Proceedings of the 2016 Winter Simulation Conference
T. M. K. Roeder, P. I. Frazier, R. Szechtman, E. Zhou, T. Huschka, and S. E. Chick, eds.

MIXED OPTIMIZATION FOR CONSTRAINED RESOURCE ALLOCATION, AN APPLICATION TO A LOCAL BUS SERVICE

Felisa J. Vázquez-Abad
Larry Fenn

Department of Computer Science, Hunter College, and
CUNY Institute for Computer Simulation, Stochastic Modeling and Optimization
New York 10065 NY, USA

ABSTRACT

The present paper follows up on Vázquez-Abad (2013), where we applied the *ghost* simulation model to a public transportation problem. The ghost simulation model replaces faster point processes (passenger arrivals) with a "fluid" model while retaining a discrete event simulation for the rest of the processes (bus dynamics). This is not an approximation, but an exact conditional expectation when the fast process is Poisson. It can be interpreted as a Filtered Monte Carlo method for fast simulation. In the current paper we develop the required theory to implement a mixed optimization procedure to find the optimal fleet size under a stationary probability constraint. It is a hybrid optimization because for each fleet size, the optimal *headway* is real-valued, while the fleet size is integer-valued. We exploit the structure of the problem to implement a stopped target tracking method combined with stochastic binary search.

1 INTRODUCTION

We propose here a mixed stochastic optimization problem that involves a stochastic binary search on an outer level, mixed with a non-linear stochastic optimization method for a stationary cost. The problem that motivated the current research was studied by Vázquez-Abad (2013) and it dates back to 2005 when the Melbourne airport needed advice to buy a new fleet of buses that bring people to and from the terminals and the parking lots. It was implicit that an optimal scheduling (here represented by a quantity $u \in \mathbb{R}^+$ called the "headway") would be applied, so it was a problem of deciding how many buses b to buy, ensuring a quality of service. In particular, they required that at least 95% of the passengers should wait less than 10 minutes for a bus. The problem has the general structure of resource allocation under risk: there is an implicit assumption that more buses will be more costly, not only because of the cost of the buses, but also because it is not efficient to pay the hours for many drivers while waiting idly at the bus depot for their turns. On the other hand, too few buses or a badly chosen value of u will increase the probability of waiting more than 10 minutes for the bus at the carparks.

The main challenge arises because of the complexity in the original model that would make the straightforward simulation-based optimization impossible to evaluate in reasonable time. The ghost model from Vázquez-Abad (2013) already provides a remarkable speed up of the simulation. However, the probability constraint that has to be estimated requires long simulations (stationary problem) for each value of the fleet size b. For such problems gradient-based optimization may be very complex if even at all practical. Instead, one can use other search methods not requiring derivative information. The general form of the stochastic constrained resource allocation problem is in Section 2. Because the goal is to find the optimal size b^*, it is apparent that spending too much CPU time optimizing for u at other values of b far away from optimal can be inefficient. Section 3 presents the proposed truncated gradient search combined with binary search to explore the control space in an efficient manner. Our method is presented in a general framework and illustrated with a simple example for which the analytic solution is known. Experimental results in Section 4 conclude the paper.

2 RESOURCE ALLOCATION UNDER A STATIONARY CONSTRAINT

The general structure of the model is of the form

$$\arg\min_{b\in\mathbb{N}} \left(\min_{u\in I:\ \mathscr{G}(u,b)\le c} C_b(u) \right) \tag{1}$$

where $I \subset \mathbb{R}^d$ is an interval, and for each value of b the function C_b is a cost function of the continuous parameter $u \in I$. The parameter b represents a "resource". The constraint function represents a "risk" associated with the model and it satisfies $\mathscr{G}(\cdot,b) \in C^2$ for any fixed b, and $\mathscr{G}(u,\cdot)$ is monotonic decreasing.

In many applications the (optimal) cost function increases with increasing resources, so the optimization problem reduces to a constraint satisfaction problem. Because $\mathscr{G}$ decreases with increasing b, then the original problem is equivalent to finding the solution of the simplified problem:

$$\arg\min_{b\in\mathbb{N}} f(b) \le c, \qquad f(b) = \min_{u\in I} \mathscr{G}(u,b). \tag{2}$$

In other words, find the minimal b such that there exists a u where $\mathscr{G}(u,b) \le c$. We will assume that $f(b)$ is strictly decreasing (if not, then multiple values of resources yield constant satisfaction and ties can be broken by minimizing the value of b).

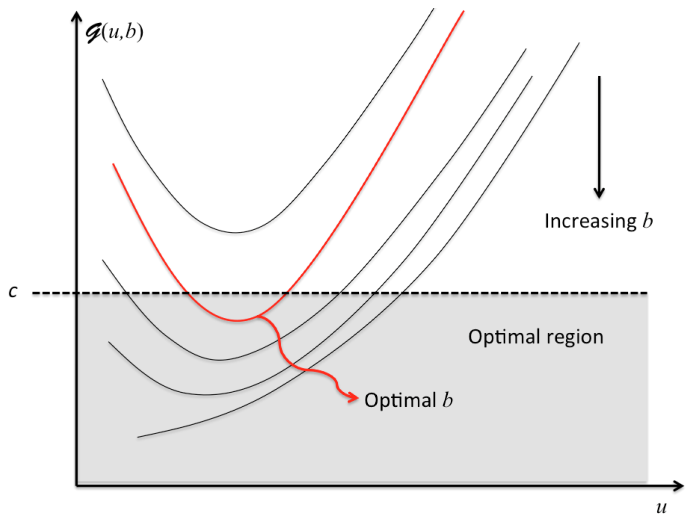


Figure 1: Illustration of the constraint $\mathscr{G}(u,b)$.

The challenge is to solve (2) when $\mathscr{G}$ is not available in closed form. We assume here that for any given values of b and u the function is a stationary average of the form

$$\mathscr{G}(u,b) = \lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} g\left(\xi_n(u,b)\right),$$

where $\{\xi_n(u,b)\}$ is a Markov process with unique stationary measure and g is a continuous and bounded function. In the general scenario (which we consider in our model), long simulations can approximate a stationary average value $\mathscr{G}(u,b)$. The focus of this research is how to efficiently find a solution with as few simulations as possible. We therefore try to find the least amount of simulations while exploring solutions, and seek early termination of sub-problems to determine the correct value of b, rather than trying to estimate $f(b)$ accurately.

This general formulation has many potential applications other than the motivating example for the public transportation system. It represents a natural generalization of resource allocation problems. Chakraborty, Das, and Magdon-Ismail (2011) and Karp and Kleinberg (2007) mention various important scenarios where these problems arise. In strategic planning, the resource b represents the various levels of investment in risky development and one seeks to find the minimal investment level that assures certain probability of success. Our formulation allows more detail in modeling this probability, which depends on various other parameters for optimal management, given an investment level. Karp and Kleinberg (2007) also mention the problem of finding the best possible placements of ads in a Google website under a lower bound on the expected click rate. Chakraborty, Das, and Magdon-Ismail (2011) mention the problem of online dynamic pricing of goods and/or services, where the seller seeks the maximum price b that ensures a certain probability of selling. Finally, drug dosage during discovery phase deals with the problem of finding a maximum dose that keeps the toxicity at a certain level. In all these problems the constraint can be modeled in further detail requiring a second level of optimal management of the resources.

Assumptions. For each value of (u,b) let $\hat{\mathscr{G}}_N(u,b)$ be an estimator of $\mathscr{G}(u,b)$ using N observations of the process of the form:

$$\hat{\mathscr{G}}_N(u,b) = \frac{1}{N}\sum_{n=1}^{N} g\left(\xi_n(u,b)\right), \tag{3}$$

(A). For any u,b the underlying Markov chain $\{\xi_k(u,b)\}$ has transition kernel $P_u(x,dx) = \mathbb{P}(\xi_{k+1}(u,b) \in dx \,|\, \xi_k(u,b) = x)$. For fixed b assume that $P_u(x,\cdot)$ is weakly continuous in x,u and that there is a unique stationary measure μ_u, with $\{\mu_u; u \in I\}$ tight.

(B). For fixed $b \in \mathbb{N}$, $\mathscr{G}(\cdot,b) \in C^2$ and it satisfies $0 < \mathscr{G}''(u,b) \leq \frac{1}{K_b}$ for all $u \in I$, so for every b there is a unique minimum $u^*(b)$.

(C). For any fixed $u \in I$, $\mathscr{G}(\cdot,u)$ is monotonic decreasing as a function of the resources b.

(D). There exists $b > 0$ such that $f(b) \leq c$, that is, the feasible region is not empty.

(E). For any choice of $b \in \mathbb{N}$ and $u \in I$ a central limit theorem holds for $\hat{\mathscr{G}}_N(u,b)$, namely there is a finite limit variance $\sigma^2_{(u,b)}$ such that $\sqrt{n}\,(\hat{\mathscr{G}}_N(u,b) - \mathscr{G}(u,b)) \Rightarrow \mathscr{N}(0,\sigma^2_{(u,b)})$.

REMARK. In our model example, under stability assumptions, the estimator has a (small) limit bias, and our results can be extended to cover this case under an additional assumption that the limit bias is smaller than $\min_b |f(b^*) - f(b)|$.

3 MIXED OPTIMIZATION VIA EARLY STOPPING

The main idea of our mixed integer/continuous optimization method is as follows. For given b we test a hypothesis $H_0 : f(b) \leq c$ to decide if the resources satisfy the constraint. If the test does not reject H_0 then we decrease b (otherwise we increase b) using a binary search strategy. In either case we keep the state of the system in memory for that value of b, because we will allow for backtracking in case that we made a wrong decision at some point in the search. From this vantage point, the method is a stochastic binary search method. The probability of correct selection (PCS) will be determined by the power of our test and the backtracking rules.

3.1 Stochastic Binary Search

For any given b, assume that the test is of the form $\hat{T}(b) \leq 0$ and that sufficient samples are taken so that $\mathbb{P}(\hat{T}(b) > 0 \,|\, H_0) = \alpha$. Statistics at different values of b are independent. The usual binary search initializes at $l_0 < r_0$ with $f(l_0) > c$, $f(r_0) \leq c$ and the algorithm works iteratively as:

$$\begin{aligned} b_n &= \left\lfloor \frac{l_n + r_n}{2} \right\rfloor \\ l_{n+1} &= l_n \mathbf{1}_{\{\hat{T}(b_n) \leq 0\}} + b_n \mathbf{1}_{\{\hat{T}(b_n) > 0\}} \\ r_{n+1} &= b_n \mathbf{1}_{\{\hat{T}(b_n) \leq 0\}} + r_n \mathbf{1}_{\{\hat{T}(b_n) > 0\}} \end{aligned}$$

Then the probability of error is greater than or equal to α, because a single mistake in the sequential hypothesis testing will suffice to discard the wrong interval and the result will be erroneous. Our backtracking scheme is based on the observation that when an error occurs at level n it is very likely that either l_{n+j} or r_{n+j} will remain fixed, as the true solution value will now be outside the bracket on one side or the other. We do the argument assuming the error is of the form $\hat{T}(b_n) > 0$ but H_0 is true for b_n. Because $b_{n+1} > b_n$ (until termination of the algorithm) it follows that $f(b_{n+1}) \leq f(b_n) < c$. Because $\mathbb{P}(\hat{T}(b_{n+1}) > 0 \,|\, H_0) = \alpha$ then the probability of k consecutive errors is α^k. If there are no more errors after the first one, then $\hat{T}(b_{n+j}) \leq 0$ and in this case, $l_{n+j} = l_n$ until termination, returning $\hat{b}^* = l_n$. Under the assumption that the first error was at level n, this means that $b^* < b_n$.

Backtracking is performed once the algorithm ends. We keep a record of the values $\{(l_n, r_n)\}$. If either boundary has been constant for τ steps prior to termination, then we identify a possible candidate level for first error to backtrack and increase the number of simulations at that point. Backtracking involves more simulations to refine the test, so that now $\mathbb{P}(\hat{T}'(b) > 0 \,|\, H_0) = \alpha' << \alpha$. If this new test swaps the decision with respect to the original one, then the regular binary search resumes with error probability α choosing new points. If the test still supports the old decision, then every time we re-use the same value of b_n we add simulations, decreasing the error probability to α'.

Lemma 1 Suppose that the binary search interval is size 2^{ζ}, and call $\textsc{pcs}(\tau)$ the probability that the selected value of b is within $2^{\tau-1}$ units from b^*. With backtracking, $\textsc{pcs}(\tau) = \mathscr{O}(\alpha^2)$.

Proof. Let Z be the index of the first error, which has a geometric distribution $Z \sim \text{Geom}(\alpha)$. The event that our procedure ends in error will be denoted by E. Using conditioning, we can find a recurrence relation for $x_n(\alpha) = \mathbb{P}(E \,|\, Z = n)$. Given $Z = j$ the procedure will end up in error when backtracking does not identify level j for the first mistake. This happens if there were further errors in the binary decisions from level $j+1$ until level ζ, since the backtracking procedure looks backwards from termination: an error between level j and level ζ will break the streak of observed constant boundary. Because of independence, the distribution of the first error after Z also has a geometric distribution with parameter α so that the probability of wrong backtracking given $Z = j$ is

$$v_j(\alpha) = \alpha \sum_{\ell=j+1}^{\zeta} (1-\alpha)^{\ell}.$$

When backtracking is done correctly to the level $Z = j$ there are two further possible sources of error: either the resampling at j gives again the wrong answer to the test (which happens now with probability $\alpha' << \alpha$) or the branching decision at level j is corrected but there is further error from level $j+1$ until the end of the procedure. In mathematical notation,

$$x_j(\alpha) = v_j(\alpha) + (1 - v_j(\alpha))\alpha' + (1 - v_j(\alpha))(1 - \alpha') \sum_{n=j+1}^{\zeta} x_n(\alpha),$$

with terminal conditions $x_j(\alpha) = 1; j \in \{\zeta, \ldots, \zeta - \tau - 1\}$ and

$$x_{\zeta-\tau}(\alpha) = \nu_{\zeta-\tau}(\alpha) + (1 - \nu_{\zeta-\tau}(\alpha))\alpha' + (1 - \nu_{\zeta-\tau}(\alpha))(1-\alpha')\,\nu_{\zeta-\tau-1}(\alpha),$$

representing the probability of error for last steps of the binary search. By construction,

$$\textsc{pcs}(\tau) = \mathbb{P}(E \,|\, Z \le \zeta - \tau) = \alpha \sum_{j=0}^{\zeta-\tau} x_j(\alpha)(1-\alpha)^{j-1}.$$

Using the recursion, we conclude by induction that $x_n(\alpha) = \mathscr{O}(\alpha)$ for all $n \le \zeta - \tau$, which proves the result. □

The expected number of iterations of this algorithm can also be expressed in terms of a recurrence. While theoretical results are helpful to provide guidance, a problem with this algorithm as stated is that every time that a backtrack is performed it may fail to identify the correct level, so that when optimal subintervals are discarded, many future backtracks may be performed by the algorithm in an attempt to get closer to the optimal value. Instead, what we implement is an algorithm that only backtracks once, however if it detects level j as the candidate for first error, it then chooses a level $j' \in \{0, \ldots, j\}$ to backtrack, with probability $\pi_{j,j'}$. We are currently in the process to find an optimal value of π in order to minimize the number of iterations of the algorithm that ensures a given $\textsc{pcs}(\tau)$.

Stochastic binary problems have been studied by Waeber, Frazier, and Henderson (2011) among others, where a Bayesian approach is used to sample at each query level. The backtracking strategy considered here is similar in approach to the approximate inference strategy shown to be nearly-optimal in Chakraborty, Das, and Magdon-Ismail (2011). The approximate inference strategy uses repeated observations of one direction to update their belief distribution differently than if the observations were more noisily scattered; this is similar to how we treat the case where one endpoint of the search interval has remained fixed as motivation to backtrack due to error (as opposed to a search where both endpoints are changing). The core difference is that the problem setting in Chakraborty, Das, and Magdon-Ismail (2011) uses a Gaussian noise model, which is a stronger assumption on the noise distribution than we have. As a result, we do not yet have any similar results of near-optimality as would be the case for the approximate inference strategy with Gaussian noise.

3.2 Hypothesis Test

Mathematically, the problem (2) is equivalent to our original formulation (1) because they have the same solution. Numerically, however, solving (2) for large complex systems such as the bus service example is inefficient, because most of the simulation time and effort is spent in finding the actual minimum $f(b) = \min_u \mathscr{G}(u,b)$. We remark that the condition used for the hypothesis test at each b is:

$$f(b) \le c \iff \exists u \in I : \mathscr{G}(u,b) \le c$$

so for fixed b, at a given value of u we test the hypothesis $H_0(u) : \mathscr{G}(u,b) \le c$ and if rejected, we build an iterative method to find:

$$\min_{u \in I:\, \mathscr{G}(u,b) > c} \frac{1}{2}(\mathscr{G}(u,b) - c)^2, \tag{4}$$

which will be stopped if $H_0(u)$ is accepted. This is an early termination of target tracking. In this section we omit dependency on b in our notation, because in this phase of the optimization b is fixed. We consider here two scenarios: when estimates of $\nabla_u \mathscr{G}$ are available and when they are not.

Gradient-based method. If gradient estimation is applicable, a gradient search method for (4) is

$$\begin{aligned} u(n+1) &= u(n) + \hat{T}(u(n))\,\gamma \widehat{\nabla_u \mathscr{G}_N}(u(n))^T (\hat{\mathscr{G}}_N(u(n)) - c) \\ \hat{T}(u(n)) &= \mathbf{1}_{\{\hat{\mathscr{G}}_N(u(n)) - q\hat{S}(n) > c\}} \end{aligned} \tag{5}$$

where $q = \Phi(1-\alpha)$ is the $(1-\alpha)$-th quantile of the normal distribution and $\hat{S}(n)$ is an estimate of the standard deviation for $\hat{\mathscr{G}}_N(u(n))$. The parameter $\gamma > 0$ is called the step size. Under our assumptions the piecewise constant interpolation process $u^\gamma(t) = u(\lfloor t/\gamma \rfloor)$ converges in distribution (as $\gamma \to 0$) to the solution $U(t)$ of the (projected) ordinary differential equation

$$\frac{dU(t)}{dt} = \mathbf{1}_{\{f(b)>c\}} \nabla_u \mathscr{G}(U(t),b)^T (\mathscr{G}(U(t) - c))$$

which has a unique zero at the optimum of (4).

Estimation of the variance is a complex problem, because although $\hat{\mathscr{G}}_N$ has approximate normal distribution, the usual sample variance is biased for Markov processes because it does not consider correlations (see (Jones 2004)). For stationary processes Alexopoulos and Seila (2001) propose to use batch means:

$$\hat{S}^2(n) = \frac{1}{W-1} \sum_{i=n-W}^{n} (Y_i - \bar{Y})^2, \tag{6}$$

which is the sample variance for W values of $Y_{n-j} = \hat{\mathscr{G}}_N(u(n-j),b), j = 0,\ldots,W$ obtained from the observation of the process over the "batches" or estimation intervals with N observations of $g(\xi_k(u),b)$, where u is kept constant. In their work (Alexopoulos and Seila 2001) consider constant parameter value u. The following lemma justifies using batch means with changing values of u whenever these follow a (convergent) stochastic approximation procedure, as in our target tracking.

Lemma 2 If $\widehat{\nabla_u \mathscr{G}_N}^T \hat{\mathscr{G}}_N$ has (uniformly) bounded variance then using (6), as $\gamma \to 0$ and $n \to \infty$ with $t = n\gamma$ fixed, $\mathbb{E}(\hat{S}^2(n)) = \mathbb{V}\text{ar}(\hat{\mathscr{G}}_N(U(t)))) + \mathscr{O}(\gamma)$, where this variance refers to the random variable (3) when the initial state has the corresponding stationary distribution $\mu_{U(t)}$.

Proof. From the recursive algorithm it follows that for $j \le W$, $\|u(n-j) - u(n)\| = \gamma Z(n)$ where $Z(n)$ has bounded variance. It follows then by the weak continuity of the transition kernel $P_u(x,dx)$ that for $j \in \{0,\ldots,W\}$ and any value of the underlying state x,

$$\mathbb{E}\left(\frac{1}{N}\sum_{k=1}^{N} g(\xi_k(u(n-j))) \mid \xi_0(u(n-j)) = x\right)^p = \mathbb{E}\left(\frac{1}{N}\sum_{k=1}^{N} g(\xi_k(u(n))) \mid \xi_0(u(n)) = x\right)^p + \mathscr{O}(\gamma),$$

for any power $p \ge 1$. The averages inside the brackets correspond to Y_{n-j} and Y_n respectively.The initial state value for each batch is the final value of the previous one, that is, $\xi_0(u(n-j)) = \xi_N(u(n-j-1))$. As $\gamma \to 0$, it follows from the tightness assumption on the process that the final value $\xi_N(u(n-j-1)$ of the previous batch will have approximately the stationary distribution with parameter value $U(t-(j-1)\gamma)$, for $t = n\gamma$ and under the weak continuity assumption, this can be replaced by the stationary distribution at the limit value $U(t)$ with an error of $\mathscr{O}(\gamma)$. Use now $p = 1$ and $p = 2$ to establish the result. □

Alexopoulos and Kim (2002) present an overview of different rules to choose increasing batch sizes N and window sizes W that can be used to establish asymptotic normality of the statistics

$$\frac{\hat{\mathscr{G}}_N(u(n)) - \mu}{\hat{S}(n)} \approx \mathscr{N}(0,1),$$

where $\mu = \mathscr{G}_N(u(n))$ is the stationary expectation of the estimate. Although we do not sequentially increase N or W, we will assume that N is large enough that the normal approximation is accurate, which will then imply that the terminal value of the algorithm for the given value of b satisfies $\mathbb{P}(\hat{T}(u(n)) > 0 \,|\, f(b) \le c) \lessapprox \alpha$.

Our next scenario deals with the problem when estimates of the gradient are not available. There are two common methods for solving problem (4). One is to replace gradients with stochastic directional derivatives (known as SPSA), and the other is to use generalizations of the Golden search method such as Nelder-Mead simplex methods. To simplify the narrative we will now focus on the one-dimensional case $u \in I \subset \mathbb{R}$.

Golden search method. The deterministic golden search method for finding the minimum of a function $\mathscr{G}(u)$ is illustrated in Figure 2. We briefly describe it here in order to explain the adaptation to the constrained stochastic version. Start with search interval $[a(0), b(0)]$ and let $u_1(0) = a(0), u_2(0) = b(0)$. At iteration n, if $\mathscr{G}(u_1(n)) < \mathscr{G}(u_2(n))$ then:

$$\begin{aligned} b(n+1) = u_2(n); \quad a(n+1) = a(n); \quad u_2(n+1) = u_1(n) \\ u_1(n+1) = b(n+1) - \phi\,(b(n+1) - a(n+1)), \end{aligned} \tag{7}$$

else

$$\begin{aligned} a(n+1) = u_1(n); \quad b(n+1) = b(n); \quad u_1(n+1) = u_2(n); \\ u_2(n+1) = a(n+1) + \phi\,(b(n+1) - a(n+1)), \end{aligned} \tag{8}$$

so that only one new evaluation for $\mathscr{G}(\cdot)$ is required at each iteration and either the subinterval $(u_2(n), b(n)]$ or the subinterval $[a(n), u_1(n))$ is discarded from further search. The algorithm usually terminates when a given tolerance $|u_2(n) - u_1(n)| \le \delta$ is achieved.

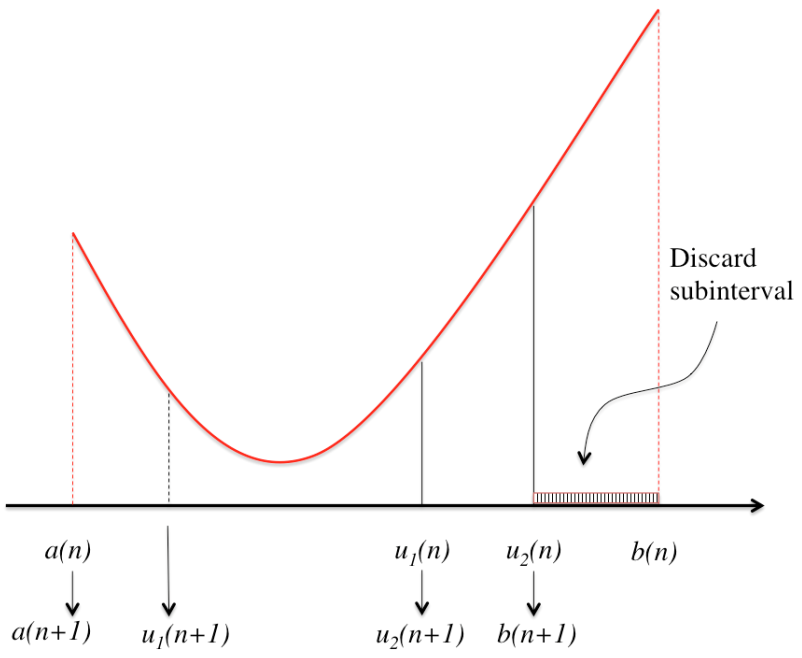


Figure 2: Illustration of one iteration of the golden search method.

The two modifications that we use in our search are (a) the implementation of a statistical comparison, for which we adapt ranking statistics of Goldsman, Kim, Marshall, and Nelson (2002) and (b) early termination for the hypothesis test $f(b) \leq c$. Because our interest is not to find the optimal value with great precision but to move fast in case the constraint is satisfied, we seek coarse precision at confidence level α.

In words, the algorithm calculates the points $u_1(n), u_2(n)$ in the same way as the deterministic version and performs a simulation with N samples at the new point. Then, while the confidence intervals for $\hat{\mathscr{G}}(u_1(n))$ and $\hat{\mathscr{G}}(u_2(n))$ overlap and are larger than a tolerance ε, it chooses which of the two points to sample for another N-long simulation and performs the next iteration. The termination condition here is that either $\hat{\mathscr{G}}(u_i(n)) - q\hat{S}_i \leq c$ for $i \in \{1,2\}$, in which case we accept $H_0 : f(b) \leq c$, or $|u_1(n) - u_2(n)| \leq \delta$, in which case we reject H_0. If further sampling is required at a point $u_i(n)$ then our simulation continues the process rather than starting again. Therefore we need to keep a "state" in memory for the three different values of u that may be kept for the next iteration, together with the corresponding sample sizes.

The description above requires comparison between two values in order to rank them. We now provide the basis for statistical comparison and optimal sampling rules (assuming sequential and not parallel processors are used). To simplify notation, given $u_1(n)$ and $u_2(n)$ we will write σ_1^2, σ_2^2 for the corresponding batch mean variances $\mathbb{V}\text{ar}(\mathscr{G}_N(u_i(n), b))$ using constant batch size N. As before, the estimator of the batch mean $\mathscr{G}_N(\cdot)$ is calculated with K batches of size N, but we will now select the number of batches to optimize the procedure. Specifically,

$$\hat{S}_i^2 = \frac{1}{K-1} \sum_{k=1}^{K} (Y_k - \bar{Y})^2$$

estimates the variance σ_i^2 of the batch means $\hat{\mathscr{G}}_N(\cdot)$ using N observations per batch. Unlike the dynamic algorithm the estimation for each point u uses the cumulative estimator and not just the final batch estimator to approximate the value $\mathscr{G}(\cdot)$, so that $\mathbb{E}(S_i^2/K) \approx \mathbb{V}\text{ar}(\hat{\mathscr{G}}_{NK}(u)) \approx \sigma_i^2$ when K batches are used for the estimation. This follows directly from Assumption (E). Given $u_1 \neq u_2$ use independent simulations to test the Golden search condition $\mathscr{G}(u_1) < \mathscr{G}(u_2)$. Using K_i batches for the estimation at $u_i, i \in \{1,2\}$ the difference satisfies:

$$\mathbb{V}\text{ar}\left(\hat{\mathscr{G}}_{NK_1}(u_1) - \hat{\mathscr{G}}_{NK_2}(u_2)\right) = \frac{\sigma_i^2}{K_1} + \frac{\sigma_2^2}{K_2}.$$

If the confidence intervals for $\mathscr{G}(u_1)$ and $\mathscr{G}(u_2)$ overlap, then further sampling is performed to increase the accuracy in the estimation. When sampling is done sequentially one must choose which one of the two points should be sampled next for one more batch. This decision is made in order to minimize the resulting variance for the difference, that is,

$$\min\left(\frac{\sigma_1^2}{K_1+1} + \frac{\sigma_2^2}{K_2} \quad , \quad \frac{\sigma_1^2}{K_1} + \frac{\sigma_2^2}{K_2+1}\right),$$

which has solution $\arg\min_{i=1,2}(\sigma_i^2/(K_1(K_1+1)))$.

The pseudocode is given below. It initializes as stated for $u_1(0), u_2(0)$ performing N-long independent simulations at these two points. If either estimator satisfies $\hat{\mathscr{G}}_N(u_i(n)) - q\hat{S}_i \leq c$ then terminate. Otherwise perform the following loop, initializing at $K_1 = K_2 = 1$ and $N_1 = N_2 = N$.

```
while $(u_2(n) - u_1(n) > \delta)$ do
    while $\left|\hat{\mathscr{G}}_{N_1}(u_1(n)) - \hat{\mathscr{G}}_{N_2}(u_2(n))\right| < q\left(\frac{\hat{S}_1}{\sqrt{K_1}} + \frac{\hat{S}_2}{\sqrt{K_2}}\right)$ do
        $k = \arg\min_{i=1,2}\left(\frac{\hat{S}_i^2}{K_i(K_i+1)}\right)$
        $K_k = K_k + 1, N_k = N_k + N$ and simulate $N$ further samples at $u_k(n)$.
        if $\hat{\mathscr{G}}_{N_k}(u_k(n)) + q\hat{S}_k/\sqrt{K_k} \leq c$ then EXIT and return $H_0$.
        else if $\max\left(c\frac{\hat{S}_1}{\sqrt{K_1}}, c\frac{\hat{S}_2}{\sqrt{K_2}}\right) < \varepsilon$ then
```

```
            indifference level reached- break **while** loop and continue.
        **end if**
    **end while**
    **if** $\hat{\mathscr{G}}_{n_1}(u_1(n)) < \hat{\mathscr{G}}_{n_2}(u_2(n))$ **then**
        Calculate equations (7) and perform $N$-long simulation at $u_1(n+1)$
    **else**
        Calculate equations (8) and perform $N$-long simulation at $u_2(n+1)$
    **end if**
    $n = n+1$
**end while**
```

4 EXPERIMENTAL RESULTS

We performed first experiments on a simple example where we know the closed form of the curves and corresponding solutions. The test function that we use is of the form

$$\mathscr{G}(u,b) = \mathbb{E}\left[0.01b + \log\left(X(u)+1\right) + \frac{4}{2X(u)+1}\right]. \tag{9}$$

where $X(u) \sim \text{Exp}(u)$ is an exponential random variable with mean $1/u$ (i.e. density function ue^{-ux}). Let u be real-valued in the domain $[.1,1]$ and b be an integer in the domain $[1,128]$. By construction, the function inside the brackets is Lipschitz continuous with respect to u w.p.1, so that the stochastic derivative is unbiased, that is,

$$\frac{d}{du}\mathscr{G}(u,b) = \mathbb{E}\left(\frac{X(u)}{u}\left(\frac{1}{X(u)+1} - \frac{8}{(2X(u)+1)^2}\right)\right).$$

This test function and derivative will be the objective function for a constrained optimization problem in a similar style to the allocation problem, with b standing in for a discrete dimension and u standing in for a continuous control parameter (for reference, the solution to this particular example problem has been determined to be located near $u = .37$ and $b = 76$.):

$$\text{Find largest } b \in [1,128] \cap \mathbb{N} \text{ where } \exists u \in [.1,1] : \mathscr{G}(u,b) < 3$$

We will examine four methods total with this example. On the u dimension, we will compare the golden section approach against gradient descent; on the b dimension, we will compare bisection with backtracking against Bayesian root finding as described in Waeber, Frazier, and Henderson (2011). The Bayesian root finding algorithm used here involves starting with a uniform distribution representing the belief of where the solution value b^* lies in the domain. At each step, the median b' of the distribution is chosen and either golden section or gradient descent is employed to judge if b' will satisfy the constraint for some u; the belief distribution for the location of b^* is updated to reflect that judgment, and a new median is sampled.

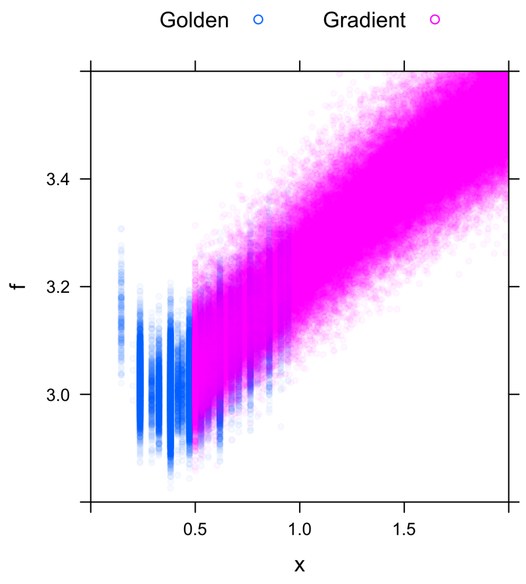
Following 4,000 separate trials for each of the four approaches, the following data was collected:

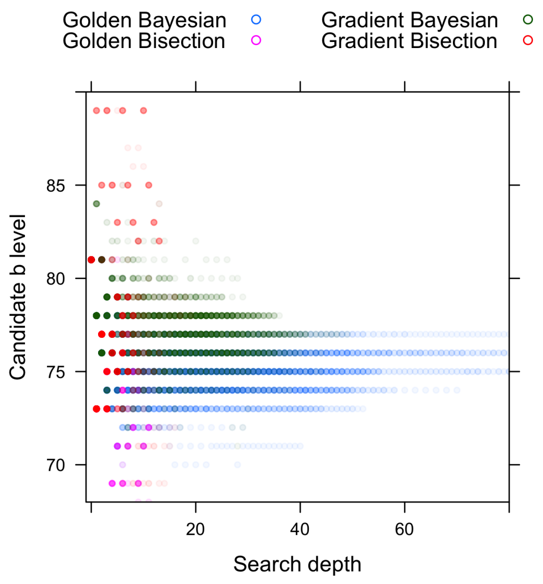
Table 1: Mean squared error, probability of correct selection, and total CPU time.

	MSE	PCS	CPU
Golden/Bayesian	2.092	.270	.172
Gradient/Bayesian	1.855	.298	1.04
Golden/Bisection	5.005	.031	.043
Gradient/Bisection	9.886	.194	.334

The primary difference with the golden section approach is that it imposes a certain structure on where the function evaluation takes place. This means that the computational cost of revisiting different resource levels b for further exploration is greatly decreased, since prior samples can be re-used. In contrast, gradient descent has no such guarantees of being able to re-use information in this manner. The difference is illustrated below in 3(a). Depending on specifics of any given problem, both bisection with backtracking and Bayesian root finding may or may not require frequent returns to previously explored resource levels. In some trials (illustrated in 3(b), the Bayesian search revisited a particular resource level over 80 times.

Between bisection with backtracking and the Bayesian root finding algorithms, it appears as if golden section is particularly well suited to the strategy used in Bayesian root finding. This may be due to the particular choice of test function and hyperparameters; in this particular case, the Bayesian algorithm would frequently request additional observations from previously-observed resource levels, which perfectly leverages the benefits of golden section. One possibility for future exploration is relating the belief levels from the Bayesian posterior to the desired levels of confidence in the hypothesis testing step: early on it will be the case that confidence levels can be relaxed, to be later updated to tighter values once the posterior distribution begins to home in on a few candidate values for the resource level (and where precision starts to grow in importance).





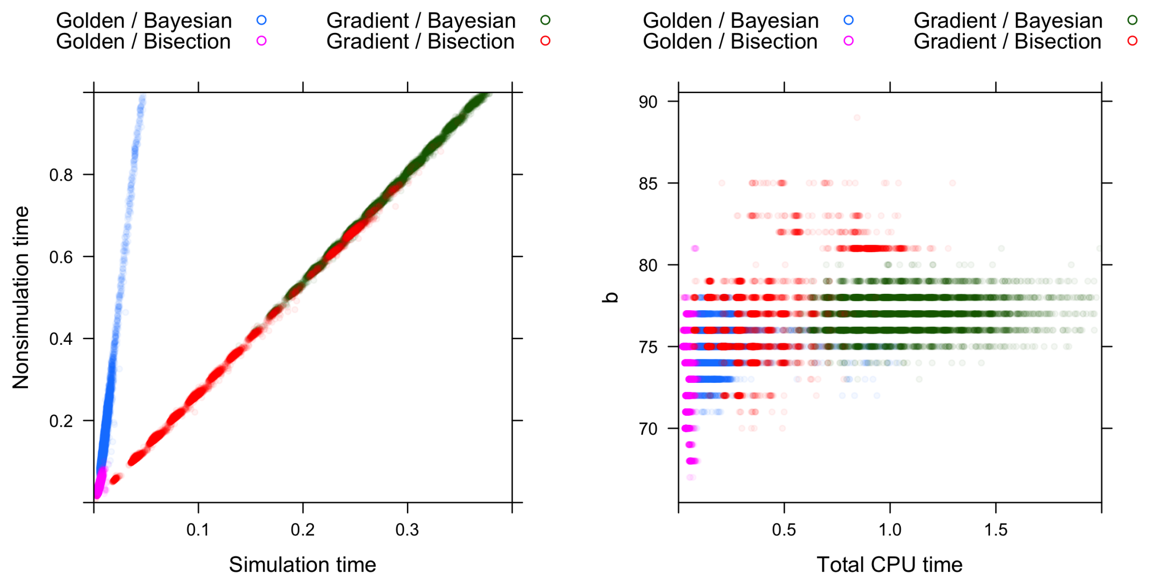
(a) Golden section search evaluates x at specific locations. (b) Bayesian search deepens as candidates are eliminated.

Figure 3: Dynamics of the different search approaches.

In contrast to gradient search (in this case using a known formula for a gradient estimate) the golden section method employs a hypothesis testing framework in its operation. This requires an increased share of computation time on non-simulation tasks as hypotheses checks have to be done at every sample. The division is illustrated in 4(a). This indicates that for problems where simulation is easy that the golden section method may be inappropriate.

In general, the Bayesian algorithm appears to take more CPU time regardless of which algorithm for u was used (see 4(b) for observed time by method); however, it also achieved much better performance overall. Similarly, the gradient descent using the unbiased stochastic derivative above outperforms golden section regardless of which algorithm for b was used. However, it is noteworthy to see that there is a substantial improvement in the golden section algorithm between the bisection algorithm and the Bayesian algorithm: the combination of golden section and Bayesian achieves similar MSE and PCS to its gradient descent counterpart.

The trade-off is that for the golden section hypothesis testing (depending on hyperparameter selection) many more samples may be needed to establish sufficiently high confidence levels to proceed. Algorithms such as gradient descent, or even simpler Robbins-Munro rely instead on properties of the underlying stochastic process, which the golden section approach does not readily appreciate. As a result, significantly more time is spent on a small set of points in the search space. Depending on problem context, this may or may not be an issue. In this particular problem, the difference is made even clearer by the readily available gradient estimator, increasing the performance of the gradient-based algorithm further (in comparison, the golden search algorithm requires the calculation of confidence intervals at each step).



(a) Division of simulation and nonsimulation time between methods.

(b) Division of total computation time and solutions between methods.

Figure 4: Observed computation time.

5 CONCLUDING REMARKS

In this paper we explore a mixed optimization method when (integer-valued) resources are to be allocated to drive the dynamics of a Markov process. Because simulations may take significant CPU time, our focus is to find a mixed procedure with the goal of increasing simulation effort close to the optimal values. We guide this by using “just enough samples” to pass a hypothesis test.

Experimental results show promise with regards to application, especially in contexts where individual simulations may be costly and gradient information is not readily available. The advantages of retaining prior sample information for future use in the course of a search algorithm remain to be explored, and overall the approaches used will be problem-dependent. In particular, dynamic strategies which transition from imprecise, early estimations into progressively more precise estimation could make maximal use of retained state information. Future research will focus on exploring the efficiency of the approaches discussed in this paper, in particular the tradeoff between probability of correct selection within tolerance and computation time.

ACKNOWLEDGMENTS

This work was partially supported by grants PSC CUNY-65653-00 43 and CUNY IRG-21-2153 grant and by the CUNY Institute for Computer Simulation, Stochastic Modeling and Simulation.

REFERENCES

Alexopoulos, C., and S.-H. Kim. 2002. "Output Data Analysis for Simulations". In *Proceedings of the 2002 Winter Simulation Conference*, edited by J. L. Snowdon, J. M. Charnes, E. Yücesan, and C.-H. Chen, 85–96. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Alexopoulos, C., and A. F. Seila. 2001. "Output Data Analysis for Simulations". In *Proceedings of the 2001 Winter Simulation Conference*, edited by B. A. Peters, J. S. Smith, D. J. Medeiros, and M. W. Rohrer, 115–122. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Chakraborty, M., S. Das, and M. Magdon-Ismail. 2011. "Near-Optimal Target Learning With Stochastic Binary Signals". In *Proceedings of the 2011 Conference on Uncertainty in Artificial Intelligence*, edited by F. G. Cozman and A. Pfeffer, 69–76. Corvallis, Oregon: Association for Uncertainty in Artificial Intelligence Press.

Goldsman, D., S. H. Kim, W. S. Marshall, and B. L. Nelson. 2002. "Ranking and Selection for Steady-State Simulation: Procedures and Perspectives". *INFORMS Journal on Computing* 14 (1): 2–19.

Jones, G. L. 2004. "On the Markov Chain Central Limit Theorem". *Probab. Surveys* 1:299–320.

Karp, R. M., and R. Kleinberg. 2007. "Noisy Binary Search and Its Applications". In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, 881–890. Philadelphia, Pennsylvania: Society for Industrial and Applied Mathematics.

Vázquez-Abad, F. 2013. "Ghost Simulation Model for Discrete Event Systems, an Application to a Local Bus Service". In *Proceedings of the 2013 Winter Simulation Conference*, edited by R. Hill, M. Kuhl, R. Pasupathy, S.-H. Kim, and A. Tolk, 655–666. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Waeber, R., P. I. Frazier, and S. G. Henderson. 2011. "A Bayesian Approach to Stochastic Root Finding". In *Proceedings of the 2011 Winter Simulation Conference*, edited by S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. Fu, 4038–4050. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

AUTHOR BIOGRAPHIES

FELISA VÁZQUEZ-ABAD is Professor of Computer Science at Hunter College of the City University New York (CUNY). She is Executive Director of the CUNY Institute for Computer Simulation, Stochastic Modeling and Optimization that she helped to create in 2013. She has a Ph.D. in Applied Mathematics from Brown University. She was a professor at the University of Montreal, Canada in 1993 until 2004 when she became a professor at the University of Melbourne, Australia, until 2009. Her interests focus on the optimization of complex systems under uncertainty, primarily to build efficient self- regulated learning systems. She has applied novel techniques for simulation and optimization in telecommunications, transportation, finance and insurance and she is interested by real life problems. Her email address is felisav@hunter.cuny.edu.

LARRY FENN finished his masters degree in Applied Mathematics at Hunter College CUNY in 2015. He is currently working for *Associated Press* as a data scientist. His interests include live signal processing and visualization. His email address is lf226@hunter.cuny.edu.