Proceedings of the 2003 Winter Simulation Conference
S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, eds

TARGETING AVIATION DELAY THROUGH SIMULATION OPTIMIZATION

Frederick Wieland
Thomas Curtis Holden

Center for Advanced Aviation Systems Development
The MITRE Corporation
7515 Colshire Drive
McLean, VA 22102, U.S.A.

ABSTRACT

Analyses of benefits due to changes in the National Airspace System (NAS) tend to focus on the delay reduction (or similar metric) given a fixed traffic schedule. In this paper, we explore the use of simulation optimization to solve for the increased traffic volume that the proposed change can support given a constant delay. The increased traffic volume as a result of the change can therefore be considered another benefit metric. As the NAS is a highly nonlinear stochastic system, the technique required to compute the increase traffic volume necessarily requires stochastic optimization methods.

1 INTRODUCTION

Simulation is an often-used technique to evaluate the benefits of planned improvements to the NAS. These improvements include avionics upgrades, new arrival and departure procedures at airports, precision navigation using satellite technology, and many other technologies. Improvements to the NAS, and the associated benefit assessment through simulation, is especially important given the future growth rates predicted for air transportation (Gentry 2003).

In evaluating the benefits of planned improvements, the usual benefit metrics are expressed in terms of delay or throughput, or simple functions that combine the two. It is quite common for a simulation scenario to be configured with a "before" and "after" case, and the change in delay, efficiency, and other metrics is computed. Those technologies and procedures showing a measurable reduction in delay or increase in efficiency, while maintaining or increasing safety, are candidates for implementation.

In the canonical study of NAS improvements, the traffic level is either held constant or grown to a volume that represents expected future air traffic growth, and then the before/after metrics are computed. In this paper, we introduce a technique that allows the value of the metrics to be held constant while computing the resulting traffic volume. In other words, this technique determines the volume of traffic with the new technology that produces the same delay (or throughput or efficiency) that is observed with current technology. It answers the question, how much traffic growth can the proposed operational change handle?

Determining the traffic growth that produces the same level of delay provides analysts an additional metric into the benefits of the proposed change. In this paper, we will concentrate on the use of two different simulation optimization techniques (simulated annealing and simultaneous perturbation stochastic approximation) to answer this question, and illustrate their use on a generic sample problem.

2 PROBLEM FORMULATION

For clarity, this paper will use *delay* as the fundamental metric for air traffic analysis. In most benefits computations, traffic volume is *input* to the simulation while the delay metric is *output* from it. When computing the traffic volume that can be supported with a given level of delay, we are reversing these two variables: delay is *input* to the simulation, while traffic volume is the *output.*

The problem is made more difficult because, for a given level of delay, the traffic schedule that can produce that delay is non-unique. This fact arises in part because air traffic scheduling is closely tied to peak hours of activity during the day. For example, system delay is virtually unchanged if more traffic is added to airports at 2:00 AM local time. However, the same amount of traffic added to a major airport at 2:00 PM can produce a measurable increase in system delay. Consequently, the increase in traffic volume has to be handled carefully, so as to reduce the spurious effects of scheduling policies on delay.

To evaluate the traffic volume's effect on system delay, we are using a new system-wide model developed by The MITRE Corporation's Center for Advanced Aviation Systems Development (CAASD). The model has a level of resolution mid-way between traditional system-wide models of the NAS (such as DPAT, NASPAC, LMINET) and more detailed NAS emulations (such as TAAM). Although developed as a sequential simulation, the model is built using the latest simulation technology on the latest hardware. Its performance is reasonable: the simulation executes the 72,000+ flights in this scenario in a few minutes.

In this study, we configured the model to handle all domestic traffic within the continental United States, as well as all international arrivals and departures. The top 31 airports are constrained to limit the number of arrivals and departures they can handle per hour, based upon runway geometry, expected weather, aircraft fleet mix, and other factors. We call these 31 airports the *modeled airports.* All other airports (those we call the *unmodeled airports)* are unconstrained with regard to their capacity.

Among many other metrics, the model computes the arrival and departure delays for a given flight schedule. The arrival and departure delays are the model-computed length of time flights wait to use a runway. The delay targeting algorithm uses the *average system delay* metric. The average system delay is the mean of the arrival and departure delays for all flights using one of the 31 modeled airports. For simplicity, we will use the term "system delay" to refer to this metric.

As flights are added to the schedule, the system delay generally increases, while removing flights from the schedule generally reduces the system delay. In this study, we developed a method using simulation optimization to increase the system delay of a particular flight schedule by increasing the number of flights in that schedule. We call the target system delay value ρ_t and the baseline system delay value ρ_b. Because we want to keep the target system delay within bounds, we require that the maximum departure or arrival delay at any of the individual airports be less than the parameter ρ_p.

3 THE SIMULATION OPTIMIZATION APPROACH

Simulation optimization is the use of search methods to find input parameter settings that improve selected output measures of a simulated system (Boesel 2001). The motivation for doing simulation optimization is to support analytical studies that use simulation to study real world systems. Applications of this technique include transportation systems, manufacturing systems, supply chains, call centers and finance (Fu 2001).

Most simulation optimization approaches include the following components: an objective function, a set of constraints, an optimization algorithm, and a simulation engine. The optimization algorithm attempts to find a minimum or maximum value for the objective function. The objective function is a wrapper for the simulation that translates parameters from the optimization algorithm to a configuration object that the simulation uses. The objective function also gathers values from the simulation output to generate a single result. The constraints define valid solutions based on the objective function input parameters and/or results.

3.1 Objective Function

The simulation optimization strategy is expressed as a constrained stochastic optimization problem. This problem has the following form:

$$\min_{\theta \in G} L(\theta), \tag{1}$$

where $L(\theta)$ is the objective function of interest, $\theta \in \Re^d$ is the parameter vector and $G \subset \Re^d$ is the constraint set on θ. In order to effectively execute the optimization, $L(\theta)$ needs to exhibit a global minimum for flight schedules that produce an system delay value that equals ρ_t. A simple relation that exhibits this behavior is:

$$\left|\rho_t - \rho_s\right|, \tag{2}$$

where ρ_s is the system delay produced by the simulation. The objective function also needs to be parameterized in some way on the simulation input. Since we want to find a flight schedule that produces a minimum of relation (2), we map the objective function parameters, θ, to the flight schedule. The mapping takes each element of θ to represent the number of arrival or departure flights to add at a particular airport over the baseline schedule. The total number of elements in θ is:

$$n_\theta = 2n_A, \tag{3}$$

where n_A is the number of capacitated airports and the factor of 2 accounts for the fact that we treat the number of arrival and departure flights separately. Based on the use of 31 modeled airports, $n_\theta = 62$. Each flight is a discrete unit; therefore, the range of each parameter element is the set of non-negative integers.

Each new departure at one of the 31 modeled airports is created by cloning a randomly chosen departure flight from the baseline flight schedule and perturbing the departure time of the new flight by a random period of time. The arrival airport is set to one of the unmodeled airports. A similar procedure is used for arrivals. This approach ensures that the original and resulting flight schedule will be structured similarly. Flights during times of peak air traffic are more likely to be cloned than those during times of very low air traffic, and hence the resulting schedule will have "peaks" and "valleys" similar to the original schedule. Producing a similar schedule for a given level of system delay is an important requirement for these types of studies.

Since each realization of a particular traffic volume θ is created by a random process, a particular instance of θ will map to many different schedules. For example, if the traffic volume θ represents an increase of five flights at airport n_i, different realizations of the random selection of which five flights to clone will result in different traffic schedules. Each of these schedules will likely map to a different value for ρ_s. This characteristic is important since randomness in the flight creation process will create noise in the objective function measurements. The noisy objective function can be expressed as:

$$y(\theta) = \left|\rho_t - \rho_s(\theta)\right| \tag{4}$$

where $\rho_s(\theta)$ is the system delay produced by the simulation for the associated values of θ. The distribution of the noise and its dependency on θ is unknown.

Air traffic models tend to be non-linear with respect to changes in the flight schedule. Because of this, the objective function will probably have many local optima (i.e. the function is probably non-convex). Like the noise, the detailed mathematical nature of the objective function is unknown.

3.2 Constraints

The objective function requires that each element of θ to be a non-negative integer. This constraint is satisfied by a simple projection operator Ψ_θ. Solutions that satisfy the optimization criteria by producing a large amount of system delay at a small number of the capacitated airports while producing very little system delay at the rest of the capacitated airports are unacceptable. A solution should produce system delay at each airport that is bound within a certain reasonable range. To address this issue we add a penalty constraint, $P(\theta)$, to the objective function that penalizes solutions which result in system delay values for individual airports that are greater than a threshold value ρ_p.

3.3 Optimization Algorithm

No single optimization algorithm has a general theoretical framework that addresses all of the characteristics of our particular objective function $y(\theta)$. These characteristics are: (probably) non-convex, measurement noise, discrete and the use of a penalty function. Since we will have to make some theoretical compromises with any algorithm, we tested two different algorithms on the optimization: fixed gain SPSA (simultaneous perturbation stochastic approximation) and τ threshold SAN (simulated annealing). We choose the fixed gain version of SPSA (Gerencsér 1999) because the values of θ for this problem are defined on the set of non-negative integers, and fixed gain SPSA is designed for optimization on discrete sets. Also, this is a fairly high dimensionality problem, $n_\theta = 62$, and the SPSA estimate of the gradient is efficient for high dimensionality θ relative to the standard finite difference procedure. We choose τ threshold SAN since the τ threshold is designed to deal with the effects of noise. Also, this is a discrete problem and SAN was developed for discrete problems (Kirkpatrick and Gellatt 1981).

SPSA is a stochastic approximation algorithm that uses an estimate of the gradient based on a random perturbation of two measurements per iteration. The fixed gain version of SPSA assumes that $L(\theta)$, where θ is defined on a discrete set, can be extended to a real valued version of $L(\theta)$. The gradient estimate is defined for this real value extension. SAN is a localized random search algorithm that accepts inferior solutions with a non-zero probability. SAN has a well-established global optimization theory, but can be slow to converge. Table 1 describes the suitability of each of the algorithms to address each of the characteristics of $y(\theta)$ in terms of the available theory.

Table 1: Suitability of Algorithms to Address Various Mathematical Characteristics of Objective Functions

Objective Function Features	**Algorithm**	
	τ-SAN	**Fixed-Gain SPSA**
Non-convex	General theory	Theory for real
Discrete	General theory	Theory for convex
Noisy	Limited theory	General theory
Penalty function	No theory	Theory for real-convex

From Table 1 it is clear that both algorithms fall short of addressing all the characteristics of $y(\theta)$. Based on a simple ranking that ignores the unknowns associated with the theory for each algorithm, it appears that τ threshold SAN is better suited to handle $y(\theta)$ than fixed gain SPSA.

In addition to the difficult objective function characteristics, the objective function is relatively time consuming to evaluate because it requires a simulation run. Each run of the simulation used to generate the results requires about 80 seconds, though this can be much longer for simulation configurations that include a large number of flights (on the order of many minutes). This runtime limits the number of iterations that can be performed in a particular time period.

The form of the penalty function used in the optimization can have an impact on the performance of the optimization. There is published theory on the use of penalty functions with SPSA on problems where the objective function is convex and the parameters are real valued (Wang and Spall 1999). Since $y(\theta)$ is discrete and is probably non-convex, this theory does not directly apply. Despite the lack of a strong theoretical foundation for the discrete case we will use the method developed in Wang and Spall (1999).

The penalty function we use for the SPSA optimization has the following form:

$$P(\theta) = \sum_{i=0}^{n_\theta} \frac{1}{\beta} \max\left(q(\theta), 0\right)^{\beta}, \tag{5}$$

where $q(\theta)$ is defined as:

$$q(\theta) = \frac{n_i\left(\rho_i^{(s)}(\theta) - \rho_p\right)}{n}. \tag{6}$$

The function $\rho_i^{(s)}(\theta)$ is the delay of all the flights arriving or departing from a particular airport (s is not a power but a superscript to indicate that this is simulation output), n_i is the total number of arriving or departing flights at the particular airport and n is the total number of flights counted for delay calculations. The value β is taken to be one for all the runs in this work. For the SPSA runs, the penalty constraint is scaled by an increasing sequence, r_k, of the optimization iteration index k. This increasing sequence is suggested by the theory presented in Wang and Spall (1999). The modified objective function with the penalty function has the following form:

$$y(\theta) = \left|\rho_t - \rho_s(\theta)\right| + r_k P(\theta). \tag{7}$$

We use the same penalty function given in (7) for the SAN optimization, but we do not scale the penalty function by the r_k sequence. We chose not to include the r_k scaling since SAN theory does not address the case where the objective function varies by iteration. By removing the r_k scaling factor the penalty function becomes part of the objective function. This addresses the problem related to a lack of theory regarding the use of penalty functions with SAN (see Table 1).

Since $P(\theta)$ penalizes the addition of delay to the system. while the core objective function relation, $\left|\rho_t - \rho_s(\theta)\right|$, can reward the addition of delay, there is a trade off between the effects of these factors on the objective function value. The challenge for the optimization algorithm is to find a value for θ that minimizes $\left|\rho_t - \rho_s(\theta)\right|$ yet keeps $P(\theta) = 0$. Since all the components of $y(\theta)$ will always be positive or zero, the theoretical minimum for $E\left[y(\theta)\right]$ is zero.

4 EXPERIMENT AND RESULTS

The benchmark scenario has an system delay of 1.37 minutes. The target system delay parameter was chosen to be $\rho_t = 3$ while the penalty function threshold was chosen to be $\rho_p = 7.5$. The goal of the optimization is to find the traffic volume that achieves an system delay of 3 minutes with a maximum departure and arrival delay of 7.5 minutes at the 31 modeled airports.

4.1 Measuring Objective Function Noise

To quantify the noise in the objective function, we computed $y(\theta)$ five times using a trivial solution. All trivial solutions take the form $\theta = [c, \ldots, c]$ where c is a non-negative integer value. The results of these runs are shown in Table 2. $X(\theta)$ is the sample mean of $y(\theta)$ and $s(\theta)$ is the sample standard deviation of $y(\theta)$. The measured noise level for the solutions chosen here varies quite a bit, but is less than 1% of the mean in all cases.

Table 2: Results of the Six Trivial Solution Runs

Values of θ	$X(\theta)$	$s(\theta)$	$s(\theta)/X(\theta)$
5	1.606374	0.006091	0.003791
10	1.591614	0.003598	0.002261
20	1.542896	0.006754	0.004378
40	1.461600	0.009288	0.006355
80	1.222824	0.005143	0.004206

Given the results in Table 2, we used the starting solution $\theta_0 = [80, \ldots, 80]$ for all runs. It is the trivial test solution with the lowest sample mean.

4.2 SPSA Results

One of the challenges with the SPSA runs was dealing with the constant step gain coefficient, a. The magnitude of the elements of many of the gradient estimates generated using the system delay target objective function are less than one. The fixed gain SPSA procedure described in Gerencsér (1999) uses the following fixed gain gradient estimate:

$\hat{g}_k^z(\hat{\theta}_k) = \lfloor \hat{g}_k(\hat{\theta}_k) \rfloor$ where $\hat{g}_k(\hat{\theta}_k)$ is the basic SPSA gradient estimate and $\lfloor \mathbf{x} \rfloor = (\lfloor x_1 \rfloor, \ldots, \lfloor x_p \rfloor)$ and $\lfloor x_i \rfloor$ indicates the integer less than or equal to x_i and closest to x_i. Based on this procedure, the product $a(\hat{g}_k^z(\hat{\theta}_k))$ would almost always be zero or a negative number regardless of the value of a since the elements of $\hat{g}_k^z(\hat{\theta}_k)$ will almost always be -1 or 0. To avoid this limitation, the objective function $y(\theta)$ was scaled by a factor a' before the floor operation was applied and the step gain was set to $a = 1$. Another open issue was the choice for the scaling sequence, r_k. Since this should be a very slowly growing sequence, we tried the following relations: $r_k = k^{1/3}$ and $r_k = \frac{1}{2}\ln(k^{1/2})$. The latter sequence was used for the work done in Hutchison and Hill (2001).

Four different SPSA runs were computed overnight (approximately 12 hours), though no run managed to achieve more than 200 objective function evaluations. All of the runs diverged (i.e., the objective function value grew rapidly) after some number of well behaved iterations. A plot of the objective function value as a function of iteration for run 2 is shown in Figure 1. Notice that the objective function value is reduced until iteration 45 and then starts to grow rapidly. These high objective function values correspond to solutions that include a very large number of additional flights. The large number of flights made the simulation runtime very long which made the average runtime/iteration for the SPSA runs quite long (on the order of many minutes).

Table 3: Summary of SPSA Runs

	a'	c	r_k	Obj. Fcn. Value
Run 1	500	1	$k^{1/3}$	Diverged
Run 2	1000	1	$k^{1/3}$	Diverged
Run 3	500	1	$\frac{1}{2}\ln(k^{1/2})$	Diverged
Run 4	1000	1	$\frac{1}{2}\ln(k^{1/2})$	Diverged

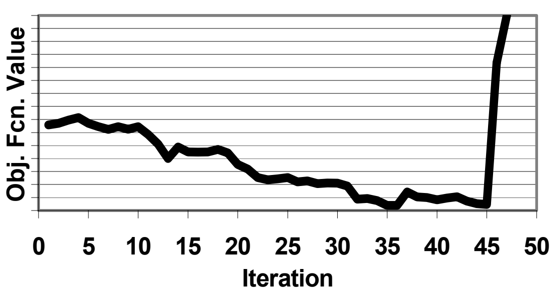

Figure 1: The Objective Function Value as a Function of Iteration for SPSA Run Number 2.

All the divergent solutions violated the penalty constraint. One problem may be that the step gain, a', is too large. If this is the problem it is not straight forward to address because with $a' = 500$ the run would often make no progress because the gradient estimate was often rounded to zero by the floor operation. Another problem may be that the scaling sequence, r_k, is too aggressive. A more fundamental problem may be that the objective function violates one or more of the conditions required for convergence of SPSA. The most likely problem is that there is a fundamental theoretical limitation. Recall from section 3.3 that there is no published theory supporting the use of penalty functions with fixed gain SPSA on non-convex objective functions. Wang and Spall (1999) noted that the estimate of the penalty function needs to be unbiased in order to guarantee convergence . In this work the gradient of the penalty function is estimated using the simultaneous perturbation technique which certainly has bias.

4.3 SAN Results

We applied a version of SAN that uses the Gaussian distribution to generate candidate solutions at each iteration. The use of the Gaussian distribution requires a temperature schedule that decreases no faster than $T(k) = T_0/\log(k)$, where T_0 is the initial temperature (Spall 2003). Each run was allowed 200 objective function evaluations (which is 200 iterations). All of the runs were able to reduce the objective function value by more than 50%, but none were able to satisfy the penalty constraint within the 200 iterations. The results for the Gaussian SAN runs are summarized in Table 4. The parameter T_{g0} is the initial generation temperature while T_{a0} is the initial acceptance temperature. The variance of the Gaussian generation distribution is determined by the generation temperature, while the Boltzmann-Gibbs state probability exponential is scaled by the inverse of the acceptance temperature. We set the τ value to 0.003 for all the runs. The choice is based on the guidelines provided in Spall (2003). We assumed the system delay target objective function contains many local minima so we chose a positive value for τ. We chose a magnitude of 0.003 because that value is on par with the measured standard deviation of the noise based on the data listed in Table 2 (section 4.1). The results of three SAN runs are shown in Table 4 below.

Because the results for the third run shown in Table 4 were the most promising, we did two more runs with the “Settings 3” parameters from Table 4. The average progress of the objective value of the current solution is plotted in Figure 2. The sample mean of the objective function

Table 4: Summary of SAN Runs

	T_{g0}	T_{a0}	Variant	Obj. fcn. value
Settings 1	100	0.1	Gaussian	0.545
Settings 2	225	0.1	Gaussian	0.399
Settings 3	900	0.1	Gaussian	0.042

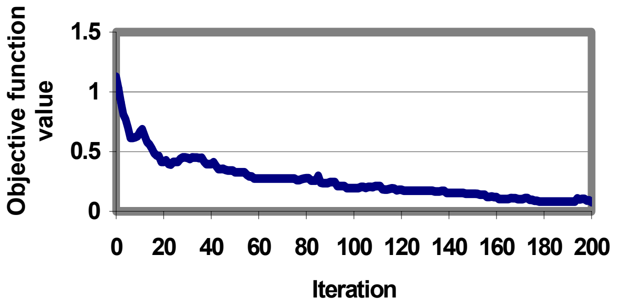

Figure 2: The Mean Objective Function Value of the SAN Solution for Three Runs with the "Settings 3" Parameters from Table 4.

of the current solution at iteration 200 is 0.0776 and the 95% confidence interval for the final objective function value is $[0.0254, 0.1299]$.

To get an idea of how well the penalty constraint was satisfied, we gathered data on the maximum departure and arrival delay at all the capacitated airports. This sample mean of the iteration 200 current solution maximum delay at any airport is 9.900 and the 95% confidence interval for this value is $[7.864, 11.94]$. The goal of the penalty constraint was to keep the maximum departure and arrival delay below 7.5. Clearly the sample mean of the maximum delay result does not satisfy this constraint, and both bounds of the 95% confidence interval are higher than 7.5.

The second run using the "Settings 3" parameters was allowed to run out to 500 iterations. Figure 3 is a plot of the objective function value of the current solution, the candidate solution and the candidate maximum arrival or departure delay at any airport for this run. It is clear from the plot that the maximum airport delay is making progress towards a solution that satisfies the penalty constraint. The penalty constraint is first satisfied by a current solution at iteration 330. The objective function value of the current solution at iteration 330 is 0.017.

4.4 Overall Results

Because the SPSA runs did not converge, we concentrate only on the SAN results here. For each of the three runs using the settings in Table 4, the number of (arrivals, departures) added to the schedule was: (4597, 4451), (5812, 6047), (5539, 5738). These volumes were distributed over the 31 modeled airports, although the details of that distribution are not shown here (for space reason).

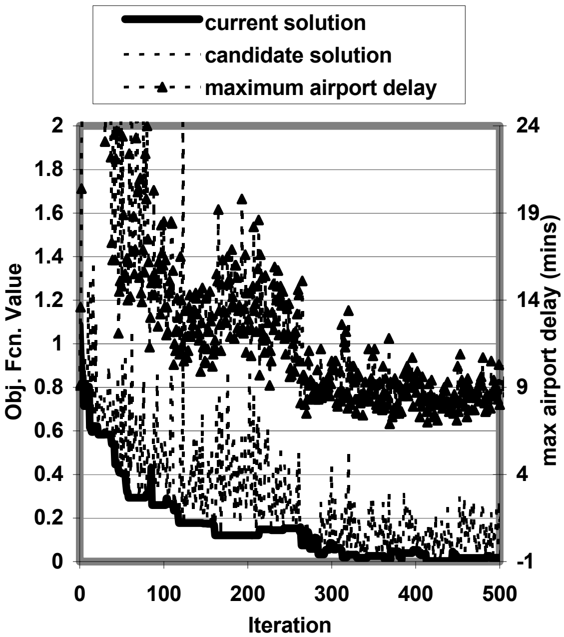

Figure 3: Plot of Details for Run 2 using the "Settings 3" Parameters from Table 4.

These three solutions differ in quality, as shown by the objective function values recorded in Table 4. If these results were the result of an actual analysis of a NAS change, we might conclude that the change could support about 4,000 to 6,000 additional flights, producing approximately the same system delay as before the change. However, as the simulation optimization technology shown here is still under investigation, we have not yet used it for an actual NAS analysis.

5 CONCLUSIONS

The results for the τ threshold SAN approach show promise while the constant gain SPSA method requires more investigation. Based on the list of potential problems for constant gain SPSA in Table 1, this is not surprising. SAN came to look even more promising than Table 1 suggests with the establishment (based on experiment) of the very low noise level and the merging of the penalty function into the objective function proper. More runs of SAN will be required to gain confidence in the procedure.

The procedure is useful for an optimization savvy analyst who can save more time by tuning the objective function parameters than by solving the problem using alternate approaches. However, this procedure is unlikely to become a turn-key solution for NAS analysis because each problem is unique and requires hand-tuning of the various parameters in the objective function. Despite the need to tune parameters, we expect that this procedure (or similar procedures) will be useful for future benefit metrics computation.

ACKNOWLEDGMENTS

We gratefully acknowledge the support of The MITRE Corporation's Center for Advanced Aviation Systems Development for this research.

The contents of this material reflect the views of the authors. Neither the Federal Aviation Administration nor the Department of Transportation makes any warranty or guarantee, or promises, expressed or implied, concerning the content or accuracy of the views expressed herein.

REFERENCES

Boesel, J., R.O. Bowden Jr., F. Glover, J.P. Kelly, and E. Westwig. 2001. Future of Simulation Optimization. In Proceeding of the 2001 Winter Simulation Conference, J.A. Joines, R.R. Barton, K. Kang, and P.A. Fishwick, eds., pp. 53-61. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Fu, M.C., 2001. Simulation Optimization. In *Proceeding of the 2001 Winter Simulation Conference,* ed. J.A. Joines, R.R. Barton, K. Kang, and P.A. Fishwick, 53-61. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Gentry, J., D. Bhadra, M. Wells, B. Hogan and F. Moreno-Hines. 2003. Building the Timetable from Bottom-Up Demand: A Micro-Econometric Approach. In *Proceedings of the 2003 Summer Computer Simulation Conference*, Montreal Canada: Society for Computer Simulation.

Gerencsér, L., S. D. Hill and Z. Vágó. 1999. Optimization over discrete sets via SPSA. In *Proceedings of the 38th Conference on Decision & Control*, 1791-1795. Phoenix, AZ.

Hutchison, D. W., and S. D. Hill. 2001. Simulation optimization of airline delay with constraints. In *Proceedings of the 2001 Winter Simulation Conference*, 1017-1022. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Kirkpatrick, S., C.D. Gellatt and M.P. Vwecchi. 1981. Optimization by simulated annealing, *Science* 220:671-680.

Spall, J. C. 2003. *Introduction to Stochastic Search and Optimization*. Hoboken, New Jersey: Wiley-Interscience.

Szu, H. and R. Hartley. 1987. Fast Simulated Annealing. *Physics Letters A* 122:157.

Wang, I.-J. and J. C. Spall. 1999. A constrained simultaneous perturbation stochastic approximation algorithm based on penalty functions. *Proceedings of the American Control Conference*, 393-399. San Diego, CA.

AUTHOR BIOGRAPHIES

FREDERICK WIELAND is a Project Team Manager at The MITRE Corporation's Center for Advanced Aviation Systems Development (CAASD), and holds a PhD in information technology/applied probability theory from George Mason University. He is the developer of numerous simulations for the U.S. Department of Defense (DoD) as well as the Federal Aviation Administration, including the system-wide model used in this study, as well as CTLS (DoD), DPAT (FAA), and others, and has done extensive research in the parallelization of large-scale simulations such as TAAM. He has been working in the simulation field for 20 years. He can be contacted by e-mail at <fwieland@mitre.org>.

THOMAS CURTIS HOLDEN is a Senior Simulation and Modeling Engineer at The MITRE Corporation's Center for Advanced Aviation Systems Development (CAASD). He can be contacted by e-mail at <tholden@mitre.org>.