Proceedings of the 2012 Winter Simulation Conference
C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. M. Uhrmacher, eds.

CONSTRUCTING ADAPTED LATTICE RULES USING PROBLEM-DEPENDENT CRITERIA

Pierre L'Ecuyer
David Munger

DIRO, Université de Montreal
C.P. 6128, Succ. Centre-Ville
Montréal (Québec), H3C 3J7, CANADA

ABSTRACT

We describe a new software tool named Lattice Builder, designed to construct lattice point sets for quasi-Monte Carlo integration via randomly-shifted lattice rules. This tool permits one to search for good lattice parameters in terms of various uniformity criteria, for an arbitrary number of points and arbitrary dimension. It also constructs lattices that are extensible in the number of points and in the dimension. A numerical illustration is given.

1 INTRODUCTION

Estimating the expectation of a random variable using Monte Carlo (MC) simulation is achieved by averaging independent realizations of that variable. Gaining a single significant digit in the MC estimate requires increasing the number of independent realizations, and thus the simulation effort, by a factor of 100. This factor can be considerably reduced by using randomized quasi-Monte Carlo (RQMC) simulation, which generates realizations of the random variable that are not independent, in contrast to MC, but that sample the probability space more evenly. In practice, with MC simulation, each realization of the random variable is produced using a certain number (the dimension of the problem) of independent (pseudo)random numbers uniformly distributed in the unit interval, or, equivalently, a point in the unit hypercube of the dimension of the problem. With RQMC simulation, the independent MC points are replaced with points that are structured to provide a better coverage of the unit hypercube, but that are uniformly distributed in that region when taken separately.

Randomly-shifted lattice rules are one prominent class of RQMC methods, in which the points in the unit hypercube are organized with a lattice structure dictated by a set of free defining parameters. For a given problem, using a fixed number of points, the amount of variance reduction provided by a randomly-shifted lattice rule depends on the choice of these parameters. The conventional approach is to consider a set of candidate lattice rules (each defined by a distinct set of parameters) and to select the one that get the best score in terms of a given uniformity criterion, that measures the quality of a lattice. Such criteria are, in turn, usually defined in terms of a certain number of free parameters called weights, and, for that reason, we call them weighted figures of merit. In practice, the choice of criterion and of its weights should be adapted to the model that needs to be simulated.

In this paper, we introduce Lattice Builder, a software tool whose purpose is to find good lattice rules according to a variety of uniformity criteria, and using different choices of construction methods. It allows practitioners to find good lattice rules adapted to their problems, for any finite dimension and number of points, at the moment they are needed. It can be used as an external program or as a software library. This contrasts with resorting to tables of parameters that can be found on websites or in published papers, where the criteria used are not necessarily adapted to one's problem, and provide less flexibility in the number of points or dimension.

978-1-4673-4781-5/12/$31.00 ©2012 IEEE

The paper is organized as follows. In Section 2, we formalize the above discussion and we present Lattice Builder. In Section 3, we illustrate how it can be used for the simulation of the payoff of financial options based on the Heston volatility model.

2 CHOOSING GOOD LATTICES RULES WITH LATTICE BUILDER

2.1 Lattice Rules

The context is the estimation of the expectation $\mu = \mathbb{E}[X]$ of a real-valued random variable X. The MC simulation of one realization of X is generally performed by drawing (pseudo)random numbers uniformly distributed in the interval $(0,1)$ and by using these to compute the value of X. Let s denote the number of these numbers required to produce one realization of X, and let us collect them into a random vector $\mathbf{U}$ uniformly distributed in $(0,1)^s$, such that $X = f(\mathbf{U})$ for some $f : (0,1)^s \to \mathbb{R}$. The n-point MC estimator for $\mu = \mathbb{E}[f(\mathbf{U})]$ is

$$\hat{\mu} = \frac{1}{n}\sum_{i=0}^{n-1} f(\mathbf{U}_i), \tag{1}$$

where $\mathbf{U}_0, \ldots, \mathbf{U}_{n-1}$ are independent realizations of $\mathbf{U}$. The RQMC estimator for μ using a randomly-shifted lattice rule has the same form as (1), but with $\mathbf{U}_0, \ldots, \mathbf{U}_{n-1}$ taken from a randomly shifted rank-1 lattice point set

$$Q_n = \{\mathbf{U}_i = (\mathbf{u}_i + \mathbf{U}) \bmod 1 \,:\, \mathbf{u}_i \in Q_n^0\},$$

where the modulo is applied coordinate-wise, $\mathbf{U}$ is called here the random shift, and where

$$Q_n^0 = \{\mathbf{u}_i = (i\mathbf{a} \bmod n)/n \,:\, i = 0, \ldots, n-1\},$$

is the deterministic lattice point set specified by the *generating vector* $\mathbf{a} = (a_1, \ldots, a_s) \in \mathbb{Z}^s$. In that case, (1) is called a *randomly-shifted rank-1 lattice rule*. For fixed n and f, the variance of $\hat{\mu}$ depends on the choice of $\mathbf{a}$ and can be significantly reduced with respect to MC if $\mathbf{a}$ is well chosen.

Lattice Builder also supports the construction of extensible lattice rules. In that case, we can consider a sequence of lattice point sets $Q_{b^0}^0 \subset \cdots \subset Q_{b^p}^0$, for a given prime base b and a maximum level p, all with the same generating vector $\mathbf{a}$. Thus, one can start estimating an integral using the b^k lattice points of Q_{b^k} for some $0 \le k \le p-1$, and then refine the estimate as needed by evaluating the next b^k lattice points of $Q_{b^{k+1}} \setminus Q_{b^k}$, and so on.

2.2 Uniformity Criteria

Lattice Builder supports a variety of uniformity criteria, appropriate for different uses. One of them is the weighted $\mathscr{P}_2$ discrepancy (Hickernell 1998a; Sloan and Joe 1994):

$$\mathscr{P}_2(Q_n^0) = \sum_{\emptyset \ne \mathfrak{u} \subseteq \{1,\ldots,s\}} \gamma_{\mathfrak{u}} \frac{1}{n}\sum_{i=0}^{n-1}\prod_{j\in\mathfrak{u}} 2\pi^2 B_2((ia_j/n) \bmod 1) \tag{2}$$

where $B_2(x) = x^2 - x + 1/6$ is the second-degree Bernoulli polynomial, and the non-negative constants $\gamma_{\mathfrak{u}}$ for the coordinate subsets (or projections) $\mathfrak{u} \subseteq \{1,\ldots,s\}$ are the projection-dependent weights, which are left to be chosen. The criterion (2) is more sensitive to a bad distribution of the points of Q_n^0 along projections $\mathfrak{u}$ associated with larger weights $\gamma_{\mathfrak{u}}$. So, when minimizing (2), the quality of these projections will likely be better than that of projections with smaller weights. Intuitively, this indicates that larger weights should be assigned to projections along which f shows more variability.

More rigorously, $f(\mathbf{x}) - \mu$ can be decomposed into a sum of functions $f_{\mathfrak{u}}(\mathbf{x})$ over non-empty subsets $\mathfrak{u} \subseteq \{1,\ldots,s\}$, where each $f_{\mathfrak{u}}$ integrates to zero and depends only on the components of $\mathbf{x}$ whose indices are

in $\mathfrak{u}$. This is the functional ANOVA decomposition (see Owen 1998 and Liu and Owen 2006 for details) of f:

$$f(\mathbf{x}) - \mu = \sum_{\emptyset \neq \mathfrak{u} \subseteq \{1,\ldots,s\}} f_{\mathfrak{u}}(\mathbf{x}).$$

It follows that (2) is a uniformity criterion in the sense that, if the first-order partial derivatives of the integrand f are square integrable, then the RQMC variance $\mathrm{Var}[\hat{\mu}_n]$ is bounded by $\mathscr{V}_2^2(f)\,\mathscr{P}_2(Q_n^0)$, where

$$\mathscr{V}_2^2(f) = \sum_{\emptyset \neq \mathfrak{u} \subseteq \{1,\ldots,s\}} \gamma_{\mathfrak{u}}^{-1} (4\pi^2)^{|\mathfrak{u}|} \int_{[0,1)^{|\mathfrak{u}|}} \left| \frac{\partial^{|\mathfrak{u}|} f_{\mathfrak{u}}(\mathbf{x})}{\partial \mathbf{x}_{\mathfrak{u}}} \right|^2 \mathrm{d}\mathbf{x},$$

is the weighted square variation of f (Hickernell 1998a; Dick, Sloan, Wang, and Wozniakowski 2004; L'Ecuyer 2009), and where $|\mathfrak{u}|$ is the cardinality of $\mathfrak{u}$, and $\partial^{|\mathfrak{u}|} f_{\mathfrak{u}}(\mathbf{x})/\partial \mathbf{x}_{\mathfrak{u}}$ denotes the mixed partial derivative of $f_{\mathfrak{u}}$ to first order with respect to x_j for each $j \in \mathfrak{u}$, which, for example, is equal to $\partial^3 f_{\{1,2,3\}}(x_1,x_2,x_3)/(\partial x_1\,\partial x_2\,\partial x_3)$ if $\mathfrak{u} = \{1,2,3\}$. We remark that an increase of $\gamma_{\mathfrak{u}}$ puts a higher premium on reducing $\mathscr{V}_2^2(f)$ significantly when f shows relatively important variability along $\mathfrak{u}$. Thus, from a practical standpoint, the weights $\gamma_{\mathfrak{u}}$ should be chosen to reflect the variability of f along their respective projections $\mathfrak{u}$, in order to make (2) a better predictor of the RQMC variance. We also remark that the values of the weights for projections of order $|\mathfrak{u}| = 1$ are irrelevant, because, for fixed n and s, all rank-1 lattices have identical one-dimensional projections consisting of n points equally spaced by a distance of $1/n$, with the first point at the origin, so their quality cannot be improved by changing $\mathbf{a}$.

The $\mathscr{P}_2$ discrepancy can be generalized as $\mathscr{P}_\alpha$ with associated variation $\mathscr{V}_\alpha$, with $\alpha > 1$, which provide a bound on the RQMC variance for integrands with different smoothness properties. Larger values of α correspond to smoother integrands in the sense that, for $\mathscr{V}_\alpha(f)$ to remain finite, f must be smoother in some sense. Lattice Builder implements the $\mathscr{P}_\alpha$ criterion for $\alpha = 2,4,6,8$, for which it takes forms similar to (2) but involving Bernoulli polynomials of degree α and different constant factors. Lattice Builder also supports the $\mathscr{R}_\alpha$ criterion (Niederreiter 1992; Hickernell and Niederreiter 2003) and uniformity criteria based on the spectral test (L'Ecuyer and Lemieux 2000).

Specifying the projection-dependent weights $\gamma_{\mathfrak{u}}$ for $\emptyset \neq \mathfrak{u} \subseteq \{1,\ldots,s\}$ means specifying $2^s - 1$ parameters. The number of these input parameters can be reduced by using special cases of the projection-dependent weights. One possibility is to specify constants $\gamma_1,\ldots,\gamma_s$, where γ_j is the weight associated to the j-th coordinate, and set $\gamma_{\mathfrak{u}} = \prod_{j \in \mathfrak{u}} \gamma_j$. These are known as product weights (Hickernell 1998a; Hickernell 1998b; Sloan and Woźniakowski 1998). Alternatively, one can specify the s constants $\Gamma_1,\ldots,\Gamma_s$ and set $\gamma_{\mathfrak{u}} = \Gamma_{|\mathfrak{u}|}$, so that the weights depend only on the order of the projection, and are known as order-dependent weights. A particular case is the order-dependent weights truncated at order k (Wang 2007), where $\Gamma_k > 0$ and $\Gamma_j = 0$ for all $j > k$. Lattice Builder evaluates the $\mathscr{P}_\alpha$ and $\mathscr{R}_\alpha$ criteria using efficient algorithms based on specific types of weights; see, for example, Cools, Kuo, and Nuyens (2006) or Nuyens and Cools (2006a).

2.3 Selection Algorithms

Lattice Builder implements a collection of algorithms for searching for good lattice rules given a uniformity criterion with its weights. To give but a few examples, a popular choice is the component-by-component (CBC) construction (Kuo and Joe 2002), which consists in incrementally selecting the generating vector one coordinate at a time. Starting with $j = 1$ up to s, this algorithm keeps $a_1,\ldots,a_{j-1}$ fixed and selects $1 \le a_j \le n-1$ coprime with n that minimizes the chosen uniformity criterion of the lattice in dimension j defined by the generating vector $(a_1,\ldots,a_j)$. A randomized variant of this algorithm (Sinescu and L'Ecuyer 2009; Wang and Sloan 2006), which considers only a given number of randomly chosen values of a_j, is also supported. Lattice Builder also implements the fast CBC algorithm (Nuyens and Cools 2006a; Nuyens and Cools 2006c; Nuyens and Cools 2006b; Cools, Kuo, and Nuyens 2006) for the case where the number of points is an integer power of a prime base. Other possibilities include exhaustive search, fully random

search, Korobov constructions and random Korobov constructions (Korobov 1959; L'Ecuyer and Lemieux 2000; Sinescu and L'Ecuyer 2012).

2.4 Using the Software

The following example illustrates how Lattice Builder can be called from the command line to perform a CBC construction of an ordinary lattice point set with $n = 2^{16} = 65,536$ points in dimension $s = 8$, based on the $\mathscr{P}_2$ discrepancy with order-dependent weights truncated at order 3, with $\Gamma_1 = 1$, $\Gamma_2 = 0.1$ and $\Gamma_3 = 0.01$:

```
latbuilder --lattice-type ordinary --size 2^16 --dimension 8 \
  --figure-of-merit sum:P2 --weights order-dependent:0:1,0.1,0.01 \
  --construction CBC
```

A trailing backslash indicates that the command continues on the next line. The first token, `latbuilder`, is the name of the executable program. The next three arguments are straightforward to interpret: they specify, respectively, the type of lattice (ordinary), the number of points $n = 2^{16}$ and the dimension $s = 8$. The `--figure-of-merit sum:P2` indicates that we want to use the $\mathscr{P}_2$ discrepancy as the uniformity criterion. The `sum:` token that precedes `P2` correspond to the $\sum_{\emptyset \neq \mathfrak{u} \subseteq \{1,\ldots,s\}}$ operator in (2); Lattice Builder also supports the maximum operator instead of the sum. The next argument specifies the weights for the $\mathscr{P}_2$ discrepancy, and consists of three colon-separated tokens: the first is the type of weights (order-dependent), the second is the default weight $\Gamma_{|\mathfrak{u}|}$ for projection orders $|\mathfrak{u}| > k$ where k is the maximum projection order for which a weight is explicitly specified by the third token, which is a comma-separated list of values of $\Gamma_1, \ldots, \Gamma_k$. In the above example, $k = 3$, and we set $\Gamma_{|\mathfrak{u}|} = 0$ for $|\mathfrak{u}| > k$ with $\Gamma_1 = 1$, $\Gamma_2 = 0.1$ and $\Gamma_3 = 0.01$. The last argument, `--construction CBC` indicates that we want to use the CBC method. The output of the above command would contain the following:

```
BEST LATTICE: lattice(2^16, [1, 19463, 17213, 14627, 24339, 21007,
  18925, 12671]): 8.38924e-06
```

which indicates that Lattice Builder found a good lattice rule with $n = 2^{16}$, with the components of $\mathbf{a}$ listed between `[` and `]`, and whose $\mathscr{P}_2$ value with the specified order-dependent weights is approximately 8.39×10^{-6}.

In simulation software, there are situations where the dimension of the problem or the required number of points, or even the appropriate weights, are not known in advance, because they are dynamically computed. In these cases, one cannot expect to find any potential combination of n, s and $\gamma_{\mathfrak{u}}$ in tables of good lattice parameters computed in advance. Lattice Builder can be directly executed by the simulation software to construct good lattice rules of the appropriate number of points and dimension, at the moment they are needed. We give an example of that in Section 3.7 below. Lattice Builder also offers a C++ application programming interface, and thus can be used as a C++ software library by other programs. It can further be extended to support different uniformity criteria, types of weights and construction methods. The software package's web page can be found from the web site of the first author.

3 SIMULATION OF THE HESTON MODEL

3.1 Description of the Problem

Giles and Waterhouse (2009) applied QMC to the multilevel path simulation (Giles 2008) of financial options using the Milstein discretization. We do the same here, but for the Heston volatility model, defined by the following two-dimensional stochastic differential equation (SDE):

$$\mathrm{d}S(t) = rS(t)\mathrm{d}t + V(t)^{1/2}S(t)\mathrm{d}B_1(t), \tag{3}$$

$$\mathrm{d}V(t) = \lambda(\sigma^2 - V(t))\mathrm{d}t + \xi V(t)^{1/2}\mathrm{d}B_2(t), \tag{4}$$

for $t \geq 0$, where $S(t)$ and $V(t)$ are, respectively, the value and the (always positive) instantaneous variance of an asset price, where (B_1, B_2) is a pair of standard Brownian motions with correlation ρ between them and where

the risk-free rate r, the long-term average variance σ^2, its rate of return to the mean λ and its volatility ξ are positive constants. Our goal is to estimate the expectation of a payoff $P = f(\{(S(t),V(t)) : 0 \le t \le T\})$, which is a function f of the process trajectory over the time interval $[0,T]$. More specifically, we consider two different payoff functions, namely the (discounted) payoff function for a European call option, $P = e^{-rT}\max(0, S(T)-K)$, where $r > 0$ is the risk-free rate and K is the strike price (a constant), and for an Asian call option, $P = e^{-rT}\max(0, \bar{S}-K)$, where $\bar{S} = (1/T)\int_0^T S(t)\mathrm{d}t$ is the continuous-time average of the process S over the time interval $[0,T]$. Our purpose is to illustrate the application of randomly-shifted lattice rules for multilevel path simulation, so we assume that P cannot be generated from its exact distribution. In our computations, we use $T = 1$, $K = 100$, $S(0) = 100$, $V(0) = 0.04$, $r = 0.05$, $\sigma = 0.2$, $\lambda = 5$, $\xi = 0.25$, and $\rho = -0.5$.

3.2 Discretization

We discretize the time and use Euler's method with time step h to generate an approximate skeleton $\{(S(t),V(t)) : t = 0,h,2h,\ldots,dh\}$ of the process $\{(S(t),V(t)) : t \in [0,T]\}$, with d time steps of length $h = T/d$, and we compute a corresponding approximate payoff. More precisely, to reduce the bias, we apply Euler's method to (S,W) instead of (S,V), where $W(t) = e^{\lambda t}(V(t)-\sigma^2)$. This gives $\mathrm{d}W(t) = e^{\lambda t}\xi V(t)^{1/2}\mathrm{d}B_2(t)$. The Euler scheme with step size h becomes

$$\widetilde{W}((j+1)h) = \widetilde{W}(jh) + e^{\lambda jh}\xi\widetilde{V}(jh)^{1/2}\sqrt{h}Z_{j,2},$$

and then, using the identity $V(t) = \sigma^2 + e^{-\lambda t}W(t)$,

$$\widetilde{V}((j+1)h) = \max\left[0, \sigma^2 + e^{-\lambda h}\left(\widetilde{V}(jh) - \sigma^2 + \xi\widetilde{V}(jh)^{1/2}\sqrt{h}Z_{j,2}\right)\right], \tag{5}$$

$$\widetilde{S}((j+1)h) = \widetilde{S}(jh) + rh\widetilde{S}(jh) + [\widetilde{V}(jh)]^{1/2}\widetilde{S}(jh)\sqrt{h}Z_{j,1}, \tag{6}$$

where the vectors $\mathbf{Z}_j = (Z_{j,1}, Z_{j,2})$ are normal with mean $\mathbf{0}$ and covariance matrix $\boldsymbol{\Sigma}$ whose elements are 1 on the diagonal and ρ elsewhere. The "max[0,...]" is to make sure that the approximation of V never becomes negative.

To generate $\mathbf{Z}_j$, we use the components of indices $2j-1$ and $2j$ of the input random vector $\mathbf{U} = (U_1,\ldots,U_s)$, where $s = 2d$, uniformly distributed in $(0,1)^s$, as follows:

$$\mathbf{Z}_j = \begin{pmatrix} Z_{j,1} \\ Z_{j,2} \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{pmatrix}}_{\mathbf{A}_\rho} \begin{pmatrix} \Phi^{-1}(U_{2j-1}) \\ \Phi^{-1}(U_{2j}) \end{pmatrix} = \begin{pmatrix} \Phi^{-1}(U_{2j-1}) \\ \rho\Phi^{-1}(U_{2j-1}) + \sqrt{1-\rho^2}\Phi^{-1}(U_{2j}) \end{pmatrix}, \tag{7}$$

where $\mathbf{A}_\rho$ is the Cholesky factorization of the variance-covariance matrix $\boldsymbol{\Sigma} = \mathbf{A}_\rho\mathbf{A}_\rho^{\mathrm{T}}$ and Φ^{-1} is the inverse of the standard normal distribution function.

For the Asian payoff, we also approximate $\bar{S}$ with its discretized average

$$\bar{S} = h\left(\frac{\widetilde{S}(0)}{2} + \frac{\widetilde{S}(T)}{2} + \sum_{j=1}^{d-1}\widetilde{S}(jh)\right).$$

3.3 Dimensional Structure of the Problem

Before we start constructing good lattice rules, we need to decide how to set the weights $\gamma_{\mathfrak{u}}$. Our previous experimentations (L'Ecuyer and Munger 2012) suggested that, as long as projections with very little variability are not given too much weight and that relevant projections are given enough weight, the variance of the RQMC estimator is not too sensitive to the exact distribution of the weights. Thus, we

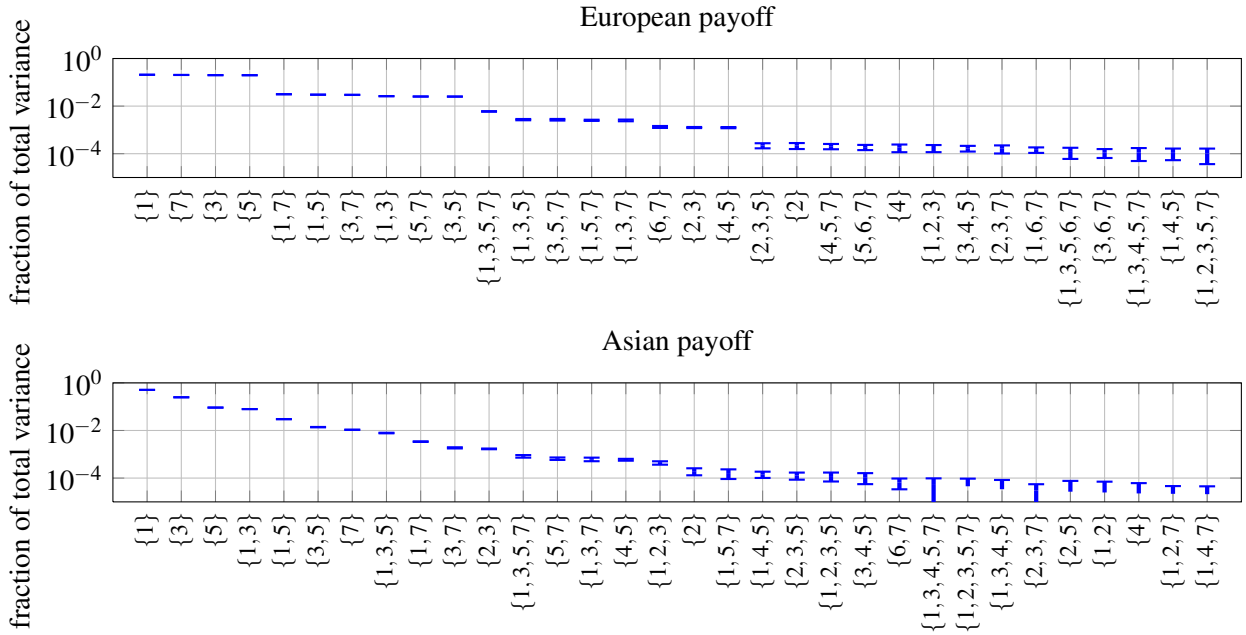

Figure 1: Distribution of the ANOVA variances across projections, for the European (top) and Asian (bottom) payoff functions with $d = 4$, for the 32 projections with the largest estimated variance. The horizontal axis lists the projection $\mathfrak{u}$ by decreasing order of estimated MC variance $\sigma^2_{\mathfrak{u}}$, plotted along the vertical axis in units of the estimated total MC variance $\mathrm{Var}[f(\mathbf{U})]$, in log scale. The span of each vertical bar corresponds to a normal confidence interval at 95 % on the variance estimate.

need to assess, at least roughly, the distribution of the variability of the integrand f across the different subsets of coordinates $\mathfrak{u}$.

We calculated the estimates $\sigma^2_{\mathfrak{u}}$ of the MC variances $\mathrm{Var}[f_{\mathfrak{u}}(\mathbf{U})]$ of the ANOVA components of f for all nonempty subsets $\mathfrak{u} \subseteq \{1,\dots,s\}$. We computed these estimates using the algorithm proposed by Sobol' and Myshetskaya (2007), with a lattice rule with $n = 2^{16}+1 = 65,537$ constructed with Lattice Builder, and using 30 independent random shifts. We decided to only allow a limited computational budget for estimating the $\sigma^2_{\mathfrak{u}}$'s, so we estimated them for a discretization with $d = 4$ time steps or, equivalently, for an integrand of dimension $s = 8$, assuming that the dimensional structure of the problem is similar for other values of d. The results are shown in Figure 1 for the European and Asian payoff functions. We see that projections involving only odd coordinates generally account for a larger portion of the variance than other projections of the same order. This can be explained from (7), where odd coordinates are used to generate $Z_{j,1}$ and $Z_{j,2}$, whereas even coordinates are involved only in generating $Z_{j,2}$. With only this observation, it would be tempting to use product weights with larger weights for odd coordinates than for even coordinates. But, in fact, some projections involving even coordinates hold much more variance than others of the same order. This can be explained as follows. We first recall that, in (5) and (6), $Z_{j,1}$ and $Z_{j,2}$ are directly used to generate increments of $\widetilde{S}$ and $\widetilde{V}$, respectively. Because the computation of $\widetilde{S}((j+1)h)$ directly depends on the values of $Z_{j,1}$ and $\widetilde{V}(jh)$, joint variations both in $Z_{j-1,2}$, which contributes to $\widetilde{V}(jh)$, and in $Z_{j,1}$ are expected to have a significant impact on the value of $\widetilde{S}((j+1)h)$. Therefore, projections containing pairs of coordinates $\{2j-2, 2j-1\}$ for any $j = 2,\dots,d$ are expected to account for more variance than other projections involving even coordinates, as confirmed by Figure 1. Such projections would not be given enough weight by product weights as considered above.

3.4 Numerical Experiments

We performed experiments with the model and parameters from Section 3.1 and the method described in Section 3.2. Given the observations from Section 3.3, we rule out product weights, but we focus on the fact that projections of lower order generally contribute a larger portion of the total variance than those of higher order. Therefore, we examine a few choices of order-dependent weights truncated at order k, of the geometric form $\gamma_{\mathfrak{u}} = \Gamma^{|\mathfrak{u}|}$ if $|\mathfrak{u}| \le k$, and $\gamma_{\mathfrak{u}} = 0$ otherwise. We consider $k = 2, 3, 4$ and $k = \infty$ (not truncated).

To measure the efficiency of estimators and compare their efficiencies, we will use the *work-normalized variance* $\mathrm{WNVar}[\hat{\mu}] = C(\hat{\mu})\,\mathrm{Var}[\hat{\mu}]$, where $C(\hat{\mu}) = dn$ represents the computing cost, taken here as the product of the number n of realizations of the sample path by the number d of time steps per realization. We estimate $\mathrm{Var}[\hat{\mu}]$ by the unbiased sample variance using $r = 1000$ independent random shifts. We take r large enough to obtain reasonable precision on the estimates in order to be able to compare the performance of different methods together. Table 1 shows the results obtained with standard MC and with rank-1 lattice rules constructed by Lattice Builder with $n = 2^{17} = 131,072$ for the European payoff and $n = 2^{16} = 65,536$ for the Asian payoff, with $d = 2^5 = 32$ in both cases. (These particular values of n and d are chosen to facilitate comparison with results from Section 3.7 below.) Note that, as opposed to MC, the rate of decrease of the RQMC variance as a function of n generally depends on n, so the associated work-normalized variance may also vary with n. Results are also shown for Korobov constructions with $\mathbf{a} = (1, a, a^2 \bmod n, \ldots, a^s \bmod n)$ taken from L'Ecuyer and Lemieux (2000), based on the spectral criterion M_{32}, which considers the uniformity of projections consisting exclusively of successive coordinates. We set $n = 131,071$ with $a = 29803$ for the European payoff and $n = 65,521$ with $a = 2469$ for the Asian one. These Korobov lattices do not perform as well as rank-1 lattices constructed by Lattice Builder with more appropriate uniformity criteria. The European payoff seems more sensitive to the choice of lattice parameters than the Asian payoff. We also tried to vary n in the experiments and we observed that it is not always the same choice of weights that yields the smallest or the largest work-normalized variances. However, the lattice rules with $\Gamma = 0.1$, which give more weight to high-dimensional projections than the other choices of weights, the lattice rules with $\Gamma = 0.01$ and $k = 2$, which give no weight to projections of dimension 3 or more, as well as the selected Korobov rules, which were constructed while ignoring many relevant projections, consistently yielded significantly larger variances in all cases.

3.5 Multilevel Formulation

To reduce the bias on an estimator for P based on this approximate process, h should be made as small possible, but the computing cost of generating $\{(S(t), V(t)) : t = 0, h, 2h, \ldots, dh\}$ is usually proportional to the number of time steps $d = T/h$. Giles (2008) developed a method, based on multigrid ideas in numerical analysis, to obtain an estimator with the same low bias and almost the same variance as an estimator based on a very fine grid (small h), with a computing effort comparable to that required with a coarse discretization (large h). The idea is to generate a basic low-variance estimator for the expectation of the approximate payoff based on a very coarse discretization by averaging a large number of realizations, then to refine the estimator by applying a series of corrections based on finer discretizations. Because the corrections are small in magnitude compared to the basic estimator, they can reach a variance comparable to that of the basic estimator using much fewer realizations. Formally, let the step sizes be $h_\ell = m^{-\ell} T$, $\ell = 0, \ldots, L$, for some integers $m \ge 2$ and $L \ge 0$. Let P_ℓ denote the (random) payoff based on $\{(S(t), V(t)) : t = 0, h_\ell, 2h_\ell, \ldots, d_\ell h_\ell\}$ with time step h_ℓ and $d_\ell = T/h_\ell = m^\ell$. Then,

$$\mathbb{E}[P_L] = \mathbb{E}[P_0] + \sum_{\ell=1}^{L} \mathbb{E}[P_\ell - P_{\ell-1}].$$

The multilevel method estimates $\mathbb{E}[P_0]$ by the average Y_0 of n_0 realizations of P_0, and estimates $\mathbb{E}[P_\ell - P_{\ell-1}]$ by the average Y_ℓ of n_ℓ realizations of $P_\ell - P_{\ell-1}$, for each ℓ. The overall estimator of $\mathbb{E}[P]$ is $Y = \sum_{\ell=0}^{L} Y_\ell$.

Table 1: Estimated $\mathrm{WNVar}[\hat{\mu}]$ for $d = 2^5 = 32$, for the European and Asian payoffs with $n = 2^{17} = 131,072$ and $n = 2^{16} = 65,536$, respectively. Under the (incorrect) assumption that $\hat{\mu}$ is normally distributed, the relative half-width of confidence intervals at 95 % is of approximately 9 %. The column labels SL and ML stand for the single-level and multilevel cases from Sections 3.4 and 3.7, respectively. Note that $\hat{\mu} = Y$ in the multilevel case. In the single level case, the line labeled Korobov corresponds to ordinary Korobov rules with $n = 131,071$ and $a = 29803$ for the European payoff and with $n = 65,521$ and $a = 2469$ for the Asian payoff. In the multilevel case, it corresponds to embedded Korobov rules with $a = 1267$. The line labeled MC shows the work-normalized variances obtained using MC instead of lattice rules.

Γ	k	European		Asian	
		SL	ML	SL	ML
0.1	∞	510	1770	150	222
0.01	∞	451	396	**101**	**124**
0.001	∞	**347**	451	104	169
0.01	2	524	653	124	213
0.01	3	378	387	104	148
0.01	4	356	**378**	110	128
Korobov		1560	586	714	216
MC		12348	3012	3635	1347

One important feature is that each realization of $P_\ell - P_{\ell-1}$ is simulated with common random numbers across the two terms. For this, we first simulate the process with time step h_ℓ to obtain P_ℓ. This requires $2m^\ell$ random numbers to simulate the increments of the two-dimensional process over the m^ℓ time steps. We denote here by $\mathbf{Z}_j^\ell$ rather than by $\mathbf{Z}_j$, defined in (7), the intermediate two-dimensional normal vector generated using the j-th pair of these random numbers. Then, to simulate the process with time step $h_{\ell-1} = mh_\ell$, we set $\mathbf{Z}_j^{\ell-1} = m^{-1/2}\sum_{k=1}^m \mathbf{Z}_{(j-1)m+k}^\ell$, hence $\mathbf{Z}_j^{\ell-1}$ is produced with common random numbers with $\mathbf{Z}_{(j-1)m+1}^\ell, \ldots, \mathbf{Z}_{jm}^\ell$ and also has the same distribution as $\mathbf{Z}_j$. However, independent random numbers are used across different values of ℓ, so that $\mathrm{Var}[Y] = \sum_{\ell=0}^L \mathrm{Var}[Y_\ell]$.

The idea from Giles and Waterhouse (2009) is to choose L and $n_0, \ldots, n_L$ in order to obtain a mean square error on Y of at most ε^2, which can be achieved by requiring that both the square bias and the variance be smaller than $\varepsilon^2/2$ (the mean square error is defined as their sum). The authors estimate the bias on Y with $(m-1)^{-1}\max\{m^{-1}|Y_{L-1}|, |Y_L|\}$ (see Giles 2008 for the details), and the variance with the unbiased sample variance. To simulate Y, the maximum level L is incremented iteratively until the bias estimate is smaller than $\varepsilon/\sqrt{2}$, and the numbers of points $n_0, \ldots, n_L$ are chosen in such a way to try to minimize the work-normalized variance $\mathrm{WNVar}[Y] = C(Y)\,\mathrm{Var}[Y]$, with $C(Y) = \sum_{\ell=0}^L C(Y_\ell) = \sum_{\ell=0}^L m^\ell n_\ell$.

For MC, n_ℓ can be chosen in advance. It is known that the MC variance of Y_ℓ decreases as n_ℓ^{-1} and it can be shown that taking n_ℓ proportional to h_ℓ minimizes $\mathrm{WNVar}[Y]$, under mild conditions, which hold for example if we use the Euler discretization and if the payoff function is Lipchitz continuous. For RQMC, the convergence rate of $\mathrm{Var}[Y_\ell]$ in terms of n_ℓ is not known precisely, so n_ℓ is chosen adaptively as follows. Until $\mathrm{Var}[Y] < \varepsilon^2/2$, the estimator Y_ℓ is refined by doubling n_ℓ for the level $\ell \in \{0, \ldots, L\}$ that has the largest potential of reduction of variance per unit cost of increasing the number of points by a factor of b. The authors measure this potential as $\mathrm{Var}[Y_\ell]/C(Y_\ell)$, assuming that most of the variance of Y_ℓ can be eliminated by increasing n_ℓ by a factor b. This adaptive refinement of $Y_0, \ldots, Y_L$ is performed every time L is incremented. The use of embedded lattice rules $Q_{b^0}^0 \subset \cdots \subset Q_{b^p}$ in base $b = 2$ allows for reusing the realizations of $P_\ell - P_{\ell-1}$ for $\ell \geq 1$, or of P_0 for $\ell = 0$, already summed into Y_ℓ before doubling n_ℓ (this also means that the r independent realizations of Y_ℓ must be stored). As we increase L, we use Lattice Builder to construct good embedded lattice rules $Q_{b^0}^0 \subset \cdots \subset Q_{b^p}$ in base $b = 2$ with $p = 16$ and in dimension m^L.

3.6 Dimensional Structure of the Multilevel Problem

In this section, we repeat the analysis from Section 3.3, but for the integrand used to compute the estimator Y_2 of $\mathbb{E}[P_2 - P_1]$ based on replicates of $P_2 - P_1$, instead of for the integrand used to compute the estimator of $\mathbb{E}[P_2]$ as was done in Section 3.3. The results are shown in Figure 2. Most of the remarks from Section 3.3 also hold here. In general, RQMC works best if the bulk of the variance is held by projections of low orders. However, here, and especially for the European payoff function, a much smaller portion of the variance is contributed from projections of order 1, compared to the single-level case of Section 3.3.

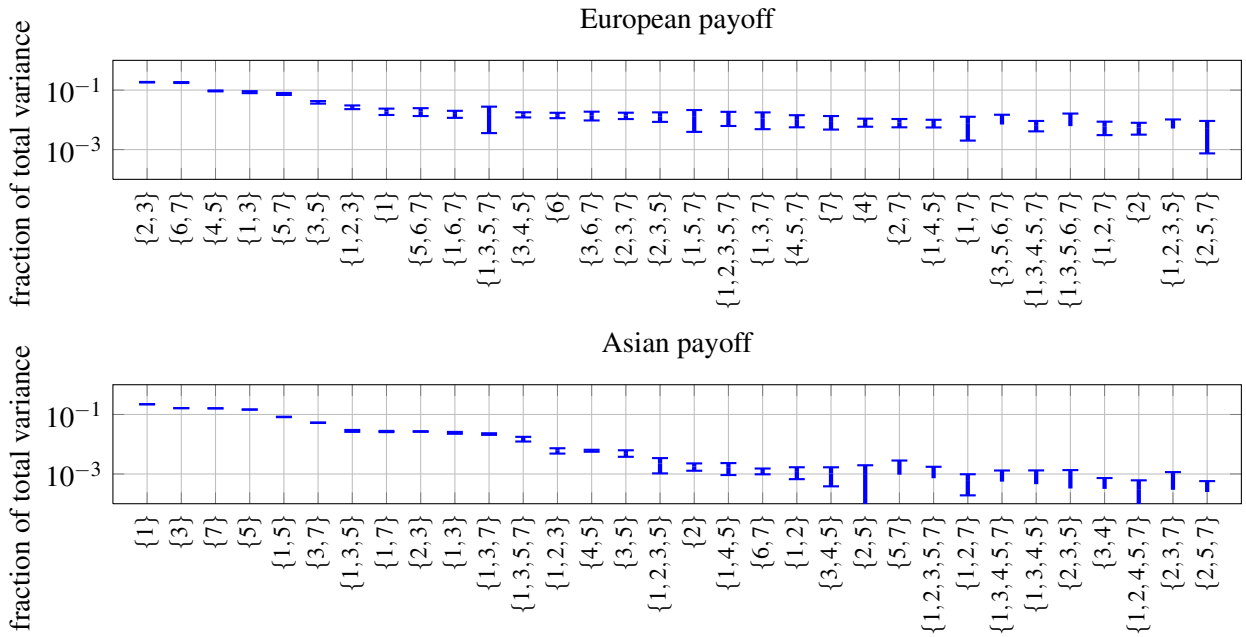

Figure 2: Distribution of the ANOVA variances across projections of $P_2 - P_1$, for the European (top) and Asian (bottom) payoff functions, for the 32 projections with the largest estimated variance. See Figure 1.

3.7 Numerical Experiments with the Multilevel Problem

Here we apply the multilevel approach described in Section 3.5 to the discretization from Section 3.2 of the problem of Section 3.1. Because the maximum value of L that will be used is not known in advance, every time L is increased, our simulation software calls Lattice Builder as illustrated in Section 2.4 to construct new good embedded lattice rules in dimension $s = 2^{L+1}$.

For $\varepsilon = 0.01$, the estimation of the European and Asian payoff converged at $L = 5$, with $Y \approx 10.46$ and 5.76, respectively. The ML columns of Table 1 give the work-normalized variances of Y using different types of weights, and also with embedded Korobov lattices with $a = 1267$, taken from Hickernell et al. (2001). In Section 3.4, we chose $d = 2^L$ where L is the highest level reached by the multilevel algorithm, and we selected n such that the total computational cost is the same as with the multilevel algorithm. The results here are qualitatively very similar to those from Section 3.4. What is surprising is that the multilevel approach consistently yields larger values of $\mathrm{WNVar}[Y]$ than those from the single-level method. To understand this, we examine the allocation of the computational effort across the different levels.

The multilevel approach relies on the decreasing magnitude of $P_\ell - P_{\ell-1}$ (and its variance) to maintain a low value of $\mathrm{Var}[Y_\ell]$ while decreasing n_ℓ with ℓ. This *permits one to maintain* the total variance under a target threshold $\varepsilon^2/2$ while reducing the total computational effort. With MC, the optimal allocation of work consists in taking $C(Y_\ell)$ constant across all levels ℓ. With RQMC, in contrast, we do not know in advance what the optimal allocation of work across the different levels is. Table 2 shows the estimates of

$\mathrm{Var}[Y_\ell]$ for the European and Asian payoffs, for MC and for lattice rules with $\Gamma = 0.01$ and $k = \infty$. The numbers displayed for MC are adjusted to correspond to the same total cost $C(Y)$ as obtained with lattice rules. For the case of lattice rules, the values of $C(Y_\ell)$ resulting from the adaptive selection of n_ℓ are also given in parentheses, and tend to increase with ℓ. The explanation for this is twofold. First, the adaptive multilevel RQMC algorithm, as originally formulated, estimates the potential of reduction of variance per unit cost in exactly the same way for lower and higher levels. However, the RQMC variance generally converges more slowly with n_ℓ in high dimension (higher levels) than in low dimension (lower levels). It follows that the adaptive algorithm overestimates the potential of reduction of variance per unit cost at higher levels, relative to that at lower levels, which results in a shift of the computational effort towards the more demanding higher levels. The second part of the explanation is that, as we have seen in Section 3.6, $P_\ell - P_{\ell-1}$ has a larger portion of its variance contributed by projections of dimension larger than unity than just P_ℓ, which makes RQMC less profitable when computing $P_\ell - P_{\ell-1}$ than when computing P_ℓ. Thus, shifting much of the RQMC computational effort toward higher levels appears to be counterproductive in the end.

Table 2: $\mathrm{Var}[Y_\ell]$ as a function of ℓ for $\varepsilon = 0.01$, for the European and Asian payoffs, using MC and rank-1 lattice rules with $\Gamma = 0.01$ and $k = \infty$ (not truncated). $C(Y_\ell)$ is given in parentheses for the case of lattice rules; for MC, $C(Y_\ell)$ is the same for all ℓ.

ℓ	European		Asian	
	MC	$\Gamma = 0.01$	MC	$\Gamma = 0.01$
0	2.3×10^{-4}	8.2×10^{-7} (2^{18})	1.5×10^{-4}	7.4×10^{-7} (2^{17})
1	5.4×10^{-6}	7.9×10^{-7} (2^{18})	8.8×10^{-5}	1.6×10^{-6} (2^{17})
2	8.0×10^{-6}	3.6×10^{-6} (2^{19})	5.6×10^{-5}	3.4×10^{-6} (2^{18})
3	2.3×10^{-5}	1.3×10^{-5} (2^{20})	4.2×10^{-5}	7.9×10^{-6} (2^{19})
4	4.0×10^{-5}	1.7×10^{-5} (2^{21})	4.3×10^{-5}	9.6×10^{-6} (2^{20})
5	5.2×10^{-5}	1.2×10^{-5} (2^{22})	4.6×10^{-5}	1.6×10^{-5} (2^{20})

4 CONCLUSION

Lattice Builder is useful to construct lattice rules on the fly, as they are needed. In our examples, the performance of RQMC was generally sensitive to the choice of criterion and weights, and it was profitable to construct our own lattice rules instead of using parameters (for the Korobov rules) obtained without targeting a specific application. Besides, we have seen that the multilevel RQMC algorithm proposed by Giles and Waterhouse (2009) did not perform as well for RQMC for our examples, possibly because of the way it allocates the computational effort across the different levels, and because the structure of the multilevel integrands is less RQMC-friendly.

ACKNOWLEDGMENTS

This work has been supported by NSERC-Canada and by a Canada Research Chair to the first author.

REFERENCES

Cools, R., F. Y. Kuo, and D. Nuyens. 2006. "Constructing embedded lattice rules for multivariate integration". *SIAM Journal on Scientific Computing* 28 (16): 2162–2188.

Dick, J., I. H. Sloan, X. Wang, and H. Wozniakowski. 2004. "Liberating the weights". *Journal of Complexity* 20 (5): 593–623.

Giles, M. B. 2008. "Multilevel Monte Carlo path simulation". *Operations Research* 56 (3): 607–617.

Giles, M. B., and B. J. Waterhouse. 2009. "Multilevel quasi-Monte Carlo path simulation". *Advanced Financial Modelling, Radon Series on Computational and Applied Mathematics*:165–181.

Hickernell, F. J. 1998a. "A Generalized Discrepancy and Quadrature Error Bound". *Mathematics of Computation* 67 (221): 299–322.

Hickernell, F. J. 1998b. "Lattice Rules: How Well do They Measure Up?". In *Random and Quasi-Random Point Sets*, edited by P. Hellekalek and G. Larcher, Volume 138 of *Lecture Notes in Statistics*, 109–166. New York: Springer-Verlag.

Hickernell, F. J., H. S. Hong, P. L'Ecuyer, and C. Lemieux. 2001. "Extensible Lattice Sequences for Quasi-Monte Carlo Quadrature". *SIAM Journal on Scientific Computing* 22 (3): 1117–1138.

Hickernell, F. J., and H. Niederreiter. 2003. "The existence of good extensible rank-1 lattices". *Journal of Complexity* 19 (3): 286–300.

Korobov, N. M. 1959. "The Approximate Computation of Multiple Integrals". *Dokl. Akad. Nauk SSSR* 124:1207–1210. in Russian.

Kuo, F. Y., and S. Joe. 2002. "Component-by-Component Construction of Good Lattice Rules with a Composite Number of Points". *Journal of Complexity* 18 (4): 943–976.

L'Ecuyer, P. 2009. "Quasi-Monte Carlo Methods with Applications in Finance". *Finance and Stochastics* 13 (3): 307–349.

L'Ecuyer, P., and C. Lemieux. 2000. "Variance Reduction via Lattice Rules". *Management Science* 46 (9): 1214–1235.

L'Ecuyer, P., and D. Munger. 2012. "On Figures of Merit for Randomly-Shifted Lattice Rules". In *Monte Carlo and Quasi-Monte Carlo Methods 2010*, edited by H. Wozniakowski and L. Plaskota, 133–159. Berlin: Springer-Verlag.

Liu, R., and A. B. Owen. 2006. "Estimating Mean Dimensionality of Analysis of Variance Decompositions". *Journal of the American Statistical Association* 101 (474): 712–721.

Niederreiter, H. 1992. *Random Number Generation and Quasi-Monte Carlo Methods*, Volume 63 of *SIAM CBMS-NSF Regional Conference Series in Applied Mathematics*. Philadelphia, PA: SIAM.

Nuyens, D., and R. Cools. 2006a. "Fast algorithms for component-by-component construction of rank-1 lattice rules in shift-invariant reproducing kernel Hilbert spaces". *Mathematics of Computation* 75:903–920.

Nuyens, D., and R. Cools. 2006b. "Fast Component-by-Component Construction, a Reprise for Different Kernels". In *Monte Carlo and Quasi-Monte Carlo Methods 2004*, edited by H. Niederreiter and D. Talay, 373–387.

Nuyens, D., and R. Cools. 2006c. "Fast component-by-component construction of rank-1 lattice rules with a non-prime number of points". *Journal of Complexity* 22:4–28.

Owen, A. B. 1998. "Latin Supercube Sampling for Very High-Dimensional Simulations". *ACM Transactions on Modeling and Computer Simulation* 8 (1): 71–102.

Sinescu, V., and P. L'Ecuyer. 2009. "On the Behavior of Weighted Star Discrepancy Bounds for Shifted Lattice Rules". In *Monte Carlo and Quasi-Monte Carlo Methods 2008*, edited by P. L'Ecuyer and A. B. Owen, 603–616. Berlin: Springer-Verlag.

Sinescu, V., and P. L'Ecuyer. 2012. "Variance Bounds and Existence Results for Randomly Shifted Lattice Rules". *Journal of Computations and Applied Mathematics* 236:3296–3307.

Sloan, I. H., and S. Joe. 1994. *Lattice Methods for Multiple Integration*. Oxford: Clarendon Press.

Sloan, I. H., and H. Woźniakowski. 1998. "When are Quasi-Monte Carlo Algorithms Efficient for High-Dimensional Integrals". *Journal of Complexity* 14:1–33.

Sobol', I. M., and E. E. Myshetskaya. 2007. "Monte Carlo Estimators for Small Sensitivity Indices". *Monte Carlo Methods and Applications* 13 (5–6): 455–465.

Wang, X. 2007. "Constructing Robust Good Lattice Rules for Computational Finance". *SIAM Journal on Scientific Computing* 29 (2): 598–621.

Wang, X., and I. H. Sloan. 2006. "Efficient Weighted Lattice Rules with Applications to Finance". *SIAM Journal on Scientific Computing* 28 (2): 728–750.

AUTHOR BIOGRAPHIES

PIERRE L'ECUYER is Professor in the Département d'Informatique et de Recherche Opérationnelle, at the Université de Montréal, Canada. He holds the Canada Research Chair in Stochastic Simulation and Optimization. He is a member of the CIRRELT and GERAD research centers. His main research interests are random number generation, quasi-Monte Carlo methods, efficiency improvement via variance reduction, sensitivity analysis and optimization of discrete-event stochastic systems, and discrete-event simulation in general. He is currently Editor-in-Chief for *ACM Transactions on Modeling and Computer Simulation*, and Associate/Area Editor for *ACM Transactions on Mathematical Software*, *Statistics and Computing*, *International Transactions in Operational Research*, and *Cryptography and Communications*. More information and his recent research articles are available at http://www.iro.umontreal.ca/∼lecuyer.

DAVID MUNGER is a post doctoral scholar in the Département d'Informatique et de Recherche Opérationnelle, at the Université de Montréal, Canada. He received his Ph.D. in physics in 2008 from the Université de Montréal. His research interests are quasi-Monte Carlo methods and their applications to simulation. His email address is david.munger@umontreal.ca.