Bayesian Simulation Optimization

Using Augmented Probability Simulation

Jason R. W. Merrick

Department of Statistical Sciences and Operations Research

Virginia Commonwealth University

Abstract

Under the subject expected utility paradigm, decisions are made by finding the alternative with the maximum expected utility. In Bayesian simulation, the probability distribution used is the distribution of a simulation output. While methods have been developed under the Bayesian paradigm for choosing the best simulated system from a discrete, finite set of alternatives, the only methods for optimization with Bayesian simulations are borrowed from classical simulation and deterministic optimization. This paper presents an algorithm for finding the decision with the maximum expected utility, based on a sampling technique, called augmented probability sampling. The algorithm is particularly appealing as both the decision and the simulation outputs are handled using sampling techniques derived from the Metropolis sampler, which is commonly used in Bayesian estimation.

(Simulation optimization; Bayesian statistics; Markov chain Monte Carlo)

1. Introduction

From a managerial perspective, the main use of simulation is to aid in making decisions. In recent years, considerable work has been performed in selecting the best simulated system and simulation (or stochastic) optimization. Figure 1 shows a simple decision problem as an influence (relevance) diagram.

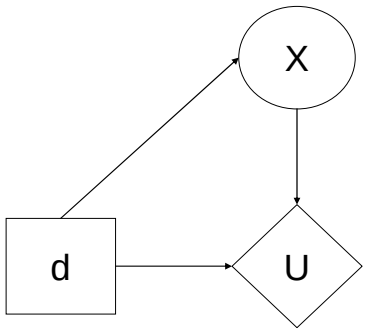

Figure 1. A simple decision problem.

To solve this problem, we define the conditional distribution of the uncertain variable given the decision d, denoted $p_d(X)$ where X is the random variable. We then use the decision maker's utility function $U(d, X)$ and the decision problem is expressed as maximization of expected utility. Formally, we wish to choose d^* from a set of alternatives Δ that maximizes,

$$\max_{d \in \Delta} \left[V(d) = \int u(d, X)\, p_d(X) dX \right]. \qquad (1)$$

In a decision made using simulation, $p_d(X)$ is the distribution under decision d of some simulation output, X, that influences the decision maker's utility. In a general decision analysis, $p_d(X)$ is a known distribution. However, in the case of simulation decision problems $p_d(X)$ cannot be pointwise calculated, but can be sampled from. Thus, $V(d)$ cannot be calculated in closed form.

If the set of possible decisions is finite, then we may solve this problem in a brute force manner by taking a large number of samples from X for a given $d \in \Delta$ and estimating $V(d)$ by the average value of $U(d, X)$ over the sample. We then choose d^* as the $d \in \Delta$ that maximizes the sample estimate

of $V(d)$. Techniques for choosing the best simulated system provide rules or heuristics for assigning the computational effort to different values of $d \in \Delta$, thus allowing the best decision to be found more quickly. Statistical procedures then assess whether there is sufficient evidence in favor of any one alternative (Matejcik and Nelson 1995; Bechhofer et al. 1995; Goldsman and Nelson 1998; Chick and Inoue 2001a 2001b).

If the set of possible decisions is not finite, whether countably or uncountably, then the problem becomes one of optimization, or searching the decision space. Most simulation software manufacturers rely on meta-heuristic approaches drawn from deterministic optimization (April et al. 2003). However, there has been significant research into purely stochastic methods.

To solve this problem, we adapt the work of Bielza et al. (1999) who offer a Markov Chain Monte Carlo (MCMC) (Gelfand and Smith 1990) approach to solving general influence diagrams like Figure 1, called augmented probability sampling. Their approach is both simple and elegant when applied to simulation optimization. Bielza et al. offer the augmented distribution defined by

$$h(d, X) \propto u(d, X)\, p_d(X) \tag{2}$$

as an artificial distribution if both the utility function and the distribution are positive and integrable. The key here is that the marginal distribution $h(d) = \int h(d, X)dX$ is in fact proportional to $V(d)$. Thus if we can sample $(d^1, X^1),\ldots,(d^n, X^n)$ from $h(d, X)$, we can convert this to a sample from the marginal distribution $h(d)$ by discarding the values for X and taking $d^1,\ldots,d^n$. We may then find the mode of $h(d)$, denoted d^*, which will in fact be the value of d that maximizes $V(d)$. The construction of this algorithm allows us to appeal to proofs of convergence for the Metropolis algorithm (Roberts and Smith 1994; Tierney 1994), thus our algorithm falls in category (1) above, with guaranteed asymptotic convergence to the optimum.

The original algorithm is obviously due to Bielza et al., but one specific form is well suited to use with simulation decisions. While the algorithm developed has guaranteed asymptotic convergence, we

would also like the mode of samples from the augmented probability distribution of the decision to converge quickly to the optimal decision. Our initial tests of the algorithm were not promising in this regard. Bielza et al. solve influence diagram problems with simple component distributions and can take very large numbers of samples in a relatively short time period. Simulation models are considerably more complex and run times can be prohibitively long if the algorithm does not converge rapidly. We provide three adaptations to the original algorithm to improve its efficiency, an adaptive scaling of the utility function, a batched modes approach akin to batched means in simulation output analysis, and an alternative that uses the augmented probability simulation to search the decision space, but then applies best simulated system techniques to choose the best decision.

In Section 2, describes the actual algorithm and discusses the requirements that guarantee asymptotic convergence. We offer modifications for improving the algorithm to achieve faster convergence in Section 3 and test the algorithm on uni-modal and bi-modal problems in Sections 4 and 5 respectively. A final example is offered involving a true discrete-event simulation, optimizing a Bayesian M/M/s queue to determine the resource allocation s. These examples can be downloaded from http://www.people.vcu.edu/~jrmerric/. Final conclusions are made and future work proposed in Section 6.

2. The APS Algorithm Applied to Simulation Optimization

Bielza et al. offer several MCMC algorithms for sampling from the artificial distribution, but one, the Metropolis algorithm (Tierney 1994), is particularly suitable for decision involving simulations. The general idea is to set up a Markov Chain that is irreducible, aperiodic and has as its limiting distribution $h(d, X)$. We sample from this Markov Chain until it converges and take additional samples as approximate samples from $h(d, X)$. Given previous samples d^{j-1} and X^{j-1}, the Metropolis algorithm uses an arbitrary probing distribution $g(d, X \mid d^{j-1}, X^{j-1})$ to sample prospective values $\overline{d}$ and $\overline{X}$ and then accepts them with probability α defined by

$$\alpha(d^{j-1}, X^{j-1}, \bar{d}, \bar{X}) = \min\left\{1, \frac{h(\bar{d}, \bar{X})}{h(d^{j-1}, X^{j-1})} \times \frac{g(\bar{d}, \bar{X} \mid d^{j-1}, X^{j-1})}{g(d^{j-1}, X^{j-1} \mid \bar{d}, \bar{X})}\right\}. \quad (3)$$

If the prospective values are rejected, then the Markov Chain remains at d^{j-1} and X^{j-1}.

However, we cannot use the Metropolis algorithm in this form as we cannot evaluate $h(d, X)$ as this involves evaluating $p_d(X)$ which represents the simulation and thus cannot be written in functional form. Instead, we define $g(d, X \mid d^{j-1}, X^{j-1})$ as $g(d \mid d^{j-1}) p_d(X)$. This means that we have a simpler probing distribution $g(d \mid d^{j-1})$ from which we draw a prospective value $\bar{d}$ and then sample $\bar{X}$ from $p_{\bar{d}}(X)$. Consider the effect these simplifications have on **Error! Reference source not found.**,

$$\begin{aligned}\alpha(d^{j-1}, X^{j-1}, \bar{d}, \bar{X}) &= \min\left\{1, \frac{h(\bar{d}, \bar{X})}{h(d^{j-1}, X^{j-1})} \times \frac{g(d^{j-1} \mid \bar{d}) p_{d^{j-1}}(X^{j-1})}{g(\bar{d} \mid d^{j-1}) p_{\bar{d}}(\bar{X})}\right\} \\ &= \min\left\{1, \frac{u(\bar{d}, \bar{X})}{u(d^{j-1}, X^{j-1})} \times \frac{g(d^{j-1} \mid \bar{d})}{g(\bar{d} \mid d^{j-1})}\right\}\end{aligned} \quad (4)$$

as $h(\bar{d}, \bar{X}) \propto u(\bar{d}, \bar{X}) p_{\bar{d}}(\bar{X})$ and $h(d^{j-1}, X^{j-1}) \propto u(d^{j-1}, X^{j-1}) p_{d^{j-1}}(X^{j-1})$. We may write our full algorithm as

1. Choose arbitrary d^0 and X^0; set $j = 1$;
2. Sample $\bar{d}$ from $g(d \mid d^{j-1})$;
3. Run the simulation under decision alternative $\bar{d}$ to obtain the output measure $\bar{X}$, i.e. sample $\bar{X}$ from $p_{\bar{d}}(X)$.
4. Evaluate $u(\bar{d}, \bar{X})$ and $u(d^{j-1}, X^{j-1})$;
5. With probability $\min\left\{1, \frac{h(\bar{d}, \bar{X})}{h(d^{j-1}, X^{j-1})} \frac{g(d^{j-1} \mid \bar{d})}{g(\bar{d} \mid d^{j-1})}\right\}$, set $(d^j, X^j) = (\bar{d}, \bar{X})$; otherwise set $(d^j, X^j) = (d^{j-1}, X^{j-1})$;
6. Set $j = j + 1$. Repeat steps 2 through 6 until convergence is practically judged.

We may achieve a further simplification if we restrict ourselves to choices for $g(d \mid d^{j-1})$ that are symmetric in the sense that $g(a \mid b) = g(b \mid a)$. This will give an acceptance probability $\alpha(d^{j-1}, X^{j-1}, \bar{d}, \bar{X}) = \min\left\{1, \frac{u(\bar{d}, \bar{X})}{u(d^{j-1}, X^{j-1})}\right\}$. Bielza et al. make suggestions for the choice of g when the set Δ is either discrete or continuous. If Δ is discrete then we may use the transition probabilities for a random walk. Suppose Δ can take any value in the integers, then we choose

$$g(i \mid j) = \begin{cases} ½, & \text{if } i = j+1 \\ ½, & \text{if } i = j-1 \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

If decisions in Δ have multiple dimensions, then we may use independent random walks in each dimension. If the set Δ is continuous, then an appropriate choice is a normal distribution of the form $g(a \mid b) = N(b, \sigma^2)$. If the decisions have multiple dimensions, then the distribution could be a multivariate normal with an appropriately chosen covariance matrix Σ.

These choices for the probing distribution give an interesting interpretation of the algorithm. We are essentially taking a random walk (or following a simple, continuous Markov process) around the space of decisions, evaluating the simulation output at each step and sometimes rejecting a given step. Considering **Error! Reference source not found.** reveals that the probability of accepting a given step is 1 if the step improves the sampled utility and less than 1 if the utility is reduced. The worse the drop in utility, the smaller the chance of accepting the step. Thus, the algorithm is not only simple, but intuitively appealing. Proofs of convergence for the Metropolis algorithm are offered in Tierney (1994) and Roberts and Smith (1994) and require only that the Markov Chain be irreducible and aperiodic. This is ensured by our condition that $u(d, X)$ be positive and integrable - $p_d(X)$ will already be a proper distribution - and by the choice of g, ensuring that there is a positive probability of eventually reaching all decisions in Δ.

Readers familiar with simulated annealing, another form of Metropolis algorithm, will notice the similarity (Johnson et al. 1989). If we were to choose an exponential utility function, defined by

$u(z) = 1 - e^{-z/R}$, where R is the risk tolerance parameter, compare the proposed next point to the best value so far and then annealed the risk tolerance to zero according to a Boltzman schedule throughout the algorithm, eventually reaching complete risk aversion, we would be performing exactly the simulated annealing approach to optimization. However, as we are decision analysts and thus choose R to represent the decision maker's attitude to risk, we must avoid altering the risk attitude of the decision maker in the final solution. Furthermore, as we are dealing with stochastic objective functions, we cannot simply record the decision with the best simulated output thus far. Instead we must record the entire sample of decisions and use this to determine the optimal decision. This means that we lose the no memory property of simulated annealing, but might allow us more flexibility in determining the optimal solution.

4. Adaptive Utility Scaling

The algorithm developed in Section 3 has guaranteed asymptotic convergence. However, we would also like the mode of samples from the augmented probability distribution of the decision to converge quickly to the optimal decision. Our initial tests of the algorithm were not promising in this regard. Bielza et al. solve influence diagram problems with simple component distributions and can take very large numbers of samples in a relatively short time period. Simulation models are considerably more complex and run times can be prohibitively long if the algorithm does not converge rapidly.

One main issue with convergence for the algorithm is the acceptance probability in **Error! Reference source not found.** given by $\min\left\{1, \frac{u(\bar{d}, \bar{X})}{u(d^{j-1}, X^{j-1})}\right\}$. If the sampled utility values are in a small range far from zero then the calculated values of the acceptance probability will all be near one. For instance, if the utility values range between 9 and 10, then the lowest possible acceptance probability will be 0.9. Although asymptotic convergence is guaranteed, the chain is then a random walk affected only infrequently by rejections based on the utility values and convergence to the optimal solution will be slow. However, if the calculated values of the acceptance probability are spread out between zero and

one, then the algorithm will be more heavily influenced by utility values. The method we offer attempt to ensure that the acceptance probabilities are spread out between zero and one.

One technique for improving the efficiency of the algorithm is affine scaling of the utility function. Specifically, we wish to scale the utility function so that utilities near, but not below, zero are encountered. Affine transformations of a utility function do not affect the decision-maker's preferences [see for example Winston (2004) page 747 for a proof], thus we are not changing the optimal solution in applying such scaling. An adaptive algorithm is desirable, where the utility function is scaled each iteration to ensure sufficient range. To achieve this, we can maintain the minimum and maximum utility calculated up to the current iteration in the algorithm, denoted by $u^{\min}$ and $u^{\max}$ and scale each utility value between zero and one using $u'(x) = \frac{u(x) - u^{\min}}{u^{\max} - u^{\min}}$. When we take the ratio in the acceptance probability calculation in **Error! Reference source not found.**, the denominators of the scaling expressions cancel, implying that we only need to record the minimum utility up to the current iteration. The algorithm can be re-stated as:

1. Choose arbitrary d^0 and X^0; set $u^{\min} = u(X^0)$; set $j = 1$;
2. Sample $\bar{d}$ from $g(d \mid d^{j-1})$;
3. Run the simulation under decision alternative $\bar{d}$ to obtain the output measure $\bar{X}$, i.e. sample $\bar{X}$ from $p_{\bar{d}}(X)$.
4. Evaluate $u(\bar{d}, \bar{X})$ and $u(d^{j-1}, X^{j-1})$;
5. If $u^{\min} > u(\bar{d}, \bar{X})$ then $u^{\min} = u(\bar{d}, \bar{X})$;
6. With probability $\min\left\{1, \frac{u(\bar{d}, \bar{X}) - u^{\min}}{u(d^{j-1}, X^{j-1}) - u^{\min}}\right\}$, set $(d^j, X^j) = (\bar{d}, \bar{X})$; otherwise set $(d^j, X^j) = (d^{j-1}, X^{j-1})$;
7. Set $j = j + 1$. Repeat steps 2 through 7 until convergence is practically judged.

While it is evident that we can perform such an affine scaling of the utility function and not effect the decision maker's preferences in terms of the expected utility, within the algorithm we are sampling from the full distribution $h(d,X) \propto u(d,X)\,p_d(X)$ not just the marginal $h(d) = \int h(d,X)dX$. Let us define $u^*(d,X) = u(d,X) - u_{\min}$, considering $u_{\min}$ to be a fixed value for now, and define

$$\begin{aligned} h^*(d,X) &\propto u^*(d,X)\,p_d(X) \\ &\propto \left(u(d,X) - u_{\min}\right) p_d(X) \\ &\propto u(d,X)\,p(X) - u_{\min}\,p_d(X). \end{aligned}$$

If we scale the utility function, then we are sampling from $h^*(d,X)$ rather than $h(d,X)$. This raises the question whether when we marginalize the sample, taking only the samples for d , can we still take the mode of this sample as the maximum expected utility decision? Consider the marginal distribution of $h^*(d,X)$ for d ,

$$\begin{aligned} h^*(X) &= \int h^*(d,X)dX \\ &\propto \int u(d,X)\,p_d(X) - u_{\min}\,p_d(X)dX \\ &\propto \int u(d,X)\,p_d(X)dX - \int u_{\min}\,p_d(X)dX \\ &\propto \int h(d,X)dX - u_{\min}\int p_d(X)dX \\ &\propto h(d) - u_{\min} \\ &\propto h(d) \end{aligned}$$

Thus the marginal distribution of $h^*(d,X)$ for d is proportional to the marginal distribution of $h(d,X)$, and thus proportional to $V(d)$ the expected utility for d . As we are sampling values for d and X , not utility values, this implies that we are not affecting our sampled values. This will be true whether we vary $u_{\min}$ over the course of the algorithm or use a fixed value. However, if we intend to use the actual utility values at any stage, we must calculate and use the natural values without scaling. We will also see in the numerical experiments that while the scaling algorithm improves the rate of rejections,

this effect reduces as low utility values are sampled over the course of the algorithm. Thus we found that re-setting the minimum utility at fixed numbers of iterations actually led to the best performance.

5. Empirical Tests with a Uni-modal Expected Utility

Consider a decision where the decision d can take any integer value. The output of interest is drawn from a normal distribution with fixed standard deviation 1, but whose mean equals $\alpha - (d-10)^2$. The prior on α is also a normal distribution with mean 400 and standard deviation 1. The decision maker's utility function on the profit is assumed to be of the exponential form, defined by $u(z) = 1 - e^{-z/R}$, where R is the risk tolerance parameter which we set to 400. Note that this utility function will be negative for $z < 0$. This will not be a problem under the modified algorithm with utility scaling. However, if this is not used, we must still scale the utility function to ensure no negative values are obtained in the algorithm. We use the random walk for the probing distribution g as shown in (5).

Note that we have set up the problem so that the optimal decision $d^* = 10$. We initialize the algorithm from Section 3, the basic algorithm, with $d^0 = 0$ and $X^0 = 300$ and performed $20,000$ iterations. Figure 1 shows the sample marginal distribution of d. Note that the mode is not 10, but 6. In fact, if we re-run the algorithm, we get values that vary around 10.

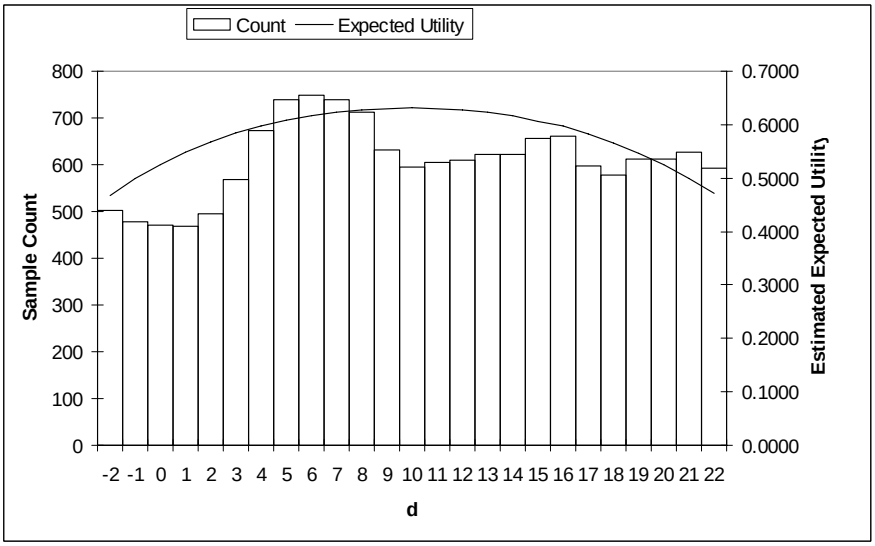

Figure 1. The marginal sample for d plotted against the estimated expected utility.

If we run the algorithm 100 times, we may examine the distribution of the mode after a given number of iterations. Figure 2 shows box plots for the 100 values for the mode every 1,000 iterations. While the median of this distribution is close to 10, there is a wide variability in the mode even after 20,000 iterations of the basic algorithm.

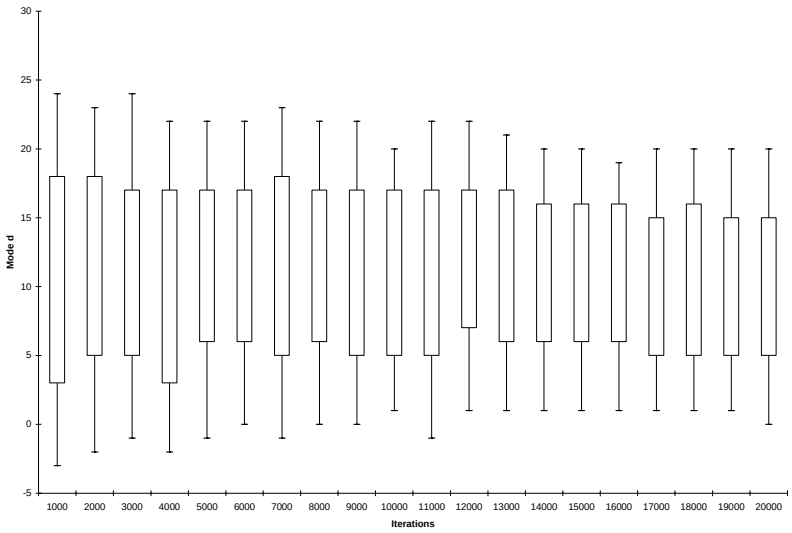

Figure 2. The distribution of the mode for increasing iterations.

In Section 4, we proposed adaptive utility scaling to improve the convergence of the algorithm. Figure 3 shows the progression of the distribution of the mode for 100 runs of the algorithm with utility scaling following. The number of iterations is counted after 1000 iterations of warm-up. The algorithm converges faster with utility scaling than the basic algorithm. However, note that the variability in the mode does not decrease significantly with additional iterations.

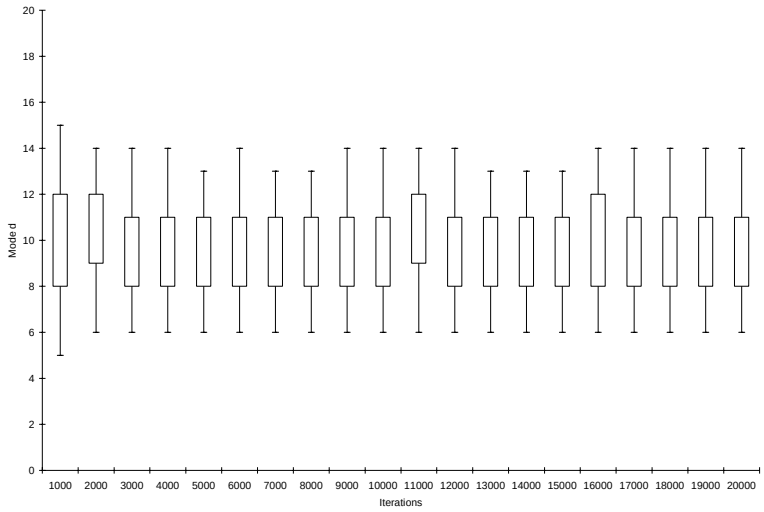

Figure 3. The distribution of the mode for increasing iterations with utility scaling.

It does not appear that we can reduce the variability by running longer Metropolis chains after warm-up. This suggests that multiple, independent chains might be more advantageous. To test this theory, we performed multiple chains, each with the same initial values but with 1000 iterations removed as warm-up making the chains effectively independent. The mode for the next 1000 iterations was then calculated for each chain and the average mode found across multiple chains. Figure 4 shows the distribution of the average mode across increasing numbers of independent chains when we repeated this procedure for 100 runs of the algorithm. The convergence now follows the square root law for averages of increasing sample sizes in the Central Limit Theorem. Note that the interquartile range for average of the modes for 20 runs of 1,000 iterations after warm-up is 0. In fact 67 of the 100 determined solutions were correct at 10 and all but one was within 1 either way. The interquartile range of the mode for a single, long chain of 20,000 iterations was 3 despite being based on the same computational effort after warm-up. In each case, the median of the 100 determined solutions is 10, the correct optimal solution. We did in fact repeat this experiment and each time got a higher interquartile range for the mode of the single long chain than the average of the modes for multiple short chains, thus this does not appear to be sampling error.

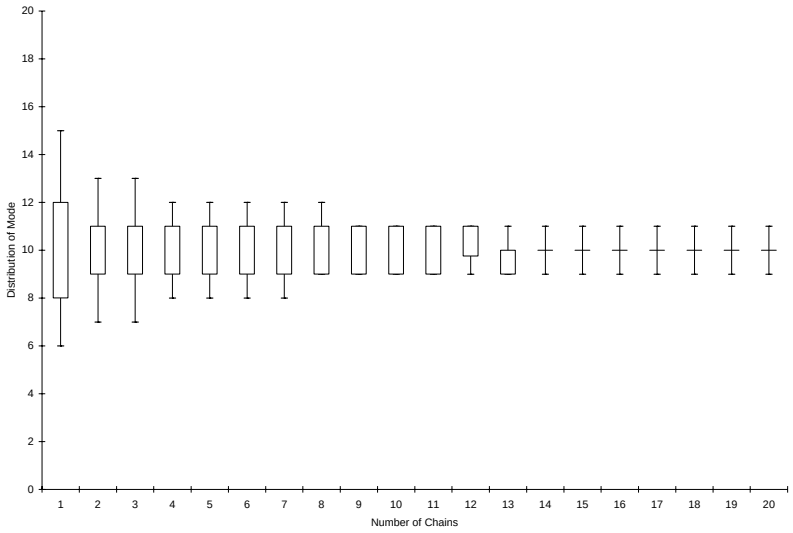

Figure 4. The distribution of the average of the modes after multiple, independent runs with utility scaling.

One problem with the multiple, independent chains approach is that while the computational effort after warm-up is the same, the warm-up period is repeated for each independent chain. In estimating means for steady state simulations, batched means are used to overcome this problem (Law and Kelton 2000 page 528). Thus, we propose batched modes. To demonstrate this procedure, we ran one long chain with a warm-up of 1,000 iterations and then 39,000 iterations of collected decision samples. Note that we went to 40,000 iterations total as this is the same total computational effort as calculating the average of the modes for 20 independent chains of 1,000 iterations with 1,000 iterations of warm-up each. Rather than finding the mode for the entire sample, as in Figure 3, we found the mode for each successive batch of 1,000 iterations. We then took the average of these the modes for these 39 batches. Figure 5 shows the progression of average of the modes for increasing numbers of batches.

Figure 5 shows that while the average of the modes of non-independent batches does appear to reach a steady state distribution, it does not converge to the optimal solution in the same manner as the average of the modes for independent runs. We note that although the newly generated decisions and simulation outputs are created independently, the utility scaling algorithm does record the lowest value of the utility found thus far. The longer a single chain becomes, the more likely it is that the algorithm will

visit decision with low simulated outputs and thus low expected utilities. This means that the minimum expected utility, $u^{\min}$, decreases over a long chain and this actually makes the acceptance probabilities higher for utility decreasing steps. This appears to be the reason that there is no benefit of obtaining additional dependent batches.

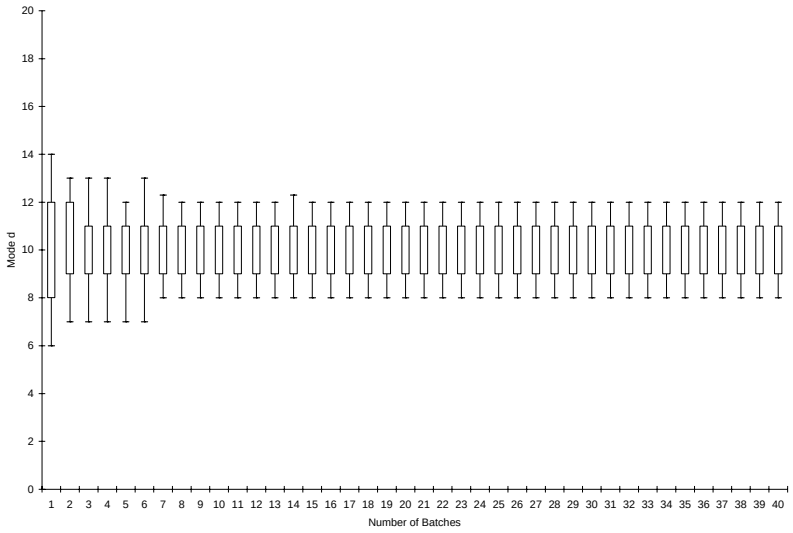

Figure 5. The distribution of the average of the modes for multiple batches with utility scaling.

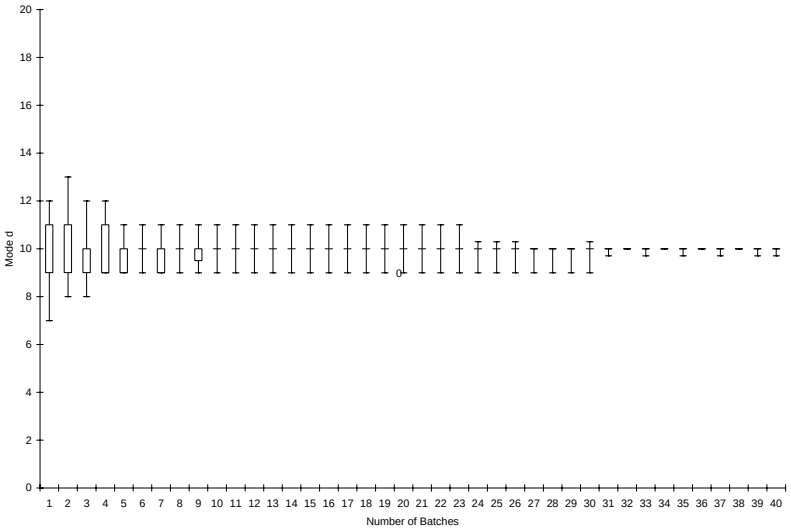

Figure 6. The distribution of the average of the modes for multiple batches for the utility scaling algorithm resetting the minimum utility every 1,000 iterations.

Thus, in Figure 6 we show the same approach but with the minimum utility $u^{\min}$ reset every 1,000 iterations. Convergence to the true optimum is now must faster, with a probability of correctly selecting $d^* = 10$ reaching 0.9 at 40 batches.

From these small empirical experiments, the utility scaling not only ensures positive utility values, but also improves the rate of convergence. Furthermore, the use of an average of modes of multiple short chains of the Metropolis algorithm further improves the accuracy for the same computational effort after warm-up and the use of batched modes does not improve accuracy for the same computational effort. However, if we reset the minimum utility at the beginning of each batch we obtain the fastest convergence of our different empirical tests. In the next section, we will examine the performance of these techniques for multi-modal expected utilities functions and the complexity this adds to the analysis of the modes for multiple short-runs.

6. Empirical Tests with a Bi-modal Expected Utility

To test the influence of our heuristics on the ability of the algorithm to search multiple modes we will introduce a multi-modal expected utility function. We change the previous example only in that the simulation output of interest is drawn from a normal distribution with fixed standard deviation of 1, but whose mean equals $\alpha - d(d-10)^2(d-45)$, and the prior on α is also a normal distribution with mean $60,000$ and standard deviation $1,000$. This function will have maxima at 3 and 37 (if d is discrete), but taking expected utilities, there will be a local solution at 3 and the global solution $d^* = 36$. As the scale of the output values are different, the risk tolerance R is assumed to be $50,000$. We initialize the algorithm at the same values with $d^0 = 0$ and $X^0 = 300$, thus the algorithm starts near the local optimum at 3, not near the global optimum at 36.

Figure 7 shows the progression of the distribution of the mode for 100 runs of the algorithm with utility scaling. Note that there is tremendous variability in the mode after 20,000 iterations. This is

because modes from some runs cluster around the global optimum at 36 and modes for other runs cluster around the local optimum at 3.

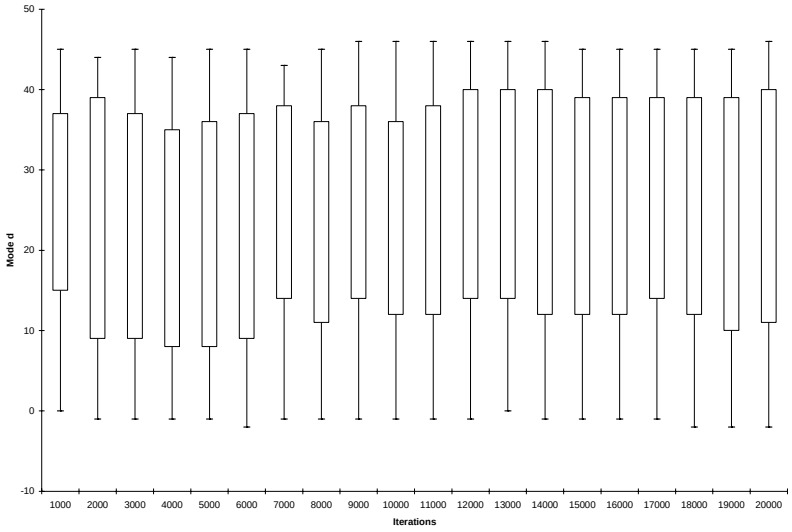

Figure 7. The distribution of the mode for increasing iterations with bi-modal response.

Figure 8 shows the progression of the sample value of the decision, d, for two independent chains of the algorithm for Example 2. Note that the upper chain converges quickly to the global optimum at 36. However, the lower chain transitions back and forth between the global optimum and the local optimum at 3. Figure 9 shows the histograms obtained by taking the marginal samples for d for the two chains shown in Figure 8; the upper chain in Figure 8 is the left histogram in Figure 9. The average of the sampled utility values for each value of d is also shown.

Figure 9 suggests two approaches to the problem. The first involves finding the multiple modes of the sample for d. However, the right histogram would still indicate that the better solution would be near the local optimum at 3. Note that the expected utility curve does show the superiority of the global optimum in each case. We have already noted that the no memory property of simulated annealing is lost with our algorithm as we have to save all sample values of d after warm-up. The additional memory of saving the utility values will not be overly expensive. Thus, a second approach is to find the value of d that has the highest average utility for the samples taken.

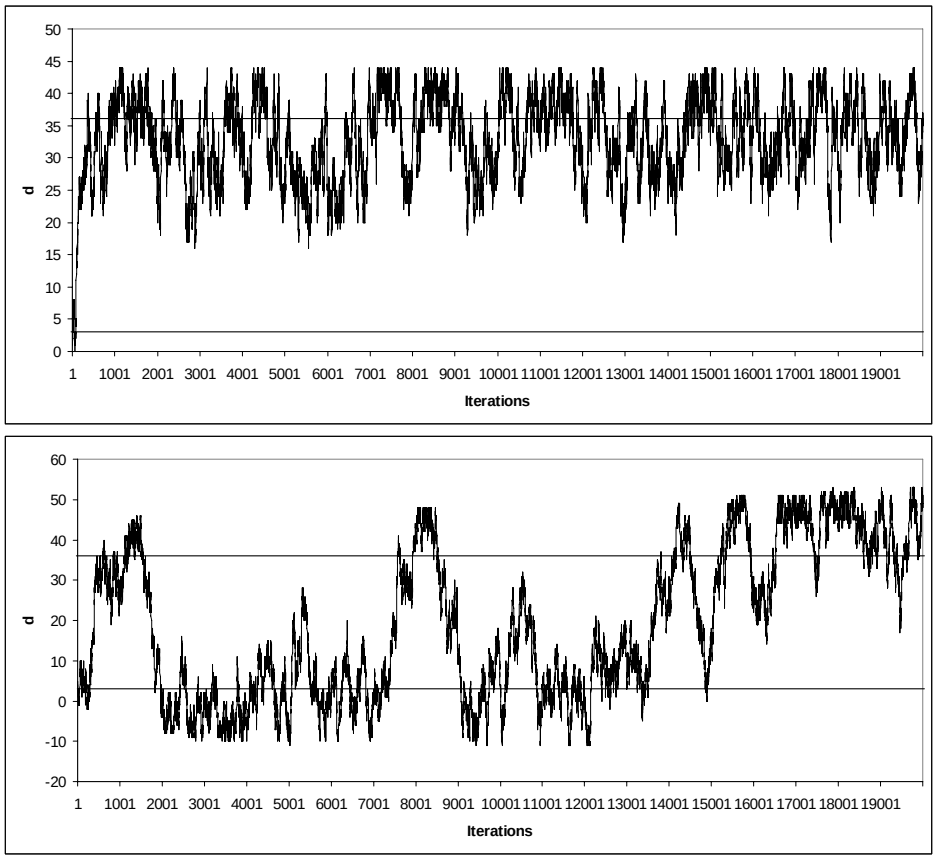

Figure 8. Two long chains for a bi-modal response one staying around the global optimum, the other switching between the local and global optima.

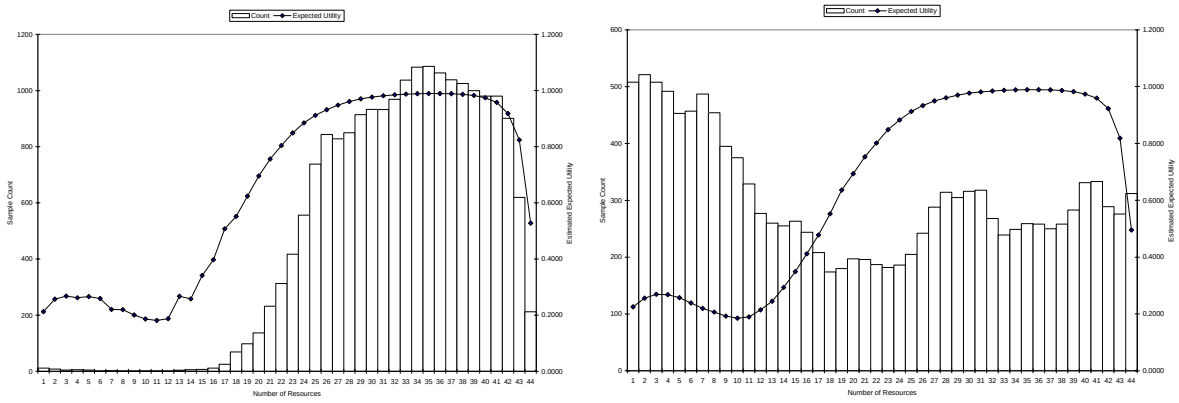

Figure 9. Histograms and Estimated Expected Utilities for the two chains shown in Figure 8.

Figure 10 shows the progression of the decision with the highest average utility. We note that simulated system selection procedures could be applied here, such as Bonferoni's probability of correct selection and indifference zone procedures (Matejcik and Nelson 1995; Bechhofer et al. 1995; Goldsman and Nelson 1998; Chick and Inoue 2001a 2001b), but we leave this for later work. The use of decision with the highest average utility converges faster than any of our previous procedures and continues to work even when $u^{\min}$ decreases over the course of a long run and the chain begins to wander as it becomes less controlled by the utilities.

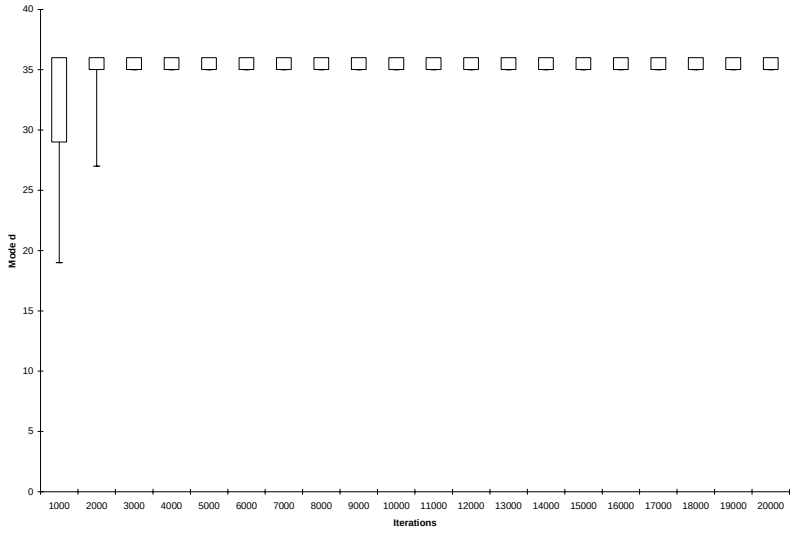

Figure 10. The distribution of the decision with the highest average utility for increasing iterations with bi-modal response.

Figure 11 shows the average of the highest average utility decision for batches of 1,000 iterations with minimum utility resetting as used in the previous section. Note that in Figure 11, the batches start after 1,000 iterations of warm-up, so the first box in Figure 11 corresponds to the second box in Figure 10 in terms of total iterations thus far. The effect of the use of batches is marginal. With one long run, we reach the true optima after 3,000 iterations with only the occasional occurrence of a 35. The probability of correct selection is estimated to be 0.55. With batches of 1,000 iterations, after warming up for 1,000 iterations, two batches allow us to reach a probability of correct selection of 0.64. We attempted to reduce the size of the batches and got more variable results as the chain may not visit the points around the

optimum with sufficient repetition during a short batch to get a reasonable estimate of the average utility at each point.

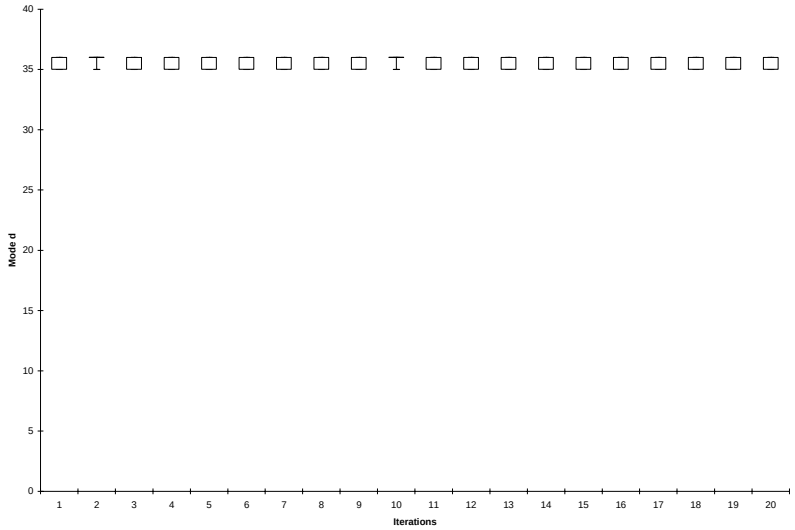

Figure 11. The distribution of the average of the decision with the highest average utility for multiple batches resetting the minimum utility every 1,000 iterations with bi-modal response.

To summarize the findings in this section, Figure 12 shows the progression of estimates of the probability of correct selection by the total number of iterations performed for the alternative procedures we have examined. Note that the batched approach has a probability of zero at 1,000 iterations as this is the warm-up period.

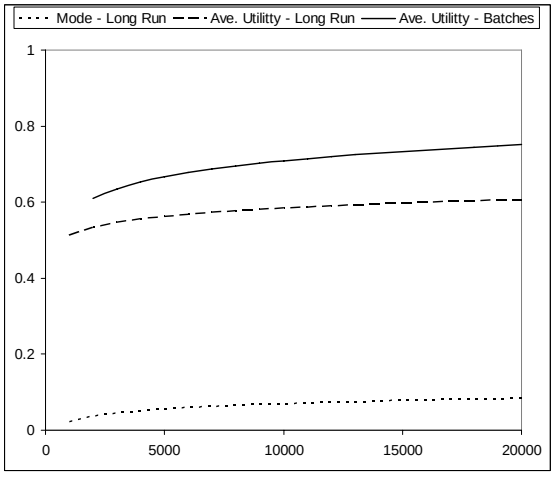

Figure 13. Estimates of the probability of correct selection ($d^* = 36$) by total number of iterations.

The batched approach taking the highest average utility in each batch achieves the highest probability of correct selection, approaching 0.75, with the incorrect selection all indicating 35 as the best decision. As mentioned previously though, the drawback with this approach is that we found that we could not reduce the size of the batches without a drop-off in accuracy.

As a comparison to the procedures developed for the uni-modal response in Example 1, we applied the highest average utility procedure to the problem in Example 1 combined with the use of batches of 1,000 iterations and resetting the minimum utility at the start of each batch. All batches achieve the correct optimal solution at 10. Even with batches of size 50, the algorithm has a probability of correct selection of 0.63 at 1 batch to 0.84 at 20 batches. The total iterations for 20 batches was 2,100 as we used a 100 iteration warm-up. This compares to a probability of 0.9 for 40 batches of 1,000 iterations and a probability of 0.43 for 2 batches of 1,000 iterations under the batched modes approach. Thus again the use of average utilities increases the rate of convergence significantly.

7. Example 3: A Discrete Event Simulation Optimization

Consider a simple, single process, discrete event simulation. We wish to determine the optimal number of resources to allocate to the process. We assume that each entity processed generates 25¢ in revenue, but each minute in the queue loses 25¢ in future revenues. The simulation is run for an 8 hour day and each resource allocated costs $25 per day. The inter-arrival times are assumed to follow an exponential distribution with expected value λ, while the service times are exponential with expected value μ. Based on the decision maker's prior beliefs and available data, the posterior distribution of λ is a gamma distribution with shape 0.5 minutes and scale 0.1 minutes, to give an expected value of 0.05 minutes or 3 seconds, and the posterior distribution of μ is a gamma distribution with shape 5 minute and scale 0.1 minutes, to give an expected value of 0.5 minutes or 30 seconds. The simulation output of interest X is the total revenue generated in an 8 hour day. The decision maker then wishes to improve profits which may be found by taking $z = X - d \times \$250$. The decision maker's utility function on the profit is assumed

to be of the exponential form, defined by $u(z) = 1 - e^{-z/R}$, where R is the risk tolerance parameter which we assume to be 1000. Note that this utility function will be negative for $z < 0$. If negative profits are possible, we must perform an affine scaling of the utility function to ensure no negative values are obtained in the algorithm.

Figure 8 shows the histogram of 1000 samples obtained for d, the number of resources to allocate with these resources.

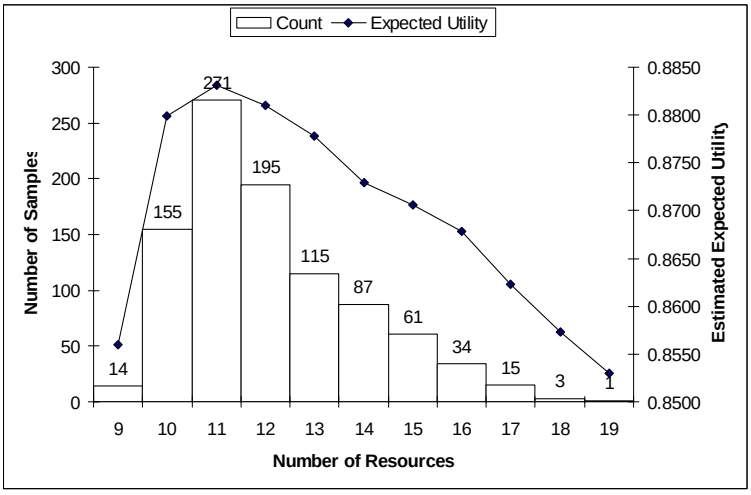

Figure 8. A histogram of the augmented distribution of d and the average utilities.

To implement the algorithm, we used a random walk generating probability for our probing distribution as described in (5), but with a boundary at 1. Thus rather than moving to 0 with probability $\frac{1}{2}$, the random walk stays at 1. This probing distribution is still symmetric as required in Section 3. The simulation in this example has been implemented in Simul8™ using Visual Logic™, outputting the resulting Markov Chain to Excel™ for analysis. The Simul8 model and Excel file can be downloaded from http://www.people.vcu.edu/~jrmerric/. We chose to use the utility scaling algorithm and found the highest average utility over 1,000 iterations. The algorithm was started with $d^0 = 3$ and $X^0 = \$2000$ and run for 1000 iterations, although the decision with the highest average utility did not change after 100 iterations. The highest average utility can be seen in Figure 8 to be for 11 resources.

6. Conclusions

The Augmented Probability Simulation algorithm developed herein is both simple to implement and heuristically appealing. Moreover, as we are already using simulation to estimate the distribution of output measures, it is attractive to use simulation to search the decision space as well. The most important feature of the algorithm in an academic sense though, is the guarantee of asymptotic convergence. Several extensions to the algorithm have been offered and tested based on small empirical experiments on abstract, academic problems. Firstly, adapting the parameters of an affine transformation of the utility function has been shown empirically to increase the speed of convergence. Secondly, the use of batched modes further increases the speed of convergence. We also found that resetting the minimum utility used to scale the utility function led to additional improvement. Lastly, we found that taking the decision with the highest average utility rather than taking the most frequent decision in the sample (and throwing away the sampled utility values) reduces the iterations necessary to find the optimal solution for both uni-modal and bi-modal response functions. We have not, however, implemented the algorithm for decisions with multiple dimensions or decision variables taking continuous values. As a more practical demonstration, we implemented the procedure for a small discrete event simulation example.

References

April, J., F. Glover, J. P. Kelly, M. Laguna. 2003. Practical introduction to simulation optimization. S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, eds. *Proc. Winter Simulation Conf.*, 71-78.

Banks, J., J. S. Carson, B. L. Nelson, D. M. Nicol. 2000. *Discrete-Event System Simulation*, 3rd ed. Prentice-Hall, Inc., Upper Saddle River, NJ.

Bechhofer, R.E., T.J. Santner, and D.M. Goldsman. 1995. *Design and Analysis of Experiments for Statistical Selection, Screening, and Multiple Comparisons*, New York: John Wiley & Sons.

Bielza, C., P. Müller, D Rios Insua. 1999. Decision analysis by augmented probability simulation. *Management Science* **45**(7) 995-1007.

Chen, C.-H. 1996. A lower bound for the correct subset-selection probability and its application to discrete event simulations. *IEEE Trans. Automatic Control* **41**(8) 1227–1231.

Nakayama, M. K. 2000. Multiple-comparisons with the best Cheng, R. 1999. Regression metamodeling in simulation using Bayesian methods. P. Farrington, H. Nembhard, D. Sturrock and G. Evans, eds. *Proc. Winter Simulation Conf.*, 330-335.

Chick, S. 1997. Bayesian analysis for simulation input and output. S. Adradottir, K. Healy, D. Withers and B., eds. *Proc. Winter Simulation Conf.*, 253-260.

----------. 2001. Input distribution selection for simulation experiments: accounting for input uncertainty. *Operations Research* **49**(5) 744-758.

Chick, S., K. Inoue. 2001a. New procedures to select the best simulated system using common random numbers. *Management Science* **47**(8) 1133-1149

----------. 2001b. New two-stage and sequential procedures for selecting the best simulated system. *Operations Research* **49**(5) 732-743.

Fu, M. C. 2001. Simulation Optimization. 2001. B. A. Peters, J. S. Smith, D. J. Medeiros, and M. W. Rohrer, eds. *Proc. Winter Simulation Conf.*, 53-61.

Gelfand A. E., A. F. M. Smith. 1990. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Society* **85**(410) 398-409.

Glynn, P. 1986. Problems in Bayesian analysis of stochastic simulation. J. Wilson, J. Henrikson and S. Roberts, eds. *Proc. Winter Simulation Conf.*, 376-383.

Goldsman, D. B.L. Nelson. 1998. *Comparing systems via simulation*. In Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice, ed. J. Banks. New York: JohnWiley & Sons.

Gupta, S. S., K. J. Miescke. 1996. Bayesian look ahead one-stage sampling allocations for selecting the best population. *Journal of Statistical Planning and Inference* **54** 229–244.

Ingber, A. L. 1989. Very fast simulated re-annealing. *Journal of Mathematical and Computer Modelling* **12** 967-973.

Johnson, D.S., C. R. Aragon, L. A. McGeoch and C. Schevon. 1989. Optimization by simulated annealing: an experimental evaluation. Part 1: graph partitioning. Operations Research 37 865-892.

Law, A. M., W. D. Kelton. 1991. *Simulation Modeling & Analysis*, 2nd ed. McGraw-Hill, Inc., New York.

Matejcik, F. J., B. L. Nelson. 1995. Using common random numbers for indifference-zone selection and multiple comparisons in simulation. *Management Science* **41**(12) 1935-1945.

Merrick, J. R. W., J. R. van Dorp and V. Dinesh. Assessing uncertainty in simulation based maritime risk assessments. Submitted to *Risk Analysis*.

Roberts, G., A. Smith. 1994. Simple conditions for the convergence of the Gibbs sampler and Metropolis-Hastings algorithms. *Stochastic Processes and Their Appl.* **49** 207–216.

Smith, A., G. Roberts. 1993. Bayesian computation via the Gibbs sampler and related Markov chain Monte Carlo methods. *J. Royal Statist. Soc., B* **55** 3–24.

Tanner, M. 1994. *Tools for statistical inference*. Springer-Verlag, New York.

Tierney, L. 1994. Markov chains for exploring posterior distributions (with discussion). *Annals of Statistics* **22** 1701–1762.

Winston, W. 2004. *Operations Research: Applications and Algorithms*. Brooks/Cole – Thomson Learning, Belmont California.

These methods fall in to four categories under various stochastic models: (1) guaranteed asymptotic convergence to the optimum; (2) guaranteed optimality under the deterministic counterpart of the model; (3) guaranteed pre-specified probability of correct selection; and (4) robust heuristics with no guarantees (Fu 2001). Some work has been performed on the question of system selection under the Bayesian paradigm (Chen 1996; Gupta and Miescke 1996; Chick and Inoue 2001a 2001b). The proposed techniques fall under category (3) above, with guaranteed pre-specified probability of correct selection. The extension to simulation optimization has not been made under the Bayesian paradigm, although the meta-heuristic and stochastic approaches in the work discussed above would be equally suitable for optimization of Bayesian simulations.

We determined several improvements to the way we used the results of the algorithm through empirical tests. These include an adaptive scaling of the utility function subtracting the minimum sampled utility value to improve the rejection rate for sampled Metropolis steps. We show that this adaptation does not affect the guaranteed convergence of the algorithm to the optimal solution. However, we found that while this adaptation improved the convergence of the algorithm in the short term, it required re-setting the minimum utility at fixed intervals to achieve the best performance. Another improvement was the use of batches similar to batched means used in steady state simulation analysis. Taking the average of the estimated optimal decision from batches of samples from one long Metropolis chain improved the rate of convergence for the algorithm. Lastly, the algorithm proposed by Bielza et al. does in fact lose the no-memory property of its deterministic counterpart, simulated annealing (Johnson et al., 1989). As all sampled values from the augmented decision distribution must be kept for calculation of the mode, we found it obvious to also record the sample utility values and make the decision by estimating the expected utility for each decision by taking the average of the sampled utility values at each sampled decision value. The accuracy of this estimate improved by the fact that $h(d)$ *is proportional to* $V(d)$*, so more samples are obtained at decisions with higher expected utilities. Thus the algorithm searches the decision space, while the final choice is made by taking the highest average utility.*

We note, however, that structural discussion of the properties of these estimators is not possible. While many authors have discussed efficient estimation of the mode (Hall et al., 1996; Hedges and Shah, 2003; Bickel, 2001), these estimators are for continuous distributions (not suitable for discrete decisions), uni-modal distributions (there may be multiple local minima) and are based mostly based on bootstrap estimates (giving no closed form structural results). However, the greatest problem in applying these results is that they assume an independent sample; this is not the case for samples taken using the Metropolis algorithm, thus biasing any independence-based results.