Randomized Quasi-Monte Carlo Simulation of Markov Chains with an Ordered State Space

Pierre L'Ecuyer[1], Christian Lécot[2], and Bruno Tuffin[3]

[1] Département d'informatique et de recherche opérationnelle, Université de Montréal, C.P. 6128, Succ. Centre-Ville, Montréal (Québec), Canada, H3C 3J7, lecuyer@iro.umontreal.ca
[2] Laboratoire de Mathématiques, Université de Savoie, 73376 Le Bourget-du-Lac Cedex, France, Christian.Lecot@univ-savoie.fr
[3] IRISA-INRIA, Campus Universitaire de Beaulieu, 35042 Rennes Cedex, France, Bruno.Tuffin@irisa.fr

Summary. We study a randomized quasi-Monte Carlo method for estimating the state distribution at each step of a Markov chain with totally ordered (discrete or continuous) state space. The number of steps in the chain can be random and unbounded. The method can be used in particular to get a low-variance unbiased estimator of the expected total cost up to some random stopping time, when state-dependent costs are paid at each step. We provide numerical illustrations where the variance reduction with respect to standard Monte Carlo is substantial.

1 Introduction

A deterministic quasi-Monte Carlo (QMC) method for estimating transient measures over a fixed number of steps, for discrete-time and discrete-state Markov chains with a totally ordered state space, was proposed and studied in [3], based on ideas of [2]. The method simulates $n = 2^k$ copies of the chain in parallel (for the same number of steps) using a $(0, 2)$-sequence in base 2. At step j of the chain, it reorders the n copies according to their states and simulates the transitions (next states) for the n copies by employing the elements nj to $nj + n - 1$ of the $(0, 2)$-sequence in place of uniform random numbers to drive the simulation. It assumes that simulating each transition of the chain requires a single uniform random variate. Convergence to the correct value was proved in [3] under a condition on the structure of the transition probability matrix of the Markov chain.

In this paper, we generalize this method to Markov chains with continuous state space, with a random and unbounded number τ of steps (this permits one to cover regenerative simulation, in particular), and for which the number d of uniform random variates that are required to generate the next state in

one step of the Markov chain can be larger than 1. The method uses randomized versions of a *single* highly-uniform (or low-discrepancy) point set of cardinality n in the d-dimensional unit cube. It provides unbiased mean and variance estimators. We have theoretical results on the rate of convergence of the variance of the mean estimator (as $n \to \infty$) only for narrow special cases, but our empirical results with a variety of examples indicate that this variance goes down much faster with the proposed method than for standard Monte Carlo (MC) simulation or for randomized QMC (RQMC) methods that use a single τd-dimensional point to simulate each sample path of the chain (as in [5, 6, 7], for example).

In the next section, we first define our Markov chain model, then we motivate and state our RQMC sampling algorithm. Section 3 contains convergence results for special settings. Section 4 illustrates the method via numerical examples where it improves the simulation efficiency (by reducing the variance) by large factors compared with standard MC.

2 The Array-RQMC Algorithm

2.1 Markov Chain Model

We consider a Markov chain that evolves over a state space $\mathcal{X}$, according to the stochastic recurrence:

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

where the $\mathbf{U}_j$ are i.i.d. random vectors uniformly distributed over the d-dimensional unit hypercube $[0,1)^d$.

We want to estimate

$$\mu = E[Y] \quad \text{where} \quad Y = \sum_{j=1}^{\tau} c_j(X_j), \tag{2}$$

each $c_j : \mathcal{X} \to \mathbb{R}$ is a *cost function*, τ is a stopping time with respect to the filtration generated by $\{(j, X_j),\ j \geq 0\}$, and we assume implicitly that $\mathcal{X}$, the φ_j's, and the c_j's satisfy appropriate measure-theoretic requirements so that all objects of interest in this paper are well-defined (so we hide the uninteresting technical details). We also assume that the functions φ_j and c_j are easy to evaluate at any given point, for each j.

The random variable Y is easy to generate by standard MC: For $j = 1, \ldots, \tau$, generate $\mathbf{U}_j \sim U[0,1)^d$, compute $X_j = \varphi_j(X_{j-1}, \mathbf{U}_j)$, and add the value of $c_j(X_j)$ to an accumulator, which at the end will contain the value of Y. This can be replicated n times independently, and the sample mean and variance of the n values of Y are unbiased estimators of the exact mean and variance of Y. From this, one can compute a confidence interval on μ.

Let $s = \sup_\omega d\tau$, where the supremum is taken over all possible sample paths ω, and $s = \infty$ if τ is unbounded. In this setting, the random variable Y can be written as a function of a sequence of s i.i.d. $U(0,1)$ random variables, say $Y = f(U_1, \ldots, U_s)$, for some complicated function f. If τ is unbounded, we assume that it at least has finite expectation.

One way of estimating μ by RQMC is to select an s-dimensional RQMC point set of cardinality n, say $\mathbf{V}_i = (U_{i,1}, \ldots, U_{i,s})$ for $i = 1, \ldots, n$, compute the average value of f over these n points, say $\bar{Y}_n$, and take it as an estimator of μ. To estimate the variance and compute a confidence interval on μ this procedure can be replicated m times, with independent randomizations of the same QMC point set. Under simple conditions on the randomization (e.g., one must have $E[f(\mathbf{V}_i)] = \mu$), the sample mean and sample variance of these m averages are unbiased estimators of the exact mean and variance of $\bar{Y}_n$. Further details on this *classical RQMC* approach can be found in [5, 6, 8] and other references given there.

Owen [8] discusses several ways of handling the case where s is large, perhaps infinite (he gives examples of situations when this happens). He proposes an RQMC variant called *Latin Supercube Sampling* (LSS), where the s coordinates are partitioned into finite subsets of sizes (say) $s_1, s_2, \ldots$, and an s_j-dimensional QMC point set $P_{n,j}$ is used for each subset j, but with the n points randomly permuted, independently across the subsets. Thus, each chain uses a random point from $P_{n,j}$ for each subset j of coordinates. If all s_j's are equal, all $P_{n,j}$'s can be the same point set. Our method bears ressemblance with LSS with $s_j = d$ for all j, but the points are assigned to the chains in a systematic manner (by sorting the chains according to their states, as described in a moment) rather than by a random permutation.

2.2 Array-RQMC for Comparable Markov Chains

We now assume (for the remainder of the paper) that $\mathcal{X} \subseteq \mathbb{R}^\ell \cup \{\infty\}$, and that a *total order* has been defined on $\mathcal{X}$, for which ∞ is the largest state. The state ∞ is an *absorbing state* used to indicate that we have reached the stopping time τ. That is, $X_j = \infty$ for $j > \tau$, and $c_j(\infty) = 0$.

The basic idea of the proposed method is to seek a good estimate of the distribution function F_j of the random variable X_j for each j. For that, we simulate n copies of the chain in parallel and estimate F_j by the empirical distribution of n values of X_j thus obtained. In contrast with classical integration or approximation methods, the states at which F_j is to be "evaluated" need not be selected in advance but are generated automatically by the RQMC algorithm according to a distribution that approximates F_j.

The *array-RQMC* algorithm works as follows. At *step 1*, we take an RQMC point set $P_n = \{\mathbf{u}_0, \ldots, \mathbf{u}_{n-1}\}$ in $[0,1)^d$, define

$$X_{i,1} = \varphi_1(x_0, \mathbf{u}_i) \qquad \text{for } i = 0, \ldots, n-1,$$

and estimate the distribution F_1 of X_1 by the empirical distribution $\hat{F}_1$ of $X_{0,1}, \ldots, X_{n-1,1}$. This gives the following approximation, where I denotes the indicator function:

$$\begin{aligned} F_1(x) &= P[X_1 \le x] \\ &= \int_{[0,1)^d} I(\varphi_1(x_0, \mathbf{u}) \le x)\, d\mathbf{u} \qquad (3) \\ &\approx \frac{1}{n} \sum_{i=0}^{n-1} I(\varphi_1(x_0, \mathbf{u}_i) \le x) \qquad (4) \\ &= \frac{1}{n} \sum_{i=0}^{n-1} I(X_{i,1} \le x) \stackrel{\text{def}}{=} \hat{F}_1(x), \end{aligned}$$

which amounts to estimating the integral (3) by RQMC in (4).

At *step* j, we use the empirical distribution $\hat{F}_{j-1}$ of $X_{0,j-1}, \ldots, X_{n-1,j-1}$ as an approximation of the distribution F_{j-1} of X_{j-1}. Let $P_n = \{\mathbf{u}_0, \ldots, \mathbf{u}_{n-1}\}$ be an RQMC point set in $[0,1)^d$ such that the $(d+1)$-dimensional point set $P'_n = \{\mathbf{u}'_i = (i/n, \mathbf{u}_i),\ 0 \le i < n\}$ is "highly uniform" (or has "low discrepancy") in $[0,1)^{d+1}$, in a sense that we leave open for the moment (a precise definition of "low discrepancy" in the asymptotic sense, as $n \to \infty$, will be adopted in the propositions of Section 3). We estimate F_j by the empirical distribution $\hat{F}_j$ of the values $X_{i,j} = \varphi_j(X_{(i),j-1}, \mathbf{u}_i)$, $i = 0, \ldots, n-1$. This can be interpreted as follows:

$$\begin{aligned} F_j(x) = P[X_j \le x] &= E[I(\varphi_j(X_{j-1}, \mathbf{U}_j) \le x)] \\ &= \int_{\mathcal{X}} \int_{[0,1)^d} I(\varphi_j(y, \mathbf{u}) \le x)\, d\mathbf{u}\, dF_{j-1}(y) \qquad (5) \\ &\approx \int_{\mathcal{X}} \int_{[0,1)^d} I(\varphi_j(y, \mathbf{u}) \le x)\, d\mathbf{u}\, d\hat{F}_{j-1}(y) \qquad (6) \\ &= \int_{[0,1)^{d+1}} I(\varphi_j(\hat{F}_{j-1}^{-1}(v), \mathbf{u}) \le x)\, d\mathbf{u}\, dv \qquad (7) \\ &\approx \frac{1}{n} \sum_{i=0}^{n-1} I(\varphi_j(\hat{F}_{j-1}^{-1}(i/n), \mathbf{u}_i) \le x) \qquad (8) \\ &= \frac{1}{n} \sum_{i=0}^{n-1} I(\varphi_j(X_{(i),j-1}, \mathbf{u}_i) \le x) \\ &= \frac{1}{n} \sum_{i=0}^{n-1} I(X_{i,j} \le x) \stackrel{\text{def}}{=} \hat{F}_j(x). \end{aligned}$$

In (6), we have replaced F_{j-1} in (5), by its approximation $\hat{F}_{j-1}$. In (8), we approximate the integral in (7) by RQMC over $[0,1)^{d+1}$ with the point set P'_n. Observe that this point set gives a perfect stratification of the distribution

$\hat{F}_{j-1}$, with exactly one observation per stratum (the strata are the jumps of $\hat{F}_{j-1}$).

Putting these pieces together, we get the following algorithm (the "for" loops are written using the C/C++/Java syntax and indentation alone indicates the scope of the loops):

Array-RQMC algorithm:

1 (Initialization). Select a d-dimensional QMC point set $\tilde{P}_n = (\tilde{\mathbf{u}}_0, \ldots, \tilde{\mathbf{u}}_{n-1})$ and a randomization of $\tilde{P}_n$ such that (a) each randomized point is uniform over $[0,1)^d$ and (b) if $P_n = (\mathbf{u}_0, \ldots, \mathbf{u}_{n-1})$ denotes the randomized version, then $P'_n = \{(i/n, \mathbf{u}_i),\ 0 \le i < n\}$ has "low discrepancy".

2 (Simulate chains). Simulate in parallel n copies of the chain, numbered $0, \ldots, n-1$, as follows:

```
For (j = 1; X_{0,j-1} < ∞; j++)
    Randomize P̃_n afresh into P_n = {u_0, ..., u_{n-1}};
    For (i = 0; i < n and X_{i,j-1} < ∞; i++)
        X_{i,j} = φ_j(X_{i,j-1}, u_i);
    Sort (and renumber) the chains for which X_{i,j} < ∞ by
        increasing order of their states;
        (The sorted states X_{0,j}, ..., X_{n-1,j} provide F̂_j).
```

3 (Output). Return the average $\bar{Y}_n$ of the n values of Y as an estimator of μ.

This entire procedure is replicated m times to estimate the variance and compute a confidence interval on μ.

3 Unbiasedness and Convergence

Proposition 1. *(a) The average* $\bar{Y}_n$ *is an* unbiased *estimator of* μ *and (b) the* empirical variance *of its* m *copies is an unbiased estimator of* $\mathrm{var}[\bar{Y}_n]$.

Proof. The successive steps of the chain use independent randomizations. Therefore, for each chain, the vectors that take place of the $\mathbf{U}_j$'s for the successive steps j of the chain in the recurrence (1) are independent random variables uniformly distributed over $[0,1)^d$. Thus, any given copy of the chain obeys the correct probabilistic model defined by (1) and (2), so the value of Y is an unbiased estimator of μ for each chain and also for the average, which proves (a). For (b), it suffices to observe that the m copies of $\bar{Y}_n$ are i.i.d. unbiased estimators of μ. □

Of course, this proposition implies that the variance of the overall average converges as $O(1/m)$ when $m \to \infty$. A more interesting question is: What about the convergence when $n \to \infty$?

The integrand $I(\varphi_j(\hat{F}_{j-1}^{-1}(v), \mathbf{u}) \le x)$ in (7) is 1 in part of the unit cube, and 0 elsewhere. The shape and complexity of the boundary between these two regions depends on $\varphi_1, \ldots, \varphi_j$. We assume that these regions are at least measurable sets. For continuous state spaces $\mathcal{X}$, the Hardy-Krause total variation of this indicator function is likely to be infinite, in which case the classical Koksma-Hlawka inequality will not be helpful to bound the integration error in (8). On the other hand, we have *proved* bounds on the convergence rate for the two following (narrow) special cases: (1) when the chain has a finite number of states (Proposition 2), and (2) when $\ell = d = 1$ and the φ_j's satisfy a number of conditions (Proposition 3). Detailed proofs of these (and other) propositions will be given in the expanded version of the paper.

Proposition 2. *Suppose that the state space $\mathcal{X}$ is finite, say $\mathcal{X} = \{1, \ldots, L\}$, that the Markov chain is stochastically increasing (i.e., $P[X_j \ge x \mid X_{j-1} = y]$ is non-decreasing in y for each j), and that at each step j, we use inversion from a single uniform to generate the next state X_j from its conditional distribution given X_{j-1} (so $d = 1$). Let $\Gamma_j = \sum_{\ell=1}^{L-1} |c_j(\ell+1) - c_j(\ell)|$, $P'_n = \{(i/n, \mathbf{u}_i),\ 0 \le i < n\}$, and suppose that the star discrepancy of P'_n satisfies $D_n^*(P'_n) = O(n^{-1} \log n)$ w.p.1 (this can easily be achieved by taking a $(0, 2)$-sequence in some base b). Then,*

$$\left| \frac{1}{n} \sum_{i=0}^{n-1} c_j(X_{i,j}) - E[c_j(X_j)] \right| \le j \Gamma_j K L n^{-1} \log n$$

for some constant K. This implies that the variance of the cost estimator for step j converges as $O((jLn^{-1} \log n)^2) = O((jL)^2 n^{-2+\epsilon})$ when $n \to \infty$.

Proposition 3. *Let $\ell = d = 1$. Define*

$$D_n(\hat{F}_j, F_j; x) := \frac{1}{n} \sum_{0 \le i < n} I(X_{i,j} \le x) - F_j(x)$$

and

$$D^*(\hat{F}_j, F_j) := \sup_{x \in \mathcal{X}} |D_n(\hat{F}_j, F_j; x)|,$$

where $I(A)$ denotes the indicator function of A. Suppose that $n = b^k$, that P'_n is a $(0, k, 2)$-net in base b, that

$$\forall j \ge 1\ \forall x \in \mathcal{X}\ \forall u \in [0, 1) \quad V(I_x \circ \varphi_j(\cdot, u)) \le 1,$$

and that there exists a sequence m_j of integers such that

$$\forall j \ge 1\ \forall x \in \mathcal{X}\ \forall y \in \mathcal{X} \quad V(I_x \circ \varphi_j(y, \cdot)) \le m_j,$$

where $V(\cdot)$ means the total variation and I_x denotes the indicator function of $A_x := \{y \in \mathcal{X} : y \le x\}$. Then,

$$D^*(\hat{F}_j, F_j) \le M_j b^{-\lfloor k/2 \rfloor} = O(jn^{-1/2})$$

where $M_j := \sum_{h=1}^{j}(m_h + 1)$. *Moreover, if* c_j *has bounded variation,*

$$\left| \frac{1}{n} \sum_{i=0}^{n-1} c_j(X_{i,j}) - E[c_j(X_j)] \right| \le V(c_j) D^*(\hat{F}_j, F_j).$$

This proposition can be proved by generalizing the arguments of [3]. It only gives a bound of $O(1/n)$ on the variance, just as for ordinary MC, but here this bound is also a *worst-case deterministic* bound on the squared error. Moreover, the variance seems to converge at a faster rate than $O(1/n)$ (empirically) in the examples that we tried, as illustrated in the next section.

4 A Numerical Illustration

4.1 An $M/M/1$ Queue with $d = 1$

Consider an $M/M/1$ queue with arrival rate $\lambda = 1$, i.e., a single-server queue with i.i.d. exponential interarrival times with mean 1 and i.i.d. exponential service times with mean $1/\mu < 1$. The utilization factor for the server is $\rho = 1/\mu$. We want to estimate the expected average waiting time of the first t customers, denoted μ_t. (This μ_t could be computed numerically without simulation; we just use this simple academic example to illustrate our method.)

Let W_j denote the waiting time of customer number j in this system, where the first customer (who arrives to the empty system) has number 0. These W_j's satisfy the well-known *Lindley's recurrence*: $W_0 = 0$ and $W_j = \max(0, W_{j-1} + S_{j-1} - A_j)$ for $j \ge 1$. We estimate μ_t by the sample average $Y = (W_0 + \cdots + W_{t-1})/t$. To compute Y, we need to generate the $2(t-1)$ random variates $S_0, A_1, \ldots, S_{t-1}, A_t$. This estimator Y (and the corresponding f in Section 2.1) is unbounded, but it has bounded variance.

We first consider a Markov chain that moves by one step each time one of these random variates is generated. That is, $X_0 = W_0$, $X_1 = W_0 + S_0$, $X_2 = W_1$, $X_3 = W_1 + S_1$, and so on. In this case, $d = 1$ and $s = 2(t-1)$. Then, we also consider the case where the chain moves by one step every $d/2$ customers (where d is even), so $X_j = W_{jd/2}$ and $s = (t-1)/d$.

We tried both *classical RQMC* and *array-RQMC* for this example, with $t = 100$. Table 1 gives the *estimated* variance reduction factors compared with standard MC, i.e., the empirical variance per observation for the MC estimator divided by that of the method considered, for some RQMC methods. For the RQMC methods, the sample variance of the averages $\bar{Y}_n$ is multiplied by n to get the variance *per observation*, i.e., per simulated copy of the Markov chain. All values are rounded to the closest integer.

If the required CPU time to simulate the mn copies of the chain with RQMC is approximately α times that required to simulate mn independent

Table 1. Empirical variance reduction factors of RQMC with respect to MC, for the average waiting time of 100 customers, estimated with $m = 100$.

Korobov, $n =$	1021	4093	16381	65521	262139
Sobol, $n =$	1024	4096	16384	65536	262144
Classical-Korobov, $\rho = 0.2$	3	4	8	10	22
Classical-Sobol, $\rho = 0.2$	1	1	2	1	15
Array-Korobov, $\rho = 0.2$	52	125	336	826	2991
Array-Sobol, $\rho = 0.2$	53	303	748	3247	7964
Classical-Korobov, $\rho = 0.5$	3	4	7	8	9
Classical-Sobol, $\rho = 0.5$	1	1	4	5	7
Array-Korobov, $\rho = 0.5$	39	133	442	810	2464
Array-Sobol, $\rho = 0.5$	72	259	1340	3642	12460
Classical-Korobov, $\rho = 0.8$	4	2	8	10	11
Classical-Sobol, $\rho = 0.8$	2	2	5	10	10
Array-Korobov, $\rho = 0.8$	80	322	1064	1329	3674
Array-Sobol, $\rho = 0.8$	208	563	2333	11860	61290

copies, and if the variance reduction factor is γ, then we can say that MC requires γ/α times more CPU time to achieve a given precision for the estimator of μ. For this example, we have $\alpha \approx 2$ for array-RQMC (due to the overhead of maintaining several copies of the chain in parallel and sorting them) and $\alpha \approx 1$ for classical RQMC.

For array-RQMC, we tried several possibilities for the d-dimensional RQMC point set P_n. Here we give the results for (a) a $(d+1)$-dimensional Korobov lattice rule with its first coordinate skipped, randomized by a random shift modulo 1 followed by a baker's transformation [1] (denoted *Array-Korobov*) and (b) the first n points of a Sobol sequence randomized by a left (upper triangular) matrix scrambling followed by a random digital shift [6, 9] (denoted *Array-Sobol*). For Array-Korobov, the multiplier a of the two-dimensional Korobov lattice rule was selected so that n is prime, a is a primitive element modulo n (this requirement is actually not needed), and a/n is close to the golden ratio. With this choice of a, the rule performs quite well in the two-dimensional spectral test. The points of the Korobov lattice are enumerated by order of the (skipped) first coordinate, so P'_n becomes the original Korobov lattice point set. For instance, if $d = 1$, the points of P_n before the shift are $0, a, 2a, \ldots, (n-1)a$ in that order. The points of the Sobol net are enumerated by order of their Gray code, which makes their enumeration faster. This is effectively equivalent to applying a permutation to their second coordinate. We repeated the same experiment, but enumerating the points in their natural order (without using the Gray code) and for $d = 1$ the resulting variance reduction was generally smaller! Thus, using the Gray code implements some form of scrambling that seems beneficial. Further investigation would be needed to understand exactly why.

For the classical RQMC, we used a randomly shifted $2(t-1)$-dimensional Korobov lattice rules using parameters taken from Table 1 of [5] (where n is prime and close to a power of 2). This is denoted by *Classical-Korobov* in the table. We also tried $2(t-1)$-dimensional Sobol nets with a left matrix scrambling and a random digital shift (denoted *Classical-Sobol*).

All the variance reduction factors were estimated by making 100×2^{18} independent simulation runs to estimate the MC variance and $m = 100$ independents replicates for each n, for each RQMC method. The simulations were performed on a standard laptop computer with 1.7 GHz processor, in Java, using the SSJ simulation library [4].

The array-RQMC methods clearly outperform both MC and classical RQMC in this example, even though classical RQMC is already significantly more efficient than MC (about 10 times more efficient for the Korobov rules with $n \geq 2^{16}$). The improvement factor is larger when the queue has more traffic (i.e., for larger ρ, which is also when the variance is larger) and larger for the Sobol nets than for the Korobov rules. For classical RQMC, it is slightly better for the Korobov rules than for the Sobol nets.

4.2 Increasing d

Table 2 gives estimated variance reduction factors of two RQMC methods compared with standard MC, when d increases, with $n \approx 2^{17}$. Here, at each step of the Markov chain, we generate d random variates to compute the waiting times of $d/2$ customers. Note that for "Classical-Korobov," the exact variance reduction factor does not depend on d; the variation observed in the table is only statistical noise. It gives an idea of the precision of our variance-improvement estimators. For Array-Sobol, the variance reduction factors decrease with d, but not so fast. The gains are still substantial even for $d = 8$, where the RQMC method approximates 9-dimensional integrals at each step of the Markov chain.

Table 2. Estimated variance reduction factors of d-dimensional classical RQMC and array-RQMC with respect to MC, for selected values of d and $n \approx 2^{17}$.

	$d=1$	$d=2$	$d=4$	$d=8$
Classical-Korobov, $\rho = 0.2$	34	30	25	24
Array-Sobol, $\rho = 0.2$	6300	3300	1700	1300
Classical-Korobov, $\rho = 0.5$	11	12	21	18
Array-Sobol, $\rho = 0.5$	7000	7000	6600	2900
Classical-Korobov, $\rho = 0.8$	11	10	12	15
Array-Sobol, $\rho = 0.8$	24000	8200	13000	10000

4.3 Random dimension: a Regenerative System

So far in this example, s was fixed to $2(t-1)$. We now modify the example so that $s = \infty$. Recall that the $M/M/1$ queue is a *regenerative* system that regenerates whenever a customer arrives to an empty system. Each regenerative cycle contains a random and unbounded number of customers. Suppose we want to estimate $\mu = E[Y]$, where we take the following two possibilities for Y: (i) the total waiting time of all customers in a regenerative cycle and (ii) the number of customers in a cycle whose waiting time exceeds c, for some constant $c > 0$. Note that changing the uniforms slightly may split or merge regenerative cycles, making Y highly discontinuous in both cases. Moreover, in the second case, Y is integer-valued, so it is not as smooth as in the first case. For our numerical illustration of case (ii), we take $c = 1$. The exact value of μ for case (i) is 1 for $\rho = 0.5$ and 16 for $\rho = 0.8$. For case (ii), it is approximately 0.368 for $\rho = 0.5$ and 3.116 for $\rho = 0.8$.

Tables 3 and 4 give the estimated variance reduction factors of array-RQMC compared with standard MC, again with $m = 100$. The improvement factors are not as large as in the two previous tables, but they are still significant and also increase with n.

Table 3. Estimated variance reduction factors of array-RQMC with respect to MC, for the regenerative example, case (i).

Korobov, $n =$	1021	4093	16381	65521	262139
Sobol, $n =$	1024	4096	16384	65536	262144
Array-Korobov, $\rho = 0.5$	4	17	47	80	174
Array-Sobol, $\rho = 0.5$	8	13	30	70	174
Array-Korobov, $\rho = 0.8$	7	9	25	36	115
Array-Sobol, $\rho = 0.8$	5	5	16	34	87

Table 4. Estimated variance reduction factors of array-RQMC with respect to MC, for the regenerative example, case (ii).

Korobov, $n =$	1021	4093	16381	65521	262139
Sobol, $n =$	1024	4096	16384	65536	262144
Array-Korobov, $\rho = 0.5$	26	62	134	281	627
Array-Sobol, $\rho = 0.5$	14	23	77	172	659
Array-Korobov, $\rho = 0.8$	12	45	109	86	415
Array-Sobol, $\rho = 0.8$	9	32	74	177	546

4.4 Summary of Other Numerical Experiments

We have performed numerical experiments with various other examples. They will be reported in the detailed version of the paper. In particular, we tried examples with multidimensional state spaces and others with integrands of high variability. Generally speaking, as expected, we observed empirically that the performance of the array-RQMC method tends to degrade when the integrand has higher variability, or when the dimension of the state space becomes larger than 1 and there is no obvious "natural order" for the states. But even in these cases, there can still be significant gains in efficiency compared with MC and classical RQMC.

For example, the payoff of an Asian option can be simulated by a Markov chain with state $X_j = (S_j, \bar{S}_j)$ where S_j is the underlying asset price at observation time j and $\bar{S}_j$ is the average of $S_1, \ldots, S_j$. The final payoff is a function of $\bar{S}_s$ only, where s is the number of observation times. One possible way of ordering the states (which is not necessarily the best way) is simply by their values of S_j. With this order, in a numerical example where the asset price evolves as a geometric Brownian motion and the number of observation times varies from 10 to 120, we observed empirical variance reduction factors (roughly) from 1500 to 40000 for array-RQMC compared with MC. With classical RQMC, the factors were (roughly) 5 to 10 times smaller.

Our empirical results suggest better convergence rates than those implied by the (worst-case) bounds that we have managed to prove. Getting better convergence bounds for the variance is a topic that certainly deserves further investigation. From the practical viewpoint, an interesting challenge would be to find good ways of ordering the states for specific classes of problems where the Markov chain has a multidimensional state space. In the future, we also intend to study the application of array-RQMC to other settings that fit a general Markov chain framework. For instance, we think of Markov chain Monte Carlo methods and stochastic approximation algorithms.

5 Acknowledgments

The work of the first author has been supported by NSERC-Canada grant No. ODGP0110050, NATEQ-Québec grant No. 02ER3218, and a Canada Research Chair. The work of the third author has been supported by EuroNGI Network of Excellence and SurePath ACI sécurité project. The paper benefited from the comments of an anonymous reviewer.

References

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

2. C. Lécot and S. Ogawa. Quasirandom walks methods. In K.-T. Fang, F. J. Hickernell, and H. Niederreiter, editors, *Monte Carlo and Quasi-Monte Carlo Methods 2000*, pages 63–85, Berlin, 2002. Springer-Verlag.
3. C. Lécot and B. Tuffin. Quasi-Monte Carlo methods for estimating transient measures of discrete time Markov chains. In H. Niederreiter, editor, *Monte Carlo and Quasi-Monte Carlo Methods 2002*, pages 329–343, Berlin, 2004. Springer-Verlag.
4. P. L'Ecuyer. *SSJ: A Java Library for Stochastic Simulation*, 2004. Software user's guide, Available at `http://www.iro.umontreal.ca/~lecuyer`.
5. P. L'Ecuyer and C. Lemieux. Variance reduction via lattice rules. *Management Science*, 46(9):1214–1235, 2000.
6. P. L'Ecuyer and C. Lemieux. Recent advances in randomized quasi-Monte Carlo methods. In M. Dror, P. L'Ecuyer, and F. Szidarovszky, editors, *Modeling Uncertainty: An Examination of Stochastic Theory, Methods, and Applications*, pages 419–474. Kluwer Academic Publishers, Boston, 2002.
7. C. Lemieux and P. L'Ecuyer. A comparison of Monte Carlo, lattice rules and other low-discrepancy point sets. In H. Niederreiter and J. Spanier, editors, *Monte Carlo and Quasi-Monte Carlo Methods 1998*, pages 326–340, Berlin, 2000. Springer-Verlag.
8. A. B. Owen. Latin supercube sampling for very high-dimensional simulations. *ACM Transactions on Modeling and Computer Simulation*, 8(1):71–102, 1998.
9. A. B. Owen. Variance with alternative scramblings of digital nets. *ACM Transactions on Modeling and Computer Simulation*, 13(4):363–378, 2003.