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.

GREEN SIMULATION WITH DATABASE MONTE CARLO

Mingbin Feng
Jeremy Staum

Department of Industrial Engineering and Management Sciences
Northwestern University
2145 Sheridan Road
Evanston, IL 60208, USA

ABSTRACT

We develop a green simulation procedure that reuses simulation outputs of the current experiment to improve the computational efficiency of future experiments. We consider practical situations where idle computational resource is available after delivering a simulation answer within given time limit. When used correctly, such idle resource can be valuable simulation investment that can benefit future experiments. In repeated simulations when the same simulation model is run with different inputs at different times, our green simulation procedure repeatedly invests idle computations into databases, which are then used to improve the accuracy of future experiments. Our numerical results show that, as more and more outputs are reused, our proposed estimator has improving accuracy within fixed time limit.

1 INTRODUCTION

In many practical applications, simulations experiments are run repeatedly with different inputs to support routine tasks. For example, in finance and insurance, derivative pricing and risk management decisions may require periodically running the same simulation model under different market conditions and business environments. Feng and Staum (2015) and Feng and Staum (2016) first introduced the *green simulation* paradigm, which reuses simulation outputs to improve computation efficiency of repeated experiments. It is argued that green simulation entails a new perspective on management of simulation experiments and turns the computational cost and expense into investment that provides future benefits. In this article, we develop a new green simulation procedure. Similar to Feng and Staum (2016), the green estimator proposed in this article can achieve arbitrary accuracy as more and more simulation investment is made. The rate of convergence is also studied.

When simulation is used as a decision support tool for routine tasks, its usage often follows a cyclic pattern: Firstly, the current state of the system, such as market price of a stock, is observed and used as the input for the simulation model. Based on the observed input a time-constrained simulation experiment is run so that an answer is delivered within the given time limit, e.g. half a working day. The cycle usually stops here in many cases, as an answer has been delivered for the current task. We observe that the computational resource remains available until the next experiment begins. So we propose adding a *simulation investment* step in the cycle so that such idle resource can be invested to improve the computational efficiency of future experiments. The split of available computations into simulation experiment and simulation investment fits the framework of database Monte Carlo (Borogovac and Vakili 2008). One important component in our green simulation procedure is the recent method of database Monte Carlo for simulation on demand (Rosenbaum and Staum 2015). This method is developed upon database Monte Carlo with control variate (Borogovac and Vakili 2008), which constructs databases of simulation outputs and then use them as quasi-control variates to improve the accuracy of subsequent experiments. Database Monte Carlo for simulation

on demand views the database construction as an one-time computational investment that benefits all future estimation tasks. After the database construction, all subsequent simulation experiments can deliver answers quickly, where the accuracy of the answers is limited by the quality of the given databases. As oppose to one-time database construction in Borogovac and Vakili (2008) and Rosenbaum and Staum (2015), our procedure repeatedly make simulation investments into databases to improve the quality of the databases over time. Consequently, in addition to the quick delivery of simulation answers, the accuracy of our proposed estimator improves over time.

The efficiency of our green simulation procedure depends on how well we can allocate the idle resources. A simulation resource allocation problem is considered to maximize the variance reduction of our green estimator. This allocation problem is similar to those studied in security pricing by simulating stochastic differential equations, such as Duffie and Glynn (1995) and Boyle, Broadie, and Glasserman (1997), among others. The solution of this allocation problem leads to a dynamic allocation policy of the idle computational resources in repeated experiments. The optimal objective value also suggests a possible rate of convergence of our green estimator. Results in our numerical example show that our green estimator has improving accuracy over time within a fixed time limit in each experiment.

2 MATHEMATICAL FRAMEWORK

This section specifies a mathematical framework for developing a green simulation procedure in the remainder of this article. We first describe simulation output as a function of inputs using the concept of random fields then present a setting of repeated experiments where the same simulation model is run repeatedly with changing inputs.

Let $\mathscr{X} \subseteq \mathbb{R}^s$ be the set of all possible inputs, or the *input space*, for a simulation model of interest. Given an input $x \in \mathscr{X}$, or a *point* in the input space, a simulation experiment is run and the simulation output is regarded as a random variable: denote it $F(x)$. In this article we consider the simulation output F as a random field (Staum 2009). Specifically, the simulation output is a measurable function $F : \mathscr{X} \times \Omega \mapsto \mathbb{R}$, where Ω is a probability space with probability measure $\mathbb{P}$ on it. The mean function $\mu : \mathscr{X} \mapsto \mathbb{R}$, which is also known as the response surface, is defined by

$$\mu(x) = \mathbb{E}[F(x)] = \int_\Omega F(x,\omega) d\mathbb{P}(\omega), \quad \forall x \in \mathscr{X}. \tag{1}$$

The covariance function $\sigma : \mathscr{X} \times \mathscr{X} \mapsto \mathbb{R}$ and the correlation function $\rho : \mathscr{X} \times \mathscr{X} \mapsto [-1,1]$ of the random field are given by

$$\sigma(x,x') := \int_\Omega F(x,\omega)F(x',\omega)d\mathbb{P} - \mu(x)\mu(x') \tag{2}$$

and

$$\rho(x,x') := \frac{\sigma(x,x')}{\sqrt{\sigma_x^2 \sigma_{x'}^2}}, \quad \text{where} \quad \sigma_x^2 = \sigma(x,x). \tag{3}$$

As is standard in analyzing Monte Carlo simulations, we assume that $\sigma_x^2 < \infty$ for all $x \in \mathscr{X}$ so that all simulation outputs have finite variance.

Given a point $x \in \mathscr{X}$, the goal is to estimate $\mu(x)$ by a simulation experiment. In some cases, such as rare event simulations, one may estimate $\mu(x)$ by running the simulation model at a different point x'. To avoid ambiguity, we refer to the point at which μ is estimated, say x, as the *prediction point* and the point at which the simulation model is run as the *design point*, whether or not these points coincide. As discussed later, in repeated experiments, the same simulation model may be run at different times, each with different design points and prediction points. A useful feature of the random field description is the probabilistic dependence between the simulation outputs $F(x)$ and $F(x')$ when the simulation is run at different points $x, x' \in \mathscr{X}$. Such dependence represents the effect of common random numbers in simulating the same model at different points. Control variates is a popular variance reduction technique

that improves simulation efficiency by leveraging dependency between random variables. Database Monte Carlo (Borogovac and Vakili 2008) extends control variates to more general settings when the control variate's mean is replaced by its estimate. Database Monte Carlo for Simulation on Demand (Rosenbaum and Staum 2015) articulates the order of implementation of Database Monte Carlo so that an accurate answer can be delivered quickly. We provide a brief review of these methods in Section 3.

We now present a setting of repeated experiment where the same simulation model is run with different prediction points at different times. Suppose that one experiment is run at time n, $n = 1,2,\cdots$; the goal of this nth experiment is to estimate $\mu(X_n)$, where X_n be the nth prediction point. We treat $\{X_n : n = 1,2,\cdots\}$ as a discrete time stochastic process taking values in the input space $\mathscr{X}$. We assume that X_n is observable at time n, but was not observable earlier. Moreover, the prediction points can only be observed but the user of the simulation model cannot control their dynamics.

We now introduce terminologies and notations for databases in repeated experiment setting. As alluded before, in the nth experiment the simulation model may be run at a design point that is different from the nth prediction point. In particular we may run simulation at a design point $\tilde{x}$ and store the outputs in one experiment then reuse these outputs to improve efficiency of future experiments. The storage of such outputs is called the *database* and the point $\tilde{x}$ is called the *database point*; for convenience we refer to a database by its database point. Let $\{\tilde{X}_k : k = 1,2,\cdots\}$ be a database sequence taking values in the input space $\mathscr{X}$. This sequence may be deterministic (e.g., low discrepancy sequence) or stochastic (e.g., a discrete time stochastic process). In later discussions it is desirable to run simulation at the same database at different times so we shall describe how the database sequence is visited. Let $s(k) : \mathbb{N} \mapsto \mathbb{N}$ be a *visit schedule* such that $\tilde{X}_{s(n)}$ is the database point at which the nth experiment is run. The visit schedule may be a deterministic function of time $k \in \mathbb{N}$ (e.g., $s(k) = \lfloor\sqrt{k}\rfloor$) or be chosen at each time n according to some policy as in Section 3.2. For simplicity we assume that $s(1) = 1$ thus the first experiment is run at the first database in the sequence and that $s(n) \leq \max\{s(k) : k < n\} + 1$ so the visit schedule does not "skip" any database in the sequence $\tilde{X}_n$; it can "revisit" any database that has been visited before. The database sequence $\{\tilde{X}_k : k = 1,2,\cdots\}$ and the design schedule $s(k)$ are both modeling choices by the user. Consequently, in contrast to prediction points, user can choose the database point $\tilde{X}_{s(n)}$ for the nth experiment. For example if one choose $\tilde{X}_n = X_n$ and $s(n) = 1$ for $n = 1,2,\cdots$ then all experiments are run at the first prediction point X_1. One contribution of this article is providing insights to choosing good database sequence $\{\tilde{X}_k\}$ and visit schedule $s(k)$ that improves computation efficiency for repeated experiments.

3 GREEN SIMULATION FOR DATABASE MONTE CARLO

In this section we develop a green simulation procedure that reuses simulation outputs to improve computational efficiency in repeated experiments. In each experiment, the proposed procedure not only delivers accurate answers quickly but also invest simulation efforts to improve the efficiency of future experiments. As more simulation investments are made, the answer delivered can be arbitrarily accurate.

We first review Database Monte Carlo for Simulation on Demand and related methods. Given a prediction point $x \in \mathscr{X}$, the *Standard Monte Carlo* (SMC) estimator for $\mu(x)$ is given by

$$\hat{\mu}_r^{SMC}(x) = \frac{1}{r}\sum_{j=1}^{r} F^{(j)}(x) \tag{4}$$

where the outputs $F^{(1)}(x),\cdots,F^{(r)}(x)$ are r independent replications of the simulation model run at point x. The SMC estimator is often used in practice for its simplicity and theoretical justifications such as the Law of Large Numbers and the Central Limit Theorem. The SMC estimator has variance σ_x^2/r but may be reduced by various variance reduction techniques that take advantages of certain properties of the simulation model. For example, control variate (CV) is a variance reduction technique that takes advantages of correlation between random variables. For simplicity of exposition we consider only single CV. Suppose

the $F(x')$ is chosen as a CV for $F(x)$, a *classical CV* estimator of $\mu(x)$ has the form

$$\hat{\mu}_r^{CV}(x;x') = \hat{\mu}_r^{SMC}(x) - \beta\left[\hat{\mu}_r^{SMC}(x') - \mu(x')\right] \tag{5}$$

where:

- The simulation is run for r independent replications: those in $\hat{\mu}_r^{SMC}(x)$ are dependent with those $\hat{\mu}_r^{SMC}(x')$ since they are simulated using common random numbers.
- The CV coefficient β is usually chosen to minimize the variance of $\hat{\mu}_r^{CV}(x;x')$. The optimal coefficient is $\beta^* = \mathbb{C}\text{ov}[F(x),F(x')]/\mathbb{V}\text{ar}[F(x')]$, provided that the required population quantities are available. The resulting estimator has zero bias and minimized variance $(1-(\rho(x,x'))^2)\sigma_x^2/r$, i.e., variance reduction of $1/(1-(\rho(x,x'))^2)$.

The classical CV (5) assumes that the CV expectation $\mu(x')$ is known, which may not be true in all applications. When $\mu(x')$ is not known, studies show that $F(x')$ can still be used as a quasi-CV (Emsermann and Simon 2002) or CV with estimated mean (Pasupathy, Schmeiser, Taaffe, and Wang 2012) by replacing $\mu(x')$ with its estimate. Database Monte Carlo with control variate (DBCV) is one such method that replaces $\mu(x')$ with $\hat{\mu}_R^{SMC}(x')$ for some large R. Mathematically, a DBCV estimator of $\mu(x)$ has the form

$$\hat{\mu}_{r,R}^{DBCV}(x;x') = \hat{\mu}_r^{SMC}(x) - \beta\left[\hat{\mu}_r^{SMC}(x') - \hat{\mu}_R^{SMC}(x')\right]. \tag{6}$$

We see that R independent replications of $F(x')$ is required for estimating $\hat{\mu}_R^{SMC}(x')$ in (6). These R replications constitute a database at x'. DBCV is effective when $R \gg r$, which is assumed in subsequent discussions. In practice the database stores the SMC estimate $\hat{\mu}_R^{SMC}(x')$ instead of the individual replications, nevertheless we say that the database has size R. Due to the use of quasi-CV, the variance-minimizing coefficient β and the resulting minimum variance are different from those of classical CV. Readers are encouraged to find details in Emsermann and Simon (2002), Borogovac and Vakili (2008), and Pasupathy, Schmeiser, Taaffe, and Wang (2012), among others. In this article we consider an approximate variance of $\hat{\mu}_r^{DBCV}(x;x')$ for simplicity in later analysis. Let $\beta = \beta(x;x')$ be the optimal CV coefficient when $F(x')$ is used as a classical CV for $F(x)$, then

$$\mathbb{V}\text{ar}[\hat{\mu}_{r,R}^{DBCV}(x;x')] = \mathbb{V}\text{ar}[\hat{\mu}_r^{CV}(x;x')] + \beta^2\mathbb{V}\text{ar}[\hat{\mu}_R^{SMC}(x')] \tag{7a}$$

$$= \frac{\left(1-(\rho(x,x'))^2\right)}{r}\sigma_x^2 + \frac{\beta^2}{R}\sigma_{x'}^2 \tag{7b}$$

$$\approx \left[\frac{1}{R} + \frac{\left(1-(\rho(x,x'))^2\right)}{r}\right]\sigma_x^2 \tag{7c}$$

where the approximation follows if $F(x)$ and $F(x')$ are similar so that $\sigma_{x'}^2 \approx \sigma_x^2$ and $\beta \approx 1$. Despite being an approximation,(7) showcase a main feature of the minimum variance of DBCV estimator (see Equation (6) in (Rosenbaum and Staum 2015)): it converges to zero as the database's size R grows to infinity and the magnitude of the correlation $\rho(x,x')$ converges to 1.

At the first glance, the DBCV estimator (6) may be computationally expensive because it requires running $R+2r$ replications of the simulation model. Database Monte Carlo for Simulation on Demand (DBSD) is an attractive procedure that uses the DBCV estimator to deliver accurate answers quickly by separating the *construction* of the database from the *estimation* of expected simulation output. A basic version of DBSD can be summarized in the following two steps:

1. *Before* the prediction point x is observed, choose a *database point* x' and run $r+R$ replications at x'(construction of a database).
2. *After* the prediction point x is observed, run r replications at x and deliver $\hat{\mu}^{DBMC}(x)$ in (6) as an estimate for $\mu(x)$ (estimation of $\mu(x)$).

In the above procedure, it takes only r replications between observing x and deliver an estimate of $\mu(x)$. Database construction is done prior to observing the prediction point and hence is not considered part of the estimation task. Moreover, database construction can potentially improve the efficiency of many future experiments and hence is one form of *simulation investment*. The computation patterns in the above DBSD procedure is applicable in many practical settings: usually there is a time limit (e.g., during working hours) between observing the input of a simulation and delivering an answer via simulation, but computational resources may be available after an answer is delivered (e.g., after working hours), when one can make simulation investments that can benefit future experiments.

3.1 Basic Green Database Monte Carlo with Control Variate

In the original proposal of DBSD, database construction is a one-time investment, i.e., the locations and sizes of databases remain unchanged once constructed. Consequently the DBCV estimator is conditionally biased given constructed databases and the variance reduction it provides is limited by the quality of the available databases. In the context of repeated experiment, we propose a green simulation procedure in which simulation investments are made repeatedly: new databases maybe constructed and existing databases may grow in size over time. At each time n, we suppose that the simulation model can run a total of $r+R$ replications: an answer must be delivered after r replications while the remaining R more replications, the simulation investment, can be run before next experiment begins. For clarity, we consider two types of simulation investment: *augmentation*, which adds more replications to an existing database and *initiation*, which runs simulation at a new database point and constructs a new database. We propose the following *Green Database Monte Carlo* (GreenDB) procedure for the nth experiment:

1. Suppose at the beginning of time n there are k_n databases $\{\tilde{X}_1, \cdots \tilde{X}_{k_n}\}$ with sizes $R_1+r, \cdots, R_{k_n}+r$; all these databases have r replications that were run using common random numbers.
2. After prediction point X_n is revealed, choose one database, say $\tilde{X}_{\tilde{k}_n}$, as a quasi-CV.
3. Run r replications at X_n using the same common random numbers as those in the databases.
4. Deliver
$$\hat{\mu}_r^{GreenDB}\left(X_n; \tilde{X}_{\tilde{k}_n}\right) = \hat{\mu}_{r,R_{k_n^*}}^{DBCV}\left(X_n; \tilde{X}_{\tilde{k}_n}\right) \tag{8}$$
as an estimate of $\mu(X_n)$, where β_n is chosen to minimize the variance of $\hat{\mu}_r^{GreenDB}\left(X_n; \tilde{X}_{\tilde{k}_n}\right)$. Estimation of β_n is discussed in Section 3.2.
5. After delivering the answer, run R replications at $\tilde{X}_{s(n)}$ based on the database sequence $\{\tilde{X}_n\}$ and visit schedule $s(n)$. This maybe an augmentation to an existing database if $s(n) \in \{1, \cdots, k_n\}$ or initiation of a new database at $\tilde{X}_{s(n)}$ otherwise.

Similar to DBSD, GreenDB delivers accurate answers quickly after a prediction point is revealed. In contrast to the one-time investment in DBSD, GreenDB makes simulation investment repeatedly in repeated experiments. As more investments are made, either existing databases grow in size or more databases are constructed, or both. As a result, GreenDB is superior than DBSD in repeated experiments.

3.2 Designing Practical Green Database Monte Carlo Procedures

Under some conditions, a well-designed GreenDB procedure can produce converging estimators similar to those in Feng and Staum (2015), which can achieve arbitrary accuracy as more and more simulation investment is made. We will supply more details to the above basic GreenDB procedure and consider designing an efficient procedure. In particular, we will specify: Construction of the initial database; Selection of quasi-CV among existing databases; Estimation of the CV coefficient $\hat{\beta}$. Lastly we consider a simulation investment allocation problem whose solution provides useful insights in choosing database sequence $\{\tilde{X}_k\}$ and visit schedule $s(k)$.

For the first database, consider the first experiment with prediction point X_1, when there is no preexisting database. Since an answer is required within r replications of the simulation model, without any other variance technique (e.g. importance sampling) one should simply run r replications at X_1 and estimate $\mu(X_1)$ by $\hat{\mu}_r^{SMC}(X_1)$. Given a database sequence $\{\tilde{X}_k\}$ and visit schedule $s(k)$, the remaining R replications of simulation investment should be run at $\tilde{X}_{s(1)} = \tilde{X}_1$. When a database is first initiated, its size is R. Note that r of these R replications should serve as common random numbers for estimations at future prediction points. The database initiated at time 1 may be a quasi-CV for future experiments. However, as discussed later, a new database may not always be candidate as quasi-CV immediately after initiation.

While selection of quasi-CV among existing databases and estimation of $\hat{\beta}$ are not our focuses, we review some theoretical criteria and practical suggestions for these tasks. Theoretically, in classical CV one should select a database $\tilde{X}_{\tilde{k}_n}$ such that $|\rho(X_n, \tilde{X}_{\tilde{k}_n})|$ is maximized to achieve maximum variance reduction. However, in general the correlation function $\rho(x,x')$ for a given simulation model may be prohibitively complicated, if possible at all, to optimize. In many practical applications, outputs of the same simulation model are highly correlated when run at similar inputs. Then, as a common practical alternative, one may choose the "nearest neighbor" database of the prediction point

$$\tilde{X}_{\tilde{k}_n} = \underset{\tilde{X}_k \in \{\tilde{X}_1, \cdots, \tilde{X}_{k_n}\}}{\arg\min} \{\|\tilde{X}_k - X_n\|_2\}. \tag{9}$$

For example, the above rule is used in the numerical examples of Rosenbaum and Staum (2015).

Estimation of $\hat{\beta}$ in DBCV has been studied by Pasupathy, Schmeiser, Taaffe, and Wang (2012) that accounts for the use of estimated mean and Avramidis and Wilson (1993) that eliminates the bias, among others. We did not consider applying those techniques to GreenDB in this article, it remains for future research. As suggested by Rosenbaum and Staum (2015), we estimate $\hat{\beta}$ by linear regression of $F^{(1)}(X_n), \cdots, F^{(r)}(X_n)$ on $F^{(1)}(\tilde{X}_{\tilde{k}_n}), \cdots, F^{(r)}(\tilde{X}_{\tilde{k}_n})$.

The database sequence $\{\tilde{X}_k\}$ and the visit schedule $s(k)$ are the two key factors in designing an efficient GreenDB procedure. In particular, we are interested in developing GreenDB estimator such that $\mathbb{V}\text{ar}[\hat{\mu}_r^{GreenDB}\left(X_n; \tilde{X}_{\tilde{k}_n}\right)] \to 0$ as $n \to \infty$. We will consider some requirements for $\{\tilde{X}_k\}$ and $s(k)$ so that the above convergence is fast. Assume that $\rho(x,x') \to 1$ as $\|x - x'\|_2 \to 0$ for all $x, x' \in \mathscr{X}$. Based on (7) we have $\mathbb{V}\text{ar}[\hat{\mu}_r^{GreenDB}\left(X_n; \tilde{X}_{\tilde{k}_n}\right)] \to 0$ as $n \to \infty$ if

(c1) $\|X_n - \tilde{X}_{\tilde{k}_n}\|_2 \to 0$ as $n \to \infty$, and
(c2) $R_{k_n^*} \to \infty$ as $n \to \infty$.

Define the nth *dispersion* (Niederreiter 1992) of the database sequence $\{\tilde{X}_k\}$ by

$$d_n := \sup_{x \in \mathscr{X}} \min_{1 \le k \le n} \|x - \tilde{X}_k\|.$$

Then we have $\|X_n - \tilde{X}_{\tilde{k}_n}\|_2 \le d_n$ for all $n = 1, 2, \cdots$. Therefore a sufficient condition for (c1) is $d_n \to 0$ as $n \to \infty$, which implies that the database sequence is space-filling and more and more databases are initiated as time progresses. Based on studies Quasi-Monte Carlo (QMC), we propose using low-discrepancy sequence in $\mathscr{X}$ for $\{\tilde{X}_n : n = 1, 2, \cdots\}$. Since a fixed simulation investment of R replications is made to database construction in each experiment. The condition (c2) requires that $s(k)$ revisits all databases infinitely many times as time progresses. We now solve a simulation investment allocation problem whose optimal solution leads to a visit schedule that satisfies the above requirements and fast convergence of the resulting GreenDB estimator. Let $C_n = nR$ be the total simulation investment budget for database construction by the end of the nth experiment. For simplicity, we assume that the user can decide the number of databases k_n and all databases have the same size R_n such that $k_n \cdot R_n = C_n$. In future research, we may investigate more sophisticated resource allocation schemes such that databases have different sizes in different regions of the input space. The goal is to find an optimal allocation so as to maximize variance reduction in (7). Then

a visit schedule $s(k)$ can be formulated using a dynamic policy that approximately matches the optimal solution k_n^* and R_n^* at each time n.

Assume that $1-(\rho(x,x'))^2 \le c_1\|x-x'\|_2^\alpha$ for some $c_1,\alpha \in \mathbb{R}_+$ for all $x,x' \in \mathscr{X}$. It follows from Niederreiter (1992) that the dispersion of some s-dimensional low-discrepancy sequence (Solbol and Halton sequences, for example) is bounded by $d_n \le c_2 k_n^{-1/s}$ for some $c_2 \in \mathbb{R}_+$. Then we have $1-(\rho(X_n,\tilde{X}_{k_n^*}))^2 \le c_1\|x-x'\|_2^\alpha \le d_n \le cN_n^{-\alpha/s}$ where $c = c_1 \cdot c_2^\alpha$. Consequently the simulation investment allocation problem is given by

$$\begin{aligned} \min_{k_n,R_n\in\mathbb{R}_+} \quad & \tfrac{1}{R_n} + \tfrac{c}{r}k_n^{-\alpha/s} \\ \text{s.t.} \quad & R_n k_n = C_n \end{aligned} \tag{10}$$

Substituting $R_n = C_n/k_n = (nR)/k_n$ into the objective then we may minimize the scaler-valued function $f(k_n) = \frac{k_n}{(nR)} + \frac{c}{r}k_n^{-\alpha/s}$. Clearly f is a convex function therefore its minimizer is given by

$$\frac{df(k_n)}{dk_n} = 0 \Rightarrow k_n^* = (nR)^{\frac{s}{\alpha+s}}\left(\frac{c\cdot\alpha}{r\cdot s}\right)^{\frac{s}{\alpha+s}}, \quad R_n^* = (nR)^{\frac{\alpha}{\alpha+s}}\left(\frac{r\cdot s}{c\cdot\alpha}\right)^{\frac{s}{\alpha+s}}. \tag{11}$$

The optimal solution (11) shows that:

1. For any $\alpha,c_1,c_2,s>0$ and for any $r,R>0$, $k_n^*\to\infty$ and $R_n^*\to\infty$ as $n\to\infty$. In other words, as more simulations are done and more simulation investments are made, both the number and the size of databases increase indefinitely. As shown in the previous discussions, these are sufficient conditions for $\mathbb{V}\mathrm{ar}[\hat{\mu}_r^{GreenDB}\left(X_n;\tilde{X}_{\tilde{k}_n}\right)] \to 0$ as $n\to\infty$. In other words, the GreenDB estimator can be arbitrarily accurate as more and more simulation investment is made.
2. For large n and fixed r,R,c_1, and c_2, the k_n^* decreases with α and R_n^* increases with α. This suggest that for larger α one should construct less databases that are sparser but with larger size. This is because for larger α the prediction point and the chosen quasi-CV may be further apart but still achieve a particular correlation level, which is bound by the dispersion of the database sequence in the allocation problem. Note that α measures the correlation of simulation outputs associated with two similar inputs: the larger the α the higher the correlation.
3. The optimal objective, and hence the variance reduction, is of order $\mathscr{O}(n^{-\alpha/(\alpha+s)})$. This suggests that, if $\sup\{\sigma_x^2 : x\in\mathscr{X}\}<\infty$, then the variance of our green estimator converges at rate $\mathscr{O}(n^{-\alpha/(\alpha+s)})$.

Exact implementation of optimal solution (11) suggests that, in the $(n+1)$th repeated experiment, invest $(R_{n+1}^* - R_{n^*})$ replications in the k_n^* existing databases and R_{n+1}^* replications in $k_{n+1}^* - k_n^*$ new databases, which raises practical difficulties: The optimal solution may not be integral; More importantly investing in multiple databases in each experiment may not be desirable. Instead, we suggest the following dynamic policy for the visit schedule Let $\{\tilde{X}_1,\cdots,\tilde{X}_{k_n}\}$ be the existing databases time n and let $R_1+r,\cdots,R_{k_n}+r$ be their sizes. The design schedule $s(n)$ at time n satisfies:

1. If $k_n < k_n^*$ where k_n^* is the optimal solution in (11), then $s(n) = k_n+1$ so a new database is initiated with size R.
2. Otherwise, if $k_n \ge k_n^*$, then $s(n) = \min\{k : R_k = \min\{R_{k'} : k' = 1,\cdots,k_n\}\}$ so one existing database with minimum size is augmented (break ties in order of initiations).

According to the above policy, simulation investment is made to one database in each experiment, which simplifies the implementation of the GreenDB procedure. By keep adding investment into the database with the minimum size, the above policy aims to balance the size in all existing database. One drawback of the above policy is that any newly initiated database has size R and only increments by at most R in each subsequent experiment. When n is large, new database may be significantly smaller than other existing databases, which results in much worse variance reduction. As a remedy, we suggest that a database

can only be a candidate for quasi-CV if its size is no more than R less than that of the first database. This additional safeguard ensures improving quality of the quasi-CV and fast convergence of the resulting GreenDB estimator. We will examine the practical performance of the GreenDB procedure by numerical examples in Section 4.

4 Illustration

To illustrate the essences of our procedure, we consider a repeated experiment that estimates the probability of a random variable exceeds a given threshold, where the random variable in question changes in each experiment. The simulation model in this experiment is simple so that the mean and covariance function of the corresponding random field can be derived analytically, which enables us to closely examine the practical performance of the proposed GreenDB procedure.

Let $\mathscr{X} = [x_{min}, x_{max}] \subset \mathbb{R}$ be the input space and $\omega \sim \text{Unif}(0,1)$ be the underlying source of randomness for a given simulation model F. For given input $x \in \mathscr{X}$ and realization ω, the simulation output is given by $F(x,\omega) = \mathbf{1}\{x \cdot \omega > \gamma\}$ where $\mathbf{1}$ is the indicator function and $\gamma \in \mathbb{R}$.

Clearly this is a 1-dimensional problem and therefore $s = 1$. For any $x, x' \in [x_{min}, x_{max}]$, the mean function for the above simulation model is given by

$$\mu(x) = Pr\left(U > \frac{\gamma}{x}\right) = 1 - \frac{\gamma}{x}, \quad \forall x \in [x_{min}, x_{max}]$$

and the covariance function is given by

$$\begin{aligned}
\sigma(x,x') &= E[F(x)\cdot F(x')] - E[F(x)]\cdot E[F(x')] \\
&= P\left(U > \tfrac{\gamma}{\min\{x,x'\}}\right) - P\left(U > \tfrac{\gamma}{x}\right)\cdot P\left(U > \tfrac{\gamma}{x'}\right) \\
&= \left(1 - \tfrac{\gamma}{\min\{x,x'\}}\right) - \left(1 - \tfrac{\gamma}{x}\right)\cdot\left(1 - \tfrac{\gamma}{x'}\right) \\
&= \left(\tfrac{\min\{x,x'\}-\gamma}{\min\{x,x'\}}\right)\cdot\left(\tfrac{\gamma}{\max\{x,x'\}}\right).
\end{aligned}$$

Then one can show that for any $x, x' \in [x_{min}, x_{max}]$

$$\begin{aligned}
(\rho(x,x'))^2 &= \frac{\sigma(x,x')}{\sqrt{\sigma_x^2\sigma_{x'}^2}} = \frac{\left(\frac{\min\{x,x'\}-\gamma}{\min\{x,x'\}}\right)^2\cdot\left(\frac{\gamma}{\max\{x,x'\}}\right)^2}{\left(\frac{\min\{x,x'\}-\gamma}{\min\{x,x'\}}\right)\cdot\left(\frac{\gamma}{\min\{x,x'\}}\right)\cdot\left(\frac{\max\{x,x'\}-\gamma}{\max\{x,x'\}}\right)\cdot\left(\frac{\gamma}{\max\{x,x'\}}\right)} \\
&= \frac{\min\{x,x'\}-\gamma}{\max\{x,x'\}-\gamma} = 1 - \frac{\max\{x,x'\}-\min\{x,x'\}}{\max\{x,x'\}-\gamma}
\end{aligned}$$

so $1-(\rho(x,x'))^2 \le c_1|x-x'|^\alpha$ for $c_1 = (x_{min}-\gamma)^{-1}$ and $\alpha = 1$. The database sequence is chosen as $\tilde{X}_n = x_{min} + (x_{max} - x_{min})h_n$ for all $n = 1,2,\cdots$ where $h_1, h_2, \cdots$ is the Halton sequence. This means the nth dispersion of the database sequence is bounded by $d_n \le 2(x_{max} - x_{min})/n$; so $c_2 = 2(x_{max} - x_{min})$. Moreover, the optimal control variate coefficient is given by

$$\beta^* = \frac{\sigma(x,x')}{\sigma_{x'}^2} = \begin{cases} \frac{x'\cdot(x-\gamma)}{x\cdot(x'-\gamma)} & \text{if} \quad x' \ge x \\ \frac{x'}{x} & \text{if} \quad x' < x \end{cases}$$

Note that for general simulation models the mean and covariance functions are too complicated to be derived analytically, so the parameters c_1, c_2, α, and β need to be estimated. This remains for future research.

Given prediction point X_n at time n, the goal of the nth simulation experiment is to estimate

$$\mu(X_n) = \mathrm{E}[F(X_n)] = Pr(X_n \cdot \omega > \gamma), \text{ where } \omega \sim \text{Unif}(0,1). \tag{12}$$

For simplicity, the prediction points are $X_n \sim \text{Unif}(x_{min}, x_{max})$, which forms a trivial stationary stochastic process with stationary distribution $X_n \sim \text{Unif}(x_{min}, x_{max})$. We suppose that $r = 200$ and $R = 1000$ to emulate practical situations (4 hours to perform estimation task and 20 hours for simulation investment). General features of the GreenDB procedure, as discussed below, remains the same for wide range of parameters $0 < \gamma < x_{min} < x_{max} < \infty$. For illustration purpose we choose $x_{min} = 10$, $x_{max} = 12$, and $\gamma = 8$ and so $k_n^* = \sqrt{10n}$.

To investigate the effectiveness of the GreenDB procedure, we performed a sequence of 1,000 experiments, each with $r = 200$ replications for estimation and $R = 1000$ for simulation investment. Using the same sample path $\{X_n : n = 1, 2, \cdots, 1000\}$, we evaluated two estimators for $\mu(X_n)$: $\mu_r^{SMC}(X_n)$ and $\mu_r^{GreenDB}(X_n; \tilde{X}_{\tilde{k}_n})$. To accurately estimate the unconditional variance, we performed such a sequence of experiments 10,000 times. These 10,000 macro-replications of the sequence of experiments have independent sample paths and simulation output. The estimated variance of an estimator $\hat{\mu}(X_n)$ of $\mu(X_n)$ was $\sum_{i=1}^{10,000}(\hat{\mu}^{(i)}(X_n^{(i)}) - \mu(X_n^{(i)}))/10,000$, where $\hat{\mu}(X_n^{(i)})$ is the value of the estimator in the nth experiment on the ith macro-replication. Due to using $10,000$ macro-replications, the standard errors of these estimated variances are less than 0.01% of the corresponding estimated variances (error bars are too narrow to show clearly in the figure).

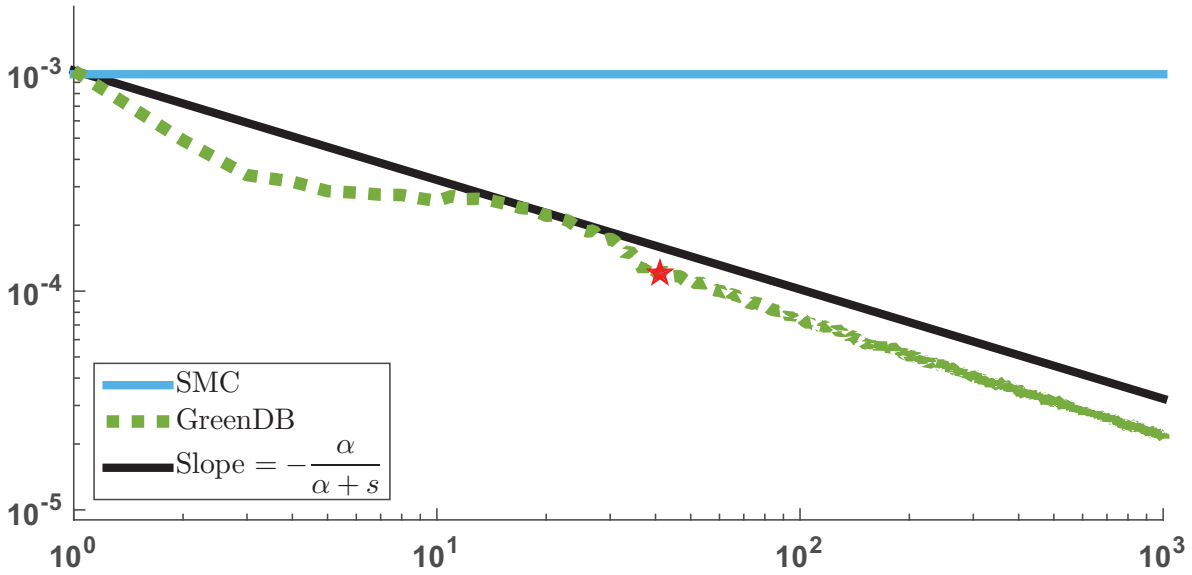

Figure 1: Log-log plot of estimated variances for Standard Monte Carlo and Green Simulation estimators for random prediction points X_n.

Figure 1 is a log-log plot of the variances of two estimators for each experiment $n = 1, 2 \cdots, 1,000$. The horizontal solid blue line is the variance of SMC estimator with r replications, and the dotted green line is the variance of our GreenDB estimator. The SMC variance forms a horizontal line because all experiments are identical the simulation model is the same and all prediction points have the same distribution. We first compare the dotted green line against the horizontal line to examine the effectiveness of our green simulation procedure. For the first experiment ($n = 1$), there is no database constructed from a previous experiment, so the GreenDB coincides with the SMC estimator. For all later experiments ($n \geq 2$), when databases are available, the GreenDB variance is less than that for the SMC variance. It is clear that the variance of the GreenDB estimator decreases over time while the SMC estimators' variance remain the same. In the 1,000th experiment, the GreenDB estimator's variance is about 50 times smaller than the

SMC estimator's variance, or equivalently a variance reduction of 50. This high accuracy for the GreenDB is achieved with the same estimation time as that for the SMC estimator, which showcases the advantage of our green simulation procedure.

A black solid line with slope $-\frac{\alpha}{\alpha+s}$ and the same intercept as the solid blue line is plotted in Figure 1 for reference. This slope is expressed in log-log scale, which indicates a convergence rate of $\mathcal{O}(n^{-\alpha/(\alpha+s)})$ for the variance of an estimator that has such slope in the figure. This is the convergence rate of the optimal objective for the simulation investment allocation problem (10). By comparing the right part of the green line (say $n \geq 100$) with the black line, it seems the GreenDB variance converges at rate $\mathcal{O}(n^{-\alpha/(\alpha+s)})$. We elaborate this point in the next paragraph.

A red star in Figure 1 splits the dotted green line into two segments that are visually different: the left segment forms a curve and the left segment forms a line. On one hand, the left segment is the warm up period in our procedure where the number and size of the databases constructed by our procedure are significantly different from the optimal solution (11). In addition, in this warm up period there are few databases so the prediction point and its closest database could be significantly different, rendering (7) an unreliable approximation. In this case the actual variance reduction is hard to predict by the optimal objective of (10). On the other hand, after warming up the right segment forms a straight line whose slope is close to the predicted value of $-\alpha/(\alpha+s)$, when comparing to the solid black line.

ACKNOWLEDGMENTS

The authors would like to thank Imry Rosenbaum for his valuable comments and suggestions.

REFERENCES

Avramidis, A. N., and J. R. Wilson. 1993. "A Splitting Scheme for Control Variates". *Operations Research Letters* 14 (4): 187–198.

Borogovac, T., and P. Vakili. 2008. "Control Variate Technique: A Constructive Approach". In *Proceedings of the 2008 Winter Simulation Conference*, 320–327. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Boyle, P., M. Broadie, and P. Glasserman. 1997. "Monte Carlo methods for security pricing". *Journal of economic dynamics and control* 21 (8): 1267–1321.

Duffie, D., and P. Glynn. 1995. "Efficient Monte Carlo simulation of security prices". *The Annals of Applied Probability*:897–905.

Emsermann, M., and B. Simon. 2002. "Improving Simulation Efficiency with Quasi Control Variates". *Stochastic Models* 18 (3): 425–488.

Feng, M., and J. Staum. 2015. "Green Simulation Designs for Repeated Experiments". In *Proceedings of the 2015 Winter Simulation Conference*, edited by L. Yilmaz, W. K. V. Chan, I.-C. Moon, and T. M. K. Roeder, 403–413. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Feng, M., and J. Staum. 2016. "Green Simulation: Reusing the Output of Repeated Experiments". Working paper. Last accessed on March 20th, 2016.

Niederreiter, H. 1992. *Random Number Generation and Quasi-Monte Carlo Methods*, Volume 63. Society for Industrial and Applied Mathematics.

Pasupathy, R., B. W. Schmeiser, M. R. Taaffe, and J. Wang. 2012. "Control-Variate Estimation using Estimated Control Means". *IIE Transactions* 44 (5): 381–385.

Rosenbaum, I., and J. Staum. 2015. "Database Monte Carlo for Simulation on Demand". In *Proceedings of the 2015 Winter Simulation Conference*, edited by L. Yilmaz, W. K. V. Chan, I.-C. Moon, and T. M. K. Roeder, 679–688. Piscataway, NJ, USA: Institute of Electrical and Electronics Engineers, Inc.

Staum, J. 2009. "Better Simulation Metamodeling: The Why, What, and How of Stochastic Kriging". In *Proceedings of the 2009 Winter Simulation Conference*, 119–133. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

AUTHOR BIOGRAPHIES

Mingbin Feng is a Ph.D. candidate in the Department of Industrial Engineering and Management Sciences at Northwestern University. He is an Associate of the Society of Actuaries (ASA). His research interest lies in the intersection between efficient simulation designs and nonlinear optimization, with focuses on financial and insurance applications. His e-mail address is benfeng@u.northwestern.edu. His website is http://users.iems.northwestern.edu/~mbfeng/.

JEREMY STAUM is an Associate Professor of Industrial Engineering and Management Sciences at Northwestern University. He coordinated the Risk Analysis track of the 2007 and 2011 Winter Simulation Conferences. His research include simulation metamodeling and optimization via simulation. His website is http://users.iems.northwestern.edu/~staum.