Proceedings of the 2004 Winter Simulation Conference
R. G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A. Peters, eds.

QUASI-MONTE CARLO METHODS IN FINANCE

Pierre L'Ecuyer

Département d'Informatique et de Recherche Opérationnelle
Université de Montréal, C.P. 6128, Succ. Centre-Ville
Montréal (Québec), H3C 3J7, CANADA

ABSTRACT

We review the basic principles of Quasi-Monte Carlo (QMC) methods, the randomizations that turn them into variance-reduction techniques, and the main classes of constructions underlying their implementations: lattice rules, digital nets, and permutations in different bases. QMC methods are designed to estimate integrals over the s-dimensional unit hypercube, for moderate or large (perhaps infinite) values of s. In principle, any stochastic simulation whose purpose is to estimate an integral fits this framework, but the methods work better for certain types of integrals than others (e.g., if the integrand can be well approximated by a sum of low-dimensional smooth functions). Such QMC-friendly integrals are encountered frequently in computational finance and risk analysis. We give examples and provide computational results that illustrate the efficiency improvement achieved.

1 INTRODUCTION

Quasi-Monte Carlo (QMC) refers to a class of deterministic numerical techniques to approximate integrals in more than 2 or 3 dimensions. The general idea is to define a set or sequence of points that covers the integration domain very uniformly, evaluate the function at these points, and approximate the integral by the average value multiplied by the volume of the integration domain.

1.1 Simulation and the Monte Carlo Method

In the vast majority of situations, the integration domain is the s-dimensional unit hypercube for some integer s, and this is the case considered in this paper. So we want to estimate an integral of the form

$$\mu = \int_{[0,1)^s} f(\mathbf{u}) d\mathbf{u}$$

for some function f. In a stochastic simulation, s corresponds to the number of calls to the underlying uniform random number generator, $\mathbf{u} = (u_0, u_1, u_2, \dots)$ is the sequence of successive numbers returned by this generator, and f represent the (usually very complicated) transformation of these numbers into the simulation output $f(\mathbf{u})$, which is assumed here to be an unbiased estimator of some constant μ. In the case where the required number of uniforms is random and unbounded, we can just take $s = \infty$.

The estimator of μ considered here has the form

$$Q_n = \frac{1}{n} \sum_{i=0}^{n-1} f(\mathbf{u}_i),$$

where $P_n = \{\mathbf{u}_0, \dots, \mathbf{u}_{n-1}\} \subset [0,1)^s$ is the point set over which the average is taken. The number of points n corresponds to the number of simulation runs.

In standard *Monte Carlo* (MC), the $\mathbf{u}_i$ are independent and uniformly distributed over $[0,1)^s$. Then, Q_n is obviously unbiased and has variance σ^2/n, where

$$\sigma^2 = \int_{[0,1)^s} f^2(\mathbf{u}) d\mathbf{u} - \mu^2.$$

If $\sigma^2 < \infty$, we have the central-limit theorem $\sqrt{n}(Q_n - \mu)/\sigma \Rightarrow N(0,1)$, so the size of a confidence interval for μ, at a fixed level, converges as at rate $O_p(\sigma/\sqrt{n})$.

1.2 Quasi-Monte Carlo and the Koksma-Hlawka Inequality

Quasi-Monte Carlo (QMC) tries to take the points $\mathbf{u}_i$ *more uniformly distributed* over $[0,1)^s$ than typical independent uniform random points (Niederreiter 1992b, Sloan and Joe 1994, L'Ecuyer and Lemieux 2002). Its traditional justification is via Koksma-Hlawka-type inequalities of the form

$$|Q_n - \mu| \le \|f - \mu\| D(P_n) \tag{1}$$

for all f in some Hilbert space of functions with norm $\|\cdot\|$, where $\|f - \mu\|$ measures the *variability* of f, $D(P_n)$ measures the *discrepancy* (or non-uniformity) of P_n, and

their definitions depend on each other (Niederreiter 1992b, Hickernell 1998a). Such inequalities provide deterministic worst-case error bounds. Infinite sequences of points $\mathbf{u}_0, \mathbf{u}_1, \ldots$ for which $D(P_n) = O(n^{-1}(\ln n)^s)$ can be constructed for any finite s. For functions with finite variation, this give a convergence rate of $|Q_n - \mu| = O(n^{-1}(\ln n)^s)$ for the error, which is asymptotically better than the $O_p(n^{-1/2})$ rate of MC. However, these bounds are not practical for several reasons (e.g., they are much too difficult to compute, and for s larger than 7 or 8, $n^{-1}(\ln n)^s < n^{-1/2}$ for all $n \geq n_0$ requires an n_0 that is much too large to be practical). Yet, QMC happens to give much more precise results than MC for certain types of integrands in hundreds of dimension or more. One explanation of this success is outlined in Section 1.4.

1.3 Randomized QMC

With deterministic QMC methods, reliable error estimates are difficult to obtain. This has motivated the introduction of *randomized QMC* (RQMC), which essentially turns QMC into a variance-reduction technique (Owen 1998, L'Ecuyer and Lemieux 2002). The idea is to randomize P_n so that: (a) it retains its high uniformity when taken as a set and (b) each individual point has the uniform distribution over $[0, 1)^s$. Under these conditions, $E[Q_n] = \mu$, and hopefully $\mathrm{Var}[Q_n]$ will be smaller with RQMC than with MC.

To estimate the variance and compute a confidence interval on μ, we can apply m independent randomizations to the same P_n, and compute $\bar{X}_m$ and $S^2_{x,m}$, the sample mean and sample variance of the m corresponding (independent) values of Q_n. Then, $\mathrm{E}[\bar{X}_m] = \mu$ and $\mathrm{E}[S^2_{x,m}] = \mathrm{Var}[Q_n] = m\mathrm{Var}[\bar{X}_m]$ (L'Ecuyer and Lemieux 2000). A confidence interval on μ can be computed in a standard way if we assume that $\sqrt{m}(\bar{X}_m - \mu)/S_{x,m}$ has approximately the Student distribution with $m - 1$ degrees of freedom.

For a fixed computing budget (i.e., a fixed value of mn), a larger m gives a more precise variance estimator, but a larger n is likely to provide a more precise estimator of μ. So a compromise must be made. This is usually done by selecting m somewhere between 5 and 25. If our aim is really to estimate the variance (e.g., to measure the efficiency improvement of RQMC over MC), then we would usually take a larger m.

For certain classes of (smooth) functions f and specific classes of (uniform) point sets, variance bounds and asymptotic variance expressions (as a function of n, for either the worst-case f or an average f in some Hilbert space) have been developed for RQMC (Owen 1997, Hickernell and Wózniakowski 2001, Heinrich, Hickernell, and Yue 2004, Owen 2003); see also Section 3.

1.4 ANOVA Decomposition and Effective Dimension

In many practical settings, the s-dimensional function f can be well approximated by a sum of low-dimensional function, i.e., functions that depend only on a small number of coordinates of $\mathbf{u}$. Then, a *sufficient* condition for QMC to be efficient is that these low-dimensional functions are integrated with small error. For example, if $s = 1000$, a direct application of (1) is essentially useless, but if f can be written approximately as a sum of three-dimensional functions (i.e., where each function depends only on three coordinates of $\mathbf{u}$ or less), then (1), with P_n replaced by its appropriate three-dimensional projection in each case, can be used to show that the integration error for each of these three-dimensional functions converges rapidly, provided that all important three-dimensional projections of P_n have fast-converging discrepancy.

For RQMC, this can be studied generally and rigourously using a *functional ANOVA decomposition* of f (Owen 1998, Liu and Owen 2003). The idea is to write f as

$$f(\mathbf{u}) = \mu + \sum_{I \subseteq \{1,\ldots,s\},\, I \neq \phi} f_I(\mathbf{u})$$

where each f_I depends only on $\{u_i,\ i \in I\}$, the f_I's integrate to zero and are orthogonal, and the variance decomposes as $\sigma^2 = \sum_{I \subseteq \{1,\ldots,s\}} \sigma_I^2$ where $\sigma_I^2 = \mathrm{Var}[f_I(\mathbf{U})]$ for $\mathbf{U}$ uniformly distributed over $[0, 1)^s$. For each set of coordinates I, let $P_n(I)$ denote the projection of P_n over the subspace determined by I. For a given function f, if $\sum_{I \in \mathcal{J}} \sigma_I^2 \approx \sigma^2$, for some "small" class $\mathcal{J}$ of subsets of $\{1, \ldots, s\}$, then it suffices to construct P_n so that the projections $P_n(I)$ are highly uniform for all $I \in \mathcal{J}$, in order to reduce the important variance terms σ_I^2.

In this context, f is said to have *effective dimension d in proportion ρ in the superposition sense* if $\sum_{|I| \leq d} \sigma_I^2 \geq \rho\sigma^2$ (Owen 1998). If ρ is close to 1, this means that f is well approximated by a sum of d-dimensional (or less) functions. The following definitions reduce even further the number of projections that have to be considered.

Sometimes, $\sum_{I \in \mathcal{J}} \sigma_I^2$ is close to σ^2 if $\mathcal{J}$ contains all the sets I formed by indices that are not too far apart, and it suffices to have good uniformity for the corresponding projections. For example, if we simulate a single queue over a long time horizon, the random numbers used to generate the interarrival times and service times of customers that are close to each other in time have a much more important interaction than those that are far away. We say that f has effective dimension d in proportion ρ in the *successive-dimensions sense* if $\sum_{I \subseteq \{i,\ldots,i+d-1\},\, 0 \leq i \leq s-d} \sigma_I^2 \geq \rho\sigma^2$ (L'Ecuyer and Lemieux 2000).

There are cases where the first few random numbers of the simulation are much more important than the others. If $\sum_{I \subseteq \{1,\ldots,d\}} \sigma_I^2 \geq \rho\sigma^2$, then f has effective dimension d in

proportion ρ in the *truncation sense* (Caflisch, Morokoff, and Owen 1997). Low effective dimension in the truncation sense can sometimes be achieved by redesigning the simulation program (i.e., the function f) in a way that the first few uniforms account for most of the variance in f (Morokoff 1998, Caflisch et al. 1997, Fox 1999).

In practice, point sets are constructed by considering the uniformity of certain sets of projections. One natural requirement is that all projections contain as many distinct points as the original point set (this rules out rectangular grids, for example). Constructions for which several projections are identical also make the analysis easier. A point set P_n in $[0,1)^s$ is called *fully projection-regular* (Sloan and Joe 1994, L'Ecuyer and Lemieux 2000) if for each non-empty $I \subseteq \{1,\ldots,s\}$, $P_n(I)$ has n distinct points. It is called *dimension-stationary* (Lemieux and L'Ecuyer 2001) if whenever $1 \le i_1 < \ldots < i_\eta < s$ and $1 \le j \le s - i_\eta$, $P_n(\{i_1,\ldots,i_\eta\}) = P_n(\{i_1+j,\ldots,i_\eta+j\})$. Thus, $P_n(I)$ depends only on the *spacings* between the indices in I.

2 MAIN QMC CONSTRUCTIONS

2.1 Digital Nets and Sequences

To start with a simple case, suppose n is fixed and $s = 1$, so we just want to place n points to cover uniformly the unit interval $[0,1)$. A simple choice for P_n in this case is $P_n = \mathbb{Z}_n/n = \{0,\ 1/n,\ \ldots,\ (n-1)/n\}$. If $s > 1$, we may want to construct P_n in a way that each one-dimensional projection is $\mathbb{Z}_n/n$. In other words, for each dimension j, the jth coordinate of the points $\mathbf{u}_i$ will visit all n values in $\mathbb{Z}_n/n$, but in a different order for the different coordinates. Such a point set is defined by selecting a different permutation of $\mathbb{Z}_n/n$ (or equivalently, of the integers in $\mathbb{Z}_n$) for each coordinate j. The permutations must be different, because otherwise all the points would lie on a diagonal line in the unit hypercube. The trick is to select the permutations in a way that P_n has high uniformity over $[0,1)^s$, in a well-defined sense (to be determined).

Linear *digital nets* (Niederreiter 1992b) are construction methods that do precisely what we just described. They are defined as follows. Let $b \ge 2$ be an arbitrary integer, usually a prime, called the *base*. To define a net of $n = b^k$ points in s dimensions, we select s *generator matrices* $\mathbf{C}_0,\ldots,\mathbf{C}_{s-1}$, which are (in theory) $\infty \times k$ matrices whose elements are in $\mathbb{Z}_b = \{0,\ldots,b-1\}$. The matrix $\mathbf{C}_j$ is used for coordinate j of all the points, for $j \ge 0$. To define the ith point $\mathbf{u}_i$, for $i = 0,\ldots,b^k-1$, write the digital expansion of i in base b and multiply the vector of its digits by $\mathbf{C}_j$, modulo b, to obtain the digits of the expansion of $u_{i,j}$, the jth coordinate of $\mathbf{u}_i$. That is,

$$
\begin{aligned}
i &= \sum_{\ell=0}^{k-1} a_{i,\ell} b^\ell, \\
\begin{pmatrix} u_{i,j,1} \\ u_{i,j,2} \\ \vdots \end{pmatrix} &= \mathbf{C}_j \begin{pmatrix} a_{i,0} \\ a_{i,1} \\ \vdots \\ a_{i,k-1} \end{pmatrix} \bmod b, \\
u_{i,j} &= \sum_{\ell=1}^{\infty} u_{i,j,\ell} b^{-\ell}, \qquad (2) \\
\mathbf{u}_i &= (u_{i,0},\ldots,u_{i,s-1}).
\end{aligned}
$$

The point set thus obtained is a *digital net in base b*. In practice, the expansion (2) is truncated to r digits for some r, so each $\mathbf{C}_j$ becomes a $r \times k$ matrix. Typically, r is equal to k, or is slightly larger, or is selected so that b^r is near 2^{31}. If the first k lines of each $\mathbf{C}_j$ form a nonsingular $k \times k$ matrix (which we shall assume henceforth), then the n output values for coordinate j, truncated to their first k fractional digits in base b, are a permutation of $\mathbb{Z}_n/n$.

To obtain an *infinite sequence* of points, it suffices to define an infinite number of columns for each $\mathbf{C}_j$. This gives a *digital sequence in base b*. Only the first k columns are needed to get the first b^k points, for any k. Likewise, to obtain an infinite-dimensional point set, it suffices to have an infinite sequence of matrices $\mathbf{C}_j$. In both cases, the sequence of columns and matrices can be defined via recurrences (each column and matrix being a function of the previous ones). Concrete implementations are the sequences of Sobol' (1967), in base 2, of Faure (1982), in prime base b, of Niederreiter (1987), and of Niederreiter and Xing (1997).

Obviously, the *projection* $P_n(I)$ of a digital net over the subspace determined by a subset of indices $I = \{i_1,\ldots,i_d\}$ is also a digital net, and similarly for a digital sequence.

If the first k lines of $\mathbf{C}_j$ form the *reflected identity* (that is, the identity matrix but with its columns taken in reverse order, so that column c contains a 1 in line $k-c+1$ and 0's elsewhere, for $0 \le c < k$) and the other lines are zero, then coordinate j runs through the values $0, 1/n, \ldots, (n-1)/n$ in that order. If the first k lines of $\mathbf{C}_j$ form the *identity* and the other lines are zero, then the first n output values are the first n elements of the *van der Corput sequence* in base b, defined as $\psi_b(0), \psi_b(1), \psi_b(2), \ldots$, where $\psi_b : \mathbb{N} \to [0,1)$ is the *radical inverse* function in base b, defined by

$$\psi_b(i) = a_0 b^{-1} + a_1 b^{-2} + \cdots + a_{k-1} b^{-k}$$

if i is a k-digit integer in base b with digital b-ary expansion

$$i = a_0 + a_1 b + \cdots + a_{k-1} b^{k-1}.$$

The first n numbers of a van der Corput sequence fill up the unit interval quite uniformly for any large enough n.

It is customary to take $\mathbf{C}_0$ as the identity for digital sequences, and the reflected identity when n is fixed in advance (so the points are enumerated by order of their first coordinate).

2.2 Lattice Rules

An interesting special case of a digital net is when $k = r = 1$. Then, $n = b$, all coordinates are multiples of $1/n$, each matrix $\mathbf{C}_j$ contains a single integer $a_j \in \mathbb{Z}_b$, and the ith point can be written as $\mathbf{u}_i = [i(a_0, a_1, \ldots, a_{s-1}) \bmod n]/n$. This is one way of defining a s-dimensional *rank-1 lattice rule*.

In general, an *integration lattice* is a vector space of the form

$$L_s = \left\{ \mathbf{v} = \sum_{j=1}^{s} h_j \mathbf{v}_j \text{ such that each } h_j \in \mathbb{Z} \right\},$$

where $\mathbf{v}_1, \ldots, \mathbf{v}_s \in \mathbb{R}^s$ are linearly independent over $\mathbb{R}$ and where L_s contains $\mathbb{Z}^s$. The QMC approximation of μ with $P_n = L_s \cap [0,1)^s$ is a called a *lattice rule* (Sloan and Joe 1994). The *rank* of L_s is the smallest r such that one can find a basis of the form $\mathbf{v}_1, \ldots, \mathbf{v}_r, \mathbf{e}_{r+1}, \cdots, \mathbf{e}_s$, where $\mathbf{e}_j$ is the jth unit vector in s-dimensions. A lattice rule of rank 1 for which $\mathbf{v}_1 = (1,\ a,\ a^2 \bmod n,\ \ldots,\ a^{s-1} \bmod n)/n$ for some $a \in \mathbb{Z}_n$ is called a *Korobov* rule. The point set of a Korobov rule can also be written as $P_n = \{(x_0, \ldots, x_{s-1})/n$ such that $x_0 \in \mathbb{Z}_n$ and $x_j = a x_{j-1} \bmod n$ for all $j > 0\}$. This is the set of all vectors of successive values produced by a linear congruential generator (LCG) with modulus n and multiplier a, from all possible initial states (including 0). This LCG recurrence offers a convenient way to enumerate the points.

2.3 Hammersley Point Sets and Halton Sequence

Digital nets and sequences can be further generalized by allowing different bases for the different coordinates, say b_j for coordinate j. For example, the point sets introduced long ago by Hammersley (1960) have

$$\mathbf{u}_i = (i/n, \psi_{b_1}(i), \psi_{b_2}(i), \ldots, \psi_{b_{s-1}}(i)),$$

for $i = 0, \ldots, n-1$, where the basis b_j used for coordinate j is the jth smallest prime number. Here, $\mathbf{C}_0$ is the reflected identity and $\mathbf{C}_j$ is the identity for all $j > 0$. The corresponding infinite sequence, proposed by Halton (1960), takes

$$\mathbf{u}_i = (\psi_{b_1}(i), \psi_{b_2}(i), \ldots, \psi_{b_s}(i))$$

for all $i \geq 0$, where b_j is again the jth smallest prime. One drawback is that b_j becomes quite large for large j. In any case, the identity matrices $\mathbf{C}_j$ could also be replaced by more general generating matrices, which may give room to improve the uniformity.

2.4 Equidistribution and t-Value

Consider a digital net in base b with $n = b^k$ points. Suppose we partition the hypercube $[0,1)^s$ into $b^{q_1+\cdots+q_s}$ rectangular boxes of the same size by partitioning the jth axis into b^{q_j} equal parts for each j, for some integers $q_j \geq 0$. We call such a partition a $\mathbf{q}$-*equidissection in base* b, where $\mathbf{q} = (q_1, \ldots, q_s)$. If each box contains the same number of points of P_n, i.e., exactly b^t points where $t = k - q_1 - \cdots - q_s$, we say that P_n is $\mathbf{q}$-*equidistributed in base* b.

The set P_n is called a (t, k, s)-*net in base* b if it is $(q_1, \ldots, q_s)$-equidistributed for all non-negative integers $q_1, \ldots, q_s$ summing to $k - t$ (Niederreiter 1992b). We call the smallest such t the t-*value* of the net. The largest ℓ for which P_n is $(\ell, \ldots, \ell)$-equidistributed is called the s-*dimensional resolution* of P_n. This value cannot exceed $\lfloor k/s \rfloor$. These definitions also apply to the projections $P_n(I)$, for $I = \{i_1, \ldots, i_d\} \subset \{1, \ldots, s\}$. The *resolution gap* of $P_n(I)$ is $\delta_I = \lfloor k/d \rfloor - \ell_I$, where ℓ_I is the d-dimensional resolution of $P_n(I)$.

As measures of uniformity for digital nets, one may consider $\Delta_{\mathcal{J}} = \max_{I \in \mathcal{J}} \delta_I$ (worst-case resolution gap) or $\max_{I \in \mathcal{J}} t^*_{|I|}/t_I$ or $\max_{I \in \mathcal{J}} (t_I - t^*_{|I|})$, where $\mathcal{J}$ is a selected class of sets I, t_I is the t-value for $P_n(I)$, $t^*_{|I|}$ a lower bound on the best possible t-value in $|I|$ dimensions, and with the convention that $0/0 = 1$. The choice of $\mathcal{J}$ is arbitrary and a matter of compromise: If $\mathcal{J}$ contains too many sets, the selection criterion will be more costly to compute and the best value that it can achieve will be larger, which means that the criterion will be less demanding for the equidistribution of the more important projections.

For lattice rules, convenient measures of uniformity can be defined in terms of the *spectral test*, in essentially the same way as for linear congruential and multiple recursive random number generators (L'Ecuyer and Lemieux 2000).

Other uniformity criteria are discussed, e.g., in Sloan and Joe (1994), Hickernell (1998b), L'Ecuyer and Lemieux (2002), Heinrich, Hickernell, and Yue (2004).

2.5 Randomizations

A simple randomization method that satisfies the two conditions (a) and (b) given in Section 1.3, for an arbirary point

set P_n, is a *random shift modulo 1*: generate a *single* point $\mathbf{u}$ uniformly over $[0,1)^s$ and add it to each point of P_n, coordinate-wise, modulo 1. In the case of a lattice rule, the lattice structure of the points is preserved by the shift.

There is also a digital version of this method, called the *random digital shift in base b*: generate again a single $\mathbf{u} = (u_1, \ldots, u_s)$ uniformly over $[0,1)^s$, write the digital expansion in base b of each of its coordinates, say $u_j = \sum_{\ell=1}^{\infty} d_{j,\ell} b^{-\ell}$, then add $d_{j,\ell}$ modulo b to the ℓth digit of the digital expansion in base b of the jth coordinate of each point $\mathbf{u}_i \in P_n$. For $b = 2$, the digit-wise addition modulo b becomes a bitwise exclusive-or, which is fast to perform on a computer. An interesting property of the digital shift in base b is that for digital nets in base b, it preserves the $\mathbf{q}$-equidistribution properties for all vectors $\mathbf{q}$. In particular, it preserves the (t, k, s)-net properties.

These random shifts provide unbiased estimators of μ with a small amount of change in the point set. For certain types of point sets, however, a deeper scrambling (i.e., more randomization) can reduce the variance because the *average* point set over the class in which we randomize can have better uniformity in the larger class (with more randomization) than in the restricted class (random shift only).

Owen (1995) has proposed a *nested uniform scrambling* randomization method, for digital nets, which randomly permutes the values $\{0, \ldots, b-1\}$ used for the digits $u_{i,j,\ell}$, independently across the coordinates and across the digits. Owen (1997) has shown that for functions whose mixed partial derivatives satisfy a Lipschitz condition, with a (t, k, s)-net scrambled in this way, the variance is in $O(n^{-3}(\log n)^s)$. However, this method requires $(1 + b + \cdots + b^{\ell-1})s$ independent permutations to scramble the first ℓ digits, and is therefore very time-consuming.

A class of less expensive approaches that perform well, for digital nets, are the *linear matrix scrambles* (Matoušek 1999, Faure and Tezuka 2002, Hong and Hickernell 2003, Owen 2003), which multiply the matrices $\mathbf{C}_j$ by a random invertible matrix $\mathbf{M}_j$, modulo b. When this is combined with a digital random shift modulo b, we get an *affine matrix scramble* (Owen 2003), which satisfies our two conditions (a) and (b). There are several variants, depending on how $\mathbf{M}_j$ is generated, and on whether $\mathbf{C}_j$ is multiplied on the left or on the right. Here we consider only multiplication by $\mathbf{M}_j$ on the left. In a version proposed by Matoušek (1999) and Hong and Hickernell (2003), which we simply call *left matrix scramble*, $\mathbf{M}_j$ is a $w \times w$ nonsingular lower-triangular matrix whose diagonal entries are generated uniformly over $\{1, \ldots, b-1\}$, whereas the entries below the diagonal are generated uniformly over $\{0, \ldots, b-1\}$, all independently. The *i-binomial matrix scramble* of Tezuka and Faure (2002) adds the constraint that all entries on any given diagonal or subdiagonal of $\mathbf{M}_j$ are identical. The *striped matrix scramble* proposed by Owen (2003) adds the constraint that in any given column, all entries below the diagonal are equal to the diagonal entry, which is generated randomly over $\{1, \ldots, b-1\}$.

A simple technique that often reduces the variance for randomly-shifted lattice rules is the *baker transformation*, which replaces each coordinate u (after the shift) by $2u$ if $u \le 1/2$ and by $2(1-u)$ if $u > 1/2$. This corresponds to streching everything by a factor of two and folding, along each coordinate. Hickernell (2002) has show that it can reduce the variance to $O(n^{-4+\epsilon})$ for non-periodic smooth functions.

Wang and Hickernell (2000) randomize the Halton sequence simply by selecting the starting point $\mathbf{u}_0$ randomly over $[0,1)^s$, and exploiting the fact that there is a simple way of getting $\psi_b(i+1)$ directly from $\psi_b(i)$, so the successive points can be generated without knowing their indices i in the original sequence. They show that this method satisfies conditions (a) and (b) of Section 1.3. In their numerical experiments, it performs much better than randomly shifting (modulo 1) the Halton sequence.

2.6 Transforming the Function f

Besides randomizing and improving the "average" uniformity of the points, another way of making RQMC more effective is to transform the function f so that its expectation and variance remain the same in the MC setting, but its effective dimension (in some sense) is reduced. In the context of financial applications, this can be achieved via bridge sampling techniques, principal component analysis, using conditional distributions, generating the random numbers in a different order, etc. (Moskowitz and Caflisch 1996, Fox 1999, Glasserman 2004). We illustrate some of these methods in our examples, in Section 3.

3 EXAMPLES

3.1 Functions of a Multinormal Vector

Suppose that our integrand of interest can be written as a function of a vector $\mathbf{Y} = (Y_1, \ldots, Y_s)$ of s normal random variables, say with mean zero (without loss of generality) and covariance matrix $\mathbf{\Sigma}$. That is, $\mu = E[g(\mathbf{Y})]$ for a known function g, and $g(\mathbf{Y})$ is the estimator. Consider, for example, a basket of c assets whose values evolve as (potentially correlated) geometric Brownian motions (GMBs). Suppose that the net payoff at time T is a function g of the assets values at fixed observation times $0 < t_1 < \cdots < t_d = T$. This covers a wide range of option types and fits the above setting with $s = cd$.

To generate $\mathbf{Y}$, it suffices to decompose the covariance matrix as $\mathbf{\Sigma} = \mathbf{C}\mathbf{C}^t$ for some matrix $\mathbf{C}$, generate a vector $\mathbf{Z} = (Z_1, \ldots, Z_s)$ of s i.i.d. $N(0,1)$ (standard normal) random variates, and return $\mathbf{Y} = \mathbf{C}\mathbf{Z}$. But there are in

general many possibilities for the choice of factorization (i.e., choice of $\mathbf{C}$). The most common one, the *Cholesky factorization*, takes $\mathbf{C}$ to be lower triangular. Another one, based on a standard *principal component analysis* (PCA), selects $\mathbf{C}$ so that Z_1 accounts for the maximum amount of variance, then Z_2 for the maximum amount of variance conditional on Z_1, and so on. In other words, the method packs the variance in the first coordinates of $\mathbf{Z}$ as much as possible. This yields $\mathbf{C} = \mathbf{P}\mathbf{D}^{1/2}$ where $\mathbf{D}$ is a diagonal matrix that contains the eigenvalues of $\mathbf{\Sigma}$ in decreasing order and $\mathbf{P}$ is an orthogonal matrix whose columns are the corresponding unit-length eigenvectors. Acworth, Broadie, and Glasserman (1997) proposed PCA as a tool to reduce the effective dimension in the truncation sense, in the context of simulating GMBs for option pricing via QMC. In their examples, it was shown to be quite effective.

This PCA technique, however, does not take into account the function g. It may turn out that with the PCA sampling scheme, $g(\mathbf{Y})$ depends very little on Z_1 and very much on Z_{25}, for example, even if Z_1 has a much greater influence on the variance of $\mathbf{Y}$. In such a situation, PCA will miss its target. A more general (and generally better) approach would be to find a decomposition $\mathbf{C}\mathbf{C}^{\mathrm{t}}$ that *minimizes* the effective dimension of the integrand $f(\mathbf{U})$ in some sense. Imai and Tan (2002, 2004) explain how to implement such an approach for the special case of an European option whose payoff is based on an geometric or arithmetic average, for a basket of assets whose prices evolve as GMBs. They write $\mathbf{C} = \mathbf{C}^{\mathrm{ch}}\mathbf{A}$ where $\mathbf{C}^{\mathrm{ch}}$ is lower triangular (the Cholesky matrix) and $\mathbf{A}$ is orthogonal, and optimize the first column of $\mathbf{A}$, then the second, and so on, trying to catch as much of the residual variance of $g(\mathbf{Y})$ at each step, i.e., so that the sum of terms with $|I| = \nu$ is as small as possible for $\nu = 1$, then for $\nu = 2$ conditional on what was selected before, etc. This technique is not necessarily easy to implement in general, however.

In the special case where $c = 1$ and where the payoff depends only on the values taken by a single Brownian motion $\{X(t),\ t \geq 0\}$ at observation times $0 = t_0 < t_1 < \cdots < t_d = T$, then the most direct way of simulating the payoff is the following *sequential sampling* (or *random walk*) method: generate $X(t_1)$, then the increment $X(t_2) - X(t_1)$, then $X(t_3) - X(t_2)$, and so on. All these increments are independent normals. A second method is *bridge sampling* (for notational simplicity, we assume here that d is a power of two): generate first $X(t_d)$, then $X(t_{d/2})$ conditional on $(X(0), X(t_d))$, then $X(t_{d/4})$ conditional on $(X(0), X(t_{d/2}))$, then $X(t_{3d/4})$ conditional on $(X(t_{d/2}), X(t_d))$, then $X(t_{d/8})$ conditional on $(X(0), X(t_{d/4}))$, and so on. These conditional distributions are all independent normals with known parameters. Each of the these two sampling methods implements (implicitly) a special case of the general decomposition outlined above, each one with its corresponding matrix $\mathbf{C}$. Intuitively, the bridge sampling is likely to pack more variance in the first few coordinates of $\mathbf{Z}$, because the first few random numbers already sketch the general shape of the trajectory whereas the last ones are only making minor adjustment to it. Using bridge sampling to reduce the effective dimension was proposed by Moskowitz and Caflisch (1996) in the context of GMB simulation via QMC. It turns out to be quite effective, as demonstrated by these authors and several others.

3.2 Numerical Illustrations with GMB Processes

We now give some numerical illustrations with examples taken from Glasserman (2004), Section 5.5.1, and from Imai and Tan (2002), but comparing a different set of RQMC methods. Consider a set of c GMB processes, $\{S_i(t),\ t \geq 0\}$, $1 \leq i \leq c$, where S_i has *drift parameter* μ_i and *volatility parameter* σ_i. That is,

$$S_i(t) = S_i(0) \exp\left[(\mu_i - \sigma_i^2/2)t + \sigma_i W_i(t)\right]$$

where W_i is a standard Brownian motion (whose increments over disjoint intervals are independent and are normal with mean 0 and variance δ over an interval of length δ). Moreover, we assume that the W_i's are correlated as follows: $\mathrm{Cov}[W_i(t+\delta) - W_i(t),\ W_j(t+\delta) - W_j(t)] = \rho_{i,j}\delta$ for all $\delta > 0$. We have an option whose discounted payoff is $e^{-rT}\max[\bar{S} - K,\ 0]$, where $\bar{S}$ can be either the geometric average

$$\bar{S}^{(\mathrm{G})} = \prod_{i=1}^{c}\prod_{j=1}^{d} S_i(t_j)^{1/cd} \tag{3}$$

or the arithmetic average

$$\bar{S}^{(\mathrm{A})} = \frac{1}{cd}\sum_{i=1}^{c}\sum_{j=1}^{d} S_i(t_j), \tag{4}$$

for fixed observation times $0 < t_1 < \cdots < t_d = T$. In all our examples, we take $t_j = jT/d$. Denoting $\mathbf{Y} = (W_1(t_1), \ldots, W_c(t_1), W_1(t_2), \ldots, W_c(t_2), \ldots, W_1(t_d), \ldots, W_c(t_d))^{\mathrm{t}}$, the element $((i-1)c + j), (i'-1)c + j')$ of Σ is $\rho_{i,i'}\sigma_i\sigma_{i'}|t_{j'} - t_{j-1}|)$ for $j' \geq j$.

Example 1 For our first numerical illustration, we take c independent assets with a single observation time, with the following parameters borrowed from Glasserman (2004): $d = 1$, $\rho_{i,j} = 0$, $T = 1$, $\sigma = 0.5$, $r = 0.05$, $S(0) = 100$, and $K = 100$. We first take $c = 5$, then $c = 10$. We consider the payoff based on the arithmetic average (4). The exact values and the MC variance per observation (the values of μ and σ^2) are approximately 11.72 and 305 for $c = 5$, and 9.207 and 145 for $c = 10$.

Table 1 gives the empirical variance reduction factors observed for (a) Sobol' nets with a random digital shift (DS)

only, (b) Sobol' nets with a left matrix scrambling (LMS) followed by DS, (c) Korobov lattice rules with a random shift modulo 1, and (d) Korobov lattice rules with a random shift modulo 1 followed by a baker transformation. The list of primitive polynomials and the direction numbers for the Sobol' sequence were taken from (Lemieux, Cieslak, and Luttmer 2004). The lattice rule parameters are from (L'Ecuyer and Lemieux 2000). The *variance reduction factor* (VRF) is defined as the Monte Carlo variance (per observation) divided by n times the variance of Q_n for the randomized QMC method. Everywhere in this paper, the RQMC variance was estimated by making $m = 100$ independent replications of the randomization. These VRFs are noisy, with a standard error of 10 percent or more. The simulations were written in Java using the SSJ library (L'Ecuyer 2004), which contains implementations of RQMC methods and facilities to generate stochastic processes encountered in finance, among other things. The list of primitive polynomials and the direction numbers for the Sobol' sequence were taken from (Lemieux, Cieslak, and Luttmer 2004). The lattice rule parameters are from (L'Ecuyer and Lemieux 2000). All random variables were generated by inversion.

Table 1: Variance Reduction Factors for Example 1 for $c = 5$ (Left Number) and $c = 10$ (Right Number)

Sobol' Nets

	$n = 2^{14}$	$n = 2^{16}$	$n = 2^{18}$
DS	953, 168	2363, 162	7156, 180
LMS+DS	733, 112	2265, 174	7058, 253

Korobov Lattice Rules

	$n = 16381$ $a = 5693$	$n = 65521$ $a = 944$	$n = 262139$ $a = 21876$
Shift	178, 74	312, 21	416, 117
Shift+baker	376, 77	440, 89	3434, 425

We see that the VRFs (i.e., efficiency gains) increase rapidly with n for the 5-dimensional problem, especially for the Sobol' net, but much less rapidly in the 10-dimensional case. For the Korobov rules, the baker transformation helps significantly, but the Sobol' nets are doing even better, with or without LMS. For $c = 5$ and $n = 2^{18}$, they reduce the variance by a factor of 7000 compared with ordinary MC. All methods require approximately the same CPU time for a given value of n. This means that they require (at least) 7000 less CPU time than MC to compute an estimator of comparable precision. We also tried the i-binomial scrambling and the striped matrix scrambling in place of LMS for the Sobol' nets and we observed no significant difference. With the Faure nets (in their original versions), we obtained much smaller efficiency improvements than with the Sobol' nets, using LMS+DS.

In Table 2, we find VRFs for the same example, except that $\rho_{i,j} = 0$ is replaced by $\rho_{i,j} = 0.4$ for $i \neq j$. All values are for $c = 10$, for which we have $\mu \approx 15.77$ and $\sigma^2 \approx 674$. We compare two ways of sampling the vector $\mathbf{Y}$ by transforming a 250-dimensional vector of independent standard normals: (a) the usual Cholesky factorization (left number in each table entry) and (b) PCA (right number in each table entry). PCA definitely outperforms the standard Cholesky factorization, and the combination of PCA with randomized Sobol' nets gives the largest VRFs.

Table 2: Variance Reduction Factors for Example 1(b) for Cholesky (Left Number) and PCA (Right Number)

Sobol' Nets

	$n = 2^{14}$	$n = 2^{16}$	$n = 2^{18}$
DS	305, 10800	640, 19280	971, 50180
LMS+DS	418, 5834	519, 13530	798, 34880

Korobov Lattice Rules

	$n = 16381$ $a = 5693$	$n = 65521$ $a = 944$	$n = 262139$ $a = 21876$
Shift	90, 890	32, 2135	198, 6276
Shift+baker	222, 6978	242, 8289	626, 16530

Example 2 For our second illustration, we slightly modify an example from Imai and Tan (2002): we take $c = 10$, $d = 25$ (they have $d = 250$), $\rho_{i,j} = 0.4$ for all $i \neq j$, $T = 1$, $\sigma_i = 0.1 + 0.4(i-1)/9$ for all i, $r = 0.04$, $S(0) = 100$, and $K = 100$. This gives a 250-dimensional integration problem. The exact value and the MC variance are $\mu \approx 5.818$ and $\sigma^2 \approx 72.3$.

The results are in Table 3, in the same format as for Table 2. They are similar. The main difference is that here we have a 250-dimensional problem instead of a 10-dimensional one, so PCA has more room to reduce the effective dimension. The VRFs are much smaller than in Table 2 with Cholesky, but the improvement provided by PCA over Cholesky is much larger.

Table 3: Variance Reduction Factors for Example 2 (250 Dimensions) with Cholesky (Left) and PCA (Right)

Sobol' Nets

	$n = 2^{14}$	$n = 2^{16}$	$n = 2^{18}$
DS	12, 4188	21, 7545	41, 12580
LMS+DS	12, 4382	24, 7951	49, 14780

Korobov Lattice Rules

	$n = 16381$ $a = 5693$	$n = 65521$ $a = 944$	$n = 262139$ $a = 21876$
Shift	15, 1128	14, 1524	10, 2141
Shift+baker	46, 2184	54, 5478	49, 4232

Example 3 Here we consider an Asian option on a single asset ($c = 1$) whose price follows a GMB process. The payoff is based on the arithmetic average (4). In this context, Kemna and Vorst (1990) suggested using the payoff

based on the geometric average (3), whose expectation is known, as a control variate (CV) to reduce the variance. Here, we look at the improvement of RQMC over MC with and without the CV, with sequential sampling, Brownian bridge sampling, and PCA, for an example with $S(0) = 100$, $T = 1$, $r = \ln(1.09)$, for $d = 10$ and 120, and for $K = 90$ and 100.

Table 4 gives estimates of the exact value μ and of the MC variance σ^2 with and without the CV. We immediately see that the CV alone (without RQMC) reduces the variance by a huge factor, especially when d is small. For $K = 90$ and $d = 10$, for example, the VRF is approximately $105.2/6.89 \times 10^{-5} \approx 1.53 \times 10^6$.

Table 4: Estimates of μ and σ^2 for Example 3

K	d	μ	σ^2 without CV	σ^2 with CV
90	10	13.844	105.2	6.89E-5
	120	11.532	41.4	8.17E-3
100	10	7.031	61.5	5.97E-5
	120	3.464	20.2	6.30E-3

Table 5 gives the VRFs of RQMC over MC, with and without the CV. In each table entry, the left value is for sequential sampling the middle one is for bridge sampling, and the right one is for PCA. The optimal CV coefficient was estimated by pilot runs for MC and for each combination of sampling scheme and RQMC method. It is important to recall that the optimal coefficient may vary a lot when we change these methods, because it depends on the estimator's variance and its covariance with the CV, which may vary significantly across the methods (Hickernell, Lemieux, and Owen 2004).

Without the CV, RQMC reduces the variance by a huge factor, especially when combined with bridge sampling or PCA. With the CV, significant *additional* VRFs are obtained by the RQMC methods on top of those obtained by the CV alone. As an illustration, for $K = 90$ and $d = 10$, Sobol' with PCA, LMS+DS, and $n = 2^{18}$ points reduces the MC+CV variance by a factor of around 45000, whereas the CV alone was already providing a VRF of around 1.53×10^6. By combining them, we reduce the variance approximately by a factor of 6.9×10^{10}. The CPU times per run are roughly the same in this case, so plain (naive) MC would take 6.9×10^{10} times more CPU time to yield an estimator with equivalent precision. For $d = 120$, PCA sampling needs around 50% more CPU time per run than MC. In most cases, the randomly shifted Korobov point sets *with the baker transformation* are competitive with the randomized Sobol' nets.

3.3 An Asian Option Under a Variance Gamma Process

We consider now an asset price that evolves according to a *variance-gamma* (VG) process S defined as follows (Madan, Carr, and Chang 1998, Avramidis, L'Ecuyer, and Tremblay 2003):

$$S(t) = S(0)\exp\{rt + B(G(t; 1, \nu), \theta, \sigma) + \omega t\},$$

where B is a *Brownian process* with drift and variance parameters θ and σ, G is a *gamma process* (a process with independent gamma increments) with mean and variance parameters 1 and ν, B and G are independent, and $\omega = \ln(1 - \theta\nu - \sigma^2\nu/2)/\nu$. We want to estimate by simulation the value of an *Asian call option*, given by $E[e^{-rT}\max(\bar{S} - K, 0)]$ where $\bar{S} = (1/d)\sum_{j=1}^{d} S(t_j)$ and $t_j = jT/d$ for $0 \le j \le d$.

Here, the vector $(S(t_1), \ldots, S(t_d))$ is not multinormal, so the general setting of the previous subsection does not apply. However, the processes G and B (and therefore S) can be generated by either sequential or bridge sampling, as explained in Avramidis, L'Ecuyer, and Tremblay (2003) and Ribeiro and Webber (2002). In the *sequential sampling* approach (BGSS), we generate $\tau_1 = G(t_1)$, $B(\tau_1)$, $\tau_2 = G(t_2)$, $B(\tau_2)$, etc., in that order. This requires the generation of d gamma variates and d normal variates, all independent. In the *bridge sampling* approach (BGBS), we generate $\tau_d = G(t_d)$, $B(\tau_d)$, $\tau_{d/2} = G(t_{d/2})$, $B(\tau_{d/2})$, $\tau_{d/4} = G(t_{d/4})$, $B(\tau_{d/4})$, $\tau_{3d/4} = G(t_{3d/4})$, $B(\tau_{3d/4})$, ..., in that order. This is easy to implement because for any given values $t_a < t < t_b$ and $\tau_a < \tau < \tau_b$, the distribution of $G(t)$ conditional on $(G(t_a), G(t_b))$ is beta with known parameters, and the distribution of $B(\tau)$ conditional on $(B(\tau_a), B(\tau_b))$ is normal with known parameters. This method requires the generation of one gamma variate, $d-1$ beta variates, and d normal variates. Yet another method, explained in Avramidis, L'Ecuyer, and Tremblay (2003), is *double gamma bridge sampling* (DGBS). It requires two gamma variates and $2d - 2$ beta variates.

Example 4 For a numerical illustration, we take the following parameters from Avramidis, L'Ecuyer, and Tremblay (2003): $\theta = -0.1436$, $\sigma = 0.12136$, $\nu = 0.3$, $r = 0.1$, $T = 1$, $K = 101$, and $S(0) = 100$. The exact value and the MC variance are $\mu \approx 6.065$ and $\sigma^2 \approx 33.22$. Table 6 gives the variance reduction factors of QMC compared with MC. DGBS provides the best improvement.

Table 5: Variance Reduction Factors for Example 3 *Without CV*, with Sequential Sampling (Left), Bridge Sampling (Middle), and PCA (Right)

Sobol' Nets *without CV*

K	d		$n = 2^{14}$	$n = 2^{16}$	$n = 2^{18}$
90	10	DS	33930, 69240, 48960	33560, 3.1E+5, 97330	1.6E+5, 9.1E+5, 4.9E+5
		LMS+DS	24290, 56290, 20110	40730, 2.2E+5, 1.7E+5	1.3E+5, 6.3E+5, 5.0E+5
	120	DS	209, 14120, 2.1E+5	318, 15080, 3.7E+5	660, 34590, 1.3E+6
		LMS+DS	334, 10350, 1.3E+5	307, 18090, 4.3E+5	406, 31090, 1.8E+6
100	10	DS	19240, 62190, 23100	18590, 1.1E+5, 1.4E+5	66240, 3.7E+5, 3.7E+5
		LMS+DS	10620, 32650, 24180	21260, 96810, 1.2E+5	42690, 3.0E+5, 3.2E+5
	120	DS	29, 3002, 69170	34, 3895, 1.6E+5	44, 5607, 4.3E+5
		LMS+DS	33, 2472, 53590	51, 3612, 1.7E+5	56, 4908, 4.5E+5

Korobov Lattice Rules *without CV*

K	d		$n = 16381$ $a = 5693$	$n = 65521$ $a = 944$	$n = 262139$ $a = 21876$
90	10	Shift	2614, 2962, 1016	6908, 5683, 541	13860, 17630, 1432
		Shift+baker	4186, 40980, 31830	78120, 4.3E+5, 32730	29380, 1.1E+6, 2.4E+5
	120	Shift	212, 1392, 3739	206, 1592, 10310	248, 3682, 38060
		Shift+baker	505, 2520, 1.2E+5	373, 11910, 3.4E+5	362, 20000, 9.1E+5
100	10	Shift	1769, 1935, 1314	1889, 4109, 818	5182, 10360, 1507
		Shift+baker	1815, 17370, 22860	28220, 28540, 17140	26680, 1.9E+5, 1.9E+5
	120	Shift	32, 682, 1731	29, 538, 5735	27, 982, 28510
		Shift+Baker	41, 1050, 32660	45, 1744, 1.3E+5	36, 4343, 4.3E+5

Sobol' Nets *with CV* (compared to MC with CV)

		$d = 10$			$d = 120$		
K		$n = 2^{14}$	$n = 2^{16}$	$n = 2^{18}$	$n = 2^{14}$	$n = 2^{16}$	$n = 2^{18}$
90	DS	60, 120, 7519	100, 232, 10950	174, 160, 88540	3, 48, 275	4, 57, 542	5, 62, 636
	LMS+DS	54, 160, 3020	100, 156, 13190	115, 200, 45260	4, 45, 350	3, 66, 423	4, 81, 572
100	DS	25, 52, 2346	41, 71, 13000	38, 148, 43230	4, 41, 160	4, 48, 173	3, 47, 504
	LMS+DS	27, 63, 3026	33, 80, 9115	47, 104, 28700	4, 43, 155	3, 40, 254	2, 48, 418

Korobov Lattice Rules *with CV* (compared to MC with CV)

		$d = 10$			$d = 120$		
K		$n = 16381$ $a = 5693$	$n = 65521$ $a = 944$	$n = 262139$ $a = 21876$	$n = 16381$ $a = 5693$	$n = 65521$ $a = 944$	$n = 262139$ $a = 21876$
90	Shift	18, 38, 114	15, 43, 46	19, 80, 108	4, 39, 93	4, 45, 83	6, 30, 118
	Shift+B.	39, 71, 7080	94, 202, 2837	79, 211, 22710	8, 46, 258	7, 61, 205	12, 42, 400
100	Shift	10, 19, 216	13, 38, 81	12, 57, 253	4, 19, 41	4, 23, 27	4, 24, 134
	Shift+B.	12, 50, 2064	36, 52, 930	47, 69, 12380	7, 29, 88	9, 36, 141	7, 47, 307

Table 6: Variance Reduction Factors for Example 4 with BGSS (Left), BGBS (middle), and DGBS (right)

Sobol' Nets

	$n = 2^{14}$	$n = 2^{16}$	$n = 2^{18}$
DS	39, 322, 903	98, 755, 1134	66, 1340, 2760
LMS+DS	56, 222, 329	125, 791, 1031	75, 1165, 2031

Korobov Lattice Rules

	$n = 16381$ $a = 5693$	$n = 65521$ $a = 944$	$n = 262139$ $a = 21876$
Shift	39, 97, 151	8, 14, 254	40, 208, 680
Shift+bak.	75, 343, 72	33, 72, 672	131, 662, 2596

ACKNOWLEDGMENTS

This work has been supported by NSERC-Canada grant No. ODGP0110050, NATEQ-Québec grant No. 02ER3218, and a Canada Research Chair to the author. Boni Abdel Chabi-Yo, Chiheb Dkhil, and Pierre-Alexandre Tremblay helped writing and running the programs for the numerical experiments.

REFERENCES

Acworth, P., M. Broadie, and P. Glasserman. 1997. A comparison of some Monte Carlo and quasi-Monte Carlo techniques for option pricing. In *Monte Carlo and Quasi-Monte Carlo Methods 1996*, ed. P. Hellekalek and H. Niederreiter, *Lecture Notes in Statistics* 127, 1–18. Springer-Verlag.

Avramidis, T., P. L'Ecuyer, and P.-A. Tremblay. 2003. Efficient simulation of gamma and variance-gamma processes. In *Proceedings of the 2003 Winter Simulation Conference*, ed. S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, 319–326. Piscataway, New Jersey: IEEE Press.

Caflisch, R. E., W. Morokoff, and A. Owen. 1997. Valuation of mortgage-backed securities using Brownian bridges to reduce effective dimension. *The Journal of Computational Finance* 1 (1): 27–46.

Faure, H. 1982. Discrépance des suites associées à un système de numération en dimension s. *Acta Arithmetica* 61:337–351.

Faure, H., and S. Tezuka. 2002. Another random scrambling of digital (t, s)-sequences. In *Monte Carlo and Quasi-Monte Carlo Methods 2000*, ed. K.-T. Fang, F. J. Hickernell, and H. Niederreiter, 242–256. Berlin: Springer-Verlag.

Fox, B. L. 1999. *Strategies for quasi-Monte Carlo*. Boston, MA: Kluwer Academic.

Glasserman, P. 2004. *Monte Carlo methods in financial engineering*. New York: Springer-Verlag.

Halton, J. H. 1960. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik* 2:84–90.

Hammersley, J. M. 1960. Monte Carlo methods for solving multivariate problems. *Annals of the New York Academy of Science* 86:844–874.

Heinrich, S., F. J. Hickernell, and R. X. Yue. 2004. Optimal quadrature for Haar wavelet spaces. *Mathematics of Computation* 73:259–277.

Hickernell, F. J. 1998a. A generalized discrepancy and quadrature error bound. *Mathematics of Computation* 67:299–322.

Hickernell, F. J. 1998b. Lattice rules: How well do they measure up? In *Random and Quasi-Random Point Sets*, ed. P. Hellekalek and G. Larcher, *Lecture Notes in Statistics* 138, 109–166. New York: Springer.

Hickernell, F. J. 2002. Obtaining $o(n^{-2+\epsilon})$ convergence for lattice quadrature rules. In *Monte Carlo and Quasi-Monte Carlo Methods 2000*, ed. K.-T. Fang, F. J. Hickernell, and H. Niederreiter, 274–289. Berlin: Springer-Verlag.

Hickernell, F. J., C. Lemieux, and A. B. Owen. 2004. Control variates for quasi-Monte Carlo. *Statistical Science*. to appear.

Hickernell, F. J., and H. Wózniakowski. 2001. The price of pessimism for multidimensional quadrature. *Journal of Complexity* 17:625–659.

Hong, H. S., and F. H. Hickernell. 2003. Algorithm 823: Implementing scrambled digital sequences. *ACM Transactions on Mathematical Software* 29:95–109.

Imai, J., and K. S. Tan. 2002. Enhanced quasi-Monte Carlo methods with dimension reduction. In *Proceedings of the 2002 Winter Simulation Conference*, ed. E. Yücesan, C. H. Chen, J. L. Snowdon, and J. M. Charnes, 1502–1510. Piscataway, New Jersey: IEEE Press.

Imai, J., and K. S. Tan. 2004. Minimizing effective dimension using linear transformation. In *Monte Carlo and Quasi-Monte Carlo Methods 2002*, ed. H. Niederreiter, 275–292. Berlin: Springer-Verlag.

Kemna, A. G. Z., and A. C. F. Vorst. 1990. A pricing method for options based on average asset values. *Journal of Banking and Finance* 14:113–129.

L'Ecuyer, P. 2004. *SSJ: A Java library for stochastic simulation*. Available at <http://www.iro.umontreal.ca/~lecuyer>.

L'Ecuyer, P., and C. Lemieux. 2000. Variance reduction via lattice rules. *Management Science* 46 (9): 1214–1235.

L'Ecuyer, P., and C. Lemieux. 2002. Recent advances in randomized quasi-Monte Carlo methods. In *Modeling Uncertainty: An Examination of Stochastic Theory, Methods, and Applications*, ed. M. Dror, P. L'Ecuyer, and F. Szidarovszki, 419–474. Boston: Kluwer Academic Publishers.

Lemieux, C., and P. L'Ecuyer. 2001. Selection criteria for lattice rules and other low-discrepancy point sets. *Mathematics and Computers in Simulation* 55 (1–3): 139–148.

Lemieux, C., M. Cieslak, and K. Luttmer. 2004. *RandQMC User's Guide: A Package for Randomized Quasi-Monte Carlo Methods in C*. Available at <http://www.math.ucalgary.ca/~lemieux>.

Liu, R., and A. B. Owen. 2003. Estimating mean dimensionality. manuscript.

Madan, D. B., P. P. Carr, and E. C. Chang. 1998. The variance gamma process and option pricing. *European Finance Review* 2:79–105.

Matoušek, J. 1999. *Geometric discrepancy: An illustrated guide*. Berlin: Springer-Verlag.

Morokoff, W. J. 1998. Generating quasi-random paths for stochastic processes. *SIAM Review* 40 (4): 765–788.

Moskowitz, B., and R. E. Caflisch. 1996. Smoothness and dimension reduction in quasi-Monte Carlo methods. *Journal of Mathematical and Computer Modeling* 23:37–54.

Niederreiter, H. 1987. Point sets and sequences with small discrepancy. *Monatshefte für Mathematik* 104:273–337.

Niederreiter, H. 1992b. *Random number generation and quasi-Monte Carlo methods*, Volume 63 of *SIAM CBMS-NSF Regional Conference Series in Applied Mathematics*. Philadelphia: SIAM.

Niederreiter, H., and C. Xing. 1997. The algebraic-geometry approach to low-discrepancy sequences. In *Monte Carlo and Quasi-Monte Carlo Methods 1996*, ed. P. Hellekalek, G. Larcher, H. Niederreiter, and P. Zinterhof, Volume 127 of *Lecture Notes in Statistics*, 139–160. New York: Springer-Verlag.

Owen, A. B. 1995. Randomly permuted (t, m, s)-nets and (t, s)-sequences. In *Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, ed. H. Niederreiter and P. J.-S. Shiue, Number 106 in *Lecture Notes in Statistics*, 299–317. Springer-Verlag.

Owen, A. B. 1997. Scrambled net variance for integrals of smooth functions. *Annals of Statistics* 25 (4): 1541–1562.

Owen, A. B. 1998. Latin supercube sampling for very high-dimensional simulations. *ACM Transactions on Modeling and Computer Simulation* 8 (1): 71–102.

Owen, A. B. 2003. Variance with alternative scramblings of digital nets. *ACM Transactions on Modeling and Computer Simulation* 13 (4): 363–378.

Ribeiro, C., and N. Webber. 2002, December. Valuing path-dependent options in the variance-gamma model by Monte Carlo with a gamma bridge. manuscript.

Sloan, I. H., and S. Joe. 1994. *Lattice methods for multiple integration*. Oxford: Clarendon Press.

Sobol', I. M. 1967. The distribution of points in a cube and the approximate evaluation of integrals. *U.S.S.R. Journal of Computational Mathematics and Mathematical Physics* 7:86–112.

Tezuka, S., and H. Faure. 2002. i-binomial scrambling of digital nets and sequences. Technical report, IBM Research, Tokyo Research Laboratory.

Wang, X., and F. J. Hickernell. 2000. Randomized Halton sequences. *Mathematical and Computer Modelling* 32:887–899.

AUTHOR BIOGRAPHY

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. 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 obtained the prestigious *E. W. R. Steacie* fellowship in 1995-97 and a *Killam* fellowship in 2001-03. His recent research articles are available on-line at <http://www.iro.umontreal.ca/~lecuyer>.