step 3, note that a smaller value of the ratio λ/T implies a greater probability of accepting $\boldsymbol{\theta}_{new}$ when this represents a local degradation in the solution. As one would expect, for any given value of the temperature T, this implies that smaller degradations are more likely to be accepted than larger degradations. However, this also implies that any given value of degradation λ is more likely to be accepted at higher temperatures than at lower temperatures. In analogy to a physical annealing process, the search process begins with at a relatively high T and subsequently decreases T according to a predefined *cooling schedule*. In this way, the search has a greater chance of avoiding capture at local optimum earlier in the search.

5 SA WITH SAN

The recursion for SA/SAN

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - a_k\hat{\mathbf{g}}_k(\boldsymbol{\theta}_k) + n_k b_k \mathbf{w}_k \tag{9}$$

adds a Monte Carlo random component to the Robbins-Monro equation (5). The scalar "bounce" gain coefficient b_k is similar to the gain coefficient a_k in equation (6) and decays to zero as $k \to \infty$. Each element of the p-vector $\mathbf{w}_k$ is an independent, identically-distributed random variable. These variables typically are normally distributed, however there is no convention for determining the distribution.

The value of the coefficient n_k is determined by Metropolis Criterion as in Step 3 in deterministic SAN. Specifically, the if difference $\lambda = L(\boldsymbol{\theta}_{k+1}) - L(\boldsymbol{\theta}_k) < 0$, then n_k=1

and the random perturbation in equation (9) is accepted for calculation of the gradient. Otherwise, a random variable U~UNIF(0,1) is generated. If $U \le e^{-\lambda/T}$, then n_k=1 and the random perturbation also is accepted. Otherwise, n_k=0 and the perturbation is rejected.

This procedure requires one additional probe of the response surface at stage k to reevaluate $L(\boldsymbol{\theta}_{k+1})$ when n_k=0. As the temperature decreases and the probability of accepting an inferior control decreases; however, n_k may simply be set to one for the remaining iterations, avoiding even the modest cost of evaluating the Metropolis Criterion. With n_k=1, the recursion in equation (9) is shown to converge probabilistically given the appropriate gain sequences such as those studied by Kushner (1987), Gelfand and Mitter (1993), Fang and Qian (1997), and Yin (1999).

Convergence of the algorithm to a local optimum may be assessed by testing the average per-iteration deviation in the control vector over the previous n points against some minimum threshold vector:

$$\frac{\sum_{l=k-n}^{k-1} \left| \boldsymbol{\theta}_k - \boldsymbol{\theta}_{k-1} \right|}{n} \le \boldsymbol{\tau} \qquad (10)$$

Alternately, or combination with inequality (10), convergence may be determined by testing the average per-iteration deviation in the scalar loss function.

6 A BRIEF OVERVIEW OF OPTQUEST

OptQuest a well-known commercial package for global optimization. OptQuest was chosen as a benchmark for empirical testing, in part, because of its recognized superior performance on a great range of problems, and, in part, as a convenience (OptQuest is bundled with the Arena 7.01 simulation suite). Because OptQuest is proprietary, the precise details of its operation are unknown. Several papers (Lagune, 1997a,b; Glover e*t al.* 2000) do provide the following general outline of its search strategy, however.

OptQuest employs a combination of scatter search and tabu search. Scatter search, similar to genetic algorithms, operates on a population of controls to determine the next control for evaluation. This control is generated by a linear combination of the reference controls mapped over the feasible region. Tabu search is superimposed over the process to prevent exploring regions of the response surface previously probed. Scatter search is an "information driven approach" deriving knowledge from the search space to finding high quality solutions.

A "Neural Network Accelerator" speeds the search by fitting and updating a neural-network model with each iteration. The neural network predicts the value of the loss function of a future search control. If this prediction indicates an unacceptably inferior solution, the corresponding control is not probed. The accelerator operates at several risk levels. For example, a risk adverse search would discard a potential probe if its predicted value is larger than three standard deviations above the best solution.

7 EMPIRICAL TESTS AND RESULTS

7.1 Testing a *Q, R* Inventory Simulation

The first test problem is a *Q,R*-inventory model. Daily demand is stochastic, as is the lead time required for a new order to be added to the current inventory. The objective is to minimize the expected yearly total cost $L(\boldsymbol{\theta})$, which is a probabilistic function of the continuous control $\boldsymbol{\theta}=[Q, R]^T$, where $Q \in [0,100]$ is the inventory re-order threshold and $R \in [0,100]$ is re-order quantity. Backordering is permitted. Fixed daily shortage and holding costs are assessed based on the inventory level at the beginning of the day.

The SA/SAN gain parameters are set using a initial guess and remain unchanged for all trials. OptQuest is executed with (the default of) five replications at each control. For this reason, comparing the five-point moving average of the SA/SAN search trajectory ($\hat{L}(\boldsymbol{\theta}_k)$ vs. $\log_{10} k$) against the OptQuest trajectory provides a similar comparison in terms of the variance of the estimate. This form of comparison is used in this an all subsequent test cases, as displayed in Figures 1-6. Initial conditions all the same for each SA/SAN and OptQuest trial.

It is important to note that for OptQuest, each iteration k requires five probes, while for SA/SAN each iteration requires either two or three probes. In addition, OptQuest is performing neural net and other computations at each iteration, while SAN is not. While approximate, *it is apparent the computation time and effort expended by OptQuest at each iteration is approximately double that for SA/SAN.*

Figure 1 shows the trajectory of five sample iterations of both OptQuest and SA/SAN. OptQuest finds minimum at \$785.74 per day, compared to that of \$826.66 for SA/SAN. SA/SAN initially improved the response more rapidly than OptQuest, but was unable to improve further after several hundred iterations. Four of the five OptQuest runs (denoted by the thicker line) have the identical trajectory, while the fifth OptQuest run follows the other four runs closely, but converges at a slightly slower rate, indicating the robustness of the algorithm.

7.2 Testing a Tandem Queuing Simulation

An entity arrives at the first stage and queues for service. Upon completion of service, the entity moves to a the second stage and again queues for service. Upon release of the second server, the entity is disposed. An entity waiting in either queue accrues a cost proportional to the duration of the wait. Waiting can be reduced by the addition of resource capacity for either server at a fixed cost per unit of

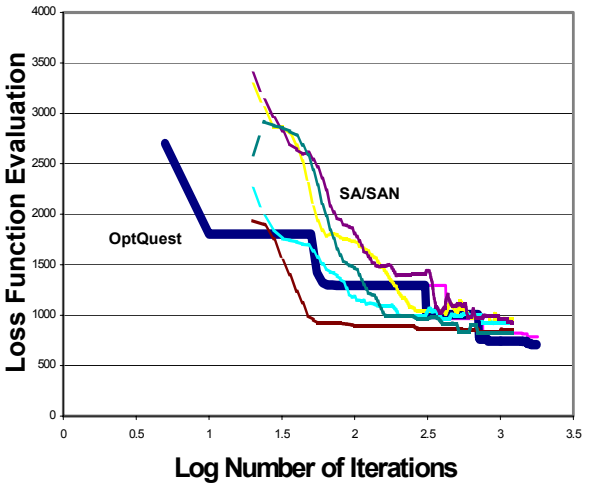

Figure 1: Comparisons for *Q-R* a Inventory Simulation

capacity. The objective is to minimize the overall cost by balancing the costs for waiting and capacity expansion.

Figure 2 summarizes the test results. SA/SAN performs poorly and does not converge after 1000 iterations. Analysis suggests that estimation error in the response does not allow for an accurate gradient estimate, despite modifications in the gain coefficients a_k and c_k..

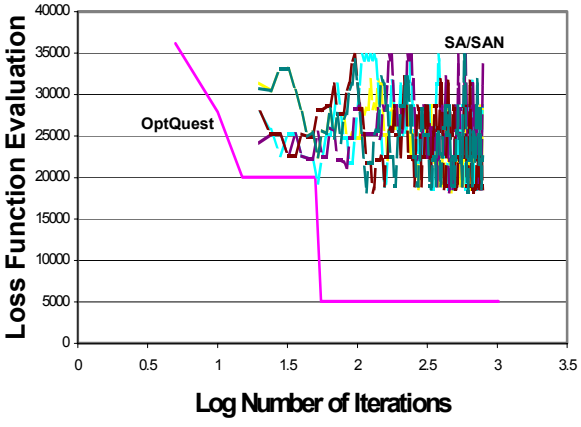

Figure 2: Comparisons for Tandem Queuing Simulation

7.3 Extensions to Original SA/SAN Algorithm

While perhaps disappointing overall, in many of the empirical tests SA/SAN did outperform OptQuest in terms of finding better solutions with far fewer probes during at least some subsequence of the search. Further empirical testing demonstrated that, as expected, SA/SAN performance is highly sensitive to the specification of the gain coefficients. Figure 3 presents results for the inventory simulation that show just how dramatic the potential performance improvement can be. By trial and error, gain values were tuned such that SA/SAN consistently outperforms OptQuest after fewer than 40 iterations using on average about half the probes at each iteration. In four of the five sample runs, SA/SAN maintains a superior solution before stalling out after several hundred iterations. In all cases SA/SAN converges to a good (if suboptimal) solution.

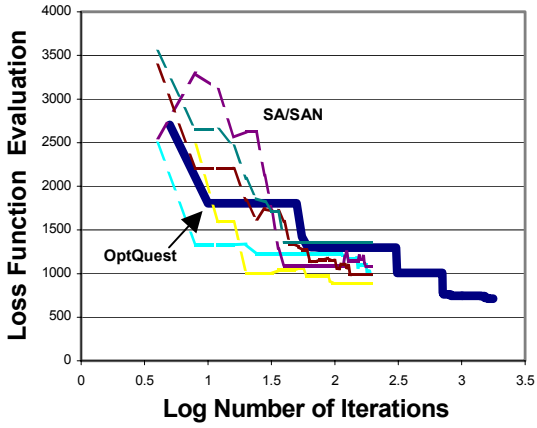

Figure 3: Comparisons After Tuning the Gain Coefficients

Clearly, OptQuest uses a significant number of initial probes to survey the search region, while SA/SAN begins improvement immediately. This is because OptQuest attempts to define the simulation response precisely, regardless of the depth of the search, using multiple probes at the same control. In contrast, SA/SAN immediately steps in the direction of improvement (or non-improvement) after the first gradient approximation. Equally important is the observation that, late in the search, OptQuest expends a very large number of additional probes to achieve only a modest improvement the solution.

The practical difficulty, of course, lies in specifying *a priori* an appropriate set of gain coefficients to direct the SA/SAN search for a given application. Further analysis of the results does provide at least some hint as to how this might eventually be achieved. At least for this particular problem, the suggestion is that SA/SAN performance benefits by selecting gains that provide rapid initial improvement, combined with quick convergence to a good (subpotimal) solution. Referring to the gain equation (6), this in fact was the result of the trial-and-error tuning. By increasing the numerator a, the initial per-iteration influence of the gradient term was increased. By also increasing α, rapid convergence was achieved.

This insight is promising and suggests the direction for future research on the optimal selection of gain coefficients. It suggests that, upon suboptimal termination, SA/SAN may be reinitialized with smaller gain coefficients to refine the precision of subsequent probes.

7.3.1 Random Multi-Start

This multi-start extension was partially explored using the inventory simulation. The SA/SAN gain coefficients were

selected *a priori* to provide an aggressive search (rapid initial improvement and early termination), but the gain values remained the same across all of the multi-starts. In the first test, the algorithm was restarted at five randomly selected initial controls.

Figure 4 summarizes the results. The individual dashed lines denote the five SA/SAN random starts; the thin solid line is the average of these results; and the thick solid line is the OptQuest search trajectory. The number associated with the dotted line indicates the order in which the multi-starts were performed. The SA/SAN results are superior to OptQuest on the second and third starts, because of the fortuitous (but entirely random) selection of initial controls. In all cases, SA/SAN yields better solutions than OptQuest after approximately 125 iterations and maintains this advantage through 725 iterations. In situations in which the cost of probes is significant, this advantage may be extremely valuable.

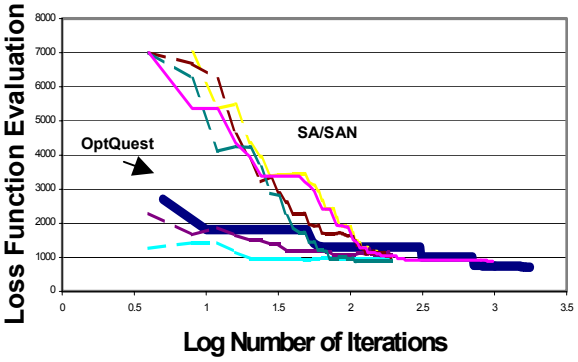

Figure 4. Comparisons Using Multiple Random Starts

7.3.2 Grid Multi-Start

In this instance, six initial controls are chosen, uniformly dispersed within the decision space. The gain coefficients are the same as those employed in the random multi-start tests. Results are shown in Figure 5.

This case yields a slightly better overall performance than the random multi-start and SA/SAN outperforms OptQuest for much of the probe budget. OptQuest has a superior solution through 35 iterations; SA/SAN wins out between 36 and 50 iterations; OptQuest regains the advantage on iterations 51 through 65; and SA/SAN finds the better solution from 66 to approximately 750 iterations.

The grid multi-start search also provides a much broader coverage of the response surface in comparison to the empirical results initially obtained. The non-restarting algorithm searches a comparatively narrow region of the search before termination. In contrast, each of the six grid multi-starts begin in widely separated regions of the response surface, quickly converging in approximately 50 iterations. This provides improved confidence in final control, given the broader coverage of the search area. Additionally, three of the six restarts converge to a similar neighborhood providing a greater degree of confidence in the optimal control parameters.

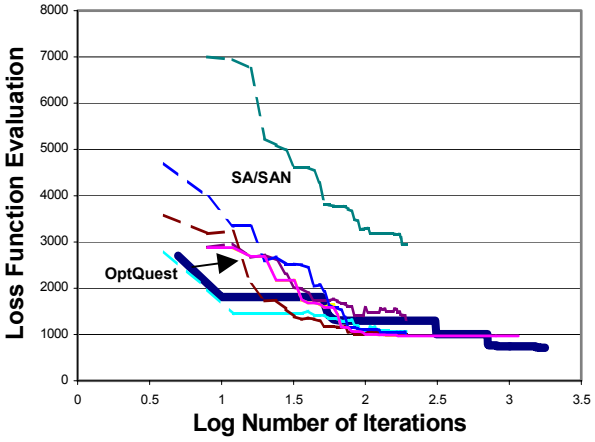

Figure 5. Comparisons Using Grid-Structure Multiple Starts

7.3.3 Random Start, Previous-Best Next-Starting Location

For this test, the first SA/SAN start is initiated at a random control. Subsequent restarts are initiated at the final control determined from the search immediately preceding. Each restart searches through 50 iterations. Results are given in Figure 6. Again, the SA/SAN summary line indicates far superior performance over OptQuest from 20 to approximately 740 iterations. Although not explored in this test, as suggested previously, performance may well be further improved by appropriately retuning the gain coefficients to improve precision with each new start.

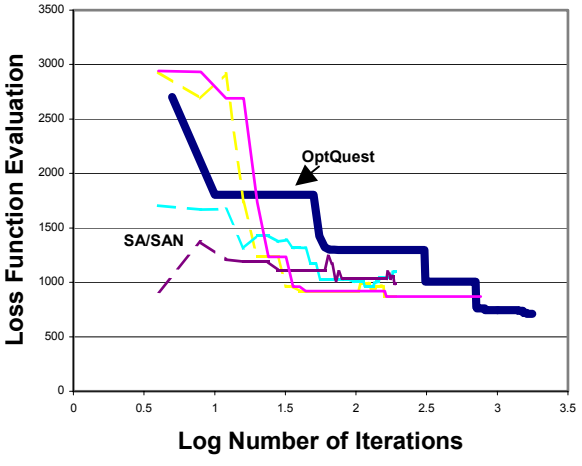

Figure 6. Comparisons Using Previous-Best Next-Starting Location

8 SUMMARY AND CONCLUSIONS

The approach adopted to compare simulated alternatives is a function of the broader systems engineering design process and depends on the specific problem at hand, the objectives of the study, and the resources available for analysis. If appropriate in this broader context, it is best to make comparisons using ranking and selection over a set of discrete design alternatives. Powerful and proven ranking-and-selection techniques are available. Given an exhaustive list of feasible possibilities *a priori*, obviously it is not possible to neglect a region of the search space (Goldsman and Nelson 2001, Swisher 1999). Further, ranking-and-selection methods provide statistical metrics as a basis for comparisons, while using a minimal number of simulation replications (Fu 2002). However, in the event that the search space is not holistically defined using a discrete set of alternatives, or the number of feasible discrete alternatives is large (Nelson *et al.* (2001) suggest thirty alternatives as a practical upper limit), simulation optimization is the preferred choice.

Among the methods available for simulation optimization, OptQuest certainly has many advantages. OptQuest is readily available and, in its commercial implementation, has only one adjustable parameter that must be determined *a priori* (the number of probes per iteration). OptQuest yields good solutions and appears to be robust.

As demonstrated in this research, however, SA/SAN can potentially outperform OptQuest and may be preferred when simulation runs are expensive and the budget for probes is limited. While the OptQuest methodology is proprietary and can not be modified by the user, SA/SAN is open source and contained within 250 lines in a C++ routine. Therefore a tradeoff clearly exists between the simplicity of OptQuest, in terms of requiring a single adjustable input parameter, and the flexible control over the algorithm and search provided by SA/SAN. Tuning the SA/SAN gain coefficients allows the user to define precisely the direction and convergence of the search, something not attainable in the commercial implementation OptQuest. Further research is needed to illuminate rules for selecting optimal SA/SAN gain coefficients, based on the structure of the application at hand and making use of information derived during the search, perhaps based on a multi-start approach.

In the multi-start approach, the gain coefficients are set to encourage rapid movement and quick convergence to a local optimum with a restricted number of surface probes. Repeatedly restarting an aggressive search appears to use surface probes more efficiently than a less aggressive, single-start approach. Furthermore, the information gained from the prior starts may provide insight into the definition of the gain coefficients for future starts, providing a method for selecting the gain coefficients. This rapid improvement approach provides a great deal of promise for continued development of SA/SAN.

Although not explored in this research, higher dimensional searches may provide a domain in which SA/SAN is preferable to OptQuest. Comparatively little public information is available regarding the scalability for OptQuest with respect to the number of control variables, although the documentation states that the performance of OptQuest quickly deteriorates beyond 100 controls. In contrast, SA/SAN should scale extremely well, requiring only two probes in the SPSA gradient approximation regardless of the number of controls. Although the bias and variance of the individual gradient estimate increases with the dimensionality of the search, SA/SAN may provide a quick and dirty approach, given a limited probe budget. SA/SAN should also be preferable to Response Surface Methodology (Fu and Hill 1997) in higher dimensional searches for the same reason.

SA/SAN exhibits the first-order Markovian property, as the $(i+1)^{st}$ control is a function of the i^{th} control alone, ignoring information collected previously in the search. Garai, Ho, and Sreenivas (1992) discuss this limitation. Another potential extension would be to augment SA/SAN with intelligent global search heuristics, such as a neural network overlay in and/or tabu search, in imitation of OptQuest.

The selection of the search algorithm is entirely a function of the system requirements and limitations. This paper suggests a significant potential for SA/SAN and invites further research into the multi-start approach.

REFERENCES

Elder, J. D. 1993. Efficient optimization through response surface modeling: a GROPE algorithm. Doctoral dissertation, Department of Systems Engineering, University of Virginia, Charlottesville, Virginia.

Fang, H. G., and M. Qian. 1997. Annealing of iterative stochastic schemes. *SIAM Journal on Applied Mathematics* 35: 1886-1907.

Fu, M. C., and S. D. Hill. 1997. Optimization of discrete event systems via simultaneous perturbation stochastic approximation. *IIE Transactions* 29: 233-243.

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

Garai, I., Y. C. Ho, and R. S. Sreenivas. 1992. Hybrid optimization-an experimental study. In *Proceedings of the 31st IEEE Conference on Decision and Control* 2: 2068-2073.

Gelfand, S., and S. K. Mitter. 1993. Metropolis-type annealing algorithms for global optimization in R^d. *SIAM Journal on Applied Mathematics* 31: 111-131.

Goldsman, D., and B. L. Nelson. 2001. Statistical selection of the best system. In *Proceedings of the 2001 Winter Simulation Conference*, ed. *B. A. Peters, J. S. Smith, D. J. Medeiros, and M. W. Rohrer,* 139-146. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Glover, F. K., and M. Laguna. 2000. The OptQuest approach to crystal ball simulation optimization [online]. Available online via <http://www.

crystalball.com/optquest/paperlist.html>

Kushner, H. J. 1987. Asymptotic global behaviour for stochastic approximation and diffusions with slowly decreasing noise effects: global minimization via Monte Carlo. *SIAM Journal on Applied Mathematics* 47: 169-185.

Laguna, M. 1997a. Optimization of complex systems with OptQuest [online]. Available online via <http://www.crystalball.com/optquest/paperlist.html>

Laguna, M. 1997b. Metaheuristic optimization with Evolver, Genocop and OptQuest [online]. Available online via <http://www.crystalball.com/optquest/paperlist.html>

Law, A. M., and W. D. Kelton. 2000. *Simulation modeling and analysis*, 3rd ed. New York: McGraw Hill.

Nelson, B. L., J. Swann, D. Goldsman, and W. Song. 2001. Simple procedures for selecting the best simulated system when the number of alternatives is large. *Operations Research* 49: 950-963.

Robbins, H., and S. Monro. 1951. A Stochastic approximation method. *Annals of Mathematical Statistics* 22: 400-407.

Spall, J. C. 1998. Implementation of the simultaneous perturbation algorithm for stochastic optimization. *IEEE Transactions on Aerospace and Electronic Systems* 34: 817-823.

Spall, J. C. 2003. *Introduction to stochastic search and optimization.* Hoboken, New Jersey: John Wiley & Sons, Inc.

Swisher, J. R. J., and S. H. Jacobson. 1999. A survey of ranking, selection, and multiple comparison procedures for discrete-event simulation. In *Proceedings of the 1999 Winter Simulation Conference*, ed. P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, 492-501. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Yin, G. 1999. Rates of convergence for a class of global stochastic optimization algorithms. *SIAM Journal on Applied Mathematics* 10: 99-120.

AUTHOR BIOGRAPHIES

MATTHEW H. JONES is employed by The Aerospace Corporation in Chantilly, VA, in the Systems Architecture and Engineering Department. He received his B.S and M.S. from the University of Virginia in Systems and Information Engineering in 2003, where he is currently a Ph.D. candidate. He is a member of INFORMS and INCOSE.

K. PRESTON WHITE, JR., is Professor of Systems and Information Engineering (SIE) at the University of Virginia. He received the B.S.E., M.S., and Ph.D. degrees from Duke University. He has held faculty appointments at Polytechnic University and Carnegie-Mellon University and served as Distinguished Visiting Professor at Newport News Shipbuilding and at SEMATECH. He is a member of INFORMS and INCOSE and a senior member of IEEE and IIE. He sits on the Board of VMASC and is listed in *Who's Who in America*. Mr. White represents IEEE/SMC on the WSC Board and current serves as Board Chairman. His e-mail address is <kpwhite@virginia.edu>.