Available online at www.sciencedirect.com
SCIENCE DIRECT®

ELSEVIER Journal of Economic Dynamics & Control 28 (2004) 2399–2426

JOURNAL OF Economic Dynamics & Control

www.elsevier.com/locate/econbase

# Randomized quasi-Monte Carlo methods in pricing securities

Giray Ökten[a,*], Warren Eastman[b]

[a] *Department of Mathematical Sciences, Ball State University, Muncie, IN 47306, USA*
[b] *Avionics Test and Analysis Corporation, Niceville, FL 32578, USA*

Received 14 August 2002; accepted 3 November 2003

**Abstract**

Today quasi-Monte Carlo methods are used successfully in computational finance and economics as an alternative to the Monte Carlo method. One drawback of these methods, however, is the lack of a practical way of error estimation. To address this issue several researchers introduced the so-called randomized quasi-Monte Carlo methods in the last decade. In this paper we will present a survey of randomized quasi-Monte Carlo methods, and compare their efficiencies with the efficiency of the Monte Carlo method in pricing certain securities. We will also investigate the effects of Box–Muller and inverse transformation techniques when they are applied to low-discrepancy sequences.

© 2004 Elsevier B.V. All rights reserved.

*JEL classification:* C15; C63; G13

*Keywords:* Randomized quasi-Monte Carlo; Box–Muller; Option pricing

## 1. Introduction

In recent years, quasi-Monte Carlo methods have enjoyed increasing popularity among researchers in finance and economics. They have been used as an alternative to Monte Carlo simulation in a variety of financial problems, for example, in pricing financial derivatives (Paskov and Traub, 1995; Traub and Papageorgiou, 1996; Ninomiya and Tezuka, 1996; Joy et al., 1996; Boyle et al., 1997; Caflisch et al., 1997; Owen and Tavella, 1997; Fishman et al., 1997; Galanti and Jung, 1997), and in econometric models (Göggelmann et al., 2000; Li and Winker, 2003). Another testament to the growing popularity of these methods is their inclusion in recent introductory textbooks

* Corresponding author. Tel.: +1-765 285 8677; fax: +1-765 285 1721.
*E-mail addresses:* gokten@cs.bsu.edu (G. Ökten), eastmanw@ix.netcom.com (W. Eastman).

in finance and economics (Judd, 1998; Jackson and Staunton, 2001). As a result of this growing interest by researchers as well as practitioners working in financial industry, various financial and mathematical software now include implementations of quasi-Monte Carlo methods.[1]

Quasi-Monte Carlo methods use the so-called low-discrepancy sequences in simulating the underlying model, in contrast to the pseudorandom sequences used by Monte Carlo. Low-discrepancy sequences do not attempt to look 'random'; they are deterministic sequences designed to have the best uniformity, or evenness, in their domain of definition. The reason for increasing interest in quasi-Monte Carlo methods is that the estimates they produce are in general more accurate than the estimates produced by standard Monte Carlo. Indeed, the quasi-Monte Carlo estimates converge to the true solution of the problem with $O(N^{-1}(\log N)^K)$ ($K$ is the 'dimension' of the problem and $N$ is the number of samples, or, random walks, generated) whereas the Monte Carlo convergence rate is only $O(N^{-1/2})$. This is certainly a desirable feature, considering that the primary objective of simulation is to obtain an approximate solution to a problem that cannot be solved analytically. For an in-depth discussion of low-discrepancy sequences and quasi-Monte Carlo methods, we refer the reader to the monograph of Niederreiter (1992).

There is, however, a drawback of the quasi-Monte Carlo method: although its estimates, in general, are more accurate than Monte Carlo estimates, there is no practical way of assessing that accuracy. In other words, the quasi-Monte Carlo method provides us with a single estimate which satisfies an error bound smaller than the Monte Carlo error bound, but the method does not provide us with a practical way of measuring the actual error of that estimate. In the Monte Carlo method, on the other hand, we can simply use the standard deviation of a number of estimates as a measure of error. To address this drawback of the quasi-Monte Carlo method, several researchers in the last decade or so introduced randomized versions of quasi-Monte Carlo methods (see, for example, L'Ecuyer and Lemieux (2002) for a discussion) that would be amenable to a statistical error analysis, yet each estimate would be at least as accurate as the original quasi-Monte Carlo method. A particular randomized quasi-Monte Carlo method, the scrambled $(t,m,K)$-nets and $(t,K)$-sequences, was discussed in a recent paper by Tan and Boyle (2000). This paper was limited to only one randomized quasi-Monte Carlo method, and it did not address an issue commonly raised by financial engineers: whether the advantages of (randomized) quasi-Monte Carlo methods by the way of increased accuracy (reduced variance of estimates) are outweighed by the substantial increase in computing time. The reason for this concern is well justified since a typical low-discrepancy sequence is computationally more expensive to generate than a pseudorandom sequence.

[1] Examples of general mathematical and financial software are *Mathematica*, Gauss library LibGRO (http://gro.creditlyonnais.fr/content/rd/home_gauss.htm), and S-Plus (S+FinMetrics module on econometric analysis is currently under development). In addition, various commercial financial software use these methods in option pricing and VaR calculations (Caminus Advanced Risk Analytics, QuantRisk.com, Quadrus Financial Software, FinDer).

In this paper, we will present a detailed survey of randomized quasi-Monte Carlo methods and evaluate their efficiencies in the context of an option pricing model. Our objective is to give a clear description of the underlying ideas so that the reader can implement the method of his/her choice in the computer without much difficulty. The efficiency comparisons will be made when the randomized methods are used to price European and Asian options. We choose these simple options for the following reasons: (i) they have analytical solutions so we can compare actual error produced by different simulation methods, (ii) they have been used as benchmark problems by many researchers in the past, providing a source of numerical results that can be used to assess the effectiveness of a newly introduced simulation method, and (iii) they are the building blocks of more complicated instruments such as basket options. The strategy of using simple models that have analytical solutions in understanding and developing numerical methods appears in a number of fields, including macroeconomics, where the Cass–Koopmans one-sector, stochastic growth model in a complete markets setting is used in developing numerical solution methods that are subsequently applied in more complicated environments.[2]

The efficiency of a simulation method is defined as the product of the standard deviation of the estimates it produces and the execution time. By comparing the efficiencies of randomized quasi-Monte Carlo methods with the Monte Carlo method, we will see if one is really better off using these 'expensive' methods, or simply should run a longer Monte Carlo simulation. In the literature, the Monte Carlo method is often regarded as a generic method where the underlying pseudorandom generator is of little significance (the name of the pseudorandom generator is not even mentioned in most of the studies). However, not all pseudorandom generators are created equal, and there is much variation in execution time and uniformity properties of different generators! For this reason, we will include two popular pseudorandom generators, drand and tt800, and treat them separately in our numerical results.

We will also investigate the use of two transformation techniques, Box–Muller method and a variant of the inverse transform method known as Moro's method, in generating normal variates. Our numerical results show that Box–Muller produces estimates with better statistical accuracy than Moro's method, which contradicts the folklore that Box–Muller should be avoided (Joy et al., 1996; Tan and Boyle, 2000).

## 2. Randomized quasi-Monte Carlo methods

It is convenient to describe randomized quasi-Monte Carlo methods in the context of numerical quadrature rules, where the problem in hand is the estimation of the integral

$$I = \int_{[0,1)^K} f(x)\,\mathrm{d}x.$$

[2] For example, see den Haan and Marcet (1990) and the other papers in the special issue of Journal of Business and Economic Statistics 1990 (Volume 8).

Both Monte Carlo and quasi-Monte Carlo methods estimate this integral using sums of the form

$$\frac{1}{N}\sum_{n=1}^{N} f(\beta^{(n)}),$$

where $\beta^{(n)}$ is a $K$-dimensional random vector from the uniform distribution on $(0,1)^K$ in the case of the Monte Carlo method, and the $n$th term of a $K$-dimensional low-discrepancy sequence in the quasi-Monte Carlo method.

In randomized quasi-Monte Carlo, we consider a family of $K$-dimensional sequences $\beta_u$ indexed by the random parameter $u$. This in turn yields a family of quadrature rules

$$Q(\beta_u) = \frac{1}{N}\sum_{n=1}^{N} f(\beta_u^{(n)})$$

and the main research problem is the study of the discrepancy of the sequence $\beta_u$, and the expected value and variance of $Q(\beta_u)$. The (star-) discrepancy of the first $N$ terms of the sequence $\beta_u$ is defined as

$$D_N^*(\beta_u) = \sup_{\alpha\in[0,1)^K}\left|\frac{A_N([0,\alpha))}{N} - \alpha_1\cdot\cdots\cdot\alpha_K\right|,$$

where $\alpha=(\alpha_1,\ldots,\alpha_K)$ and $A_N([0,\alpha))$ is the number of elements $\beta_u^{(1)},\ldots,\beta_u^{(N)}$ that belong to the interval $[0,\alpha)=[0,\alpha_1)\times\cdots\times[0,\alpha_K)$. Intuitively, the discrepancy of a sequence measures its deviation from the 'ideal' distribution it is supposed to follow. We want the discrepancy to be as small as possible, since the accuracy of our quadrature rule depends on it by the Koksma–Hlawka inequality

$$|Q(\beta_u) - I| \leqslant V_{HK}(f) D_N^*(\beta_u).$$

This inequality suggests that the quadrature rule is more accurate if the terms $V_{HK}(f)$ and $D_N^*(\beta_u)$ are smaller. The first term, $V_{HK}(f)$, is the variation of $f$ in the sense of Hardy and Krause (see Niederreiter (1992) for its definition). This term measures the variability in the function values, whereas the discrepancy term measures the variability of the underlying sequence (the nodes) from the ideal distribution. The accuracy of a quasi-Monte Carlo quadrature rule can be improved by 'rewriting' the function so that the variation term is reduced, or by constructing sequences that have smaller discrepancy. In this paper we will be mostly concerned with the latter approach. A discussion of variation reduction methods can be found in Ökten (1999). We also note that there are various generalizations of discrepancy and the Koksma–Hlawka inequality (see Hickernell (2000) for various notions of discrepancy), and one generalization will be given later in the paper (see Definition 5).

In general, the quadrature rules indexed by the random parameter $u$ are unbiased, i.e.,

$$E[Q(\beta_u)] = I$$

and $I$ is estimated by the sample mean

$$\frac{Q(\beta_{u_1}) + \cdots + Q(\beta_{u_M})}{M}.$$

The random parameter $u$ will typically have uniform distribution on $(0,1)^K$, and $u_1,\ldots,u_M$ are independent samples from $u$.

## 2.1. Scrambled $(t,m,K)$-nets and $(t,K)$-sequences

In the theory of quasi-Monte Carlo methods, a distinction is made between infinite sequences of points and finite point sets (nets). Constructing a finite point set of size $N$ with a small discrepancy $D_N^*$ is in a way simpler than constructing an infinite sequence of points where the first $N$ terms have small discrepancy *consistently* as $N$ varies. One particular type of low-discrepancy nets and sequences is the so-called $(t,m,K)$-nets and $(t,K)$-sequences (Niederreiter, 1992). In this section we will discuss a method called *scrambled nets and sequences*, introduced by Owen (1995). The method takes a $(t,K)$-sequence (or, a net), call it $\beta$, and constructs a randomized version of it, call it $\beta_u$, by an elaborate algorithm that randomly scrambles the digits of the numbers in $\beta$. We will explain how this algorithm works by giving simple examples that will illustrate the underlying ideas. For further details, the reader should consult Owen (1995, 1997a,b). We also note that this method was the method considered by Tan and Boyle (2000).

Let $K \geqslant 1, b \geqslant 2$ be integers. An elementary interval in base $b$ is a subinterval of $[0,1)^K$ of the form

$$\prod_{j=1}^{K} \left[ \frac{a_j}{b^{d_j}}, \frac{a_j+1}{b^{d_j}} \right),$$

where integers $a_j, d_j$ satisfy $d_j \geqslant 0$ and $0 \leqslant a_j < b^{d_j}$. For $0 \leqslant t \leqslant m$, a finite sequence of $b^m$ points in $[0,1)^K$ is a $(t,m,K)$-net in base $b$ if every elementary interval in base $b$ of volume $b^{t-m}$ contains exactly $b^t$ points of the sequence. An infinite sequence of points $q_1, q_2, \ldots$ is called a $(t,K)$-sequence in base $b$ if the finite sequence $q_{kb^m+1}, \ldots, q_{(k+1)b^m}$ is a $(t,m,K)$-net in base $b$ for all $k \geqslant 0$ and $m \geqslant t$. There are several methods of constructing nets and sequences, and some of these constructions give the so-called van der Corput sequence, Sobol' sequence, Faure sequence, and Niederreiter sequence.[3] A detailed investigation of the construction of nets and sequences is given by Niederreiter (1992).

The main objective in the quasi-Monte Carlo theory is to find the best way to generate points in $[0,1)^K$ as evenly as possible. Evenness is measured by the discrepancy of the sequence (or, net, if we only generate a finite set of points). In contrast, randomness is the key notion in the Monte Carlo theory, and the main objective there is to find the best way to generate points that look random. Intuitively, we can achieve evenness if our quasi-Monte Carlo points are generated in such a way that there are no large gaps, or volumes in $[0,1)^K$, that do not contain a point. In other words, if

[3] Computer codes implementing these low-discrepancy sequences can be found at www.mcqmc.org.

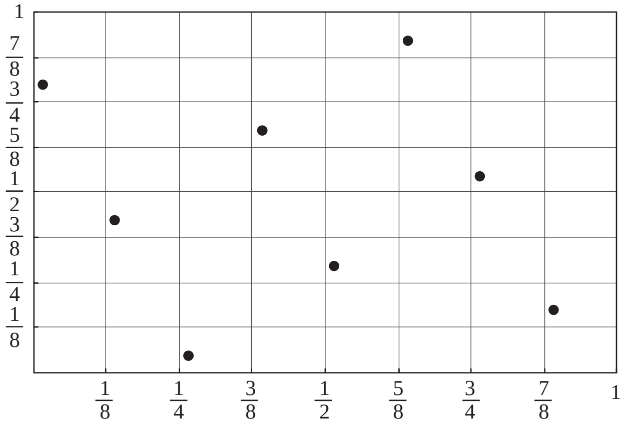

Fig. 1. (0,3,2)-net in base 2.

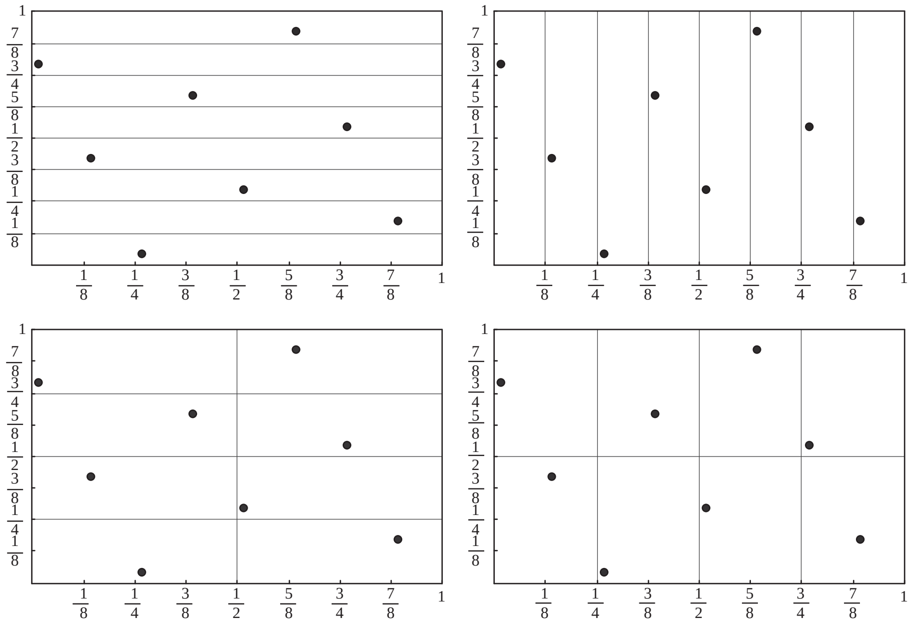

Fig. 2. Elementary intervals of area 1/8.

we think of $[0, 1)^K$ as a region composed of smaller 'building blocks' (the elementary intervals) then we want to have the 'right' proportion of points in each building block. To develop an intuitive understanding of this, we will now examine plots of quasi-Monte Carlo nets in dimension 2, together with the elementary intervals. In Fig. 1 we plot a $(0, 3, 2)$-net in base 2, and in Fig. 2 we plot the net together with

Table 1

| | $x$-coord. | $y$-coord. |
|---|---|---|
| $\left(\frac{1}{64}, \frac{51}{64}\right)$ | 000001 | 110011 |
| $\left(\frac{33}{64}, \frac{19}{64}\right)$ | 100001 | 010011 |
| $\left(\frac{17}{64}, \frac{3}{64}\right)$ | 010001 | 000011 |
| $\left(\frac{49}{64}, \frac{35}{64}\right)$ | 110001 | 100011 |
| $\left(\frac{9}{64}, \frac{27}{64}\right)$ | 001001 | 011011 |
| $\left(\frac{41}{64}, \frac{59}{64}\right)$ | 101001 | 111011 |
| $\left(\frac{25}{64}, \frac{43}{64}\right)$ | 011001 | 101011 |
| $\left(\frac{57}{64}, \frac{11}{64}\right)$ | 111001 | 001011 |

Table 2

| | $x$-coord. | $y$-coord. |
|---|---|---|
| $\left(\frac{33}{64}, \frac{51}{64}\right)$ | **1**00001 | 110011 |
| $\left(\frac{1}{64}, \frac{19}{64}\right)$ | **0**00001 | 010011 |
| $\left(\frac{49}{64}, \frac{3}{64}\right)$ | **1**10001 | 000011 |
| $\left(\frac{17}{64}, \frac{35}{64}\right)$ | **0**10001 | 100011 |
| $\left(\frac{41}{64}, \frac{27}{64}\right)$ | **1**01001 | 011011 |
| $\left(\frac{9}{64}, \frac{59}{64}\right)$ | **0**01001 | 111011 |
| $\left(\frac{57}{64}, \frac{43}{64}\right)$ | **1**11001 | 101011 |
| $\left(\frac{25}{64}, \frac{11}{64}\right)$ | **0**11001 | 001011 |

the elementary intervals of area $2^{-3}$ to illustrate that exactly one point (there are eight points total and eight elementary intervals, so the 'right' proportion is one point for each interval) falls in every elementary interval. The coordinates of the points in the $(0,3,2)$-net and their base 2 decimal expansions are given in Table 1.

We now describe Owen's randomization method using our $(0,3,2)$-net. The first step is to 'scramble' the first coordinates. This is done by picking a random permutation of the digits $\{0,1\}$; say $0 \to 1$ and $1 \to 0$, and then applying this permutation to the first digit of the $x$-coordinates to get the data given in Table 2. The new digits are typed in bold in Table 2, and the new point set is plotted in Fig. 3. Comparison of Figs. 1 and 3 shows that permuting the first digit has the effect of swapping the two half-rectangles ($[0,1/2]\times[0,1]$ and $[1/2,1]\times[0,1]$) in the original net.

The next step will use two random permutations to scramble the second digits. Assume that the first permutation is $0 \to 1$ and $1 \to 0$, and the second permutation

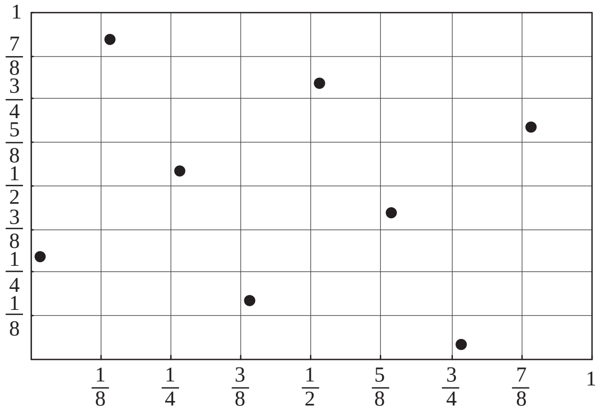
Fig. 3. Scrambling the first digits.

Table 3

| | $x$-coord. | $y$-coord. |
|---|---|---|
| $\left(\frac{33}{64}, \frac{51}{64}\right)$ | 100001 | 110011 |
| $\left(\frac{17}{64}, \frac{19}{64}\right)$ | 0**1**0001 | 010011 |
| $\left(\frac{49}{64}, \frac{3}{64}\right)$ | 110001 | 000011 |
| $\left(\frac{1}{64}, \frac{35}{64}\right)$ | 0**0**0001 | 100011 |
| $\left(\frac{41}{64}, \frac{27}{64}\right)$ | 101001 | 011011 |
| $\left(\frac{25}{64}, \frac{59}{64}\right)$ | 0**1**1001 | 111011 |
| $\left(\frac{57}{64}, \frac{43}{64}\right)$ | 111001 | 101011 |
| $\left(\frac{9}{64}, \frac{11}{64}\right)$ | 0**0**1001 | 001011 |

is the identity permutation $0 \to 0$ and $1 \to 1$. We will use the first permutation to scramble those numbers whose first digits are 0 in Table 2 (or, whose digits are 1 in Table 1). See Table 3 where the new digits are typed in bold. Geometrically, the effect of this permutation is to swap the rectangles $[0, 1/4] \times [0, 1]$ and $[1/4, 1/2] \times [0, 1]$ in Fig. 3.

The second permutation is then applied to those numbers in Table 2 whose first digits are 1. Since the second permutation is the identity permutation, this does not change anything. We obtain Fig. 4 after the scrambling of the first 2 digits.

If we continue this process until all digits are scrambled, and then scramble the $y$-coordinates of points in a similar manner, then we obtain a scrambled version of the $(0, 3, 2)$-net. However, in a practical computation, one can only scramble a finite number of digits $k^*$. Owen (1995) makes the following suggestions: When scrambling a $(t, m, K)$-net, take $k^*$ large enough so that $b^{-k^*}$ is small compared to the error in

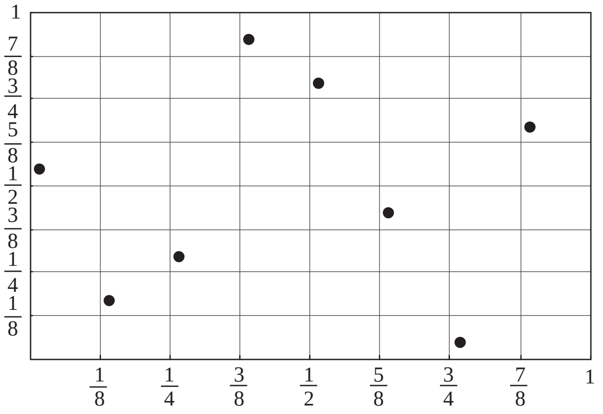

Fig. 4. Scrambling the second digits.

approximating real numbers on the computer, or let $k^* = M$ if at most $b^M$ points will be ever used (with some additional modifications in the latter choice). The number of permutations needed to scramble a $(t,m,K)$-net in base $b$ is then $K \times (1+b+\cdots+b^{k^*})$. This number becomes very large quickly, and causes computational difficulties, for large $K, b$, or $k^*$.

In order to use scrambled nets and sequences in high-dimensional option pricing problems, Tan and Boyle (2000) suggested a modification of Owen's algorithm in which one would scramble only a small number of digits $k$, and then use the last random permutations generated to scramble the rest of the digits. They called this approach 'level $k$ randomization'. We will use this randomization approach in our numerical work in Section 3.[4]

Let $\beta_u$ denote a scrambled $(t,K)$-sequence in base $b$. Owen (1995) proved

**Theorem 1.** *The scrambled sequence $\beta_u$ is a $(t,K)$-sequence in base $b$ with probability* 1. *Furthermore the resulting quadrature rules satisfy*

1. $\mathrm{E}[Q(\beta_u)] = I$,
2. $Var(Q(\beta_u)) = \mathrm{O}\left(\dfrac{(\log N)^{2K}}{N^2}\right)$.

Additional properties of scrambled nets and sequences are established by Owen (1997a, b), Hickernell (1996), and Hickernell and Hong (1999). These results provide certain improvements of the previous theorem. For example, for 'smooth' functions Owen (1997b) shows that the variance of the quadrature rule based on scrambled nets can be as small as $\mathrm{O}(N^{-3}(\log N)^{K-1})$. A central limit theorem for fully scrambled nets was proved by Loh (1996).

[4] The way we generate permutations is adapted from a computer implementation of the scrambling algorithm written by Hozumi Morohosi (http://www.misojiro.t.u-tokyo.ac.jp/~morohosi/researchE.html).

Table 4

| | $x$-coord. | $y$-coord. |
|---|---|---|
| $(\frac{9}{16}, \frac{53}{64})$ | 100100 | 110101 |
| $(\frac{1}{16}, \frac{21}{64})$ | 000100 | 010101 |
| $(\frac{13}{16}, \frac{5}{64})$ | 110100 | 000101 |
| $(\frac{5}{16}, \frac{37}{64})$ | 010100 | 100101 |
| $(\frac{11}{16}, \frac{29}{64})$ | 101100 | 011101 |
| $(\frac{3}{16}, \frac{61}{64})$ | 001100 | 111101 |
| $(\frac{15}{16}, \frac{45}{64})$ | 111100 | 101101 |
| $(\frac{7}{16}, \frac{13}{64})$ | 011100 | 001101 |

## 2.2. Random digit-scrambling

Owen's scrambling method, as we discussed in the previous section, is computationally very demanding. Matoušek (1998) introduced an alternative scrambling approach which is efficient, and yet satisfies some of the theoretical properties of Owen's scrambled nets and sequences. The idea is to limit the amount of randomness in Owen's method in return for efficiency. Instead of generating independent permutations for every value of the previous digits, we will now generate one permutation for each digit. For example, consider the (0,3,2)-net in Table 1. There are 6 digits in base 2 for each coordinate. We generate 6 independent permutations at random to scramble each digit of the $x$-coordinates and another 6 permutations for the $y$-coordinates. Say the following permutations are selected: $\varphi, id, id, \varphi, id, \varphi, id, id, id, \varphi, \varphi, id$, where $\varphi$ is the permutation that sends 0 to 1 and 1 to 0, and $id$ is the identity permutation. We then apply the first 6 permutations to each digit of the $x$-coordinates of the points in the net, and the other 6 permutations are similarly applied to the digits of the $y$-coordinates. For example, the $x$-coordinate of the first point in Table 1 is 000001. Applying the first 6 permutations to the digits we obtain: $\varphi(0), id(0), id(0), \varphi(0), id(0), \varphi(1) =$ 100100. The next 6 permutations applied to the $y$-coordinate of the same point gives 110011. Continuing this for the other points of the net we obtain the points given in Table 4.

In Fig. 5 we plot the new point set. Note that the net structure is preserved by the permutations.

Matoušek (1998) calls this approach *random digit scrambling*. He also presents a number of efficient ways to generate the permutations. If we denote the $j$th permutation that is applied to digit $a_j$ by $\pi_j$, then a simple choice is given by

$$\pi_j(a_j) = h_j a_j + g_j, \tag{1}$$

where $h_j \in \{1, 2, \ldots, b-1\}$ and $g_j \in \{0, 1, \ldots, b-1\}$ are integers to be selected at random and the arithmetic is done modulo $b$. Here $b$ is the prime base of the net. Another

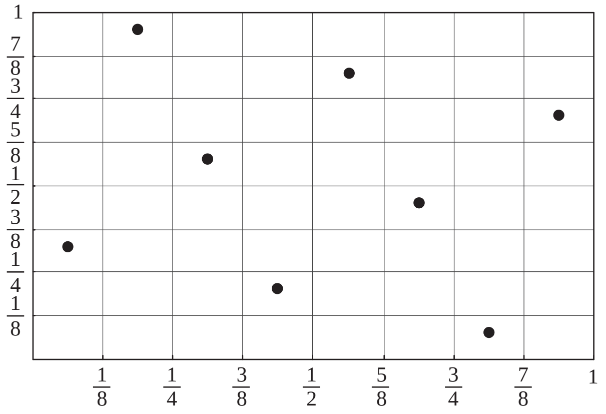

Fig. 5. Random digit-scrambling.

choice is given by

$$\pi_j(a_j) = \sum_{i=1}^{j} h_{ij} a_i + g_j, \tag{2}$$

a variant of which was used by Tezuka (1995) in generating 'generalized Faure sequences'. Here the arithmetic is again in modulo $b$, $g_j$ and $h_{ij} (i < j)$ are selected at random and independently from the uniform distribution on $\{0, 1, \ldots, b-1\}$, and $h_{jj}$ are selected at random and independently from the uniform distribution on $\{1, \ldots, b-1\}$. Note that in matrix form this equation can be written as

$$\pi(a) = H^{\mathrm{T}} a + g$$

where $H^{\mathrm{T}}$ is a nonsingular lower-triangular matrix.

Matoušek (1998) shows that both choices ((1) and (2)) for generating permutations result in scrambled nets and sequences that satisfy, in particular, the theoretical analysis given in Hickernell (1996), and many other subsequent studies as mentioned in Hong and Hickernell (2003). These theoretical results show that the mean square $L^2$ discrepancy of the alternative scrambling approaches is the same as the original scrambling approach of Owen. However, the variance of the resulting quadrature rule could be worse. Owen (2003) gives an example, and a detailed analysis of this issue, together with a discussion of various scrambling algorithms. A description of the scrambling approach (2) when applied to digital sequences is also given in Hong and Hickernell (2003). In the numerical results of Section 3, we will implement (2) which is called *random linear scrambling* in Matoušek (1998).

## 2.3. Randomly shifted low-discrepancy sequences

Now we are going to describe a randomization method which is very simple to implement. The idea is to randomize the underlying low-discrepancy sequence by shifting

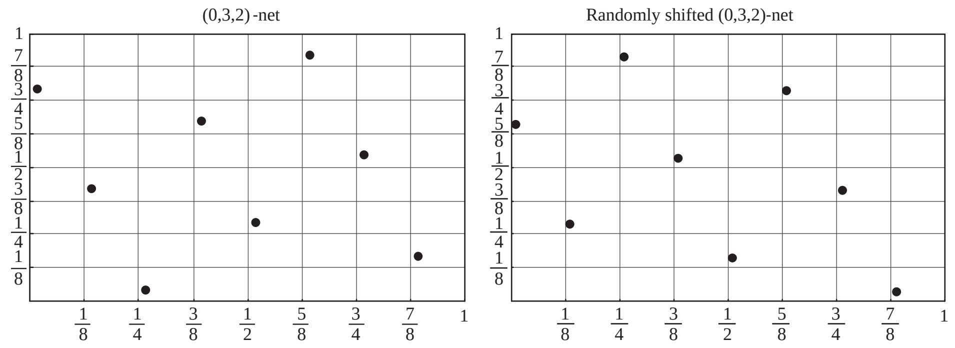

Fig. 6. A (0,3,2)-net and its randomly shifted version.

its elements by a random vector. To this end, let $\{q^{(n)}\}_{n=1}^{\infty}$ be a $K$-dimensional low-discrepancy sequence, and $u$ be a random vector selected from the uniform distribution on $(0,1)^K$. We obtain a randomly shifted version of $\{q^{(n)}\}_{n=1}^{\infty}$ by adding $u$ to each vector $q^{(n)}$ (componentwise) and taking the fractional part of the sum. To give a formal definition, we define an operation $\oplus$ on $K$-dimensional vectors $x=(x_1,\dots,x_K), y=(y_1,\dots,y_K)$ by

$$x \oplus y = (\{x_1+y_1\},\dots,\{x_K+y_K\}), \tag{3}$$

where braces about a number in (3) denote its fractional part. A randomly shifted version of $\{q^{(n)}\}_{n=1}^{\infty}$ is then given by $\{q^{(n)}\oplus u\}_{n=1}^{\infty}$.

The idea of random shifting was first introduced by Cranley and Patterson (1976) in the context of good lattice points, and later Joe (1990) applied the idea to general lattice rules. Tuffin (1996) considered random shifting of any low-discrepancy sequence, and studied the discrepancy of the shifted sequence.

We now present an example. In Fig. 6 we plot our $(0,3,2)$-net in base 2, and a randomly shifted version of it. The random vector $u$ is $(0.74, 0.62)$. As it can be seen from the graph, a randomly shifted net is not necessarily a net itself: The elementary interval $[1/4,1/2)\times[0,1/2)$ does not contain a point in the second plot.

One geometric interpretation of random shifting is given by Owen (1998): Consider a particular one-dimensional projection of a randomly shifted sequence, and plot the points (projected components) onto a circle by simply identifying the endpoints of the unit interval $[0,1)$. The randomly shifted points (components) are then obtained from the original components of the net by a rotation to the right by a distance of $u$.

Using the notation we introduced in Section 2, the randomization approach used by Tuffin (1996) defines the sequence $\beta_u=\{\beta_u^{(n)}\}_{n=1}^{\infty}$ as

$$\beta_u^{(n)} = q^{(n)} \oplus u. \tag{4}$$

Tuffin (1996) proved the following results.

**Theorem 2.** *The discrepancy of a shifted low-discrepancy sequence satisfies*

1. $D_N(\beta_u) \leqslant 2^K D_N(q)$,
   *and the corresponding quadrature rules satisfy*
2. $\mathrm{E}[Q(\beta_u)] = I$,
3. $Var(Q(\beta_u)) = \mathrm{O}\left(\frac{(\log N)^{2K}}{N^2}\right)$.

A detailed investigation of the variance of this quadrature rule is given by Morohosi and Fushimi (2000). Using the theory of $b$-adic $K$-dimensional Haar functions, the authors obtain expansions for $Var(Q(\beta_u))$ when $Q(\beta_u)$ is obtained using the random shifting method, and Owen's scrambling method. They conclude that the variance of scrambled sequences might be smaller than the variance of randomly shifted sequences, due to possible cancellations in the Haar expansion of the variance of the former method. They also provide numerical results where the opposite happens in Morohosi and Fushimi (2002) and Morohosi et al. (2000). In these two papers, the authors present extensive numerical results that compare scrambled and randomly shifted Sobol', Faure, and Niederreiter sequences, when they are used to price several options (barrier, European, Asian, and lookback options) and mortgage-backed securities. The numerical results suggest that random shifting gives smaller variance than scrambling when the base of the $(t, K)$-sequence is 'small'. We refer the reader to these papers for details.

## *2.4. Random-start Halton sequences*

The Halton sequence is one of the classical low-discrepancy sequences. The $n$th term of the Halton sequence in bases $b_1, \ldots, b_K$ (pairwise relatively prime integers greater than or equal to 2) is a vector of $K$ components, denoted by

$$q^{(n)} = (\phi_{b_1}(n), \ldots, \phi_{b_K}(n)). \tag{5}$$

The $i$th component of this vector is calculated as follows: First we write $n$ in its base $b_i$ expansion as

$$n = a_0 + a_1 b_i + a_2 b_i^2 + \cdots + a_l b_i^l,$$

where the coefficients $a_j$ are between 0 and $b_i - 1$, and $l$ is the greatest integer less than or equal to $\log_{b_i} n$. We then evaluate the *radical-inverse* function in base $b_i$ at $n$, i.e., compute

$$\phi_{b_i}(n) = \frac{a_0}{b_i} + \frac{a_1}{b_i^2} + \cdots + \frac{a_l}{b_i^{l+1}}. \tag{6}$$

Repeating this calculation for all $i = 1, \ldots, K$, we obtain vector (5). The one-dimensional projections of the Halton sequence are known as the van der Corput sequences. In other words, the one-dimensional sequence $\{\phi_{b_i}(n)\}_{n=1}^{\infty}$ is the van der Corput sequence in base $b_i$.

An alternative description of the Halton sequence based on the von Neuman–Kakutani transformation was introduced by Lambert (1985) and Struckmeier (1995) used this transformation to introduce a fast way to generate the sequence. We now describe this alternative approach. Although our treatment will be algebraic, a geometric approach is given by Lambert (1985).

We first define the rightward carry addition in base $b$ of a number $x \in [0,1]$ and $1/b$. Consider the base $b$ expansion of $x$

$$x = \frac{x_0}{b} + \frac{x_1}{b^2} + \cdots + \frac{x_l}{b^l}.$$

The rightward carry sum of $x$ and $1/b$ is

$$x \underset{R}{+} \frac{1}{b} = \frac{1+u_m}{b^{m+1}} + \sum_{k>m} \frac{u_k}{b^{k+1}}.$$

The von Neuman–Kakutani transformation $T_b : [0,1) \to [0,1)$ in base $b$ is defined as

$$T_b(x) = x \underset{R}{+} \frac{1}{b}.$$

The van der Corput sequence in base $b$ is then simply the orbit of 0 under $T_b$, i.e., the sequence $\{T_b^1(0), T_b^2(0), \ldots\}$. For example, if $b = 1/2$, it can be verified that

$$\{T_{1/2}^1(0), T_{1/2}^2(0), T_{1/2}^3(0) \cdots\} = \left\{\frac{1}{2}, \frac{1}{4}, \frac{3}{4}, \cdots\right\}.$$

Here $T_b^n$ is the $n$th iteration of the mapping, i.e., $T_b^n(x) = T_b(x)$ if $n=1$, and $T_b(T_b^{n-1}(x))$ if $n > 1$.

The von Neuman–Kakutani transformation can be generalized easily to the multidimensional case. If $\mathbf{b} = (b_1, \ldots, b_K)$ is a vector of positive integers that are pairwise relatively prime and $\mathbf{x} = (x_1, \ldots, x_K)$ is a vector in $[0,1)^K$, then we define the $K$-dimensional von Neuman–Kakutani transformation by

$$T_{\mathbf{b}}(\mathbf{x}) = (T_{b_1}(x_1), \ldots, T_{b_K}(x_K)). \tag{7}$$

The orbit of the vector $(0, \ldots, 0)$ under $T_{\mathbf{b}}$ gives the Halton sequence in bases $b_1, \ldots, b_K$. In general, we can consider the orbit of any vector $\mathbf{x}$ in $[0,1)^K$ under $T_{\mathbf{b}}$, and use the resulting sequence $\{T_{\mathbf{b}}^n(\mathbf{x})\}_{n=1}^{\infty}$ to obtain a quadrature rule. These sequences were called *generalized Halton sequences* by Struckmeier (1995).

Wang and Hickernell (2000) observe that if $\mathbf{x} = (x_1, \ldots, x_K)$ where $x_i = 0.d_0^{(i)} d_1^{(i)} \cdots d_n^{(i)}$ (in base $b_i$), and $\phi_{b_i}^{-1}(0.d_0^{(i)} d_1^{(i)} \cdots d_n^{(i)}) = m^{(i)}$ (see Eq. (6)) then

$$T_{\mathbf{b}}^n(\mathbf{x}) = (\phi_{b_1}(m^{(1)} + n), \ldots, \phi_{b_K}(m^{(K)} + n)). \tag{8}$$

In other words, the sequence $\{T_{\mathbf{b}}^1(\mathbf{x}), T_{\mathbf{b}}^2(\mathbf{x}), \ldots\}$ is simply the Halton sequence in bases $b_1, \ldots, b_K$ (see (5)) skipped by the integer vector $(m^{(1)}, \ldots, m^{(K)})$.

Struckmeier (1995) discussed the idea of independently selecting random starting values $\mathbf{x}$ in (7). He used generalized Halton sequences with random starting values to estimate the expected value $\mathrm{E}[xyz]$, where $x, y, z$ are uniform random variables on $(0,1)$. He then computed the variance of several such estimates, although he did not justify the validity of such computations. A probability model that justifies the computation of

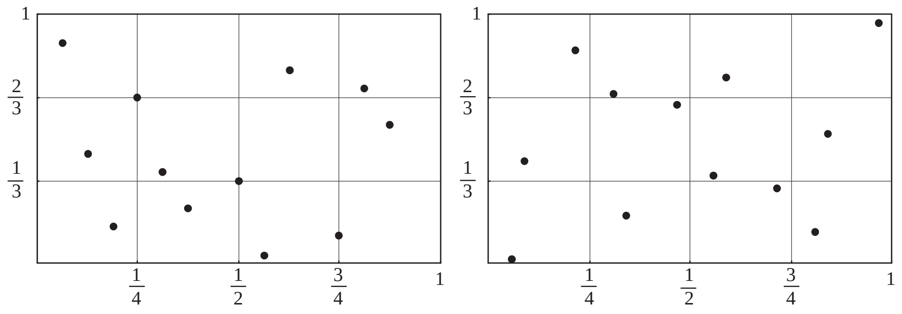

Fig. 7. A two-dimensional random-start Halton sequence.

variances was introduced by Wang and Hickernell (2000). They considered the family of sequences $\beta_u = \{\beta_u^{(n)}\}_{n=1}^{\infty}$ where

$$\beta_u^{(n)} = T_{\mathbf{b}}^n(u) \tag{9}$$

and $u$ is a random vector from the uniform distribution on $[0,1]^K$. They called these sequences, *random-start Halton sequences*. The discrepancy of $\beta_u^{(n)}$ satisfies

$$D_N(\beta_u) \leqslant \frac{1}{N}\left(1 + \prod_{i=1}^{K}(b_i - 1)\,\frac{\log(b_i N)}{\log b_i}\right)$$

(see Bouleau and Lépingle, 1994, p. 77) and the analysis of the resulting quadrature rules is given by the following theorem by Wang and Hickernell (2000).

**Theorem 3.** *The quadrature rules based on random-start Halton sequences satisfy*

1. $\mathrm{E}[Q(\beta_u)] = I$,
2. $Var(Q(\beta_u)) = \mathrm{O}\left(\dfrac{(\log N)^{2K}}{N^2}\right)$.

In Fig. 7, we plot the first 12 elements of the Halton sequence (left) and a random-start Halton sequence (right), in bases 2 and 3. The random-start Halton is obtained by skipping $m^{(1)} = 28709564263$ terms for the first component and $m^{(2)} = 26896075533$ terms for the second component.

## 2.5. *Finite random sampling from low-discrepancy sequences*

Unlike the randomization methods discussed earlier, this approach samples from a family of *finitely* many quadrature rules. Let $\Omega$ consist of $m$ low-discrepancy sequences, $q_1, \ldots, q_m$, where each sequence is $K$-dimensional. Furthermore let $u$ have discrete uniform random distribution on $\{1, \ldots, m\}$. Then every randomization of this approach simply selects a random $q_u$ from the set $\Omega$ to construct the corresponding quadrature rule. Using the notation of Section 2, we have $\beta_u = q_u$. Ökten (2002) showed

**Theorem 4.** *The quadrature rules based on the finite random sampling approach satisfy*

1. $|\mathrm{E}[Q(\beta_u)] - I| \leqslant \mathrm{O}\left(\frac{(\log N)^K}{N}\right)$,
2. $Var(Q(\beta_u)) = \mathrm{O}\left(\dfrac{(\log N)^{2K}}{N^2}\right)$.

This approach produces estimates that are asymptotically unbiased as a result of sampling from a finite set of low-discrepancy sequences. To populate $\Omega$ we will use scrambled Halton sequences of Faure (1992). Each randomization of this approach will select $K$ different prime numbers at random from a list of the first $L$ prime numbers, and construct the corresponding scrambled Halton sequence, which will be then used in the simulation to obtain a single estimate. Other choices for populating $\Omega$ are discussed in Ökten (2002).

In the next section, we will apply randomized quasi-Monte Carlo methods to problems whose solutions can be found exactly by analytical methods. We will construct confidence intervals for each randomized method, and then verify the accuracy of these intervals by checking how often they contain the true solution. We note that the finite random sampling method, unlike the other randomized methods, has a bias that satisfies an upper bound with the same order as the standard deviation of its estimates (from the above theorem). Although the accuracy of confidence intervals constructed for this method in the numerical results of Section 3 was as good as the other methods, we remark that the theoretical results as provided by the above theorem do not lead to asymptotically justified confidence intervals.

## 3. Applications to option pricing

### 3.1. Generating asset price paths

We assume that the asset price $S_t$ follows the risk-neutral geometric Brownian motion with variance $\sigma^2$ and drift parameter $\mu = r - \sigma^2/2$, where $r$ is the risk-free interest rate. We let $x_t = \ln(S_t)$ and generate the (logarithm of) price paths using

$$x_i = x_{i-1} + \mu(T/K) + \sigma\sqrt{T/K}z_i, \;\; i = 1,\ldots,K, \tag{10}$$

where $T$ is the expiry, $K$ is the number of time steps used in the discretization of the time interval $(0,T)$ so that $\Delta t = T/K$, and $z_i$ is a standard normal random variable.

In the Monte Carlo and randomized quasi-Monte Carlo methods, $N$ price paths $w_1,\ldots,w_N$ are generated and the present value of the average option value is computed. To generate the $n$th price path $w_n = (x_1^{(n)},\ldots,x_K^{(n)})$, one needs to sample from $\mathrm{N}(0,1)$

and generate numbers $z_1^{(n)},\dots,z_K^{(n)}$ needed in (10). In the Monte Carlo method, this is done by generating pseudorandom numbers $\xi_1,\dots,\xi_K$ from the uniform distribution on $(0,1)$ and then using a normal random variable generation method (such as the Box–Muller or inverse transform technique) to obtain $z_1^{(n)},\dots,z_K^{(n)}$. In randomized quasi-Monte Carlo methods, one uses the $n$th term of the $K$-dimensional sequence $\beta_u$ to generate the $n$th price path. If we denote the components of the $n$th term of this sequence by $q_1,\dots,q_K$, then a normal random variable generation method is applied to these numbers to obtain $z_1^{(n)},\dots,z_K^{(n)}$.

### 3.2. Numerical results

In the following, we will use randomized quasi-Monte Carlo methods and the Monte Carlo method with two popular pseudorandom generators, in pricing the standard European call option and geometric Asian call option. We deliberately choose option pricing problems that can be solved analytically so that we can assess the exact error of the simulation methods, and use this information to compare them numerically.

#### 3.2.1. European call option

Our first example is the standard European call option. The risk free interest rate is 0.1, the exercise price is 10, the initial price of the underlying is 10, volatility is 0.4 and the expiry is 0.25. The Black and Scholes value for the option is 0.916291. We first want to test the hypothesis that the option estimates obtained from randomized quasi-Monte Carlo methods have approximately a normal distribution with mean the Black and Scholes value for the option price. We generate two sets of 1600 estimates for the option value using each of the following randomized quasi-Monte Carlo methods: (i) the scrambled Faure sequence (ScrFaure) with level 2 randomization, as described in Section 2.1, (ii) the linear scrambled Faure sequence (LinScrFaure), as described in Section 2.2, (iii) the randomly shifted Halton sequence (RShiftHalton), as described in Section 2.3, (iv) the random-start Halton sequence (RStartHalton), as described in Section 2.4, and (v) the finite random sampling method where $\Omega$ consists of scrambled Halton sequences (RS-scrHalton), as described in Section 2.5. The first set of 1600 estimates is generated when the Box–Muller method is used (see Rubinstein, 1981) to simulate the normal variables, and in the second set, a variant of the inverse transform method due to B. Moro is used (see Joy et al., 1996). The number of random walks (price paths) $N$ is set to 200 000. The number of prices, or time steps, is $K = 20$. The number of primes $L$ is set to 40 in the RS-method.

Fig. 8 plots the relative frequency histograms of the option estimates. To each histogram, we superimpose the normal density curve whose mean is the Black & Scholes value of the option, and whose variance is the sample variance of the data. The graphs suggest that the ScrFaure, LinScrFaure, and RS-scrHalton estimates, when generated using the inverse transform method, tend to underestimate the true option value. There is no visual indication of any bias in the other two methods. We now use a number of statistical tests for a quantitative investigation of the normality assumption and bias.

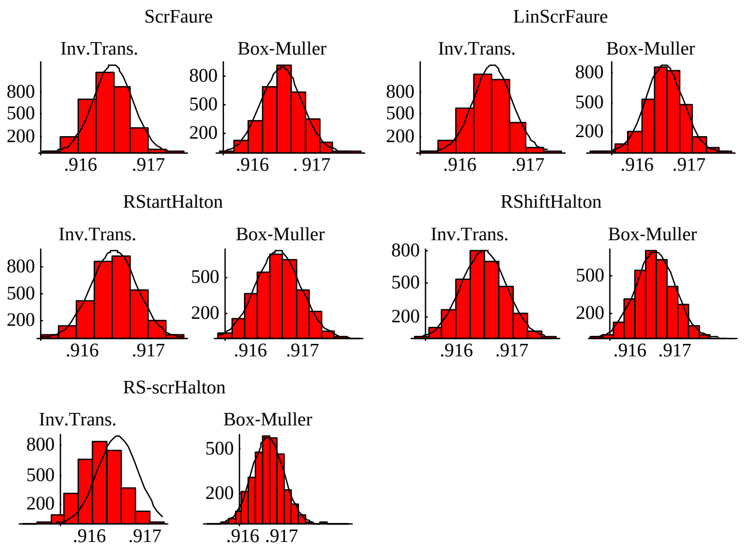

Fig. 8. Histograms of option estimates.

We conduct three tests as recommended by D'Agostino and Stephens (1986): (i) Anderson–Darling $A^2$ test to check the normality assumption with known mean, (ii) Anscombe–Glynn test to check the kurtosis, and (iii) $S_U$ approximation method due to D'Agostino to check the skewness. The following table gives the $A^2$ values.

| $A^2$ *Statistic* | Box–Muller | Inv.Trans. |
|---|---|---|
| ScrFaure (level 2) | 3.03 | 56.71 |
| LinScrFaure | 3.967 | 42.16 |
| RShiftHalton | 0.751 | 1.831 |
| RStartHalton | 0.702 | 1.918 |
| RS-scrHalton ($L=40$) | 2.858 | 352.0 |

At the 10% significance level, the critical value for the $A^2$ statistic is 1.743, and at the 2.5% level, the critical value is 2.898. We make the following observations:

1. The Box–Muller method produces lower $A^2$ values than the inverse transform method for all randomized quasi-Monte Carlo methods. The difference is quite substantial especially for the ScrFaure, LinScrFaure, and RS-scrHalton methods. The high $A^2$ values obtained by the inverse transform technique indicate a substantial bias, which was visible in the histograms. Indeed, if we drop our requirement that the empirical distribution has mean equal to the option value (the theoretical mean), and use Anderson–Darling test with *unknown mean*, then we obtain the following

$A^2$ values:

| Inverse transform | $A^2$ Statistic (unknown mean) |
|---|---|
| ScrFaure (level 2) | 0.793 |
| LinScrFaure | 0.332 |
| RS-scrHalton ($L = 40$) | 0.467 |

These are perfect values much less than the 10% critical value 1.743.

2. At the 10% level, the methods that are not rejected are: RShiftHalton (Box–Muller) and RStartHalton (Box–Muller). At the 2.5% level, the inverse transform implementations of these methods pass the test, as well as the RS-scrHalton (Box–Muller) method. The ScrFaure (level 2) and LinScrFaure methods fail both tests.

We now examine the kurtosis and skewness of the data. In the first table, the numbers in bold are the kurtosis values that do not fail the two-sided Anscombe–Glynn test for normality at the 10% level.

| Kurtosis | Box–Muller | Inv.Trans. |
|---|---|---|
| ScrFaure (level 2) | **2.90** | 2.80 |
| LinScrFaure | 3.35 | **2.86** |
| RShiftHalton | **3.01** | 2.78 |
| RStartHalton | **2.85** | **3.01** |
| RS-scrHalton ($L = 40$) | 3.59 | **2.99** |

In the next table, the numbers in bold are the skewness values that do not fail the $S_U$ approximation test for normality, at the 10% level. The two-sided 10% significance level for 1600 data values is 0.101.

| Skewness | Box–Muller | Inv.Trans. |
|---|---|---|
| ScrFaure (level 2) | **0.011** | **−0.064** |
| LinScrFaure | **−0.032** | **−0.002** |
| RShiftHalton | **0.07** | **0.013** |
| RStartHalton | 0.13 | **−0.054** |
| RS-scrHalton ($L = 40$) | **0.012** | −0.16 |

The Box–Muller and inverse transform methods produce mostly acceptable kurtosis and skewness values, although we see slight deviations from normality for some randomized quasi-Monte Carlo methods.

We now use the 1600 estimates generated by the Box–Muller and inverse transform methods, to construct forty 90% confidence intervals, where each interval is constructed using 40 estimates. The intervals are obtained using the Student's $t$, bootstrap-$t$, bootstrap percentile and bootstrap $\mathrm{BC}_a$ methods. For a discussion of bootstrap methods see Efron and Tibshirani (1993). In general, bootstrap confidence intervals are used when little is known about the underlying distribution. The following tables display the number of confidence intervals that contain the true option value, followed by the average width of the intervals, when the intervals are constructed using the Box–Muller and inverse transform method, respectively.

| Box–Muller | | | | |
|---|---|---|---|---|
| Coverage, avg.width | Student's $t$ | Boot.-$t$ | Boot.perc. | Boot.BC$_a$ |
| ScrFaure (level 2) | 36, 2.4e-4 | 36, 2.4e-4 | 35, 2.3e-4 | 35, 2.3e-4 |
| LinScrFaure | 31, 2.4e-4 | 31, 2.4e-4 | 30, 2.3e-4 | 31, 2.3e-4 |
| RShiftHalton | 34, 3.0e-4 | 34, 3.0e-4 | 32, 2.9e-4 | 34, 2.9e-4 |
| RStartHalton | 36, 2.9e-4 | 36, 2.9e-4 | 36, 2.8e-4 | 37, 2.8e-4 |
| RS-scrHalton ($L = 40$) | 36, 3.4e-4 | 35, 3.7e-4 | 35, 3.6e-4 | 35, 3.6e-4 |
| | | | | |
| Inverse transform | | | | |
| Coverage, avg.width | Student's $t$ | Boot.-$t$ | Boot.perc. | Boot.BC$_a$ |
| ScrFaure (level 2) | 20, 1.8e-4 | 20, 1.8e-4 | 20, 1.8e-4 | 20, 1.8e-4 |
| LinScrFaure | 22, 1.8e-4 | 22, 1.8e-4 | 21, 1.8e-4 | 22, 1.8e-4 |
| RShiftHalton | 36, 2.6e-4 | 36, 2.7e-4 | 35, 2.6e-4 | 35, 2.6e-4 |
| RStartHalton | 35, 2.1e-4 | 36, 2.1e-4 | 33, 2.1e-4 | 36, 2.1e-4 |
| RS-scrHalton ($L = 40$) | 0, 2.4e-4 | 0, 2.4e-4 | 0, 2.3e-4 | 0, 2.3e-4 |

We make the following observations:

1. The number of intervals that contain the true option value is close to the theoretical value 36 for all methods except LinScrFaure, when the Box–Muller method is used. The inverse transform method gives a very low coverage for the ScrFaure and LinScrFaure methods and zero coverage for the RS-scrHalton method.
2. The inverse transform method produces intervals that are narrower (relative improvement varies between 10% and 35%) than the Box–Muller method.
3. There seems to be no significant difference between different confidence interval generation methods. The largest variation occurs in the RS-scrHalton method (first table), and Student's $t$ approach gives the smallest average width.
4. The average width of the confidence intervals is smallest for the ScrFaure (level 2) and LinScrFaure methods. The former, however, has significantly better coverage in the Box–Muller implementation.

We now summarize the results of some additional numerical work. We computed the Kolmogorov–Smirnov statistic for our data, and found that the Box–Muller method consistently produced smaller values than the inverse transform method, as in the $A^2$ statistic. We also repeated all our calculations when the number of price paths was $N = 10^5$, and obtained similar results to those reported above. We also computed the $A^2$ statistic, kurtosis and skewness, and confidence interval coverage when two pseudorandom generators were used: (i) drand; a popular linear congruential generator used in ANSI C library, and (ii) tt800; a recent generator based on generalized feedback shift-register methods (Matsumoto and Kurita, 1994). In the following table, BM and IT stand for the Box–Muller and inverse transform methods.

| | $A^2$ *BM* | $A^2$ *IT* | Kurtosis *BM* | Kurtosis *IT* | Skewness *BM* | Skewness *IT* |
|---|---|---|---|---|---|---|
| drand | 0.66 | 1.69 | 2.84 | 3.00 | −0.0018 | 0.071 |
| tt800 | 0.31 | 0.31 | 3.01 | 2.92 | 0.022 | 0.060 |

The Box–Muller method still produces a slightly lower $A^2$ value in the case of the drand generator. There is no difference between the $A^2$ values for the tt800 generator. The kurtosis and skewness results do not suggest a definite advantage of any of the methods.

As a result of our statistical tests, we decide to use the Box–Muller method in generating normal random variates. Our results indicate that both Box–Muller and inverse transform methods produce fairly normal distributions (based on the kurtosis and skewness results), however, there is significant bias in the inverse transform method for certain randomized quasi-Monte Carlo methods (based on the $A^2$ test with known mean and confidence interval results). These findings are rather surprising, since it is often reported in the literature that the Box–Muller method should be avoided. For example, Joy et al. (1996) writes (footnote 6, p. 930)

> Note that it is incorrect to use the standard Box–Muller transform to map $\varepsilon_n$ to the unit interval. This is because the Box–Muller transform does not preserve the matrix on $I^s$ and thus fails to preserve the low-discrepancy of the original Faure sequence. In other words, if we use the Box–Muller transform, then the even spacing of the Faure sequence will be scrambled, resulting in the loss of our low error bound.

Whether a random variable generation method is 'better' than another, when the underlying number sequence is a low-discrepancy sequence, is related to the discrepancy of the transformed sequence. The $F$-star discrepancy of a transformed sequence is defined as follows:

**Definition 5.** The $F$-star discrepancy of the data $x_1,\dots,x_N$, where $F(x)$ is a distribution function, is defined by

$$D_N^*(x_i;F) = \sup_{\alpha\in[0,1)^K}\left|\frac{A_N([0,\alpha))}{N} - F(\alpha)\right|, \tag{11}$$

where $\alpha=(\alpha_1,\dots,\alpha_K)$ and $A_N([0,\alpha))$ is the number of elements $x_1,\dots,x_N$ that belong to the interval $[0,\alpha)=[0,\alpha_1)\times\cdots\times[0,\alpha_K)$.

The corresponding Koksma–Hlawka inequality to the notion of $F$-star discrepancy is

**Theorem 6.** *Let $f$ be of bounded variation on $[0,1]^K$ in the sense of Hardy and Krause and $\{x_i\}\in[0,1)^K$, and $F$ be a distribution function on $[0,1)^K$. Then for any*

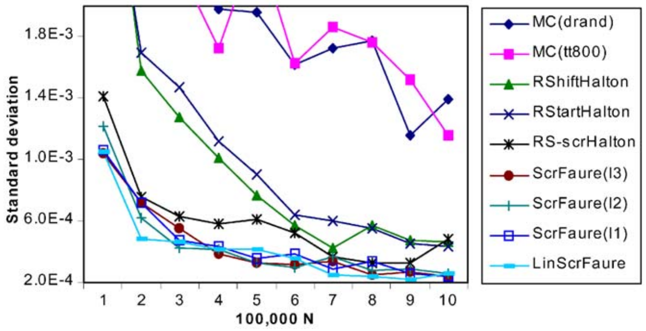

Fig. 9. European Call option – standard deviation of estimates.

*$N > 0$*

$$\left|\frac{1}{N}\sum_{i=1}^{N} f(x_i) - \int_{[0,1)^K} f(x)\,\mathrm{d}F(x)\right| \leqslant V_{HK}(f) D_N^*(x_i;F), \tag{12}$$

*where $V_{HK}(f)$ is the variation of $f$ in the sense of Hardy and Krause.*

We observe that the Kolmogorov–Smirnov statistic for the data $x_1,\dots,x_N$ is the $F$-star discrepancy of the data, and the $A^2$ statistic can similarly be regarded as a weighted discrepancy of the data. In other words, our empirical results suggest that the Box–Muller method in fact produces data with a smaller calculated discrepancy, than the one produced by Moro's approximation of the inverse transform method. This, in turn, suggests that the Box–Muller method is 'better', since it makes the error, i.e., the right-hand side of inequality (12) smaller, for the data sets we have considered.

Now we turn our attention to the standard deviation of the option estimates produced by the Monte Carlo methods with drand and tt800 generators, and the randomized quasi-Monte Carlo methods. For a given $N$ ($N = 10^5, 2\times 10^5, \dots, 10^6$), 40 option estimates are computed by independent randomizations of each method. We then calculate the standard deviation of these 40 estimates. The option parameters are as above, except for the number of time steps, which is increased to 40. We are in particular interested in observing the performance of the randomized quasi-Monte Carlo methods as the number of time steps, i.e., the dimension of the problem, increases, since quasi-Monte Carlo methods are often reported to lose their competitive 'edge' over the Monte Carlo method in high-dimensional problems. In Fig. 9, we plot the standard deviation of Monte Carlo and randomized quasi-Monte Carlo estimates.

We make the following observations:

1. Monte Carlo methods produce the largest standard deviation, and there does not seem to be a significant difference between generators tt800 and drand.
2. Methods RShiftHalton and RStartHalton come next in the graph, producing smaller standard deviation than the Monte Carlo methods. RShiftHalton gives a slightly smaller standard deviation than RStartHalton.

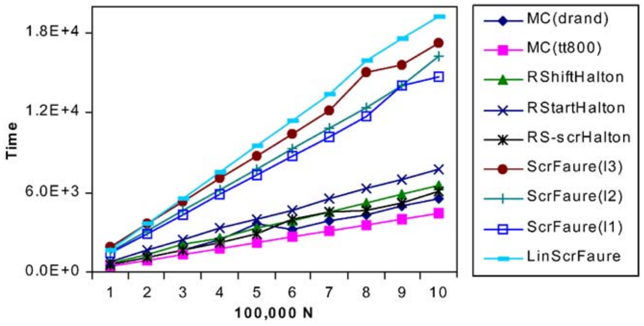

Fig. 10. European call option – timing results.

3. The standard deviation of RS-scrHalton method is smaller than RShiftHalton and RStartHalton, but larger than the ScrFaure and LinScrFaure implementations.
4. For most of the samples, the ScrFaure method with a higher level of randomization produces smaller standard deviation, as expected. However, there are a few samples where the opposite is observed. This is certainly not very strange, given the probabilistic nature of the scrambling algorithm. There does not seem to be a clear separation between the LinScrFaure and various ScrFaure implementations, although the former method gives the smallest standard deviation for half of the sample sizes.

In a practical implementation of any simulation method, one is interested in not only the standard deviation of its estimates, but also the computation time. A standard notion of efficiency of a Monte Carlo method is defined as the standard deviation (or, variance) multiplied by time. In Fig. 10, we plot the time (in seconds) spent to compute the 40 estimates whose standard deviations were reported above. The pseudorandom generators are the fastest, but RShiftHalton, RStartHalton, and RS-scrHalton follow them closely. Not surprisingly, LinScrFaure and ScrFaure implementations are the slowest. We note that after level 3, the ScrFaure implementations become much slower than LinScrFaure. In general, when the dimension is small and the level of randomization is low, the ScrFaure implementation can be faster than LinScrFaure. This is due to the fact that the former method involves the multiplication of an upper diagonal matrix with a vector in generating the random walks, and thus more efficient than the latter method where the matrix is no longer upper diagonal. As the dimension and the level of randomization increases, ScrFaure quickly loses its advantage due to the geometrically increasing number of permutations that have to be computed.

Finally in Fig. 11, we plot the efficiency of the simulation methods. The Monte Carlo methods have the worst efficiency although they are the fastest of all methods. The RS-scrHalton has the best efficiency among all methods and all sample sizes considered. The rest of the methods are not easily distinguishable, although RShiftHalton separates itself from the rest after 500,000 random walks. In the ScrFaure implementations, level 2 randomization has a better efficiency than level 3 randomization for every sample

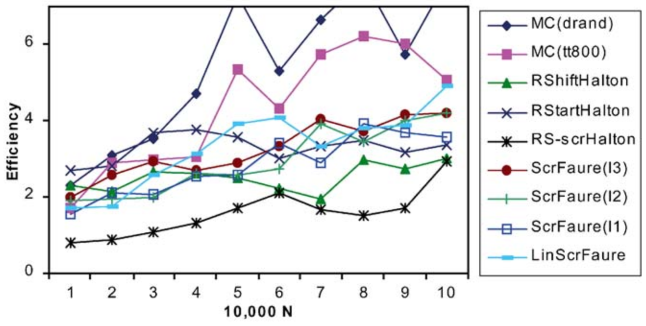

Fig. 11. European call option – efficiency of the simulation methods.

size. There is no winner between level 1 and level 2 randomizations. When levels 1 and 3 are compared, we see the former give a better efficiency for all except two sample sizes. In conclusion, level 2 randomization seems to be the optimal choice in terms of efficiency.

### 3.2.2. Geometric Asian call option

To eliminate any possibility of bias in the selection of option parameters, we decide to select them at random in this example. For the distribution of parameters, we follow Tan and Boyle (2000), and fix the strike price at 100, and assume the initial asset price is uniformly distributed between 50 and 150. The volatility is uniformly distributed between 0.1 and 0.6, and the interest rate is uniform between 0.05 and 0.15. The expiry is uniform between 6 months and 2 years. We take the number of time steps to be $K = 80$.

Since RS-scrHalton is the only method that produces asymptotically unbiased estimates, we repeat the investigation of the distribution of its estimates using the $A^2$ test with known mean. We generate 1600 estimates using the Box–Muller method. The sampling is done from $L = 100$ prime numbers. The following table displays $A^2$ values for a variety of choices for the number of price paths $N$:

| | $N = 2 \times 10^5$ | $N = 3 \times 10^5$ | $N = 4 \times 10^5$ | $N = 5 \times 10^5$ |
|---|---|---|---|---|
| $A^2$ | 30 | 22 | 42 | 0.6 |

The $A^2$ statistic attains values at an acceptable level only after 400 000 price paths. These results underline the bias in the RS-estimates and the fact that $N$ should be ‘sufficiently’ large to conduct a valid statistical analysis of its estimates. The inverse transform method produces much larger $A^2$ values for RS-estimates as in the case of the European call option example.

In Fig. 12 we plot the standard deviation of 40 estimates for each $N$ ($N = 10^5$, $2 \times 10^5, \ldots, 10^6$). Forty estimates are obtained by independent randomizations of each

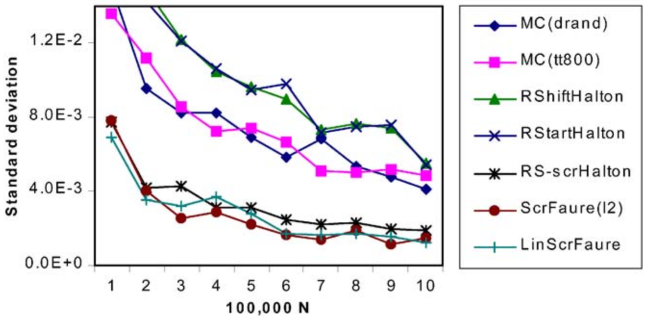

Fig. 12. Geometric Asian option – standard deviation of estimates.

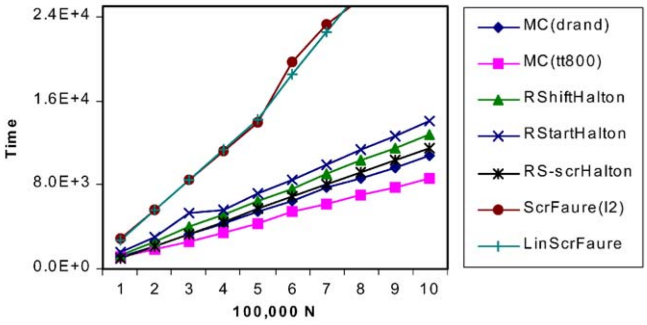

Fig. 13. Geometric Asian option – timing results.

method. In the RS-scrHalton method we take $L = 100$, and in ScrFaure, the level of randomization is 2.

It is interesting to note that the RShiftHalton and RStartHalton methods now produce a higher standard deviation than the two Monte Carlo methods; a complete reversal of what we observed in the European call example. This is probably a consequence of the increased dimension of the problem, from $K = 40$ (in the European call problem) to $K = 80$. The ScrFaure (level 2) and LinScrFaure methods have slightly smaller variance than the RS-scrHalton method (except for one sample size for LinScrFaure).

In Figs. 13 and 14, we plot the time and efficiency of the methods. The timing results are similar to those obtained for the European call option. The RS-scrHalton method has the best efficiency; however we emphasize that this method failed the $A^2$ test for sample sizes up to 400 000 (see the previous table). The Monte Carlo methods have now a better efficiency than the RShiftHalton and RStartHalton methods. The efficiency of the generator tt800 is better than drand and LinScrFaure except for one data point for the former, and two data points for the latter method. The ScrFaure efficiency is between the Monte Carlo methods.

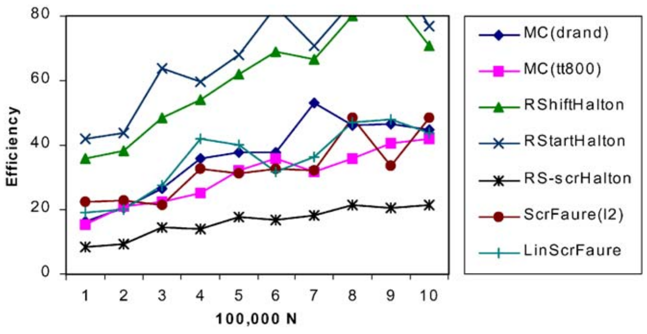

Fig. 14. Geometric Asian option – efficiency of the simulation methods.

## 4. Conclusion

Our numerical results suggest that in low-dimensional problems, the randomized quasi-Monte Carlo methods can offer gains not only in error reduction but also in efficiency, when compared to the Monte Carlo method. In higher dimensions, we observed that the RShiftHalton and RStartHalton methods produced estimates with larger variance than the Monte Carlo methods. The ScrFaure, LinScrFaure, and RS-scrHalton methods still provide estimates with smaller variance, but except for RS-scrHalton, the increased execution time cancels out gains in error reduction. The asymptotically unbiased RS-scrHalton method gave the best efficiency results, however, the sample size should be sufficiently large for the inherent bias to stay within acceptable levels. We note that our efficiency results constitute by no means the final word on this issue, and they are subject to change with faster implementations of the randomized quasi-Monte Carlo methods.

The adverse effects of high dimensionality to the quasi-Monte Carlo method can be lessened by using techniques that reduce the 'effective' dimension of the problem. Examples are the Brownian bridge technique (see Caflisch et al. (1997), who also discuss the notion of effective dimension) and the principal components technique (see Acworth et al., 1998). Our numerical results suggest that these methods may be needed to improve the quasi-Monte estimates in dimensions as small as 80.

We also observed that the Box–Muller transform consistently provided estimates with better statistical accuracy than that of the inverse transform method, in the case of randomized quasi-Monte Carlo methods. The interaction of random variable generation techniques (inverse transform, Box–Muller, etc.) with low-discrepancy sequences poses interesting research questions.

## Acknowledgements

We thank the anonymous referee for his/her helpful suggestions.

## References

Acworth, P., Broadie, M., Glasserman, P., 1998. A comparison of some Monte Carlo and quasi-Monte Carlo techniques for option pricing. In: Niederreiter, H., Hellekalek, P., Larcher, G., Zinterhof, P. (Eds.), Monte Carlo and Quasi-Monte Carlo Methods 1996, Proceedings of a Conference at the University of Salzburg, Austria, July 9–12, 1996, Springer, New York, pp. 1–18.

Bouleau, N., Lépingle, D., 1994. Numerical Methods for Stochastic Processes. Wiley, New York.

Boyle, P., Broadie, M., Glasserman, P., 1997. Monte Carlo methods for security pricing. Journal of Economic Dynamics and Control 21, 1267–1321.

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

Cranley, R., Patterson, T.N.L., 1976. Randomization of number theoretic methods for multiple integration. SIAM Journal of Numerical Analysis 13, 904–914.

D'Agostino, R.B., Stephens, M.A. (Eds.), 1986. Goodness-of-Fit Techniques. Marcel Dekker, New York.

den Haan, W.J., Marcet, A., 1990. Solving the stochastic growth model by parameterizing expectations. Journal of Business and Economic Statistics 8, 31–34.

Efron, B., Tibshirani, R.J., 1993. An Introduction to the Bootstrap. Chapman & Hall, New York.

Faure, H., 1992. Good permutations for extreme discrepancy. Journal of Number Theory 42, 47–56.

Fishman, V., Fitton, P., Galperin, Y., 1997. Hybrid low-discrepancy sequences: effective path reduction for yield curve scenario generation. Journal of Fixed Income 7, 75–84.

Galanti, S., Jung, A., 1997. Low-discrepancy sequences: Monte Carlo simulation of option prices. Journal of Derivatives 5, 63–83.

Göggelmann, K., Winker, P., Schellhorn, M., Franz, W., 2000. Quasi-Monte Carlo methods in stochastic simulations: an application to policy simulations using a disequilibrium model of the West German economy 1960–1994. Empirical Economics 25, 247–259.

Hickernell, F.J., 1996. The mean square discrepancy of randomized nets. ACM Transactions on Modeling and Computer Simulation 6, 274–296.

Hickernell, F.J., 2000. What affects the accuracy of quasi-Monte Carlo quadrature? In: Niederreiter, H., Spanier, J. (Eds.), Monte Carlo and Quasi-Monte Carlo Methods 1998, Proceedings of a Conference held at the Claremont Graduate University, Claremont, CA, USA, June 22–26, 1998, Springer, New York, pp. 16–55.

Hickernell, F.J., Hong, H.S., 1999. The asymptotic efficiency of randomized nets for quadrature. Mathematics of Computation 68, 767–791.

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

Jackson, M., Staunton, M., 2001. Advanced Modelling in Finance. Wiley, New York.

Joe, S., 1990. Randomization of lattice rules for numerical multiple integration. Journal of Computational and Applied Mathematics 31, 299–304.

Joy, C., Boyle, P.P., Tan, K.S., 1996. Quasi-Monte Carlo methods in numerical finance. Management Science 42, 926–938.

Judd, K.L., 1998. Numerical Methods in Economics. MIT, London.

Lambert, J.P., 1985. Quasi-Monte Carlo, low discrepancy sequences, and ergodic transformations. Journal of Computational and Applied Mathematics 12&13, 419–423.

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

Li, X.J., Winker, P., 2003. Time series simulation with quasi Monte Carlo methods. Computational Economics 21, 23–43.

Loh, W.-L., 1996. A combinatorial central limit theorem for randomized orthogonal array sampling designs. Annals of Statistics 24, 1209–1224.

Matoušek, J., 1998. On the $L_2$-discrepancy for anchored boxes. Journal of Complexity 14, 527–556.

Matsumoto, M., Kurita, Y., 1994. Twisted GFSR generators II. ACM Transactions on Modeling and Computer Simulation 4, 254–266.

Morohosi, H., Fushimi, M., 2000. A practical approach to the error estimation of quasi-Monte Carlo integrations. In: Niederreiter, H., Spanier, J. (Eds.), Monte Carlo and Quasi-Monte Carlo Methods 1998, Proceedings of a Conference held at the Claremont Graduate University, Claremont, CA, USA, June 22 –26, 1998, Springer, New York, pp. 377–390.

Morohosi, H., Fushimi, M., 2002. Experimental studies on the error estimation of the numerical integration by generalized Niederreiter sequences. Operations Research and Its Applications, Proceedings of the Fourth International Symposium, ISORA'02, World Publishing, New York, pp. 46–53.

Morohosi, H., Kobayashi, K., Fushimi, M., 2000. Estimating the error of quasi-Monte Carlo integrations—experimental studies in financial problems. Research Institute for Mathematical Sciences, Kyoto University, Kokyuroku, pp. 1–11.

Niederreiter, H., 1992. Random Number Generation and Quasi-Monte Carlo Methods. SIAM, Philadelphia, PA.

Ninomiya, S., Tezuka, S., 1996. Toward real-time pricing of complex financial derivatives. Applied Mathematical Finance 3, 1–20.

Owen, A.B., 1995. Randomly permuted $(t,m,s)$-nets and $(t,s)$-sequences. In: Niederreiter, H., Shiue, P.J.-S. (Eds.), Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing, Proceedings of a Conference at the University of Nevada, Las Vegas, NV, USA, June 23–25, 1994, Lecture Notes in Statistics, Vol. 106. Springer, New York, pp. 299–317.

Owen, A.B., 1997a. Monte Carlo variance of scrambled net quadrature. SIAM Journal on Numerical Analysis 34, 1884–1910.

Owen, A.B., 1997b. Scrambled net variance for integrals of smooth functions. Annals of Statistics 25, 1541–1562.

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

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

Owen, A.B., Tavella, D., 1997. Scrambled nets for value at risk calculations. In: Grayling, S. (Ed.), VaR: Understanding and Applying Value-at-Risk. RISK, London.

Ökten, G., 1999. Error reduction techniques in quasi-Monte Carlo integration. Mathematical and Computer Modelling 30, 61–69.

Ökten, G., 2002. Random sampling from low-discrepancy sequences: applications to option pricing. Mathematical and Computer Modelling 35, 1221–1234.

Paskov, S.H., Traub, J.F., 1995. Faster valuation of financial derivatives. Journal of Portfolio Management 22, 113–120.

Rubinstein, Y.R., 1981. Simulation And The Monte Carlo Method. Wiley, New York.

Struckmeier, J., 1995. Fast generation of low-discrepancy sequences. Journal of Computational and Applied Mathematics 61, 29–41.

Tan, K.S., Boyle, P.P., 2000. Applications of randomized low discrepancy sequences to the valuation of complex securities. Journal of Economic Dynamics and Control 24, 1747–1782.

Tezuka, S., 1995. Uniform Random Numbers: Theory and Practice. Kluwer Academic, Dordrecht.

Traub, F.J., Papageorgiou, A., 1996. Beating Monte Carlo. Risk 63–65.

Tuffin, B., 1996. On the use of low discrepancy sequences in Monte Carlo methods. Monte Carlo Methods and Applications 2, 295–320.

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