Randomized Quasi-Monte Carlo: An Introduction for Practitioners

Pierre L'Ecuyer

Randomized Quasi-Monte Carlo: An Introduction for Practitioners

Pierre L'Ecuyer

Abstract We survey basic ideas and results on randomized quasi-Monte Carlo (RQMC) methods, discuss their practical aspects, and give numerical illustrations. RQMC can improve accuracy compared with standard Monte Carlo (MC) when estimating an integral interpreted as a mathematical expectation. RQMC estimators are unbiased and their variance converges at a faster rate (under certain conditions) than MC estimators, as a function of the sample size. Variants of RQMC also work for the simulation of Markov chains, for function approximation and optimization, for solving partial differential equations, etc. In this introductory survey, we look at how RQMC point sets and sequences are constructed, how we measure their uniformity, why they can work for high-dimensional integrals, and how can they work when simulating Markov chains over a large number of steps.

1 Introduction

We consider a setting in which Monte Carlo (MC) or quasi-Monte Carlo (QMC) is used to estimate the expectation $\mu = \mathbb{E}[X]$ of a random variable X defined over a probability space $(\Omega, \mathscr{F}, \mathbb{P})$. We assume that $\omega \in \Omega$ can be identified with a sequence of s independent $\mathscr{U}(0,1)$ random variables (uniform over $(0,1)$) for some integer $s > 0$, so we can write $X = f(\mathbf{U})$ and

$$\mu = \int_0^1 \cdots \int_0^1 f(u_1, \ldots, u_s)\, du_1 \cdots du_s = \int_{(0,1)^s} f(\mathbf{u})\, d\mathbf{u} = \mathbb{E}[f(\mathbf{U})], \qquad (1)$$

Pierre L'Ecuyer
Département d'Informatique et de Recherche Opérationnelle, Université de Montréal, Canada, and Inria, Rennes, France, lecuyer@iro.umontreal.ca

for some function $f : \Omega = (0,1)^s \to \mathbb{R}$, where $\mathbf{u} = (u_1, \ldots, u_s) \in (0,1)^s$, and $\mathbf{U} \sim \mathscr{U}(0,1)^s$ (uniform over the unit hypercube). We can allow s to be random and unbounded; then this model is very general [28].

The standard *Monte Carlo* (MC) method estimates μ by

$$\bar{X}_n = \frac{1}{n} \sum_{i=0}^{n-1} X_i \tag{2}$$

where $X_i = f(\mathbf{U}_i)$ and $\mathbf{U}_0, \ldots, \mathbf{U}_{n-1}$ are n independent $\mathscr{U}(0,1)^s$ random vectors. In implementations, these $\mathbf{U}_i$ are replaced by vectors of "random numbers" that drive the simulation, but the MC theory developed under the above probabilistic assumptions still works well. We have $\mathbb{E}[\bar{X}_n] = \mu$ and $\mathrm{Var}[\bar{X}_n] = \sigma^2/n$ where $\sigma^2 := \int_{(0,1)^s} f^2(\mathbf{u})\, d\mathbf{u} - \mu^2$. If $\sigma^2 < \infty$ then when $n \to \infty$ we have $\bar{X}_n \to \mu$ with probability 1 by the strong law of large numbers and $\sqrt{n}(\bar{X}_n - \mu)/S_n \Rightarrow \mathscr{N}(0,1)$ (the standard normal distribution) by the usual central limit theorem (CLT), where $S_n^2 = \frac{1}{n-1} \sum_{i=0}^{n-1} (X_i - \bar{X}_n)^2$. This CLT is invoked routinely to compute a confidence interval on μ based on a normal approximation. The width of this confidence interval is asymptotically proportional to $\sigma/\sqrt{n}$, which means that for each additional decimal digit of accuracy on μ, one must multiply the sample size n by 100.

Quasi-Monte Carlo (QMC) replaces the independent random points $\mathbf{U}_i$ by a set of *deterministic* points $P_n = \{\mathbf{u}_0, \ldots, \mathbf{u}_{n-1}\}$ that cover $[0,1)^s$ more evenly. (Here we include 0 in the interval because some of these deterministic points often have coordinates at 0.) It estimates μ by

$$\bar{\mu}_n = \frac{1}{n} \sum_{i=0}^{n-1} f(\mathbf{u}_i).$$

Roughly speaking, P_n is called a *highly-uniform point set* or *low-discrepancy point set* if some measure of *discrepancy* between the empirical distribution of P_n and the uniform distribution converges to 0 faster than $\mathscr{O}(n^{-1/2})$, which is the typical rate for independent random points, when $n \to \infty$. QMC theory defines several types of discrepancies, usually by defining function spaces (often Hilbert spaces) $\mathscr{H}$ in which by applying the Cauchy-Schwarz inequality, one obtains the worst-case error bound

$$|\bar{\mu}_n - \mu| \le D(P_n) V(f) \tag{3}$$

for all $f \in \mathscr{H}$, where $V(f) = \|f - \mu\|_{\mathscr{H}}$ is the norm of $f - \mu$ in $\mathscr{H}$ (it measures the *variation* of f), and $D(P_n)$ is the *discrepancy* measure of P_n associated with this space [9, 18, 50]. For any fixed $f \in \mathscr{H}$ with $V(f) \neq 0$, this error bound converges at the same rate as $D(P_n)$. The error itself sometimes converges faster than the bound. To capture this, Hickernell [21] considers a setting in which (3) is transformed into an equality by introducing a third multiplicative factor on the right side. This new factor is the *confounding* between f and the empirical distribution of the points $\mathbf{u}_i$; it is always in $[0,1]$ and is equal to the left term divided by the right term in (3). The resulting equality is named a *trio identity*.

In a well-known special case, $D(P_n)$ is the *star discrepancy* $D^*(P_n)$, defined as follows: for each rectangular box $[\mathbf{0},\mathbf{u})$ with opposite corners at $\mathbf{0}$ and $\mathbf{u}$, let $\Delta(\mathbf{u})$ be the absolute difference between the volume of the box and the fraction of P_n that fall in that box. Then define $D^*(P_n)$ as the supremum of $\Delta(\mathbf{u})$ over all $\mathbf{u} \in (0,1)^s$. There are known explicit constructions that can provide a P_n for each n, for which $D^*(P_n) = \mathcal{O}(n^{-1}(\ln n)^{s-1})$. Variants of these constructions provide an infinite sequence of points for which $D^*(P_n) = \mathcal{O}(n^{-1}(\ln n)^s)$ if P_n comprises the first n points of the sequence. One important limitation of (3) for this case is that the corresponding $V(f)$, known as the Hardy-Krause variation of f, is infinite as soon as f has a discontinuity that is not aligned with the axes. Computing $D^*(P_n)$ explicitly is also difficult: the best known algorithms are polynomial in n but exponential in s; see [12] for a coverage of various types of discrepancies and their computation.

There are other interesting (Hilbert) spaces of periodic smooth functions for which the corresponding $D(P_n)$ in (3) converges as $\mathcal{O}(n^{-\alpha+\varepsilon})$ for any $\varepsilon > 0$, for some $\alpha > 0$ that depends on the space and can be arbitrarily large. The main construction methods for P_n are *lattice rules* and *digital nets*. We discuss them later and give examples.

With deterministic QMC, it is hard to estimate the integration error in practice. *Randomized quasi-Monte Carlo* (RQMC) randomizes the QMC points in a way that for the RQMC point set $P_n = \{\mathbf{U}_0, \ldots, \mathbf{U}_{n-1}\} \subset (0,1)^s$ (which is now random),

(i) each point $\mathbf{U}_i$ has the uniform distribution over $(0,1)^s$;
(ii) P_n as a whole is a low-discrepancy point set.

This turns QMC into a variance-reduction method. The RQMC estimator

$$\hat{\mu}_{n,\text{rqmc}} = \frac{1}{n}\sum_{i=0}^{n-1} f(\mathbf{U}_i) \tag{4}$$

is an unbiased estimator of μ, with variance

$$\text{Var}[\hat{\mu}_{n,\text{rqmc}}] = \frac{\text{Var}[f(\mathbf{U}_i)]}{n} + \frac{2}{n^2}\sum_{i<j}\text{Cov}[f(\mathbf{U}_i), f(\mathbf{U}_j)]. \tag{5}$$

We want to make the last sum as negative as possible, by inducing pairwise negative covariance. Well-known ways of creating such negative correlation include antithetic variates (with $n = 2$), Latin hypercube sampling (LHS), and stratification. The first two can reduce the variance under some conditions, but do not improve the $\mathcal{O}(n^{-1/2})$ MC rate. Stratification and other RQMC methods based for example on lattice rules and digital nets can improve the rate; see Section 2. Some RQMC methods provide a better convergence rate than the squared worst-case error, because the average over the randomizations is sometimes better than the worst case [17, 48, 56].

Note that because of the nonzero covariances, we cannot use the sample variance of the $f(\mathbf{U}_i)$ to estimate $\text{Var}[\hat{\mu}_{n,\text{rqmc}}]$ as for MC. To estimate the variance, we can simulate m independent realizations $X_1, \ldots, X_m$ of $\hat{\mu}_{n,\text{rqmc}}$, then estimate μ and $\text{Var}[\hat{\mu}_{n,\text{rqmc}}]$ by their sample mean $\bar{X}_m$ and sample variance S_m^2. This can be used to estimate the integration error. It may seem natural to compute a confidence interval

by assuming that $\bar{X}_m$ is approximately normally distributed, but one should be careful: The CLT holds in general for $m \to \infty$, but for fixed m and $n \to \infty$ it holds only for a few RQMC methods [38, 45]. When applying RQMC to estimate μ, for a given total computing budget mn, we prefer n as large as possible to benefit from the faster convergence rate in n, and then m is small (e.g., 10 or 20) and $\bar{X}_m$ may be far from normally distributed. We give an example in Section 4.1.

In the remainder of this tutorial, we focus on RQMC to estimate an integral on the unit cube, and we assume that the goal is to reduce the variance. For simplicity, we assume that s and n are fixed. There are constructions (not discussed here) in which the point sets can be expanded dynamically in dimension by adding new coordinates and in size by adding new points without changing the existing points. There are settings in which the criterion to minimize is not the variance. It can be the worst-case error or the average error for some class of functions, for example. This may give rise to various kinds of Hilbert (or Banach) spaces and discrepancies that we do not examine here. We look at just a few RQMC settings to give illustrations. Another interesting fact for practitioners is that the faster convergence rates of RQMC are proved under conditions on f that are often not satisfied in applications, but despite this, RQMC often reduces the variance by significant factors in those applications, so it is worth giving it a try and comparing RQMC vs MC empirical variances.

RQMC methods are best-known for estimating an integral, but they can effectively apply much more widely, for example to estimate the derivative of an expectation, or a function of several expectations, or a quantile, or a density, or the optimizer of a parameterized expectation or function, etc. This tutorial does not (and cannot) cover every aspect of RQMC. It gives more emphasis to what the author knows best. For more on QMC and RQMC, see for example [9, 33, 34, 28, 41, 50, 55, 58]. Tutorials on more advanced QMC topics can be found elsewhere in this book.

2 RQMC point set constructions

2.1 Stratification

A first approach to obtain a negative sum of covariances in (5) is to stratify the unit hypercube [16]. We partition axis j in $k_j \geq 1$ equal parts, for $j = 1, \ldots, s$. This determines $n = k_1 \cdots k_s$ rectangular boxes of volume $1/n$. Then we draw n random points, one per box, independently and uniformly in each box. Fig. 1 (left) gives an illustration with $s = 2$, $k_1 = 12$, $k_2 = 8$, and $n = 12 \times 8 = 96$.

The stratified estimator in the general case is

$$X_{\mathrm{s},n} = \frac{1}{n} \sum_{j=0}^{n-1} f(\mathbf{U}_j).$$

The crude MC variance with n points can be decomposed as

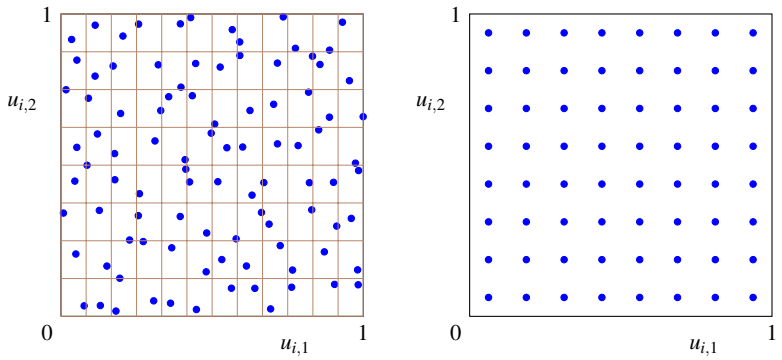

Fig. 1 An illustration of stratified sampling (left) and a midpoint rule (right) over $(0,1)^2$.

$$\mathrm{Var}[\bar{X}_n] = \mathrm{Var}[X_{\mathrm{s},n}] + \frac{1}{n}\sum_{j=0}^{n-1}(\mu_j - \mu)^2$$

where μ_j is the mean over box j. The more the μ_j's differ, the more the variance is reduced. Stratification provides an unbiased estimator and never increases the variance. One can estimate the variance by replicating the scheme $m \geq 2$ times, computing the empirical variance in each box, and averaging. If f' is continuous and bounded, and all k_j are equal to k (so $n = k^s$), then by using a Taylor expansion in each box one can show [16] that $\mathrm{Var}[X_{\mathrm{s},n}] = \mathcal{O}(n^{-1-2/s})$. This may provide a significant improvement when s is small, but for large s, the rate is not much better than for MC, and the method quickly becomes impractical because n increases exponentially with k. Nevertheless, it is sometimes effective to apply stratification to just a few important (selected) coordinates.

It is interesting to note that for f' continuous and bounded, a (deterministic) multivariate midpoint rule which takes one point at the center of each box, as shown in the right panel of Fig. 1 for $k_1 = k_2 = 8$, gives the same rate as stratification for the worst-case square integration error. For the midpoint rule, each one-dimensional projection of P_n has only d distinct points, each two-dimensional projection has only d^2 distinct points, etc. This means that for integrating functions that depend (mostly) on just a few of the s coordinates, many of the n points are identical with respect to those important coordinates, so the scheme becomes inefficient.

2.2 Lattice rules

An *integration lattice* is a vector space of the form

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

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 vectors of integers. A *lattice rule* is a QMC method that takes $P_n = \{\mathbf{u}_0, \ldots, \mathbf{u}_{n-1}\} = L_s \cap [0,1)^s$. It has *rank 1* if we can write $\mathbf{u}_i = i\mathbf{v}_1 \bmod 1$ for $i = 0, \ldots, n-1$, where $n\mathbf{v}_1 = \mathbf{a} = (a_1, \ldots, a_s) \in \{0, 1, \ldots, n-1\}^s$. These are the most widely used rules in practice. We have a *Korobov rule* if $\mathbf{a} = (1, a, a^2 \mod n, \ldots, a^{s-1} \mod n)$ for some integer a such that $1 \le a < n$.

Fig. 2 shows the points of a two-dimensional Korobov lattice rule with $n = 101$ and $\mathbf{a} = (1, 12)$ on the left and $\mathbf{a} = (1, 51)$ on the right. In both cases the points have a lattice structure. They are very evenly spread over the unit square for $\mathbf{a} = (1, 12)$, but for $\mathbf{a} = (1, 51)$ all the points of P_n lie on two parallel straight lines! Thus, the joint choice of n and $\mathbf{a}$ must be done carefully.

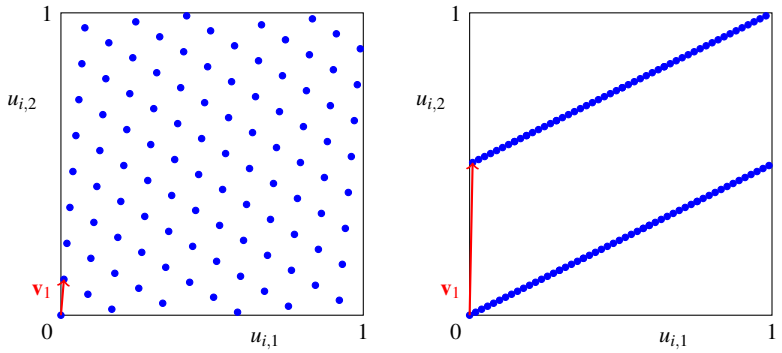

Fig. 2 An integration lattice with $s = 2$, $n = 101$, with $\mathbf{v}_1 = (1, 12)/n$ on the left and $\mathbf{v}_1 = (1, 51)/n$ on the right. In both cases, we can take $\mathbf{v}_2 = (0, 1)$.

A lattice rule can be turned into an RQMC method simply by shifting the lattice randomly, modulo 1, with respect to each coordinate [7, 33]. One generates a single point $\mathbf{U}$ uniformly in $(0,1)^s$, and adds it to each point of P_n modulo 1, coordinate-wise. This satisfies the two RQMC conditions. Fig. 3 gives an example in which $\mathbf{U} = (0.40, 0.08)$, for the lattice rule of Fig. 2.

A good randomly-shifted lattice rule provides an unbiased estimator with points that seem to cover the unit cube more evenly than independent points, but does it make the variance converge faster with n than MC? The answer is yes, under some conditions on the integrand f. Suppose f has Fourier expansion

$$f(\mathbf{u}) = \sum_{\mathbf{h} \in \mathbb{Z}^s} \hat{f}(\mathbf{h}) e^{2\pi\sqrt{-1}\mathbf{h}^t\mathbf{u}}.$$

For a *randomly shifted lattice*, the exact variance is always (see [33]):

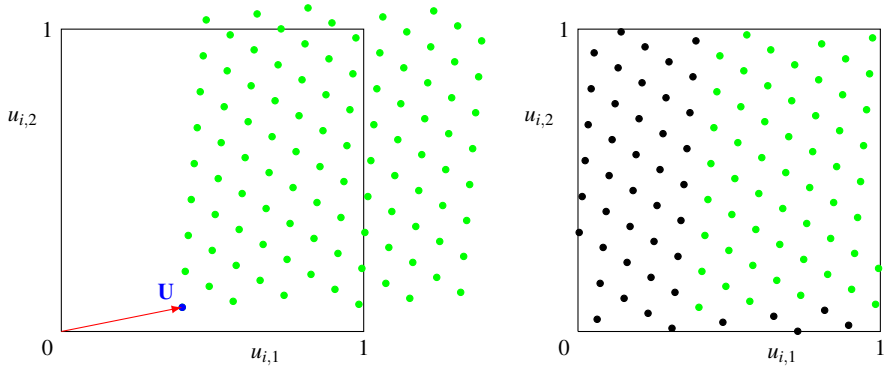

Fig. 3 Random shift modulo 1 for the lattice of Fig. 2 (left), with $\mathbf{U} = (0.40, 0.08)$.

$$\mathrm{Var}[\hat{\mu}_{n,\mathrm{rqmc}}] = \sum_{\mathbf{0}\neq\mathbf{h}\in L_s^*} |\hat{f}(\mathbf{h})|^2, \tag{6}$$

where $L_s^* = \{\mathbf{h} \in \mathbb{R}^s : \mathbf{h}^t\mathbf{v} \in \mathbb{Z} \text{ for all } \mathbf{v} \in L_s\} \subseteq \mathbb{Z}^s$ is the *dual lattice*. Thus, from the viewpoint of variance reduction, an optimal lattice for any given f is one that minimizes (6). But finding it for a given f is generally too hard and unrealistic.

Let $\alpha > 0$ be an even integer. If f has *square-integrable mixed partial derivatives* up to order $\alpha/2 > 0$, and the periodic continuation of its derivatives up to order $\alpha/2 - 1$ is *continuous across the boundaries of the unit cube modulo 1*, then it is known that $|\hat{f}(\mathbf{h})|^2 = \mathcal{O}((\max(1,h_1)\cdots\max(1,h_s))^{-\alpha})$. It is also known that for any $\varepsilon > 0$, there is always a vector $\mathbf{v}_1 = \mathbf{v}_1(n)$ such that

$$\mathscr{P}_\alpha := \sum_{\mathbf{0}\neq\mathbf{h}\in L_s^*} (\max(1,h_1)\cdots\max(1,h_s))^{-\alpha} = \mathcal{O}(n^{-\alpha+\varepsilon}). \tag{7}$$

This $\mathscr{P}_\alpha$ has been proposed long ago as a figure of merit, often with $\alpha = 2$ [58]. It is the variance for a *worst-case* f having $|\hat{f}(\mathbf{h})|^2 = (\max(1,|h_1|)\cdots\max(1,|h_s|))^{-\alpha}$. A larger α means a smoother f and a faster convergence rate. This $\mathscr{P}_\alpha$ is defined by an infinite sum, which cannot be computed exactly in general. However, when α is an even integer, the worst-case f is

$$f^*(\mathbf{u}) = \sum_{\mathrm{u}\subseteq\{1,\ldots,s\}} \prod_{j\in\mathrm{u}} \frac{(2\pi)^{\alpha/2}}{(\alpha/2)!} B_{\alpha/2}(u_j)$$

where $B_{\alpha/2}$ is the Bernoulli polynomial of degree $\alpha/2$ (e.g., $B_1(u) = u - 1/2$), and $\mathscr{P}_\alpha$ can be written as a finite sum that is easy to compute and can be used as a criterion to search for good lattices; see (9) in Section 3, where we give a more general version.

Thus, under the above conditions on the periodic continuation of f, for all $\varepsilon > 0$ there is a sequence of integration lattices indexed by n for which the variance converges as $\mathcal{O}(n^{-\alpha+\varepsilon})$. What if f does not satisfy these conditions? One can often change f to a continuous and/or smoother function that integrates to the same μ over $[0,1)^s$, usually via a change of variable. For example, suppose f is continuous in $[0,1)$, but discontinuous at the boundary, i.e., $f(\ldots,u_j=0,\ldots) \neq f(\cdots,u_j=1,\ldots)$. To simplify the exposition, suppose f is a function of a single real variable u (the other ones are fixed). Consider the change of variable $v=\varphi(u)=2u$ if $u \leq 1/2$ and $1-2u$ if $u > 1/2$. It is known as the *baker* transformation [20]: it stretches the points by a factor of 2, from $[0,1)$ to $[0,2)$, and folds back the segment $[1,2)$ to $[1,0)$. Its impact on the RQMC estimator (4) is exactly equivalent to compressing the graph of f horizontally by a factor of 1/2, and then making a mirror copy on the interval $[1/2,1)$. The transformed f is a continuous function. Fig. 4 gives an illustration. In practice, it is more convenient to apply the baker transformation to the randomized points $\mathbf{U}_i$ instead of changing f. Higher-order transformations can also make the derivatives (up any given order) continuous and improve the asymptotic rate even further, but they often increases the variance for "practical" values of n by increasing $V(f)$, so are not necessarily better in the end.

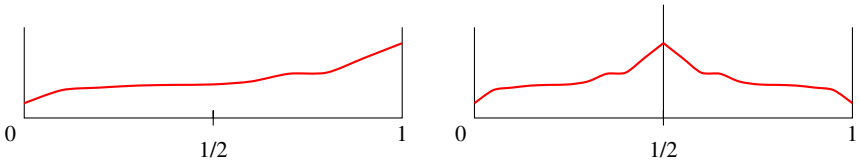

Fig. 4 Applying the baker transformation to the points $\mathbf{U}_i$ is equivalent to transforming f as shown.

Note that the worst-case function for the bounds we discussed is not necessarily representative of what happens in applications. Also, the *hidden factor in the $\mathcal{O}$ may increase quickly with s*, so the rate result in (7) is not very useful for large s. To get a bound that is uniform in s, the Fourier coefficients must decrease faster with the dimension and "size" of vectors $\mathbf{h}$; that is, f must be "smoother" in high-dimensional projections [10, 59, 60]. This is typically what happens in applications for which RQMC is effective. The criterion (9) will take that into account.

2.3 Digital nets

Niederreiter [50] defines a *digital net in base b* as follows. Choose the base b, usually a prime number or a power of a prime, and an integer $k > 0$. For $i = 0,\ldots,b^k-1$ and $j = 1,\ldots,s$, put

$$i = a_{i,0} + a_{i,1}b + \cdots + a_{i,k-1}b^{k-1} \;=\; a_{i,k-1}\cdots a_{i,1}a_{i,0},$$
$$\begin{pmatrix} u_{i,j,1} \\ \vdots \\ u_{i,j,w} \end{pmatrix} = \mathbf{C}_j \begin{pmatrix} a_{i,0} \\ \vdots \\ a_{i,k-1} \end{pmatrix} \bmod b,$$
$$u_{i,j} = \sum_{\ell=1}^{w} u_{i,j,\ell} b^{-\ell}, \qquad \mathbf{u}_i = (u_{i,1}, \ldots, u_{i,s}),$$

where the *generating matrices* $\mathbf{C}_j$ are $w \times k$ with elements in $\mathbb{Z}_b$. This gives $n = b^k$ points. In practice, w and k are finite, but there is no limit. The definition in [50] is actually more general: One can define bijections between $\mathbb{Z}_b$ and some ring R, and perform the multiplication in R. Assuming that each $\mathbf{C}_j$ has full rank, each one-dimensional projection truncated to its first k digits is $\mathbb{Z}_n/n = \{0, 1/n, \ldots, (n-1)/n\}$. That is, each $\mathbf{C}_j$ defines a permutation of $\mathbb{Z}_n/n$.

If each $\mathbf{C}_j$ is defined with an infinite number of columns, then we have an infinite sequence of points, called a *digital sequence in base b*. One can always take the first $n = b^k$ points of a digital sequence to define a digital net, for any k.

Measuring uniformity. A standard way of measuring the uniformity of a digital net in base b with $n = b^k$ points is as follows [34, 50]. Suppose we divide axis j in b^{q_j} equal parts for some integer $q_j \geq 0$, for each j. This determines a partition of $[0,1)^s$ into $2^{q_1+\cdots+q_s}$ rectangles of equal sizes. If each rectangle contains exactly the same number of points from P_n, we say that P_n is $(q_1, \ldots, q_s)$-*equidistributed in base b*. This occurs if and only if the matrix formed by the first q_1 rows of $\mathbf{C}_1$, the first q_2 rows of $\mathbf{C}_2$, ..., the first q_s rows of $\mathbf{C}_s$, is of full rank (mod b). The $(q_1, \ldots, q_s)$-equidistribution can be verified by constructing this matrix and checking its rank. We say that P_n is a (t,k,s)-*net in base b* if and only if it is $(q_1, \ldots, q_s)$-equidistributed whenever $q_1 + \cdots + q_s = k - t$. This is possible for $t = 0$ only if $b \geq s - 1$. The t-*value* of a digital net is the smallest t for which it is a (t,k,s)-net.

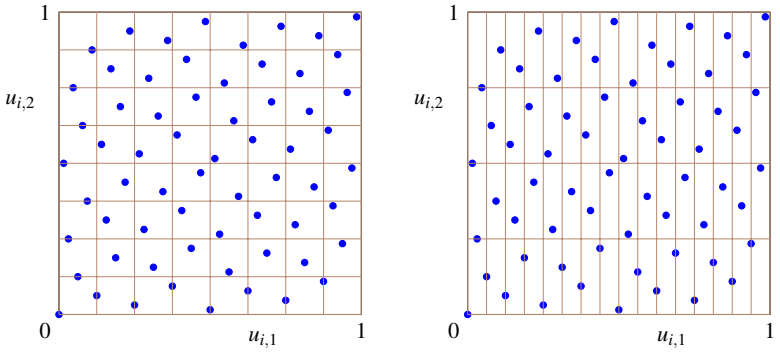

Fig. 5 The Hammersley point set (or Sobol net with appended first coordinate) for $s = 2$ and $n = 64$

Fig. 5 gives an example of a $(0,6,2)$-net in base 2. The equidistribution can be observed on the left with $q_1 = q_2 = 3$ and on the right with $q_1 = 4$ and $q_2 = 2$. This point set is obtained by taking $\mathbf{C}_2$ as the identity matrix and $\mathbf{C}_1$ as the reverse identity (with 1's on the descending diagonal). The points are enumerated by their first coordinate and the second coordinate follows the van der Corput sequence in base 2. Many of the points sit exactly on the left or bottom boundary of their rectangle in Fig. 5, because only the first six bits of each coordinate can be nonzero. For any integer $k > 0$, this construction (with these $\mathbf{C}_1$ and $\mathbf{C}_2$) is a $(0,k,2)$-net in base 2; it is the two-dimensional *Hammersley point set*.

An infinite sequence $\{\mathbf{u}_0, \mathbf{u}_1, \dots\}$ in $[0,1)^s$ is a (t,s)-*sequence in base b* if for all $k > 0$ and $\nu \geq 0$, $Q(k,\nu) = \{\mathbf{u}_i : i = \nu b^k, \dots, (\nu+1)b^k - 1\}$, is a (t,k,s)-net in base b. This is possible for $t = 0$ only if $b \geq s$.

A key property that connects digital nets with the star discrepancy and QMC error bounds is that for fixed s, if P_n is a (t,k,s)-net in base b for $k = 1,2,3,\dots$ and t is bounded, then $D^*(P_n) = \mathcal{O}(n^{-1}(\log n)^{s-1})$, and for any f having finite Hardy-Krause variation $V(f)$, the error bound (3) with these point sets converges at this same rate.

Specific constructions. The most popular (and oldest) specific instance of digital sequence was proposed by Sobol' [61], in base $b = 2$. Each binary matrix $\mathbf{C}_j$ is upper triangular with ones on the main diagonal. The bits above the diagonal in any given column form the binary expansion of an integer called a *direction number*. The first few direction numbers are selected and the following columns are determined by a bitwise linear recurrence across the columns. The choice of the initial direction numbers is important for the quality of the points. For original values proposed by Sobol', in particular, the uniformity of several low-dimensional projections are very poor. Better direction numbers are proposed in [26, 42], for example. Fig. 6 shows the first 64 points of the Sobol sequence in base 2, for which $\mathbf{C}_1$ is the identity matrix. These points form a $(0,6,2)$-net in base 2, just like the Hammersley points of Fig. 5.

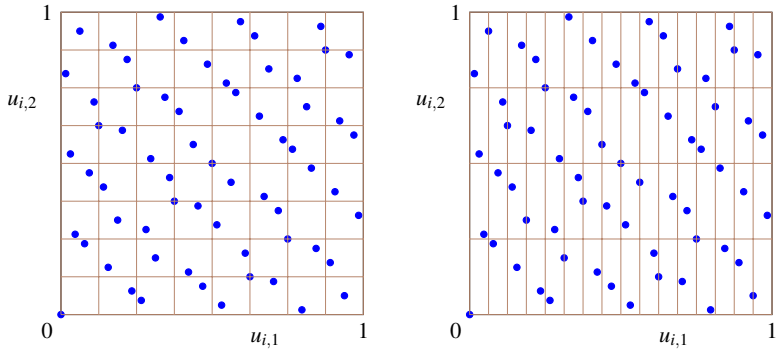

Fig. 6 The first $n = 64$ Sobol points in $s = 2$ dimensions

Faure [14] proposed digital sequences in base b for any prime b by taking $\mathbf{C}_j$ as the $(j-1)$th power of the Pascal matrix, modulo b. He proved that the resulting sequence is a $(0,s)$-sequence in base b for any $s \le b$. The latter condition is a practical limitation, because it imposes the choice of a large base when s is large. Also, the arithmetic modulo a prime $b > 2$ is less convenient and generally slower on computers than arithmetic modulo 2. Other sequences and nets for arbitrary prime power bases and nets in base 2 with better t-values than those of Sobol' are also available (see, e.g., [9, 51]), but they rarely outperform Sobol' points in applications.

For all these sequences, if we fix $n = b^k$, we can take the first coordinate of point i as i/n, which corresponds to taking $\mathbf{C}_1$ as the reflected identity matrix, and then take $\mathbf{C}_{j+1}$ as the old $\mathbf{C}_j$, for $j \ge 1$. It turns out that by doing this with a (t,s)-sequence, we can obtain a $(t,k,s+1)$-net for any k. That is, we gain one coordinate in the net. By doing this with the Sobol' sequence with $s = 1$, we obtain the Hammersley net illustrated in Fig. 5. Other types of digital net constructions can be found in [9, 43, 52] and the references given there.

Randomization. If we apply a random shift modulo 1 to a digital net, the equidistribution properties are not preserved in general. However, a *random digital shift* in base b preserves them and also satisfies the two criteria that define RQMC. It works as follows. As for the ordinary random shift, we generate a single $\mathbf{U} = (U_1, \ldots, U_s) \sim \mathscr{U}[0,1)^s$ where $U_j = \sum_{\ell=1}^{w} U_{j,\ell} b^{-\ell}$. For coordinate j of point i, before the shift we have $u_{i,j} = \sum_{\ell=1}^{w} u_{i,j,\ell} b^{-\ell}$. The digital shift replaces each $u_{i,j}$ by $\tilde{U}_{i,j} = \sum_{\ell=1}^{w} [(u_{i,j,\ell} + U_{j,\ell}) \bmod b] b^{-\ell}$. It is not difficult to show that if P_n is $(q_1, \ldots, q_s)$-equidistributed in base b before the shift, it retains this property after the shift. Moreover, if $w = \infty$, each randomized point $\tilde{\mathbf{U}}_i$ has the uniform distribution over $(0,1)^s$. As a result, if f has finite Hardy-Krause variation and we use (t,k,s)-nets with fixed s and bounded t for RQMC with a random digital shift, the estimator $\hat{\mu}_{n,\mathrm{rqmc}}$ is unbiased and by squaring the worst-case error we immediately find that its variance converges as $\mathscr{O}(n^{-2}(\log n)^{2(s-1)})$. (Better rates are obtained for certain classes of smooth functions and certain types of randomizations; see below.)

In base $b = 2$, the digital shift consists in applying a bitwise XOR between $\mathbf{U}$ and each $\mathbf{u}_i$. To illustrate how the equidistribution is preserved, take for example $k_1 = 3$ and $k_2 = 5$. For the given $\mathbf{U}$, the bits marked as "C" in the result have been flipped and those still marked with $*$ are unchanged:

$$\begin{aligned} \mathbf{u}_i &= (0.\star\star\star,\ \ 0.\star\star\star\star\star)_2 \\ \mathbf{U} &= (0.101,\ \ 0.01011)_2 \\ \mathbf{u}_i \oplus \mathbf{U} &= (0.\mathrm{C}\star\mathrm{C},\ \ 0.\star\mathrm{C}\star\mathrm{CC})_2 \end{aligned}$$

If the eight considered bits for $\mathbf{u}_i$ take each of the 2^8 possible configurations exactly the same number of times when $i = 0, \ldots, n-1$, then this also holds for $\mathbf{u}_i \oplus \mathbf{U}$. More generally, for a digital net in base 2 with $n = 2^k$ points in s dimensions, this preservation holds for any $\mathbf{U}$ and any non-negative integers $k_1, \ldots, k_s$ such that $k_1 + \cdots + k_s \le k$. Fig. 7 shows a digital shift in base 2 with

$$\mathbf{U} = (0.10100101100..., 0.01011001100...)_2$$

applied to the Hammersley points (Sobol' net with one extra coordinate) of Fig. 5. For this given $\mathbf{U}$, for each point $\mathbf{u}_i$ we flip the first, third, sixth, ..., bits of the first coordinate, and we flip the second, fourth, fifth, ..., bits of the second coordinate. The figure shows what happens when we flip the first bit of the first coordinate (top right); it permutes the left half with the right half. If the second bit in $\mathbf{U}$ was a 1 (here it is not), we would also permute the first (shaded) quarter with the second and the third (shaded) with the fourth. Since the third bit in $\mathbf{U}$ is 1, we flip the third bit of the first coordinate of each $\mathbf{u}_i$, The lower left plot shows the points after this flip, which has permuted each lightly colored vertical stripe with the yellow one on its right. After doing all the permutations specified by $\mathbf{U}$ for the first coordinate, we do the same with the second coordinate. The points after the full digital shift are shown in the lower right. Equidistribution is preserved because for each relevant partition in dyadic rectangular boxes, the digital shift only permutes those boxes (by pairs).

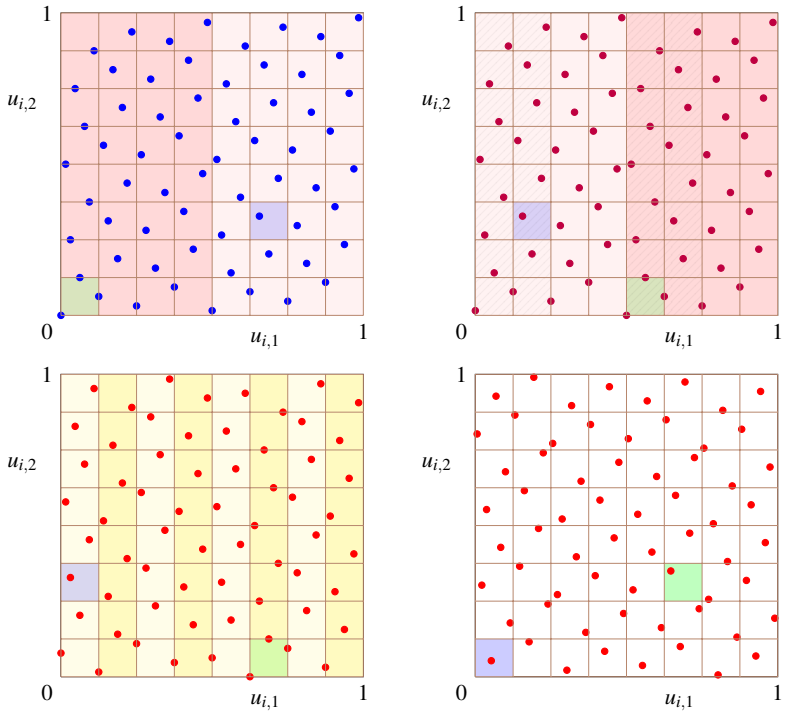

Fig. 7 A random digital shift with $\mathbf{U} = (0.10100101100..., 0.01011001100...)_2$ applied to a Sobol net with 64 points. The original points are in upper left. Flipping the first bit of the first coordinate permutes the left half with the right one, giving the upper right plot. Then flipping the third bit of the first coordinate gives the lower left plot. The lower right shows the points after the full digital shift of the two coordinates. One can follow the movement of the green and blue boxes during the permutations. In the end, those two square boxes are permuted.

A more powerful but more costly randomization for digital nets is the *nested uniform scramble* (NUS) of Owen [53, 54]. The difference with the digital shift (in base 2) is as follows. For any given coordinate, with probability 1/2 we flip the first bit of all points, just as before, then for the left half, we flip the second bit of all points with probability 1/2, and we do the same for the right half, but *independently*. That is, in the top right of Fig. 7, we would permute the first shaded area (first quarter) with the light area on its right (second quarter) with probability 1/2, and independently we would permute the second shaded area (third quarter) with the pink area on its right (fourth quarter). Then we do this recursively, and we repeat for each coordinate. For instance, in the lower left Fig. 7, we would use four independent random bits, one for each pair of successive (light, yellow) columns. Doing permutations like this for 31 bits or more would be very time-consuming, but in fact one can do it for the first k bits only, and then generate the other bits randomly and independently across points and coordinates. From a statistical viewpoint this is equivalent to NUS and less time-consuming [48]. NUS also works in general base b: for each digit of each coordinate, we generate a random permutation of b elements to permute the points according to this digit. Owen proved that under sufficient smoothness condition on f, for digital nets in base b with fixed s and bounded t, with this scrambling the variance converges as $\mathcal{O}(n^{-3}(\log n)^{s-1})$, which is better than for the random digital shift.

There are other types of permutations which are less costly than NUS and may help reduce the variance further compared with just a random digital shift. One popular example is the (left) *linear matrix scramble* [48, 64, 56]: left-multiply each matrix $\mathbf{C}_j$ by a random $w \times w$ matrix $\mathbf{M}_j$, non-singular and lower triangular, mod b. With this scrambling just by itself, each point does not have the uniform distribution (e.g., the point $\mathbf{0}$ is unchanged), but one can apply a random digital shift in base b after the matrix scramble to obtain an RQMC scheme. There are other types of linear matrix scrambles, such as the stripe scramble, ibinomial scramble, etc.; see [23, 48, 49, 56]. The proved convergence rate of the variance with these scrambles is generally the same as with a random digital shift alone.

3 Anova decomposition

Filling the unit hypercube very evenly requires an excessive number of points in large dimension. For example, in $s = 100$ dimensions, it takes $n = 2^{100}$ points to have one in each quadrant; this is unrealistic. The reason why RQMC might still work for large s is because f can often be well approximated by a sum of low-dimensional functions, and RQMC with a well-chosen point set can integrate these low-dimensional functions with small error. A standard way of formalizing this is as follows [44, 62].

An *ANOVA decomposition* of $f(\mathbf{u}) = f(u_1, \ldots, u_s)$ can be written as

$$f(\mathbf{u}) = \sum_{\mathfrak{u} \subseteq \{1,\ldots,s\}} f_{\mathfrak{u}}(\mathbf{u}) = \mu + \sum_{i=1}^{s} f_{\{i\}}(u_i) + \sum_{i,j=1}^{s} f_{\{i,j\}}(u_i, u_j) + \cdots \tag{8}$$

where

$$f_{\mathfrak{u}}(\mathbf{u}) = \int_{[0,1)^{|\bar{\mathfrak{u}}|}} f(\mathbf{u})\,\mathrm{d}\mathbf{u}_{\bar{\mathfrak{u}}} - \sum_{\mathfrak{v}\subset\mathfrak{u}} f_{\mathfrak{v}}(\mathbf{u}_{\mathfrak{v}}),$$

and the *Monte Carlo variance* decomposes accordingly as

$$\sigma^2 = \sum_{\mathfrak{u}\subseteq\{1,\ldots,s\}} \sigma^2_{\mathfrak{u}}, \quad \text{where } \sigma^2_{\mathfrak{u}} = \mathrm{Var}[f_{\mathfrak{u}}(\mathbf{U})].$$

Getting a rough estimate of the variance $\sigma^2_{\mathfrak{u}}$ captured by each subset $\mathfrak{u}$ of coordinates suffices to define relevant uniformity criteria that give more weight to the more important projections. The $\sigma^2_{\mathfrak{u}}$ can be estimated by MC or RQMC; see [47, 57].

One example of this is the following weighted version of $\mathscr{P}_\alpha$, with projection-dependent weights $\gamma_{\mathfrak{u}}$, in which $\mathfrak{u}(\mathbf{h}) = \mathfrak{u}(h_1,\ldots,h_s) = \{j : h_j \neq 0\}$:

$$\mathscr{P}_{\gamma,\alpha} = \sum_{\mathbf{0}\neq\mathbf{h}\in L^*_s} \gamma_{\mathfrak{u}(\mathbf{h})} (\max(1,|h_1|)\cdots\max(1,|h_s|))^{-\alpha}.$$

If $\alpha/2$ is a positive integer, for a lattice rule with $\mathbf{u}_i = (u_{i,1},\ldots,u_{i,s})$, we have

$$\mathscr{P}_{\gamma,\alpha} = \sum_{\emptyset\neq\mathfrak{u}\subseteq\{1,\ldots,s\}} \frac{1}{n}\sum_{i=0}^{n-1} \gamma_{\mathfrak{u}} \left[\frac{-(-4\pi^2)^{\alpha/2}}{(\alpha)!}\right]^{|\mathfrak{u}|} \prod_{j\in\mathfrak{u}} B_\alpha(u_{i,j}), \tag{9}$$

and the corresponding *variation* (squared) is

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

for $f : [0,1)^s \to \mathbb{R}$ smooth enough. Then,

$$\mathrm{Var}[\hat{\mu}_{n,\mathrm{rqmc}}] = \sum_{\mathfrak{u}\subseteq\{1,\ldots,s\}} \mathrm{Var}[\hat{\mu}_{n,\mathrm{rqmc}}(f_{\mathfrak{u}})] \leq V^2_\gamma(f)\,\mathscr{P}_{\gamma,\alpha}. \tag{10}$$

This $\mathscr{P}_{\gamma,\alpha}$ with properly chosen α and weights $\gamma_{\mathfrak{u}}$ is a good practical choice of figure of merit for lattice rules [10, 19]. The weights $\gamma_{\mathfrak{u}}$ are usually chosen to have a specific form with just a few parameters, such as order-dependent or product weights [35, 60]. The *Lattice Builder* software [36] permits one to search for good lattices for arbitrary n, s, and weights, using various figures of merit, under various constraints.

4 Examples

The numerical results reported in this paper were obtained using the Java library SSJ [29], which offers tools for RQMC and stochastic simulation in general. In the examples, all random variates are generated by inversion.

4.1 A stochastic activity network

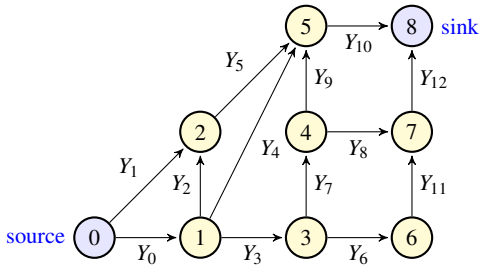

Fig. 8 A stochastic activity network

This example is from [2, 33]. We consider the stochastic activity network in Fig. 8, in which Y_j is the length of arc j, for $j = 0, \ldots, 12$. The Y_j are assumed independent with cdf's F_j given in [2, Section 4.1] and [35], and we generate them by inversion: $Y_j = F_j^{-1}(U_j)$ where $U_j \sim U(0,1)$. Let T be the (random) length of the longest path from node 0 to node 8. We compare RQMC and MC for two estimation problems: (a) estimating $\mathbb{P}[T > x]$ for some constant x and (b) estimating $\mathbb{E}[T]$.

To estimate $\mathbb{E}[T]$ we simply use T, so $s = 13$. To estimate $\mathbb{P}[T > x]$, we consider two base MC estimators. The first one is $X = \mathbb{I}[T > x]$ (where $\mathbb{I}[\cdot]$ is the indicator function) and the second one uses conditional Monte Carlo (CMC) as follows. We generate the Y_j's only for the 8 arcs that *do not* belong to the cut $\mathscr{L} = \{4,5,6,8,9\}$, and replace $\mathbb{I}[T > x]$ by its *conditional expectation* given those Y_j's,

$$X_{\text{CMC}} = \mathbb{P}\left[T > x \mid \{Y_j : j \notin \mathscr{L}\}\right] = 1 - \prod_{j \in \mathscr{L}} \mathbb{P}[Y_j \le x - P_j] \tag{11}$$

where P_j is the length of the longest path that goes through edge j when we put $Y_j = 0$. This X_{CMC} is easy to compute, as explained in [33], and is guaranteed to have smaller variance than the indicator (the first one) under MC. A more important advantage under RQMC is that CMC makes the estimator continuous as a function of the U_j's, whereas the first one is discontinuous. It also reduces the dimension s from 13 to 8. Fig. 9 shows the impact of CMC on the ANOVA decomposition. For each estimator, the length of the white box is proportional to the variance captured by one-dimensional projections, the second lightest box is for the two-dimensional projections, etc. CMC pushes much of the variance to the projections over one and two dimensions. The variance components $\sigma_{\mathfrak{u}}^2$ were estimated using the algorithm of [63] with RQMC (100 independent shifts of a lattice rule with $n = 2^{20} - 3$ points); see [35, Section 8] for the details. The bounds (3) and (10) are useless here because the mixed derivatives are not defined everywhere, so the variation is infinite. It will be interesting to see that RQMC nevertheless improves the variance. This is frequent in applications.

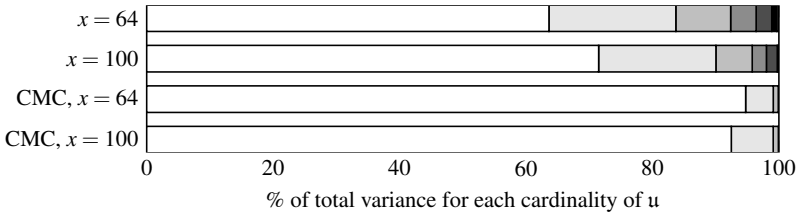

Fig. 9 ANOVA Variance captured by each projection order for estimators of $\mathbb{P}[T > x]$ for the stochastic activity network example

In an experiment reported in [35], an integration lattice of rank 1 was constructed for 50 different prime values of n ranging from $2^5 - 1$ to $2^{22} - 3$, roughly uniformly spread in log scale. For each n a generating vector $\mathbf{a}$ was found based on the criterion (9), with weights selected based on estimates of the ANOVA components. The variance was estimated for each n and a linear regression was fitted for $\log \mathrm{Var}[\hat{\mu}_{n,\mathrm{rqmc}}]$ vs $\log n$ to estimate the rate ν for which $\mathrm{Var}[\hat{\mu}_{n,\mathrm{rqmc}}] \propto n^{-\nu}$ in this range of values of n. For the estimation of $\mathbb{P}[T > x]$ with $x = 60$, for example, we found $\nu \approx 1.2$ for the standard estimator (indicator) and $\nu \approx 1.5$ with CMC. The log-log plot follows pretty much a straight line. Based on an interpolation from the regression model, for $n \approx 2^{20}$, RQMC reduces the variance approximately by a factor of 27 with the standard estimator and 4400 with CMC. This shows that the combined smoothing of f and dimension reduction provided by CMC has a large impact on the effectiveness of RQMC. For the estimation of $\mathbb{E}[T]$ (without CMC), we had $\nu \approx 1.47$. Table 1 reports the results for other RQMC schemes, namely Sobol' points with a linear matrix scramble and a random digital shift (LMS+DS), and Sobol' points with NUS. We see that the choice of scrambling makes almost no difference in this example.

Table 1 Empirical convergence rate ν and $\log_{10}$ of empirical variance with $n \approx 2^{20}$ (in parentheses) for the stochastic activity network example

	$\mathbb{P}[T > 60]$, MC		$\mathbb{P}[T > 60]$, CMC		$\mathbb{E}[T]$, MC	
	ν	$\log_{10}$ Var	ν	$\log_{10}$ Var	ν	$\log_{10}$ Var
Independent	1.00	(-6.6)	1.00	(-7.3)	1.04	(-3.5)
Lattice	1.20	(-8.0)	1.51	(-10.9)	1.47	(-6.3)
Sobol+LMS+DS	1.20	(-7.5)	1.68	(-11.2)	1.36	(-6.2)
Sobol+NUS	1.18	(-8.0)	1.65	(-11.2)	1.37	(-6.3)

Fig. 10 shows histograms for $m = 10000$ replications of the MC estimator with $n = 8191$ (left) and for a particular randomly shifted lattice rule with $n = 8191$ (right, taken from [38]). The MC histogram resembles a normal distribution, as expected, but the second one is far from normal. A confidence interval based on a normality assumption is certainly inappropriate in this case. The limiting distribution of an RQMC estimator based on a randomly-shifted lattice rule when $n \to \infty$ is analyzed

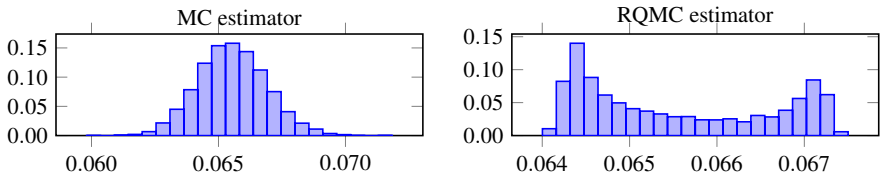

Fig. 10 Histogram of MC and RQMC estimators with $n = 8191$ for the stochastic activity network with CMC, for $x = 100$, based on $m = 10000$ replications of each estimator. The RQMC estimator is a randomly shifted lattice rule with $n = 8191$.

in [38]; the properly scaled limiting distribution is usually a spline, not a normal distribution. For a digital net with a digital random shift, the CLT does not apply either (in one dimension it is equivalent to a randomly-shifted lattice), but the CLT does apply for a digital net with NUS [45].

4.2 A financial option under a variance-gamma process

Alternative sampling schemes. Consider an asset price that evolves according to a *geometric variance-gamma* (GVG) process S defined by [3, 4, 46]:

$$S(t) = S(0)\exp\left[rt + X(G(t;1,\sqsubseteq),\mu,\sigma) + \omega t\right],$$

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

The realization of $Y(t)$ (and of $S(t)$) at the d observation points t_j can be generated in the following three ways (among others), as explained in [3]: Brownian and gamma sequential sampling (BGSS), Brownian and gamma bridge sampling (BGBS), and difference of gammas bridge sampling (DGBS). BGSS generates $\tau_1 = G(t_1)$, then $X(\tau_1)$, then $\tau_2 - \tau_1 = G(t_2) - G(t_1)$, then $X(\tau_2) - X(\tau_1)$, etc., in that order. This requires d gamma variates and d normal variates, so the dimension is $s = 2d$. BGBS generates $\tau_d = G(t_d)$, $X(\tau_d)$, $\tau_{d/2} = G(t_{d/2})$, $X(\tau_{d/2})$, $\tau_{d/4} = G(t_{d/4})$, $X(\tau_{d/4})$, $\tau_{3d/4} = G(t_{3d/4})$, $X(\tau_{3d/4})$, ..., in that order, exploiting the fact that 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 $X(\tau)$ conditional on $(X(\tau_a), X(\tau_b))$ is normal with known parameters. BGBS requires one gamma variate, $d-1$ beta variates, and d normal variates, so $s = 2d$. DGBS uses the fact that $\{S(t), t \ge 0\}$ can be written as a difference of two gamma processes, which can be simulated via a bridge (conditional) approach as for BGBS. This requires two

gamma variates and $2d-2$ beta variates. When d is a power of 2, all the beta variates are symmetric, and for that case there is a fast inversion algorithm [40]. The idea of the bridge constructions is to reformulate the integrand f in a way that more of the variance is captured by the low-dimensional projections on the first few coordinates of the points (the first few coordinates already determine a sketch of the trajectory), to make RQMC more effective.

For a numerical illustration, we take the following parameters from [4]: $\theta = -0.1436$, $\sigma = 0.12136$, $\nu = 0.3$, $r = 0.1$, $T = 1$, $K = 101$, and $S(0) = 100$. The exact value and the MC variance are $\mu \approx 5.725$ and $\sigma^2 \approx 29.89$. Table 2 compares the variance rates ν for RQMC (estimated from experiments with $n = 2^9, \ldots, 2^{21}$) and MC (for which $\nu = 1$ and the variance is the same for all sampling schemes). RQMC improves ν and reduces the variance in all three cases, BGBS gives the best rate empirically, and for $n = 2^{20}$ it reduces the variance by a factor of about 1000.

Table 2 Empirical convergence rate ν and $\log_{10}$ of empirical variance for $n = 2^{20}$ (in parentheses) for the option pricing example under a GVG process

	BGSS		BGBS		DGBS	
	ν	$\log_{10}$ Var	ν	$\log_{10}$ Var	ν	$\log_{10}$ Var
Independent	1.00	(-4.5)	1.00	(-4.5)	1.00	(-4.5)
Sobol+LMS+DS	1.26	(-6.5)	1.42	(-7.6)	1.34	(-7.6)

For comparison, we ran a similar experiment with the GVG process replaced by a geometric Brownian motion (GBM) process, for which

$$S(t) = S(0)\exp\left[(r-\sigma^2/2)t + X(t,0,\sigma)\right],$$

with the same parameter values (ν and θ are not used). We tried three sampling methods: sequential sampling (SS), Brownian bridge sampling (BBS), and Brownian sampling with principal component analysis (PCA). SS and BBS work as in the GVG case. For PCA, to generate the multivariate normal vector $(X(t_1), \ldots, X(t_d))$, we do a principal component decomposition of its covariance matrix, say $\Sigma = \mathbf{A}\mathbf{A}^{\mathsf{t}}$, and return $\mathbf{X} = \mathbf{A}\mathbf{Z}$ where $\mathbf{Z}$ is a vector of d independent standard normals. With this decomposition, the first few coordinates of $\mathbf{Z}$ (i.e., the first few coordinates of the RQMC points) capture a large fraction of the variance, even larger than with BBS. The results are in Table 3. SS does not improve the variance rate very much, BSS does better, and PCA much better. For PCA, $\nu \approx 1.9$ and the variance is reduced by a factor of over two millions for $n = 2^{20}$. Similar improvements were obtained in [35] with lattice rules constructed by Lattice Builder [36].

Option pricing examples with multiple correlated assets, in up to 250 dimensions, and in which PCA is very effective, can be found in [24, 27, 28]. For more on changing the sampling method of a Brownian motion to reduce the effective dimension and make RQMC more effective, see, e.g., [5, 25].

Control variates. There are various ways of reducing the RQMC variance by making the integrand smoother. Using control variates (CVs) is one of them. For

Table 3 Empirical convergence rate ν and $\log_{10}$ of empirical variance (in parentheses) for the option pricing example under a GBM process

		BSS		BBS		BPCA	
		ν	$\log_{10}$ Var	ν	$\log_{10}$ Var	ν	$\log_{10}$ Var
No CV	Independent	1.00	(-4.5)	1.00	(-4.5)	1.00	(-4.5)
	Sobol+LMS+DS	1.19	(-7.2)	1.42	(-8.8)	1.90	(-11.0)
With CV	Independent	1.00	(-7.8)	1.00	(-7.8)	1.00	(-7.8)
	Sobol+LMS+DS	1.21	(-9.5)	1.17	(-10.1)	1.37	(-11.1)

the Asian option under the GBM process, for instance, if we replace the arithmetic average $\bar{S}$ by a geometric average $\tilde{S} = (\prod_{j=1}^{d} S(t_j))^{1/d}$, there is a closed-form formula for the expected payoff, so this payoff can be used as a CV with either MC or RQMC [22, 33]. This is very effective in this example, especially when T and the volatility σ are not too large. Table 3 show some results for when we add this CV.

Importance sampling. Financial options for which the payoff is zero most of the time and takes a large or significant value only once in a while are not rare. Many options are indeed like insurance contracts, in which a nonzero claim is a *rare event* (it has a small probability). The contract value is then an integral whose integrand (the payoff function) has a peak in a small region and is zero elsewhere. MC performs poorly in this situation, because an excessively large sample size n might be required to obtain a sufficiently large number of occurrences of the rare event, and RQMC does not solve this problem. One appropriate tool then is *importance sampling* (IS), which can be seen as a change of variable that, when done properly, flattens out the integrand to reduce its variation. It can help both MC and RQMC.

We illustrate this with an ordinary European call option under the GVG model. We have $d = 1$ and $\bar{S} = S(t_1)$ in the payoff. We also take $t_1 = 1$ and $K = 180$, so a positive payoff is a rare event. We simulate the VG process via DGBS. To apply IS, we will increase the mean of G^+ and reduce the mean of G^- so $G^+(1) - G^-(1)$ takes larger values. A standard IS strategy for this is exponential twisting (see [1]): we multiply the density $g^+(x)$ of $G^+(1)$ by $e^{\theta x}$ and the density $g^-(y)$ of $G^-(1)$ by $e^{-\theta y}$, for some constant $\theta \geq 0$, and the re-normalize those densities. They are still gamma, but with different means. Then the (unbiased) IS estimator is the payoff multiplied by the likelihood ratio L of the old density product $g^+(y)g^-(y)$ divided by the new one. Since the payoff is nonzero only when $S(1) \geq K$, i.e., when $r + \omega + G^+(1) - G^-(1) = \ln[S(1)/S(0)] > \ln(K/S(0))$, we (heuristically) choose θ so that $\mathbb{E}[G^+(1) - G^-(1)] = \ln(K/S(0)) - r - \omega$ under IS. We write the expectation as a (monotone) function of θ and find the root θ^* of the equation numerically. For $K = 180$, we find $\theta^* = 25.56$ and $\rho \approx 0.26$. We can do even better, as follows. First generate $G^-(1)$ under IS with θ^*, and then generate $G^+(1)$ from its original gamma distribution but conditional on $G^+(1) > G^-(1) + \ln(K/S(0)) - r - \omega$ (a truncated gamma). This way the payoff is positive with probability 1 and the resulting IS estimator has an even smaller variation. We call the previous one IS-twist, this one IS-twist-cond, and the original estimator no-IS. We can use RQMC with any of these three estimators. We try a Sobol' net with one extra coordinate (i.e., the

Hammersley point set) with $n = 2^{16}$ randomized by LMS+DS, the same point set with NUS, and also a randomly shifted lattice rule with a baker's transformation, in two dimensions, with the same n. With this we find that the option price is 1.601×10^{-4} (the given digits here are significant). Table 4 reports the empirical variances for all the estimators discussed above, with both MC and RQMC. For RQMC, the variance of $\hat{\mu}_{\text{rqmc},n}$ is multiplied by n to obtain the *variance per run* (for a fair comparison with MC). We see that without IS, RQMC does not reduce the variance by much, and the standard deviation is more than 300 times the mean in all cases. The combination of IS and RQMC has a synergistic effect: IS first makes the function flatter, then RQMC can shine. The difference between the three RQMC methods here is not significant.

Table 4 Empirical variances per run for the European call option under a GVG process, with and without IS, with MC and RQMC

	MC	Sobol+LMS+DS	Sobol+NUS	Lattice+S+B
no-IS	1.8×10^{-3}	6.3×10^{-4}	6.0×10^{-4}	7.8×10^{-4}
IS-twist	1.0×10^{-7}	1.9×10^{-11}	1.3×10^{-11}	1.1×10^{-11}
IS-twist-cond	2.8×10^{-8}	7.1×10^{-13}	7.4×10^{-13}	7.4×10^{-13}

5 RQMC for Markov chains

When simulating a Markov chain over a large number of steps, if we use RQMC in a standard way, the dimension s will be large and RQMC is then likely to become ineffective. If each step requires d uniform random numbers and we simulate τ steps, then we have $s = \tau d$. The *Array-RQMC* method, which we now summarize, has been designed for this situation [31, 32, 37].

Consider a Markov chain with state space $\mathscr{X} \subseteq \mathbb{R}^\ell$, which evolves as

$$X_0 = x_0, \qquad X_j = \varphi_j(X_{j-1}, \mathbf{U}_j),\ j \geq 1,$$

where the $\mathbf{U}_j$ are i.i.d. uniform over $(0,1)^d$. Want to estimate

$$\mu = \mathbb{E}[Y] \quad \text{where} \quad Y = \sum_{j=1}^{\tau} g_j(X_j).$$

Ordinary MC or RQMC would produce n realizations of Y and take the average. Each realization requires $s = \tau d$ uniforms.

The idea of Array-RQMC is to simulate an *array* (or population) of n chains in parallel, in a way that at any given step j, there is small discrepancy between the empirical distribution of the n states $S_{n,j} = \{X_{0,j}, \ldots, X_{n-1,j}\}$ and the theoretical distribution of X_j. At each step, an RQMC point set is used to advance all the chains by one step.

To provide insight about the method, it is useful to assume for simplification that $X_j \sim U(0,1)^\ell$ for all j. This can be achieved conceptually by a change of variable, and is not necessary for implementing the method. We estimate

$$\mu_j = \mathbb{E}[g_j(X_j)] = \mathbb{E}[g_j(\varphi_j(X_{j-1}, \mathbf{U}))] = \int_{[0,1)^{\ell+d}} g_j(\varphi_j(\mathbf{x}, \mathbf{u}))d\mathbf{x}d\mathbf{u}$$

by

$$\hat{\mu}_{\text{arqmc},j,n} = \frac{1}{n}\sum_{i=0}^{n-1} g_j(X_{i,j}) = \frac{1}{n}\sum_{i=0}^{n-1} g_j(\varphi_j(X_{i,j-1}, \mathbf{U}_{i,j})).$$

This is (roughly) RQMC with the point set $Q_n = \{(X_{i,j-1}, \mathbf{U}_{i,j}), 0 \le i < n\}$. We want Q_n to have low discrepancy (LD) over $[0,1)^{\ell+d}$.

However, we do not choose the $X_{i,j-1}$'s in Q_n: they come from the simulation. We can select a low discrepancy point set $\tilde{Q}_n = \{(\mathbf{w}_0, \mathbf{U}_{0,j}), \ldots, (\mathbf{w}_{n-1}, \mathbf{U}_{n-1,j})\}$, in which the $\mathbf{w}_i \in [0,1)^\ell$ are fixed and each $\mathbf{U}_{i,j} \sim U(0,1)^d$. Then a key operation is to permute the states $X_{i,j-1}$ so that $X_{\pi_j(i),j-1}$ is "close" to $\mathbf{w}_i$ for each i (low discrepancy between the two sets), and compute $X_{i,j} = \varphi_j(X_{\pi_j(i),j-1}, \mathbf{U}_{i,j})$ for each i. In particular, if $\ell = 1$, we can take $\mathbf{w}_i = (i+0.5)/n$ and just sort the states in increasing order. For $\ell > 1$, there are various ways to define the matching (multivariate sorts). At the end, we return the average $\bar{Y}_n = \hat{\mu}_{\text{arqmc},n} = \sum_{j=1}^{\tau} \hat{\mu}_{\text{arqmc},j,n}$ as an estimator of μ.

The array-RQMC estimator satisfies [32]: (i) $\bar{Y}_n$ is an *unbiased* estimator of μ, and (ii) the *empirical variance* of m independent realizations of $\bar{Y}_n$ is an unbiased estimator of $\text{Var}[\bar{Y}_n]$. Known convergence rate results for special cases are summarized in [36]. For example, it is proved in [32] that for $\ell = 1$ and if the RQMC points form a stratified sample, the variance converges as $\mathcal{O}(n^{-3/2})$. In higher dimensions, it is show in [13] under some conditions that the worst-case error converges as $\mathcal{O}(n^{-1/(\ell+1)})$. In a sequential QMC setting (with particle filters) in which a digital net with NUS is used for RQMC and a Hilbert curve sort for mapping the states to the points, Gerber and Chopin [15] show that the variance is $o(n^{-1})$. Applications in finance and computer graphics can be found in [31, 66]. There are combinations with splitting techniques (multilevel and without levels), with importance sampling, and with weight windows (related to particle filters) [8, 30], combination with "coupling from the past" for exact sampling [39], and combination with approximate dynamic programming and for optimal stopping problems [11].

Examples in which the observed convergence rate for the empirical variance is $\mathcal{O}(n^{-2})$, or even $\mathcal{O}(n^{-3})$ in some cases, and does not depend on τ, can be found in [31, 37]. For example, when pricing an Asian call option under a geometric Brownian motion, we observe $\mathcal{O}(n^{-2})$ convergence for the variance regardless of the number τ of observation times that determine the payoff [31], both with lattice rules and Sobol' points, while with stratification the observed rate is more like $\mathcal{O}(n^{-3/2})$. For this example, the state is in $d = 2$ dimensions and the RQMC points are in 3 dimensions.

A different way of using QMC to simulate Markov chains is studied in [6, 65]. The main idea is to use an approximation of a completely uniformly distributed (CUD) sequence, implemented by taking successive overlapping vectors of output values

produced by a small linear random number generator as a source of "randomness" to simulate the chain (one vector per step).

Acknowledgements This work has been supported by a Canada Research Chair, an Inria International Chair, a NSERC Discovery Grant, to the author. It was presented at the MCQMC conference with the help of SAMSI funding. David Munger made Figures 10 and 9.

References

1. Asmussen, S., Glynn, P.W.: Stochastic Simulation. Springer-Verlag, New York (2007)
2. Avramidis, A.N., Wilson, J.R.: Correlation-induction techniques for estimating quantiles in simulation experiments. Operations Research **46**(4), 574–591 (1998)
3. Avramidis, T., L'Ecuyer, P.: Efficient Monte Carlo and quasi-Monte Carlo option pricing under the variance-gamma model. Management Science **52**(12), 1930–1944 (2006)
4. Avramidis, T., L'Ecuyer, P., Tremblay, P.A.: Efficient simulation of gamma and variance-gamma processes. In: Proceedings of the 2003 Winter Simulation Conference, pp. 319–326. IEEE Press, Piscataway, New Jersey (2003)
5. Caflisch, R.E., Morokoff, W., Owen, A.: Valuation of mortgage-backed securities using Brownian bridges to reduce effective dimension. J. of Computational Finance **1**(1), 27–46 (1997)
6. Chen, S., Dick, J., Owen, A.B.: Consistency of Markov chain quasi-Monte Carlo on continuous state spaces. The Annals of Statistics **39**(2), 673–701 (2011)
7. Cranley, R., Patterson, T.N.L.: Randomization of number theoretic methods for multiple integration. SIAM Journal on Numerical Analysis **13**(6), 904–914 (1976)
8. Demers, V., L'Ecuyer, P., Tuffin, B.: A combination of randomized quasi-Monte Carlo with splitting for rare-event simulation. In: Proceedings of the 2005 European Simulation and Modeling Conference, pp. 25–32. EUROSIS, Ghent, Belgium (2005)
9. Dick, J., Pillichshammer, F.: Digital Nets and Sequences: Discrepancy Theory and Quasi-Monte Carlo Integration. Cambridge University Press, Cambridge, U.K. (2010)
10. Dick, J., Sloan, I.H., Wang, X., Woźniakowski, H.: Good lattice rules in weighted Korobov spaces with general weights. Numerische Mathematik **103**, 63–97 (2006)
11. Dion, M., L'Ecuyer, P.: American option pricing with randomized quasi-Monte Carlo simulations. In: Proceedings of the 2010 Winter Simulation Conference, pp. 2705–2720 (2010)
12. Doerr, C., Gnewuch, M., Wahlström, M.: Calculation of discrepancy measures and applications. In: W. Chen, A. Srivastav, G. Travaglini (eds.) A Panorama of Discrepancy Theory, pp. 621–678. Springer (2014)
13. El Haddad, R., Lécot, C., L'Ecuyer, P., Nassif, N.: Quasi-Monte Carlo methods for Markov chains with continuous multidimensional state space. Mathematics and Computers in Simulation **81**, 560–567 (2010)
14. Faure, H.: Discrépance des suites associées à un système de numération en dimension *s*. Acta Arithmetica **61**, 337–351 (1982)
15. Gerber, M., Chopin, N.: Sequential quasi-Monte Carlo. Journal of the Royal Statistical Society, Series B **77**(Part 3), 509–579 (2015)
16. Haber, S.: A modified Monte Carlo quadrature. Mathematics of Computation **19**, 361–368 (1966)
17. Hickernell, F.J.: The mean square discrepancy of randomized nets. ACM Transactions on Modeling and Computer Simulation **6**(4), 274–296 (1996)
18. Hickernell, F.J.: A generalized discrepancy and quadrature error bound. Mathematics of Computation **67**(221), 299–322 (1998)
19. Hickernell, F.J.: What affects the accuracy of quasi-Monte Carlo quadrature? In: H. Niederreiter, J. Spanier (eds.) Monte Carlo and Quasi-Monte Carlo Methods 1998, pp. 16–55. Springer-Verlag, Berlin (2000)

20. Hickernell, F.J.: Obtaining $O(N^{-2+\varepsilon})$ convergence for lattice quadrature rules. In: K.T. Fang, F.J. Hickernell, H. Niederreiter (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2000, pp. 274–289. Springer-Verlag, Berlin (2002)
21. Hickernell, F.J.: Error analysis for quasi-Monte Carlo methods. In: P.W. Glynn, A.B. Owen (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2016 (2017)
22. Hickernell, F.J., Lemieux, C., Owen, A.B.: Control variates for quasi-Monte Carlo. Statistical Science **20**(1), 1–31 (2005)
23. Hong, H.S., Hickernell, F.H.: Algorithm 823: Implementing scrambled digital sequences. ACM Transactions on Mathematical Software **29**, 95–109 (2003)
24. Imai, J., Tan, K.S.: Enhanced quasi-Monte Carlo methods with dimension reduction. In: E. Yücesan, C.H. Chen, J.L. Snowdon, J.M. Charnes (eds.) Proceedings of the 2002 Winter Simulation Conference, pp. 1502–1510. IEEE Press, Piscataway, New Jersey (2002)
25. Imai, J., Tan, K.S.: A general dimension reduction technique for derivative pricing. Journal of Computational Finance **10**(2), 129–155 (2006)
26. Joe, S., Kuo, F.Y.: Constructing Sobol sequences with better two-dimensional projections. SIAM Journal on Scientific Computing **30**(5), 2635–2654 (2008)
27. L'Ecuyer, P.: Quasi-Monte Carlo methods in finance. In: Proceedings of the 2004 Winter Simulation Conference. IEEE Press, Piscataway, New Jersey (2004)
28. L'Ecuyer, P.: Quasi-Monte Carlo methods with applications in finance. Finance and Stochastics **13**(3), 307–349 (2009)
29. L'Ecuyer, P.: SSJ: Stochastic simulation in Java (2016). http://simul.iro.umontreal.ca/ssj/
30. L'Ecuyer, P., Demers, V., Tuffin, B.: Rare-events, splitting, and quasi-Monte Carlo. ACM Transactions on Modeling and Computer Simulation **17**(2), Article 9 (2007)
31. L'Ecuyer, P., Lécot, C., L'Archevêque-Gaudet, A.: On array-RQMC for Markov chains: Mapping alternatives and convergence rates. In: P. L'Ecuyer, A.B. Owen (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2008, pp. 485–500. Springer-Verlag, Berlin (2009)
32. L'Ecuyer, P., Lécot, C., Tuffin, B.: A randomized quasi-Monte Carlo simulation method for Markov chains. Operations Research **56**(4), 958–975 (2008)
33. L'Ecuyer, P., Lemieux, C.: Variance reduction via lattice rules. Management Science **46**(9), 1214–1235 (2000)
34. L'Ecuyer, P., Lemieux, C.: Recent advances in randomized quasi-Monte Carlo methods. In: M. Dror, P. L'Ecuyer, F. Szidarovszky (eds.) Modeling Uncertainty: An Examination of Stochastic Theory, Methods, and Applications, pp. 419–474. Kluwer Academic, Boston (2002)
35. L'Ecuyer, P., Munger, D.: On figures of merit for randomly-shifted lattice rules. In: H. Woźniakowski, L. Plaskota (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2010, pp. 133–159. Springer-Verlag, Berlin (2012)
36. L'Ecuyer, P., Munger, D.: Algorithm 958: Lattice builder: A general software tool for constructing rank-1 lattice rules. ACM Trans. on Mathematical Software **42**(2), Article 15 (2016)
37. L'Ecuyer, P., Munger, D., Lécot, C., Tuffin, B.: Sorting methods and convergence rates for array-rqmc: Some empirical comparisons. Mathematics and Computers in Simulation (2016). http://dx.doi.org/10.1016/j.matcom.2016.07.010
38. L'Ecuyer, P., Munger, D., Tuffin, B.: On the distribution of integration error by randomly-shifted lattice rules. Electronic Journal of Statistics **4**, 950–993 (2010)
39. L'Ecuyer, P., Sanvido, C.: Coupling from the past with randomized quasi-Monte Carlo. Mathematics and Computers in Simulation **81**(3), 476–489 (2010)
40. L'Ecuyer, P., Simard, R.: Inverting the symmetrical beta distribution. ACM Transactions on Mathematical Software **32**(4), 509–520 (2006)
41. Lemieux, C.: Monte Carlo and Quasi-Monte Carlo Sampling. Springer-Verlag (2009)
42. Lemieux, C., Cieslak, M., Luttmer, K.: RandQMC User's Guide: A Package for Randomized Quasi-Monte Carlo Methods in C (2004). Software user's guide, available at http://www.math.uwaterloo.ca/~clemieux/randqmc.html
43. Lemieux, C., L'Ecuyer, P.: Randomized polynomial lattice rules for multivariate integration and simulation. SIAM Journal on Scientific Computing **24**(5), 1768–1789 (2003)
44. Liu, R., Owen, A.B.: Estimating mean dimensionality of analysis of variance decompositions. Journal of the American Statistical Association **101**(474), 712–721 (2006)

45. Loh, W.L.: On the asymptotic distribution of scramble nets quadratures. Annals of Statistics **31**, 1282–1324 (2003)
46. Madan, D.B., Carr, P.P., Chang, E.C.: The variance gamma process and option pricing. European Finance Review **2**, 79–105 (1998)
47. Mara, T.A., Rakoto, J.O.: Comparison of some efficient methods to evaluate the main effect of computer model factors. J. of Statistical Computation and Simulation **78**(2), 167–178 (2008)
48. Matoušek, J.: On the L_2-discrepancy for anchored boxes. J. of Complexity **14**, 527–556 (1998)
49. Matoušek, J.: Geometric Discrepancy: An Illustrated Guide. Springer-Verlag, Berlin (1999)
50. Niederreiter, H.: Random Number Generation and Quasi-Monte Carlo Methods, *SIAM CBMS-NSF Reg. Conf. Series in Applied Mathematics*, vol. 63. SIAM (1992)
51. Niederreiter, H., Xing, C.: Nets, (t,s)-sequences, and algebraic geometry. In: P. Hellekalek, G. Larcher (eds.) Random and Quasi-Random Point Sets, *Lecture Notes in Statistics*, vol. 138, pp. 267–302. Springer, New York, NY (1998)
52. Nuyens, D.: The construction of good lattice rules and polynomial lattice rules. In: P. Kritzer, H. Niederreiter, F. Pillichshammer, A. Winterhof (eds.) Uniform Distribution and Quasi-Monte Carlo Methods: Discrepancy, Integration and Applications, pp. 223–255. De Gruyter (2014)
53. Owen, A.B.: Monte Carlo variance of scrambled equidistribution quadrature. SIAM Journal on Numerical Analysis **34**(5), 1884–1910 (1997)
54. Owen, A.B.: Scrambled net variance for integrals of smooth functions. Annals of Statistics **25**(4), 1541–1562 (1997)
55. Owen, A.B.: Latin supercube sampling for very high-dimensional simulations. ACM Transactions on Modeling and Computer Simulation **8**(1), 71–102 (1998)
56. Owen, A.B.: Variance with alternative scramblings of digital nets. ACM Transactions on Modeling and Computer Simulation **13**(4), 363–378 (2003)
57. Owen, A.B.: Better estimation of small sobol sensitivity indices. ACM Transactions on Modeling and Computer Simulation **23**(2), Article 11 (2013)
58. Sloan, I.H., Joe, S.: Lattice Methods for Multiple Integration. Clarendon Press, Oxford (1994)
59. Sloan, I.H., Woźniakowski, H.: When are quasi-Monte Carlo algorithms efficient for high-dimensional integrals. Journal of Complexity **14**, 1–33 (1998)
60. Sloan, I.H., Woźniakowski, H.: Tractability of multivariate integration for weighted Korobov classes. Journal of Complexity **17**(4), 697–721 (2001)
61. Sobol', I.M.: The distribution of points in a cube and the approximate evaluation of integrals. U.S.S.R. Comput. Math. and Math. Phys. **7**(4), 86–112 (1967)
62. Sobol', I.M.: Sensitivity indices for nonlinear mathematical models. Mathematical Modeling and Computational Experiment **1**, 407–414 (1993)
63. Sobol', I.M., Myshetskaya, E.E.: Monte Carlo estimators for small sensitivity indices. Monte Carlo Methods and Applications **13**(5–6), 455–465 (2007)
64. Tezuka, S.: Uniform Random Numbers: Theory and Practice. Kluwer Academic (1995)
65. Tribble, S.D., Owen, A.B.: Constructions of weakly CUD sequences for MCMC. Electronic Journal of Statistics **2**, 634–660 (2008)
66. Wächter, C., Keller, A.: Efficient simultaneous simulation of Markov chains. In: A. Keller, S. Heinrich, H. Niederreiter (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2006, pp. 669–684. Springer-Verlag, Berlin (2008)