MARKOV CHAIN MONTE CARLO ALGORITHMS USING COMPLETELY UNIFORMLY DISTRIBUTED DRIVING SEQUENCES

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF STATISTICS
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Seth D. Tribble

June 2007

Abstract

The advantage of low-discrepancy sequences in lieu of random numbers for simple independent Monte Carlo sampling is well-known. This procedure, known as quasi-Monte Carlo (QMC), yields an integration error that decays at a superior rate to that obtained by IID sampling, by the Koksma-Hlawka inequality. For the class of Markov chain Monte Carlo (MCMC) samplers, little literature has been produced examining the use of low-discrepancy sequences, and previous experiments have offered no theoretical validation for this practice. The central result in this work is the establishment of conditions under which low-discrepancy sequences can be used for consistent MCMC estimation. This condition of completely uniform distribution (CUD) applies to a series of sequences that look like full outputs of a small random number generator. A strategy for the incorporation of these sequences into a general MCMC sampling scheme is thoroughly developed here, with attention to the preservation of the CUD condition. The use of these sequences in a few MCMC examples shows reductions in estimate error that are most significant in Gibbs samplers. From these examples, the empirical benefits of CUD sequences in MCMC sampling are immense, although no analog of the Koksma-Hlawka inequality has been produced for MCMC to provide a general theoretical corroboration of these improvements.

Acknowledgments

I would like to thank:

- Art Owen for his excellent mentorship and inspiration in bringing this material forward
- Trevor Hastie, Simon Jackman, Wing Wong and Guenther Walther for providing a wide range of perspective on the work done and its broader potential
- David Vansuch, Emily Tribble, Gail Tribble and David Tribble for their unwavering support
- The Department of Defense
- The National Science Foundation

This work has been supported by the National Defense Science and Engineering Graduate Fellowship and by National Science Foundation grants DMS-0604939 and DMS-0306612.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The practice of Markov chain Monte Carlo is one whose theoretical validation has existed since the seminal paper of Metropolis et al [29] in 1953 (and in greater generality from Hastings [12] in 1970). At that time, the need was present to simulate sample values from a distribution that could not be sampled directly. As computing power grows and complex data structures become more manageable, Markov chain Monte Carlo is likely to become a more familiar tool to those analyzing these data. The general Metropolis-Hastings algorithm (of which Gibbs sampling is a special case) is almost synonymous with Markov chain Monte Carlo, and its traditional justification relies on the assumption of the use independent uniform variables. The assumption that we can create truly independent and uniformly distributed variables is sustainable for problems of ample size, thanks to advances in random number generation.

The research on Markov chain Monte Carlo and the research on quasi-Monte Carlo, in which independent uniform sampling is replaced by the use of a point sequence chosen to approximate the uniform distribution more closely, have been almost exclusively separate. It is easy to believe that the delicate framework that guarantees the consistency of the Metropolis-Hastings algorithm would be difficult to maintain through a sequence of chosen points, because the samples are now dependent, and, to a diminishing degree as one looks further in the past, the current state of the Markov chain is influenced by the variates used in any previous step.

Most of the sparse previous efforts to put QMC points in a Markov chain sampler or a sequential Monte Carlo method (such as particle filters or Brownian path sampling) do so in a way that the statistical dependence of successive points in a QMC sequence is not carried over to statistical dependence among successive values used to simulate the chain (see [35] and [34]). Liao [23] runs a Gibbs sampler using a QMC point to drive each step, but the order of the points is randomized. Chaudary [5] uses QMC to weight the sample points from a Metropolis-Hastings algorithm by a neighboring point, but the underlying chain is still simulated by IID sampling. Some important aspects of the work here draw inspiration from recent efforts of Niederreiter [32], who first proposed the full output of a small random number generator as a QMC rule, and of Lemieux and L'Écuyer [21], who use such a sequence on the simulation of an infinite-dimensional process. Markov chain Monte Carlo is equivalent to infinite-dimensional sampling using an overlapping sequence of variates.

The central goal of this work is the development of a theory that justifies the use of these full-period outputs of small random number generators in MCMC. There is some intuition that this strategy may work: because the marginal dependence on the past of a sample point in an MCMC algorithm decays, a nice distribution among the values close together in the driving sequence of the chain is most essential. This "nice distribution" is the goal of a good random number generator.

This work begins with a cursory background on Monte Carlo, quasi-Monte Carlo and Markov chain Monte Carlo in Chapter 2, with particular focus on the aspects of the process suited to the goals above. Chapters 3 and 4 establish a necessary and sufficient consistency condition on the use of non-IID point sets in an arbitrary Metropolis-Hastings sampler; this condition is quite restrictive, but fortunately the method of using small random number generators satisfies the condition for the classes of generators examined. For a single infinite sequence, this condition is called a completely uniform distribution (CUD), in which the blocks of s consecutive values form a sequence whose empirical distribution approaches the uniform distribution on the s-dimensional hypercube for ALL dimensions s. The proof that this yields consistent estimates is from Owen & Tribble [37], and it generalizes work of Chentsov [6] on a simpler Markov chain construction. Chapter 4 begins to develop a strategy for using the full RNG sequences discussed above. In Chapter 5, this strategy culminates in a closed form for the variate sequence recommended for use in driving the MCMC algorithm.

All experiments and results appear in Chapter 6. A reader who is not so familiar with Markov chain Monte Carlo or the central results of [37] is encouraged to refer to these examples to facilitate understanding of the method and its motivations.

What is not present in this work is a rate of convergence for the estimation procedure given above. This is perhaps the largest obstacle to a widespread acceptance of this method. As is shown in the simulation results, particularly for the Gibbs sampler, the performance of the full RNG output sequences is markedly better than that of IID sample points at times, even for problems of much larger dimension than is guaranteed by the theoretical bounds. This is similar to the use of quasi-Monte Carlo in a simple independent sampling scheme, where the Koksma-Hlawka inequality provides an error rate highly sensitive to the dimension of the sampling space. Just as in the independence scenario, there are ways to codify the improvement beyond these conservative bounds using a functional ANOVA decomposition, as is done in [41] and [25]. Still, it is a loftier task to produce a single useful rate of estimate convergence in the Markov chain setting.

1.1 New Results

Much of the paper is a restatement or slight expansion of the results in [37] and [42]. The following new results are the most significant:

1. The entire discussion of Tausworthe and linear feedback shift register generator sequences, its incorporation into the framework of CUD arrays, and multiple strategies for its randomization, in Sections 4.5.2 and 5.2

2. A safe scheme for a more smooth use of the generator sequence, such that error cancellation is augmented, as discussed in Section 5.1

3. A more careful demonstration that the randomized CUD arrays of interest are array-WCUD, in Section 5.2

4. A more complex Metropolis-Hastings example where the method is not as nice, in Section 6.4

5. A smoother Metropolis-Hastings-type algorithm introduced in Section 7.1.2

The first two items are the most essential in the advancement of the method beyond what is seen in [37] or [42]. The last item is interesting, but there is still much exploratory work to be done to see if the method provides a general advantage.

Chapter 2

Background

The purpose of this section is both to make the reader familiar with the fundamental aspects of Markov chain Monte Carlo (MCMC) algorithms and quasi-Monte Carlo (QMC) techniques and to introduce notation of significant use throughout this work.

2.1 Simple Monte Carlo Estimation

We are given a probability distribution π on a state space $\mathbf{S}$, and a function $f : \mathbf{S} \to \mathbb{R}$. The task of Monte Carlo estimation is to construct an estimate of $E_\pi[f(X)]$ (X is a π-distributed random variable on $\mathbf{S}$). Frequently an explicit solution for this expectation is readily obtained. This state space $\mathbf{S}$ will be discrete or continuous in all cases examined here, and in these respective circumstances, we denote by $\pi(\omega)$ the probability mass function or density function of the distribution π at a state ω.

In cases where $\mathbf{S}$ is $\mathbb{R}$ or a Jordan measurable subset of $\mathbb{R}$ and the function f is well-behaved, classical quadrature methods are a clear choice for the estimation of $E_\pi[f(X)]$, when an analytical solution is unavailable. In cases where $\mathbf{S}$ is a space of considerably larger dimension, the number of function evaluations required to implement the analogous quadrature is far larger. The increased difficulty of basic quadrature in higher dimensions can be codified through the rate of decay of the quadrature error; if ϵ_n is the absolute error of an estimate of $E_\pi[f(X)]$ based on an n-point quadrature rule for a well-behaved function f, we note that the rate of decay of ϵ_n is far slower for higher dimensions (most methods yield error rate $O(n^{-r/d})$ for some constant r). Simple Monte Carlo estimation is a randomized procedure that is clearly beneficial in high dimensions, as the absolute error of the estimate is $O_p(n^{-1/2})$ for a state space of any dimension (assuming f has finite variance over π). In simple Monte Carlo, we generate a sequence of values $X^{(1)}, \ldots, X^{(n)}$ with the practically sustainable assumption that these values are mutually independent and that each X_i has distribution π. In this case we take as our Monte Carlo estimate of $E_\pi[f(X)]$ the sample mean of f over our generated point set:

$$\frac{1}{n}\sum_{i=1}^{n} f(X^{(i)}) \approx E_\pi[f(X)] \tag{2.1.1}$$

Fundamental results in probability and statistics validate this procedure given the assumption that the point set really is an IID π-distributed sample. The strong law of large numbers guarantees almost sure (a.s.) consistency of our estimate:

$$\frac{1}{n}\sum_{i=1}^{n} f(X^{(i)}) \to E_\pi[f(X)] \qquad \text{a.s.} \tag{2.1.2}$$

In terms of finding the error rate, we note by the Central Limit Theorem that when f has finite variance σ^2 over π,

$$\sqrt{n}\left[\frac{1}{n}\sum_{i=1}^{n} f(X^{(i)})\right] \xrightarrow{\mathcal{L}} N(0,\sigma^2). \tag{2.1.3}$$

Hence the absolute error of our Monte Carlo estimate is $O_p(n^{-1/2})$. Only in cases where f is unbounded but a.s. finite can the task of determining whether f is of finite variance become difficult.

2.1.1 Random Number Generation

The assumption that one can create a set of values with independent π distributions is one that has become more viable with the advent of computing power and is crucial in the justification of simple Monte Carlo estimation. The task of generating a set of independent values with distribution π is usually divided into two parts:

1. Generate a set of values $U^{(1)}, \ldots, U^{(n)}$ assumed to be i.i.d. $U[0,1)^d$
2. Transform $U^{(i)}$ to yield $X^{(i)}$ with distribution π

Note that the first step is equivalent to generating a set of nd values assumed to be independent uniforms on $[0,1)$. The goal of a "random number generator" is this first step, to which much attention has been given. As the capacity grows for working with samples of increasing size, algorithms of increasing sophistication have been designed to produce a sequence that is practically indistinguishable in law from a sequence of independent uniforms. These algorithms usually operate via a recursive formula such that the next number generated is determined by the last number or last several numbers. As each number is identified to finite precision, a recursive generator must be periodic. If the sample size is larger than the period of the generator then clearly an illusion of independence is impossible to maintain; the presence of discernible patterns gives rise to an informal notion that to maintain pseudo-independence, a sample from a RNG should not exceed the square root of the period in size.

A "good" RNG should be able to produce blocks of points that look like independent uniforms; i.e., the empirical distribution of the entire set of s-dimensional blocks obtained by s consecutive outputs of the generator should be close, in some sense, to the uniform distribution over $[0,1)^s$. The idea of relating an empirical distribution to the uniform distribution will be developed in the introduction to quasi-Monte Carlo and will play a central role in much of the findings discussed later.

In this work, one large RNG is used to create practically IID uniform samples. This generator (due to [28]), commonly known as the "Mersenne Twister," has a period of $2^{19937}-1$ and has optimal equidistribution property in 623-dimensional output blocks to 32-bit accuracy. (In other words, the $2^{623\cdot 32}$ bits formed from the leading 32 bits of 623 successive outputs of the generator take on each value in $\{0,1\}^{623\cdot 32}$ the same number of times, except the all-zero combination, which appears one fewer time.) For the simulations studied herein, the assumption of independence and uniformity among values taken from this generator appears safe.

2.1.2 Transformations

In the continuous univariate setting, the usual method of using a uniform variate U to generate a target distribution with cumulative density function (CDF) $F(x)$ is to take $X = F^{-1}(U)$, the image of the uniform under the inverse of the CDF:

$$P(F^{-1}(U) \le x) = P(U \le F(x)) = F(x). \tag{2.1.4}$$

A brief review of the most commonly seen continuous univariate distributions makes it clear that even when the density of a distribution is known, the closed

Table 2.1: Acceptance/Rejection Sampling Algorithm

```
Simple Acceptance/Rejection Sampling
1 | Generate uniform $U^{(1)}$
2 | Transform $U^{(1)}$ to a $g$-distributed variable $G$
3 | Generate uniform $U^{(2)}$
4 | If $U^{(2)} < \frac{\pi(G)}{cg(G)}$:
  |     Set $X$ to $G$
  | Else:
  |     Go to step 1
5 | Return $X$
```

form expression of its CDF or inverse CDF is frequently not available. Univariate Gaussian and Gamma distributions are such cases. In such cases as these, some easily evaluated expressions are available that convert a uniform into its appropriate quantile to a negligible precision for the vast majority of $[0,1)$; these formulae are in operation in such functions as "qnorm" and "qgamma" in R, which come respectively from [44] and [1]. In arbitrary cases where the inverse CDF is not easily obtained or sufficiently approximated, alternative methods are necessary. See [9] for more details.

One method that is usually available for any continuous distribution with an identifiable density π is that of acceptance/rejection sampling. What is required is a distribution (with density g) for which sampling is easy by a simple uniform transformation, with the condition that

$$\sup_{x \in \mathbb{R}} \frac{\pi(x)}{g(x)} = c < \infty.$$

The algorithm for generating a π-distributed variable is in Table 2.1, with all generated uniforms independent.

The number of uniform variables required to generate X is twice a geometric variable with parameter $1/c$; note this number is unbounded.

In the multivariate setting, it is valid to generate each univariate component by its conditional distribution on the components already generated, with the first component sampled from its marginal distribution. For a multivariate Gaussian distribution with covariance matrix Σ, a common method of generation is a case of the above practice. Using univariate normal generation, a multivariate normal with identity covariance matrix is created and then transformed under the linear operator $\Sigma^{1/2}$, obtained via the Cholesky decomposition of Σ.

2.2 Quasi-Monte Carlo

The simple Monte Carlo estimate of $E_\pi[f(X)]$ is the sample mean of f on a set of points $X^{(1)}, \ldots, X^{(n)}$; equivalently, this estimate is the expectation of f on the empirical measure of this point set. For f with a bounded variation condition, we can justify the consistency of this estimation procedure by the convergence of the empirical measure to the distribution π. As we transform a uniform variable $U^{(i)} \sim U[0,1)^d$ to get $X^{(i)}$, this convergence is equivalent to the convergence of the empirical measure of $U^{(1)}, \ldots, U^{(n)}$ to that of the uniform measure on $[0,1)^d$.

The convergence of the empirical measure of a uniform sample to the uniform distribution is characterized by a notion of discrepancy, which is a multivariate generalization of the Kolmogorov-Smirnov distance. We develop the notion of discrepancy as follows.

For points $y, z \in [0,1]^d$, denote by $[y,z]$ the rectilinear box with every edge parallel to some axis and opposite corners at y and z (i.e. the Cartesian product of intervals $[\min(y_i, z_i), \max(y_i, z_i)]$). The uniform measure of $[y,z]$ is its Euclidean volume:

$$V([y,z]) \triangleq \prod_{i=1}^{d} |z_i - y_i|. \tag{2.2.1}$$

The empirical measure of $[y,z]$ over a point set $u^{(1)}, \ldots, u^{(n)}$ is the fraction of points in the box:

$$\hat{V}_n([y,z]) \triangleq \frac{1}{n}\sum_{i=1}^{n} I_{u^{(i)} \in [y,z]}. \tag{2.2.2}$$

Here the focus will be on "anchored" boxes, where z is the corner opposite the origin ($y = \mathbf{0}$). A notion of local discrepancy comes from the absolute difference between the uniform and empirical measure of $[\mathbf{0}, z]$:

$$\delta_n(z, u^{(1)}, \ldots, u^{(n)}) \triangleq |\hat{V}_n([\mathbf{0}, z]) - V([\mathbf{0}, z])|. \tag{2.2.3}$$

An overall notion of the deviation from uniformity of the point set is obtained by finding the supremum of this local discrepancy. We call this value the star discrepancy:

Definition 2.2.1. The star discrepancy of a point set is the supremum of its local discrepancy over all anchored boxes:

$$D_n^*(u^{(1)}, \ldots, u^{(n)}) \triangleq \sup_{z \in [0,1]^d} \delta_n(z). \tag{2.2.4}$$

The star is used to specify that only anchored boxes are examined. An analogous unanchored discrepancy takes the supremum of the volume difference over all boxes $[y,z]$. When it is certain which point set is under examination, its inclusion as an argument may be suppressed for simplicity of notation. Future references to "discrepancy" will indicate global star discrepancy unless otherwise specified.

An analogous derivation to that used to derive the null distribution in a Kolmogorov-Smirnov test verifies that the empirical measure of the first n values in an IID uniform sequence converges to the uniform measure with a $n^{-1/2+\epsilon}$ rate:

$$D_n^* = O_p(n^{-1/2}(\log\log n)). \tag{2.2.5}$$

This rate of decay of the star discrepancy of a point set bears relevance to the use of that point set in constructing Monte Carlo estimates. First we must note that evaluating f on the π-distributed value $X^{(i)}$ is equivalent to evaluating $f \circ \xi$ on the d-dimensional uniform $U^{(i)}$, where ξ is the aforementioned transformation function used to generate π-distributed variables. Thus we can assume that our Monte Carlo task is the evaluation of a function f over the $U[0,1)^d$ distribution.

For estimating the integral of f over the unit hypercube in d dimensions, integration error relates to discrepancy by the Koksma-Hlawka inequality:

Theorem 2.2.2 (Koksma-Hlawka Inequality). *The absolute error of integration using point set $u^{(1)}, \dots, u^{(n)}$ obeys the inequality*

$$\left| \int_{[0,1)^d} f \; dU - \frac{1}{n}\sum_{i=1}^{n} f(u^{(i)}) \right| \leq D_n^* V_{HK}(f), \tag{2.2.6}$$

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

The set of functions with finite Hardy-Krause variation includes bounded continuous functions f with the condition $|f(x) - f(y)| \leq C|x - y|$ for some constant C. Functions with discontinuity are generally of infinite Hardy-Krause variation; hence it is a widely held belief that QMC only “works” on continuous integrands. For a thorough treatment of Hardy-Krause variation, see [36]. When f has finite variation, the rate of decay of the discrepancy bounds the rate of decay of the Monte Carlo error.

In IID sampling, the $O_p(n^{-1/2})$ absolute error rate found from (2.1.3) is corroborated by (2.2.5) and (2.2.6).

The actual practice of quasi-Monte Carlo (QMC) is done in response to the desire to improve error rate beyond $O(n^{-1/2})$. According to the Koksma-Hlawka inequality, the use of a point set with a quicker discrepancy decay as our sample of quasi-uniform variables will lead to quicker convergence of the resulting estimate. QMC replaces the IID uniform sequence with a deterministic "low-discrepancy sequence" that provides a more even cover of the unit hypercube than is likely for an independent random sample. In d dimensions, many sequences are known that have discrepancy $O(n^{-1}(\log n)^d)$. For fixed d, this rate is faster than $n^{-1+\epsilon}$ for any $\epsilon > 0$; it is common practice to write that these low-discrepancy sequences have discrepancy $O(n^{-1+\epsilon})$.

2.2.1 Low-discrepancy Sequences: Examples

The more sizable deviations of empirical measure from uniform measure occur when large clusters or voids of points appear. An intuitive way to choose a point set that minimizes this effect is to create induce a regular spacing between points. This gives rise to the use of integration lattices as low-discrepancy sequences. An integration lattice on $[0,1)^d$ of size n is defined by a multiplier $\mathbf{g} = (g_1, \ldots, g_d) \in \mathbb{Z}^d$:

$$u^{(i)} \triangleq \frac{1}{n}[i\mathbf{g}(\text{mod } n)] \tag{2.2.7}$$

where the modulus is applied coordinatewise. A good choice of $\mathbf{g}$ exists for each prime n such that the discrepancy of the lattice is $\leq Cn^{-1}(\log n)^d$ for some C constant with respect to n.

Notice that an integration lattice contains the origin. In many examples we want to avoid sampling too closely to the corners of the cube for our sample size, and certainly the actual corner can yield severe problems. We also note that there is one deterministic estimate obtained from the use of a lattice. A well-used habit that addresses both of these concerns is a randomization of the whole set of QMC points. A good randomization is one such that the image of a single point in the set is uniform on the cube under the random transformation, but the joint low-discrepancy property is preserved.

In the lattice case, a randomization that preserves the lattice spacing is due to [7] and is appropriately known as a Cranley-Patterson rotation. A single uniform variable U in the unit cube is taken, and every point is shifted by U with a "wrap-around" (coordinates shifted above 1 are moved to their mod 1 residue). For example, the point $(0.8, 0.6, 0.4)$ shifted by the random vector $(0.2921, 0.6623, 0.3010)$ becomes $(0.0921, 0.2623, 0.7010)$.

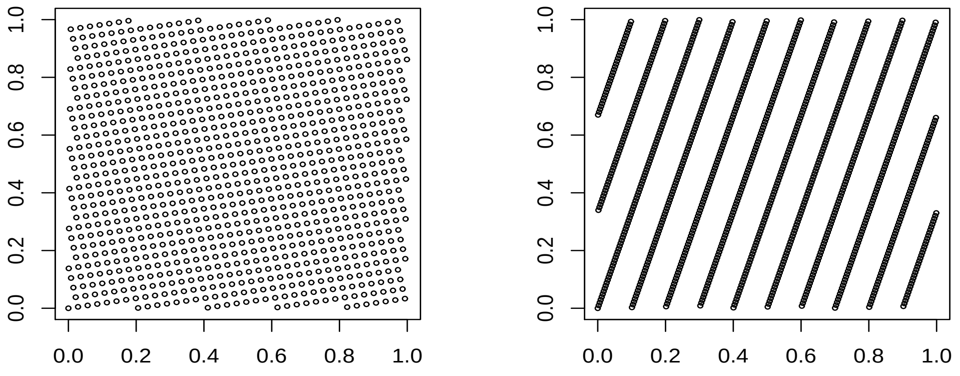
Since each point is marginally uniform, the estimate constructed from a randomized QMC point set is unbiased. An approximate variability of an estimate can be constructed using multiple replications of the procedure with independent randomizations.

As seen in [40] and [33], the star discrepancy of a lattice (which can be difficult to compute) is related to several "figures of merit," which are frequently used to select good integration lattices of a certain size. See Figure 2.1 for a look at a "good" and a "bad" lattice in two dimensions.

A single infinite sequence with desirable discrepancy is known as the Halton sequence. We note that positive integer i has a unique base b representation:

$$i = \sum_{j=0}^{\infty} a_j b^j \tag{2.2.8}$$

Figure 2.1: 2-dimensional integration lattices



for $a_j \in \{0, 1, \ldots, b-1\}$. We can define an inversion function

$$\phi_b(i) \triangleq \sum_{j=0}^{\infty} a_j b^{-(j+1)} \tag{2.2.9}$$

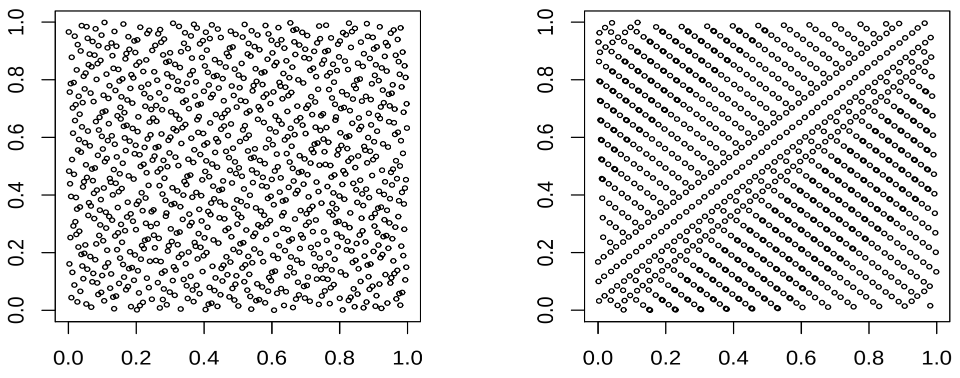
that maps each integer to a value in $[0, 1)$. The Halton sequence is defined as

$$u^{(i)} \triangleq (\phi_{b_1}(i), \ldots, \phi_{b_d}(i)), \tag{2.2.10}$$

where $b_1, \ldots, b_d$ are different bases, usually the first d prime numbers. This sequence has discrepancy $O(n^{-1}(\log n)^d)$ if the bases are different primes, and it is clear that lower bases create point sets with lower discrepancy; see Figure 2.2.

These sequences are just a few simple examples in the large canon of sequences used in QMC sampling. A relatively thorough treatment of common QMC practices is found in [33].

Figure 2.2: 2-dimensional projections of the Halton sequence, first 1000 points. The left graph has components corresponding to prime bases 2 and 3, while the right graph has components corresponding to 27 and 29.



2.3 Markov Chain Monte Carlo

The use of Markov chain Monte Carlo (MCMC) is most often in cases where the construction of an IID sample of points under a target distribution π is impossible. As was seen in the brief background on transforming uniform numbers, the knowledge of the density function of the desired distribution is usually sufficient to create an independent sampling scheme by means of acceptance/rejection sampling, although this may be computationally costly.

2.3.1 Metropolis-Hastings Algorithms

The first instances of MCMC sampling addressed a problem in which the target distribution density is proportional to a known energy function. The algorithm due to

Table 2.2: Metropolis-Hastings Algorithm

```
The Metropolis-Hastings Algorithm
1 | Begin at $X^{(0)} \in \mathbf{S}$
2 | Given $X^{(i)}$, generate $Y^{(i+1)}$
  |         Transition proposal density $q(X^{(i)}, \cdot)$
3 | Generate $U^{(i+1)} \sim U[0,1)$
4 | For $A(x,y) = \min\left(\frac{\pi(y)q(y,x)}{\pi(x)q(x,y)}, 1\right)$
  |         **If** $U^{(i+1)} < A(X^{(i)}, Y^{(i+1)})$
  |                 Set $X^{(i+1)}$ to $Y^{(i+1)}$
  |         **Else**
  |                 Set $X^{(i+1)}$ to $X^{(i)}$
5 | Repeat steps 2-4 $K + n$ times
6 | Return $X^{(K+1)}, \ldots, X^{(K+n)}$
```

[29] and its generalization by [12], accordingly known as the Metropolis-Hastings algorithm, creates a Markov chain whose values converge in distribution to an arbitrary distribution π and requires only the knowledge of the ratio of the density at two states x and y. (Equivalently, one can create the algorithm if a function on the state space proportional to the density function is known.)

The Monte Carlo sample drawn from this algorithm for the purpose of estimation is usually all values in the chain beyond a "burn-in" period (such that every point in the sample is considered approximately marginally π-distributed). The values in this sample are no longer independent, but the consistency of the estimate drawn from this sample is now verified by ergodic theory instead of the law of large numbers. The mechanics of the general algorithm are outlined in Table 2.2. First for every state x we need a transition distribution Q_x on the state space with density denoted by $q(x, \cdot)$.

The only restriction on the proposal densities $q(x, \cdot)$ is that all states communicate, and the choice of proposal densities can affect the ease of simulating the chain and the speed through which the chain ranges over the state space. Issues relating to the choice of proposal densities have warranted significant study, although for the most part, this topic will not be discussed here. The value $A(X^{(i)}, Y^{(i+1)})$ in the algorithm above is known as the acceptance probability, as it is the chance of the chain moving to $Y^{(i+1)}$ versus staying at $X^{(i)}$. It is important to note that if the proposed value is not accepted, then the previous value is repeated in the sample. This is essential for consistency of the estimate constructed from the sample, as is obvious from a 2-state space with nonuniform target distribution: a sample without repeated values would have an empirical measure converging to the uniform distribution on the two states.

To verify that this chain has stationary distribution π, we note that for two distinct states $x, y \in \mathbf{S}$, the overall transition kernel is given by $q(x,y)A(x,y)$, and from the definition of $A(x,y)$ in step 4 of the algorithm, it is easy to verify that reversibility holds for this chain:

$$\pi(x)q(x,y)A(x,y) = \pi(y)q(y,x)A(y,x), \qquad \forall\ x \neq y. \tag{2.3.1}$$

Alternate definitions for $A(x,y)$ exist which still yield the reversibility condition. These are acceptable for Metropolis-Hastings, although the one given here is most frequently used, as it minimizes rejections.

Commonly seen sets of proposal distributions include the “random walk” sampler, in which $q(x,y)$ is a symmetric density on y about the starting value x and the distributions look the same about x for all x (i.e., $q(x, y-x)$ does not depend on x), and the “independence” sampler, where $q(x,y)$ is the same for all x.

2.3.2 Gibbs Sampling

The claim has been made that all valid MCMC methods for approximate sampling from a stationary distribution are instances of a Metropolis-Hastings algorithm or a mild extension thereof. The most familiar method of MCMC to many is known as the Gibbs sampler, which may seem not to fit in the Metropolis-Hastings scheme at first glance. In truth, the Gibbs sampler can be reconciled under this framework, and much of the theoretical results established in the remainder of this work for Metropolis-Hastings extend to the Gibbs sampler quite easily.

The Gibbs sampler, whose necessity emerges naturally in problems in Bayesian modeling, looks to generate a value $\theta = (\theta_1, \ldots, \theta_d)$ under a joint distribution when the usable information known about this distribution is its full set of conditional distributions; i.e., $P(\theta_i|\theta_1, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots, \theta_d)$ is known for all i. We assume that drawing from these conditional distributions can be done; there are instances where acceptance/rejection sampling or even Metropolis-Hastings sampling is done to generate points from these conditional distributions. The basic step in the algorithm updates a single component by keeping the others fixed and drawing from the conditional distribution of this component given the others. These components are often updated in cyclic fashion, although a random choice is sometimes taken as well. We will keep focus on the cyclic scan algorithm, which is detailed in Table 2.3.

Note that this algorithm only returns points taken d steps apart, such that each component is updated exactly once before the next point in the sample is taken. A sample that takes every point can also be used for consistent estimation, and so either choice is valid. To reconcile with the Metropolis-Hastings framework, we note that for a single step, if we use the conditional distribution in step 3 of the algorithm as our proposal density, our accepance probability is always 1. The only distinction now

Table 2.3: Gibbs Sampler

The Gibbs Sampler	
1	Start with $\theta^{(0)}$
2	Given $\theta^{(i)} = (\theta_1^{(i)}, \ldots, \theta_d^{(i)})$ take $s \equiv i + 1 (\text{mod } d)$
3	Draw $\tilde{\theta}_s$ from $P(\theta_s \| \theta_1^{(i)}, \ldots, \theta_{s-1}^{(i)}, \theta_{s+1}^{(i)}, \ldots, \theta_d^{(i)})$
4	Set $\theta^{(i+1)}$ to $(\theta_1^{(i)}, \ldots, \theta_{s-1}^{(i)}, \tilde{\theta}_s, \theta_{s+1}^{(i)}, \ldots, \theta_d^{(i)})$
5	Repeat steps 2-4 $K + nd$ times
6	Return $\theta^{(K+d)}, \theta^{(K+2d)}, \ldots, \theta^{(K+nd)}$

is that the proposal densities change every step. We can view every d steps as having one common proposal distribution (with acceptance probability still 1). If we take every point instead of every d points, we can view our sample as the combination of d interlocking Metropolis-Hastings chains.

Chapter 3

Foundation: MCQMC

The main goal that this work has sought to develop is the application of randomized QMC sequences to the general MCMC sampling scheme, such that we can obtain benefits analogous to those of QMC in independence sampling. Here we denote this practice as Markov chain quasi-Monte Carlo, or MCQMC. Most of the key results in this section that establish a condition for valid MCQMC appear in some detail in [37] and [42]. It will become clear by the following results that the question of validity, which is synonymous with estimate consistency, is addressed to sufficient satisfaction. The question of superiority to IID sampling is a far more difficult one, as there is a dependence structure between successive updates to a Markov chain. The notion that a significant dependence structure among the variates used k steps apart can create problems for sizable k gives rise to a new "curse of dimensionality" that reduces the advantage of low-discrepancy sequences.

There is a distinction that should be clarified before the structure of the MCQMC algorithm is established: the use of QMC sequences in MCMC estimation is not done to accelerate convergence to the stationary distribution. Much attention is given to ways in which convergence can be verified and ways in which slow-mixing or frequently "stuck" chains can be accelerated; neither question is a closed case by any means. However, the chief contribution of MCQMC is not the acceleration of convergence. Rather, on the assumption on convergence, the aim of MCQMC is to create a more balanced sample of the space for improving estimate accuracy, in the same way that QMC is done to cover the cube more evenly than by IID sampling.

3.1 MCQMC Notation

In independent Monte Carlo estimation, the use of a QMC sequence in lieu of pseudorandom numbers (assumed to be IID) seems immediately clear: use each value in the sequence as a sample point. It is less obvious how one might go about using a QMC sequence in a Metropolis-Hastings sampler. A definitive answer is not given here, but it will help in future discussion to establish a notation on the values used at different stages of an MCMC algorithm.

Recall in a Metropolis-Hastings sampling scheme that to simulate a step in the chain, two actions that require random variates must occur: the generation of a proposal value y from a transition proposal density, and the generation of an acceptance/rejection decision based on the acceptance probability. For now, we assume that with probability 1, at most $d-1$ independent univariate uniforms are required to generate a variable from the transition proposal density (for any starting state). Clearly only one univariate uniform is needed to generate the decision, and so each step in the chain requires (at most) d univariate uniforms.

Similarly in the Gibbs sampler, we assume that a bounded number of variates is required to update ALL the coordinates once, regardless of the starting values used in the conditional distribution sampling. We assume this bounding number is d in this case, as no acceptance/rejection step is necessary. Then d univariate values are needed to generate the next point in the sample for any MCMC scheme of interest. We call this a d-dimensional MCMC sampler.

After a burn-in period (if it is so desired), we wish to generate a sample of size N, and so we run the chain through N steps. The univariate values needed to effect these N steps will be stored in the "variate matrix", which is indexed as follows:

$$\begin{bmatrix} u^{(1)} & u^{(2)} & \cdots & u^{(d)} \\ u^{(d+1)} & u^{(d+2)} & \cdots & u^{(2d)} \\ \vdots & \vdots & \ddots & \vdots \\ u^{((N-1)d+1)} & u^{((N-1)d+2)} & \cdots & u^{(Nd)} \end{bmatrix} \tag{3.1.1}$$

The sequence $u^{(1)}, u^{(2)}, \ldots, u^{(Nd)}$ of univariate values in the variate matrix will be referred to as the "driving sequence" of the MCMC algorithm.

Frequently it will be of interest to look at blocks of this sequence as variates in a hypercube. To that effect, we define for any $i < j$ the notation $\mathbf{u}_{i:j} \triangleq (u^{(i)}, u^{(i+1)}, \ldots, u^{(j)})$, the $(j-i+1)$-dimensional point with coordinates taken from the univariate sequence.

The mth row $\mathbf{u}_{((m-1)d+1):(md)}$ of the variate matrix is used to generate the mth sample point of the chain. As this is a Markov chain, we can define a Markov transition function:

$$X^{(m)} = \phi(X^{(m-1)}, \mathbf{u}_{((m-1)d+1):(md)}) \tag{3.1.2}$$

We will want to investigate the relationship of the univariate values in the coordinates of successive multivariate values, and so we define for a set of points $x^{(1)}, \ldots, x^{(n)} \in [0,1)^d$ the unpacking function $\mathcal{U}(x^{(1)}, \ldots, x^{(n)})$ to return the sequence of univariate values $u^{(1)}, \ldots, u^{(nd)}$ such that $\mathbf{u}_{((m-1)d+1):(md)} = x^{(m)}$.

3.2 Completely Uniform Distribution

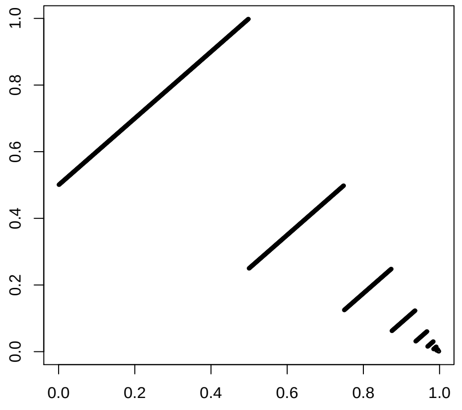
We assume a d-dimensional MCMC sampler is our candidate for receiving QMC updates in lieu of psuedorandom values, and we are able to construct a d-dimensional QMC sequence $x^{(1)}, x^{(2)}, \ldots, x^{(N)}$. The clearest way to use the QMC points is to make each row of the variate matrix a QMC point, such that $\mathbf{u}_{((m-1)d+1):(md)} = x^{(m)}$. In this scheme, each QMC point is used to generate one step in the chain. For this method of inclusion, many QMC point sets will lead to comically inaccurate results.

Recall the Halton sequence with bases the first d primes, and suppose we wish to use this point set in a random walk Metropolis-Hastings algorithm. Assuming the standard method of generating the proposal value from a point in $[0,1)^{d-1}$, the proposal will have a smaller first coordinate than the current value if the first variate used (which lies in the first column of the variate matrix) is less than 0.5. Likewise, the first coordinate of the proposal will be larger than the current value if the first variate is greater than 0.5. The use of the Halton sequence in d bases would establish the first column of the variate matrix as the base 2 sequence, known as the van der Corput sequence. This sequence is as follows:

$$1/2, 1/4, 3/4, 1/8, 5/8, 3/8, 7/8, 1/16, 9/16\ldots$$

Note that it alternates above and below 1/2. Thus the proposals alternately move up and down in the first coordinate. The chain is prevented from moving into the tails of the marginal distribution of this first coordinate, and so consistency from a sample generated in this fashion obviously fails for a variety of functions. Alternative

Figure 3.1: Lag plot of van der Corput sequence.



methods of interlaying QMC points of different dimension may be even worse, such as an example in [30], where a particle intended to undergo symmetric random walk moves in the same direction at every step.

The clear culprit in the grave errors obtained in this crude attempt at MCQMC is the relationship between successive points in the QMC sequence. The practical notion of independence among successive values in a good random number generator output is not upheld here. We can codify this notion by looking at the properties of the unpacked sequence $u^{(1)}, \ldots, u^{(nd)} = \mathcal{U}(x^{(1)}, \ldots, x^{(n)})$. For the Halton sequence, the relationship between $u^{(i)}$ and $u^{(i+d)}$ for all values of i is not one that approximates uniformity on the unit square, as seen in Figure 3.1.

It is easy to conceive scenarios in which a lack of approximate uniformity in $[0,1)^m$ among $x^{(i)}, x^{(i+k_1)}, \ldots, x^{(i+k_m)}$ for some set of lags $(k_1, \ldots, k_m)$ can create a similar problem for producing a consistent sample. Sequences that avoid this problem are such that the points created by blocks of d successive variates fill the unit hypercube

$[0, 1)^d$ in a uniform fashion. In an asymptotic sense, to approach uniformity is to have a discrepancy that decays to 0. This motivates the proposition that the following sequence condition is essential for incorporation of a sequence in an MCMC sampler:

Definition 3.2.1. A sequence $u^{(1)}, u^{(2)}, \ldots$ is completely uniformly distributed (CUD) if for every integer $s \geq 1$, the sequence $x^{(1)}, x^{(2)}, \ldots$ of s-blocks ($x^{(i)} = \mathbf{u}_{i:(i+s-1)}$) satisfies:

$$D_n^*(x^{(1)}, \ldots, x^{(n)}) \longrightarrow 0 \qquad \text{as } n \to \infty. \tag{3.2.1}$$

The concept of CUD sequences originated in [17], and a survey of many CUD sequence constructions is given in [22]. The CUD property is given as a definition of randomness in [16]. Note there is no condition of uniform convergence of the discrepancy to 0 over all dimensions s. This definition applies to deterministic sequences; it serves us to create a similar definition for random sequences.

Definition 3.2.2. A random sequence $u^{(1)}, u^{(2)}, \ldots$ is weakly completely uniformly distributed (WCUD) if for every integer $s \geq 1$, the sequence $x^{(1)}, x^{(2)}, \ldots$ of s-blocks ($x^{(i)} = \mathbf{u}_{i:(i+s-1)}$) has the following condition of convergence in probability:

$$D_n^*(x^{(1)}, \ldots, x^{(n)}) \xrightarrow{P} 0 \qquad \text{as } n \to \infty. \tag{3.2.2}$$

To validate the use of (W)CUD sequences in an MCMC sampler, we generalize a result of Chentsov [6] derived for a simpler class of Markov chain simulations. Before the mechanics of this result and its proof are discussed, it is helpful to note the following lemma, shown in [16]:

Lemma 3.2.3. *The sequence $u^{(1)}, u^{(2)}, \ldots$ is CUD if and only if for arbitrary integers $s \geq l \geq 1$, the sequence $\{z^{(i)}\}$ of s-tuples defined by $z^{(i)} = (u^{(is-l+i)}, u^{(is-l+2)}, \ldots, u^{(is-l+s)})$*

satisfies

$$D_n^*(z^{(i)}, \dots, z^{(n)}) \longrightarrow 0 \qquad \text{as } n \to \infty. \tag{3.2.3}$$

An analogous equivalence holds for WCUD sequences.

The lemma establishes that a CUD property has good balance in both its overlapping blocks and its nonoverlapping blocks of arbitrary offset. The "if" statement is easy to verify via Slutsky's Theorem, but the "only if" statment is less obvious.

3.3 The Main Consistency Theorem

The following result, as mentioned earlier, generalizes a result of Chentsov from what he calls a "standard construction" for Markov chain simulation on a finite state space. The following proof is contained in [37], but it will be repeated here, as the result is the foundation on which any future results rest. The generalization to Metropolis-Hastings sampling requires some assumptions. The most restrictive is that our state space $\mathbf{S}$ is finite; the necessity of this restriction is readily evident in the proof.

A milder assumption satisfied by all feasible sampling schemes is a regularity condition on the proposal mechanisms in the Metropolis-Hastings algorithm:

Definition 3.3.1. The proposals of a Metropolis-Hastings algorithm are regular if and only if for any states $k, l \in \mathbf{S}$ and time i, the set

$$\mathcal{A}_{kl}^{(i)} \triangleq \{(u^{(id+1)}, \dots, u^{(id+d-1)} | Y^{(i+1)} = l \text{ when } X^{(i)} = k\} \tag{3.3.1}$$

is Jordan measurable.

A Jordan measurable set is one whose indicator function is Riemann integrable. For a starting state k at time i, the hypercube $[0,1)^{d-1}$ can be divided into regions $\mathcal{A}_{kl}^{(i)}$ of variates whose use to generate the next proposal would propose state l; regularity implies that each of these sets is Jordan measurable. Note these sets are usually the same for all i (the proposals are homogenous).

See Appendix A for a treatment of Jordan measurable sets in the unit hypercube. Included are the results that finite unions and tensor products of Jordan measurable sets are also Jordan measurable.

Lemma 3.3.2. *If regularity of proposals holds, for any states $k, l \in \mathbf{S}$ and time i, the overall transition sets defined as*

$$\mathcal{S}_{kl}^{(i)} \triangleq \{u^{(id+1)}, \ldots, u^{(id+d)} | X^{(i+1)} = l \text{ when } X^{(i)} = k\} \tag{3.3.2}$$

are Jordan measurable.

Proof. For $k \neq l$, $\mathcal{S}_{kl}^{(i)} = \mathcal{A}_{kl}^{(i)} \times [0, A(k,l))$, the product of two Jordan measurable sets. $\mathcal{S}_{kk}^{(i)} = \left(\bigcup_{l \in \mathbf{S} \setminus \{k\}} \mathcal{A}_{kl}^{(i)} \times [A(k,l), 1)\right) \cup \mathcal{A}_{kk}^{(i)} \times [0,1)$ (assuming $[1,1) = \emptyset$). These are Jordan measurable due to Theorems A.1.4 and A.1.5. □

The central theorem indicates that the replacement of IID points by a CUD sequence preserves the consistency of a Metropolis-Hastings sampler. In this finite-state setting, consistency holds if for any state $\omega \in \mathbf{S}$ and any starting state $X^{(0)} = \omega_0$:

$$\hat{\pi}_n(\omega) \triangleq \frac{1}{n} \sum_{i=1}^{n} 1_{X^{(i)}=\omega} \to \pi(\omega). \tag{3.3.3}$$

Similarly, weak consistency holds if for any state $\omega \in \mathbf{S}$ and $\epsilon > 0$, under any starting state ω_0:

$$P(|\hat{\pi}_n(\omega) - \pi(\omega)| > \epsilon \mid X^{(0)} = \omega_0) \to 0. \tag{3.3.4}$$

Theorem 3.3.3. *Suppose* $\mathbf{S} = \{\omega_1, \ldots, \omega_K\}$ *is finite and a sequence* $u^{(1)}, u^{(2)}, \ldots$ *is used to run a Metropolis-Hastings sampler with regular homogenous proposals. Assume the resulting sample is weakly consistent if the* $u^{(i)}$ *are IID* $U[0,1)$*, such that (3.3.4) holds. Then if the* $u^{(i)}$ *form a CUD sequence, the consistency result (3.3.3) holds. Similarly, if* $u^{(i)}$ *are a WCUD sequence, (3.3.4) holds.*

Proof. For a given value of $X^{(0)}$, the empirical measure $\hat{\pi}_n(\omega)$ is completely determined by the variates $u^{(1)}, \ldots, u^{(nd)}$. We wish to look at regions in $[0,1)^{nd}$ which are "problematic" in the sense that the empirical measure of a state is not close to its target value. For a tolerance $\epsilon > 0$, we define for each starting state and target state the region

$$\mathcal{T}_{lkn}(\epsilon) \triangleq \{(u^{(1)}, \ldots, u^{(nd)}) \mid |\hat{\pi}_n(\omega_k) - \pi(\omega_k)| > \epsilon \text{ when } X^{(0)} = \omega_l\}.$$

These regions are Jordan measurable by Theorem A.1.4 as they are the finite unions of the sets in (3.3.2). Because the volume of $\mathcal{T}_{lkn}(\epsilon)$ is the probability under IID sampling that $|\hat{\pi}_n(\omega_k) - \pi(\omega_k)| > \epsilon$ when $X^{(0)} = \omega_l$, the validity of (3.3.4) under IID sampling means that for any k, l:

$$V(\mathcal{T}_{lkn}(\epsilon)) \longrightarrow 0 \qquad \text{as } n \to \infty. \tag{3.3.5}$$

So we pick an m sufficiently large that for all k, l, $\text{Vol}(\mathcal{T}_{lkm}(\epsilon)) < \epsilon/K$, which we can do due to the finite number of states. We now define $\mathcal{T}_{km}(\epsilon) \triangleq \bigcup_{l \in \mathbf{S}} \mathcal{T}_{lkm}(\epsilon)$, the region that samples ω_k "badly" for at least one starting state. This set is Jordan measurable as well and has volume $< \epsilon$.

For ω_k and m we define an indicator $Z^{(i)}$ of tail behavior of our CUD sequence $\{u^{(i)}\}$ in the following fashion:

$$Z^{(i)} \triangleq 1_{\mathbf{u}_{((i-1)d+1):((i-1)d+md)} \in \mathcal{T}_{km}(\epsilon)}.$$

We also define the empirical measure $\hat{\pi}_{i,m}(\omega_k)$ on the corresponding block of m points in our sample:

$$\hat{\pi}_{i,m}(\omega_k) \triangleq \frac{1}{m} \sum_{j=0}^{m-1} 1_{X^{(i+j)}=\omega_k}.$$

Note that if $Z^{(i)} = 0$, $|\hat{\pi}_{i,m}(\omega_k) - \pi(\omega_k)| < \epsilon$ (although the converse is not true depending on $X^{(i-1)}$). Because the sequence $u^{(1)}, u^{(2)}, \ldots$ is CUD, we have by Lemmas 3.2.3 and A.2.2:

$$\frac{1}{n} \sum_{i=1}^{n} Z^{(i)} \to \mathrm{Vol}(\mathcal{T}_{km}(\epsilon)). \tag{3.3.6}$$

We dissect our overall empirical law on n points as follows:

$$\hat{\pi}_n(\omega_k) = \frac{1}{n} \sum_{i=1}^{n} \hat{\pi}_{i,m}(\omega_k) + \frac{1}{n} \sum_{j=1}^{m-1} \left[1_{X^{(m-j)}=\omega_k} - 1_{X^{(n+m-j)}=\omega_k} \right]. \tag{3.3.7}$$

The latter term in the above decomposition is bounded in magnitude by m/n. Now we use the triangle inequality and striate over $Z^{(i)}$:

$$\begin{aligned}
|\hat{\pi}_n(\omega_k) - \pi(\omega_k)| &\leq \frac{1}{n}\sum_{i=1}^{n} Z^{(i)}|\hat{\pi}_{i,m}(\omega_k) - \pi(\omega_k)| \\
&\quad + \frac{1}{n}\sum_{i=1}^{n}(1 - Z^{(i)})|\hat{\pi}_{i,m}(\omega_k) - \pi(\omega_k)| + \frac{m}{n} \\
&\leq \frac{1}{n}\sum_{i=1}^{n} Z^{(i)} + \epsilon + \frac{m}{n} && (3.3.8) \\
&\to \mathrm{Vol}(\mathcal{T}_{km}(\epsilon)) + \epsilon \qquad (\text{as } n \to \infty) \\
&\leq 2\epsilon. && (3.3.9)
\end{aligned}$$

As ϵ is arbitrary, (3.3.3) is established for the CUD case. If $\{u^{(i)}\}$ is WCUD, (3.3.8) still holds w.p. 1, but now

$$\frac{1}{n}\sum_{i=1}^{n} Z^{(i)} \xrightarrow{P} \mathrm{Vol}(\mathcal{T}_{km}(\omega_k)). \tag{3.3.10}$$

So for $n > m/\epsilon$,

$$P(|\hat{\pi}_n(\omega_k) - \pi(\omega_k)| > 3\epsilon) \leq P\left(\frac{1}{n}\sum_{i=1}^{n} Z^{(i)} > \epsilon\right) \to 0 \tag{3.3.11}$$

and so (3.3.4) is established for the WCUD case. □

Clearly there are cases where a non-CUD sequence still provides consistency in the sense of (3.3.3), but for a non-CUD sequence, it is easy to construct ad hoc a Metropolis-Hastings sampler on which (3.3.3) fails. Hence a general practice of MCQMC designed to adapt to an arbitrary sampling scheme should use CUD variates.

For the Gibbs sampler, the lack of an acceptance-rejection step and the nonhomogeneity of proposals are the only distinctions that need be addressed. Without the acceptance-rejection step, the Jordan measurable proposals assumption tautologically gives the Jordan measurable transitions. If we take every output in the Gibbs sampler as opposed to every dth output, there is a nonhomogeneity issue in the proposals, but again this is easily satisfied by viewing the sample as d interlocking samples, each of which is consistent, and so the average of these samples is consistent as well.

Chapter 4

CUD Sequences in Practice

4.1 Low Discrepancy

The results of the previous chapter provide a general condition on the valid use of QMC sequences in a Metropolis-Hastings sampler. The replacement of IID points by a CUD or a weakly CUD sequence leads to a consistent estimate (and note that IID points are weakly CUD), and any other choice of sequence fails for some Metropolis-Hastings construction. But not much has been said concerning the actual rate of decay of s-dimensional discrepancy in a CUD sequence for any given s. The ultimate goal is to create estimates with lower variability than those obtained via IID sampling; therefore we wish to create a "balance" along the sequence of variates, just as is done with QMC for regular independent Monte Carlo sampling.

It is apparent that for a QMC sequence like the Halton sequence, in which the relationship between successive points does not support an assumption of independence, that something must be done which eliminates this relationship. There are two intuitive notions of how this can be done:

1. Randomize the order of the points in the sequence
2. Choose a sequence whose successive points have more uniform distribution

The first method was proposed by Liao [23] on a series of Gibbs sampling schemes for fitting Bayesian models. The resulting estimates of the quantiles of the marginal posterior distributions showed lower variability than those obtained with IID sampling. No theoretical argument was given that this procedure is consistent in some sense or that the variance is reduced; this chapter will demonstrate the former claim (consistency). A nonrigorous argument for this method says that the larger the sample size, the more the points look like independent random points, except that after the entire sequence is used, each component has seen a set of update variates that are more evenly spaced than one would expect from independent uniforms. Because of this final balance, Liao's proposal may be likely to improve on IID sampling. Still, a notion of a better-than-random approximation of uniformity across successive points may provide further improvement, if it is possible.

The notion of consistency does not make sense for a single finite sequence, but the implementation of Liao's proposed method requires a clear choice of a finite simulation length before the randomization can occur. In addition, it may be difficult for a single infinite sequence to maintain a more uniform appearance in different dimensions simultaneously (e.g., see the CUD constructions given in [22]). Therefore the need arises to incorporate the use of finite sequences into the theoretical framework given in the previous chapter. To this effect, we will define classes of finite sequences of increasing length such that the limits which characterize the CUD property and consistency will be over the increase in sequence length.

The goal in constructing sequences that improve on IID sampling is a lower discrepancy in many dimensions ("s-dimensional discrepancy" is the discrepancy of the sequence of s-blocks formed from concatenating s successive values in the original sequence). Clearly for a finite sequence of length N, the s-dimensional discrepancy can only be good for $s \ll N$. Even an infinite sequence will not a discrepancy decay faster than $n^{-1/2}$ in every dimension. The reason that this line of inquiry still seems worthwhile is that the importance of uniformity in s dimensions decays as s increases. Another nonrigorous argument says that for a Markov chain that is mixing appropriately, the Markov transition function given in (3.1.2) can be expressed as

$$X^{(i)} = \phi_m(X^{(i-m)}, \mathbf{u}) \approx \psi(\mathbf{u}), \tag{4.1.1}$$

where $\mathbf{u}$ is an md-dimensional uniform variate. This approximation gets better as m increases, and so at some point when the approximation error is negligible, the Markov chain sampler is like an independent sampler of md dimensions. Thus if we find a sampling scheme whose s-blocks are well-distributed for $s \leq md$, it is likely to provide less variable estimates than those given by IID sampling.

Much of the work contained here is also found in [42]; some of the proofs are reproduced here, but more details on the remaining results are contained there.

4.2 Useful Lemmas

In working with discrepancy in multiple dimensions, it is useful to note the relationship between discrepancies of the s-blocks of a sequence.

Lemma 4.2.1. *For a sequence $u^{(1)}, u^{(2)}, \ldots \in [0,1)$, define $y^{(i)} = (u^{(i)}, u^{(i+1)}, \ldots, u^{(i+s_1-1)})$ and $z^{(i)} = (u^{(i)}, u^{(i+1)}, \ldots, u^{(i+s_2-1)})$, where $s_1 < s_2$. Then the following inequality holds:*

$$D_n^*(y^{(1)}, \ldots, y^{(n)}) \leq D_n^*(z(1), \ldots, z^{(n)}). \tag{4.2.1}$$

Proof. For a box $[\mathbf{0}, b] \subset [0,1)^{s_1}$, the empirical measure of $[\mathbf{0}, b]$ on $\{y^{(1)}, \ldots, y^{(n)}\}$ is the same as the empirical measure of $[\mathbf{0}, b] \times [0,1]^{(s_2-s_1)}$ on $\{z^{(1)}, \ldots, z^{(n)}\}$. These sets have the same Jordan measure, and so the local discrepancies are equal. Hence

$$\delta_n([\mathbf{0}, b], y^{(1)}, \ldots, y^{(n)}) \leq \sup_{m \in [0,1]^{(s_2-s_1)}} \delta_n([\mathbf{0}, b] \times [\mathbf{0}, m], z^{(1)}, \ldots, z^{(n)}). \tag{4.2.2}$$

Taking the supremum of both sides over all b, (4.2.1) follows. □

Lemma 4.2.2. *For a sequence $x^{(1)}, x^{(2)}, \ldots \in [0,1)^s$ and fixed integer m, the star discrepancies of this sequence satisfy the following inequality:*

$$\left| D_{n+m}^* - D_n^* \right| \leq \frac{m}{n+m}. \tag{4.2.3}$$

Proof. For an arbitrary box B, suppose k of the first n points in the sequence lie in B. Then the number of the first $n+m$ points that lie in B is in the set $\{k, k+1, \ldots, k+m\}$. Thus the difference of the empirical measures of B on the first n points and the first $n+m$ points is bounded above by

$$\max\left(\frac{m(n-k)}{n(n+m)}, \frac{mk}{n(n+m)}\right), \tag{4.2.4}$$

which is at most $m/(n+m)$. By the triangle inequality, the local discrepancies of B differ by at most $m/(n+m)$, and (4.2.3) follows. □

It is easier to verify the convergence of local discrepancy to 0 than to verify the convergence of star discrepancy to 0 directly. The following lemma establishes a useful equivalence that helps to establish the latter.

Lemma 4.2.3. *For a sequence $x^{(1)}, x^{(2)}, \ldots \in [0,1)^d$, if for arbitrary $z \in [0,1]^d$,*

$$\delta_n(z, x^{(1)}, \ldots, x^{(n)}) \to 0, \tag{4.2.5}$$

then convergence for star discrepancy holds as well:

$$D_n^*(x^{(1)}, \ldots, x^{(n)}) \to 0. \tag{4.2.6}$$

For a random sequence $x^{(1)}, x^{(2)}, \ldots$, if the convergence in (4.2.5) holds in probability, then (4.2.6) holds in probability as well.

Proof. For arbitrary $\epsilon > 0$, choose positive integer $M > 1/\epsilon$ and define lattice $\mathcal{L}$ to be the set of points whose coordinates are integer multiples of $1/(2dM)$ between 0 and 1 (inclusive). For arbitrary $z \in [0,1]^d$, there are points $z^{(1)}, z^{(2)} \in \mathcal{L}$ such that $[\mathbf{0}, z^{(1)}] \subseteq [\mathbf{0}, z] \subseteq [\mathbf{0}, z^{(2)}]$ and $z_i^{(2)} - z_i^{(1)} < \epsilon/(2d)$ for all i.

Note $V([\mathbf{0}, z^{(2)}]) - V([\mathbf{0}, z^{(1)}]) < \epsilon/2$, as the difference of these sets is contained in d blocks of volume $\epsilon/(2d)$. $V([\mathbf{0}, z])$ is contained in the interval $[V([\mathbf{0}, z^{(1)}]), V([\mathbf{0}, z^{(2)}])]$. By the nested nature of the boxes,

$$\hat{V}_n([\mathbf{0}, z^{(1)}]) - V([\mathbf{0}, z]) \leq \hat{V}_n([\mathbf{0}, z]) - V([\mathbf{0}, z]) \leq \hat{V}_n([\mathbf{0}, z^{(2)}]) - V([\mathbf{0}, z]). \tag{4.2.7}$$

Applying the triangle inequality to the left and right ends of the above inequality,

$$-\epsilon/2 - \delta_n([\mathbf{0}, z^{(1)}]) < \hat{V}_n([\mathbf{0}, z]) - V([\mathbf{0}, z]) < \epsilon/2 + \delta_n([\mathbf{0}, z^{(2)}]), \tag{4.2.8}$$

and so $D_n^* < \epsilon/2 + \max_{y \in \mathcal{L}} \delta_n([\mathbf{0}, y])$. As $\mathcal{L}$ is finite, (4.2.5) yields that, for a deterministic sequence, $\limsup D_n^* < \epsilon/2$. As ϵ is arbitrary, (4.2.6) follows. For a random sequence, convergence in probability implies that $P(\max_{y \in \mathcal{L}} \delta_n([\mathbf{0}, y]) > \epsilon/2) \to 0$, and so $P(D_n^* > \epsilon) \to 0$, and (4.2.6) holds in probability. $\square$

4.3 Triangular Arrays

As we would like to use a finite (W)CUD sequence to generate an MCMC sample, it is important to incorporate the use of a finite sequence into the CUD framework. We can define a class $\mathcal{C}$ of sequences of lengths $N_1 < N_2 < \ldots \to \infty$. We will denote the jth value of the ith sequence as $u_{N_i}^{(j)}$.

Definition 4.3.1. The class $\mathcal{C}$ of sequences is a CUD triangular array (array-CUD) if for arbitrary dimension s,

$$\lim_{i \to \infty} D^*_{N_i - s + 1}\Big((u_{N_i}^{(1)}, \ldots, u_{N_i}^{(s)}), \ldots, (u_{N_i}^{(N_i - s + 1)}, \ldots, u_{N_i}^{(N_i)}) \Big) = 0. \tag{4.3.1}$$

Similarly, a class of random sequences is array-WCUD if for arbitrary s, the limit in (4.3.1) holds in probability.

Many results that held for a single CUD sequence hold for a CUD triangular array. Most importantly, the central Theorem 3.3.3 extends to CUD arrays, where for a Metropolis-Hastings sampler of dimension d, the first $\lfloor N_i/d \rfloor \cdot d$ elements of row i of the array $\{u_{N_i}^{(1)}, \ldots, u_{N_i}^{(N_i)}\}$ are used to generate a sample of size $\lfloor N_i/d \rfloor$.

Theorem 4.3.2. *Let $M_i = \lfloor N_i/d \rfloor$. Suppose for arbitrary starting value $X^{(0)}$ the sequence $\{u_{N_i}^{(1)}, \ldots, u_{N_i}^{(M_i d)}\}$ is used as a driving sequence for a finite-state Metropolis-Hastings sampler under which (3.3.4) holds with an IID driving sequence. For the* *resulting sample* $X_{N_i}^{(1)}, \ldots, X_{N_i}^{(M_i)}$ *and the resulting empirical measure*

$$\hat{\pi}_{N_i}(\omega) \triangleq \frac{1}{M_i} \sum_{j=1}^{M_i} \mathbf{1}_{\{X_{N_i}^{(j)}=\omega\}}, \tag{4.3.2}$$

the following convergence result holds when the triangular array is CUD:

$$\hat{\pi}_{N_i}(\omega) \to \pi(\omega) \qquad \forall\ \omega \in \mathbf{S}. \tag{4.3.3}$$

If the triangular array is WCUD, convergence in probability holds:

$$\hat{\pi}_{N_i}(\omega) \xrightarrow{P} \pi(\omega) \qquad \forall\ \omega \in \mathbf{S}. \tag{4.3.4}$$

The proof of this theorem includes only a few minor modifications of the proof of Theorem 3.3.3. This result is also fundamental in justifying the use of certain sequences in Metropolis-Hastings sampler. Subsequent sections will develop specific CUD triangular arrays that will be useful for MCQMC.

Lemma 4.2.3, which verifies the equivalence of local discrepancy decay and star discrepancy decay, has a clear analog for arrays. Lemma 3.2.3, which establishes the equivalence of a CUD property for overlapping and nonoverlapping s-tuples, also has an analog for arrays (note that this extension is necessary for Theorem 4.3.2). However, to verify that certain classes of sequences are array-WCUD, we need the following stronger result, which is not obvious for WCUD arrays. Its proof appears in [42].

Theorem 4.3.3. *For some infinite subset* $\mathcal{D} \subseteq \mathbf{N}$, *suppose that a triangular array* *satisfies for any $s \in \mathcal{D}$ and $\epsilon > 0$:*

$$\lim_{i\to\infty} P\left[D_M^*\left((u_{N_i}^{(1)},\ldots,u_{N_1}^{(s)}),(u_{N_i}^{(s+1)},\ldots,u_{N_1}^{(2s)}),\ldots,(u_{N_1}^{((M-1)s+1)},\ldots,u_{N_i}^{(Ms)})\right) > \epsilon\right] = 0, \tag{4.3.5}$$

where $M = \lfloor N_i/s \rfloor$. Then the triangular array is array-WCUD.

The lemma says that to show an array-WCUD property, it is only necessary to verify the discrepancy decay in probability of the nonoverlapping s-tuples for s in an infinite subset of the positive integers. This subset often contains only integer multiples of a common integer s_0.

4.4 Liao's Method

Recall the proposal of Liao that takes a low-discrepancy sequence in d dimensions and randomly permutes the sequence to use in a d-dimensional Gibbs sampler. This method requires a selection beforehand of the simulation length N, as the permutation step does not permit extensibility. A theoretical validation of this method (in terms of consistency) is now available through the above framework on WCUD arrays.

We take a low-discrepancy sequence $a^{(1)},\ldots,a^{(N)} \in [0,1)^d$ and a random permutation τ on the integers $\{1,2,\ldots,N\}$. The random sequence with elements $u^{((i-1)d+j)} \triangleq a_j^{(\tau(i))}$ for all $i \in \{1,2,\ldots,N\}$ and $j \in \{1,2,\ldots,d\}$ is one of a triangular array of sequences of lengths Nd for fixed d and all positive integers N.

The following theorem is due to [42]:

Theorem 4.4.1. *Suppose D_N^* is the discrepancy of the sequence $a^{(1)},\ldots,a^{(N)} \in [0,1)^d$ in d dimensions. Then for arbitrary dimension s, the sequence $z^{(1)},\ldots,z^{(\tilde{N})} \in [0,1)^s$*

obtained by

$$z^{(i)} = \mathbf{u}_{((i-1)s+1):(is)} \tag{4.4.1}$$

for $\tilde{N} = \lfloor (Nd)/s \rfloor$ *satisfies, for arbitrary* $z \in [0,1)^s$, $\epsilon > 0$*:*

$$P\Big(\delta_{\tilde{N}}(z; z^{(1)}, \ldots, z^{(\tilde{N})}) > \epsilon\Big) = O(N^{-1} + D_N^*). \tag{4.4.2}$$

Although the result is not so surprising, the proof is quite complex, and so it is not restated here. The proof uses the Markov inequality as its final step, as it bounds the mean of the squared local discrepancy.

Corollary 4.4.2. *The random triangular array emerging from Liao's method is WCUD. Consequently, weak consistency of MCQMC estimates holds when Liao's method generates the driving sequence.*

Proof. By applying Lemma 4.2.3 to the result of the above theorem, the WCUD property is verified, and so Theorem 4.3.2 yields weak consistency. □

Notice that there is nothing that requires the dimension of the permuted points to match the dimension of the MCMC algorithm. The natural impulse is to match these dimensions, as the variates in each column of the variate matrix are more evenly spaced than is expected with IID sampling.

4.5 Random Number Generators Revisited

The goal of Liao's method was to provide an overall balance on the variate rows used to update each step but make the successive variate rows look essentially independent. To fully carry over the benefits of QMC sampling to the Markov chain case, we would like to improve on pseudo-independence in the relationship between successive variate rows, and such a sequence would have low s-dimensional discrepancy for values of s greater than d. In other words, the s-tuples formed from consecutive s-blocks of the entire sequence would have an approximate uniform distribution on $[0, 1)^s$. This criterion is equivalent to the criterion used to indicate that a random number generator is "good." For example, recall the equidistribution property of the Mersenne Twister of [28], such that for all $s \leq 623$, the 32 leading bits of all components of an s-block evenly cover $\{0, 1\}^{32s}$. We must run through the full period to see this even distribution; certainly we have no interest in running our Metropolis-Hastings chain to a length anywhere near $2^{19937} - 1$.

So the candidates for MCQMC endorsed by this line of reasoning are sequences of much smaller periods that may pass as random number generators in smaller capacity. The random number generator of period N produces a sequence of the form $u^{(1)}, \dots, u^{(N)}, u^{(1)}, \dots$. Clearly as a single sequence, this is not CUD. However, we can look at classes of random number generators of increasing sizes, and in cases where an array-CUD property holds, the use of the full random number generator output in a MCQMC algorithm is justifiable.

Recall the variance matrix given in (3.1.1). Liao's method is expected to provide some variance reduction due to a "balance in the columns," whereby the set of variates used to update each component is evenly distributed across $[0, 1)$. If the full output of a small random number generator with period N is used exactly once in populating the variate matrix (discarding the $N(\text{mod } d)$ variates left over), the columns of the variate matrix see at most N/d values from the generator, and the balance in the columns is not notably better than that via IID sampling. So the actual sequence that should be used is the full output from the random number generator repeated exactly d

times. Assuming that N and d are relatively prime (the case in which this is false will be discussed later), the columns of the variate matrix are $u^{(i)}, u^{(i+d)}, \ldots, u^{(i+(N-1)d)}$, where the indices are equivalent to their mod N residue. These columns include each value in the generator exactly once, and so the notion of balance in the columns is upheld here. If the array-CUD property applies to a class of generator sequences, Slutsky's Theorem and Lemma 4.2.2 verify that the array-CUD property applies to the class of these generator sequences repeated d times.

An additional benefit is that the s-tuples of consecutive values in the columns include each of $(u^{(i)}, u^{(i+d)}, \ldots, u^{(i+(s-1)d)})$ exactly once (assuming a sort of "wrap-around" from the bottom of the matrix to the top). Thus the last s variates used to update a given component have a better-than-IID balance in $[0,1)^s$ as well, assuming the random number generator is "good." This notion will be made more concrete for specific classes of generators.

4.5.1 The Korobov Lattice

One well-known recursive pseudorandom number generator with desirable properties for MCQMC is the multiplicative congruential generator (MCG). For a prime number M, the generator assumes all values in the set $\{1/M, 2/M, \ldots, (M-1)/M\}$ exactly once. The order of the output sequence is determined by powers of an integer a, for some $1 \leq a \leq (M-1)$ such that

$$a^n \equiv 1 (\text{mod } M) \tag{4.5.1}$$

has exactly one solution in n $(n = M-1)$. Such a value a is known as a primitive root of the prime M. It is well-known (see [3], e.g.) that the number of primitive roots for M is positive; more specifically, it is equal to $\phi(M-1)$, where ϕ is Euler's totient function (which maps an integer n to the number of positive integers k less than n such that $\gcd(k,n)=1$).

For the sequence $r^{(n)} = a^n (\text{mod } M)$, which is equivalent to the recursion

$$r^{(n)} = ar^{(n-1)}(\text{mod } M), \qquad r^{(0)} = 1, \tag{4.5.2}$$

the sequence formed by $x^{(n)} = r^{(n)}/M$ hits each value in $\{1/M, 2/M, \ldots, (M-1)/M\}$ exactly once before repeating. So we take the sequence $x^{(1)}, \ldots, x^{(M-1)}$ as the full output sequence of the generator.

A look at the consecutive s-tuples in this sequence for $s < M$ reveals that each is the mod 1 residue of an integer multiple of the vector $(1, g, g^2, \ldots, g^{s-1})$, and so the full set of s-tuples from the generator sequence forms an integration lattice (introduced in Section 2.2.1). The use of these sequences then guarantees some notion of even spacing in s dimensions for all s, although, as is evident in two-dimensional projections, some lattices are better than others.

For an infinite subset of primes, we would like to define, on each member of this set, a generator sequence of this type such that we can verify an array-CUD property on the resulting collection of sequences. A simultaneous discrepancy bound in dimension s and period $N = M-1$ is useful in verifying this property. Niederreiter [31] derives the result that for a fixed choice of s and M, at least one primitive root exists such that the s-dimensional discrepancy of the resulting sequence satisfies:

$$D^*_{M-1} < \frac{1}{M-1}\left(1 + \frac{(M-2)(s-1)}{\phi(M-1)}\right)\left(\frac{2}{\pi}\log M + \frac{7}{5}\right)^s. \tag{4.5.3}$$

The totient function obeys the following limit ($\gamma \approx 0.5772$ is the Euler-Mascheroni constant):

$$\liminf_{n\to\infty} \frac{\phi(n)}{n} \log(\log(n)) = e^{-\gamma}, \tag{4.5.4}$$

so for some MCG sequence on prime M above a threshold $M_0 > 0$:

$$D^*_{M-1} < \frac{As}{M} \log(\log(M))(\log M)^s, \tag{4.5.5}$$

where A is a positive constant. Note that the choice of generator for each s is not necessarily the same. Still, we can use this fact to verify an array-CUD property.

Theorem 4.5.1. *For an infinite subset $\mathcal{M}$ of the primes, one can define for each $M \in \mathcal{M}$ a multiplicative congruential generator such that the collection of full output sequences from these generators is array-CUD.*

Proof. For each M, we choose a dimension s such that $s(M) = o([\log M/\log\log M]^\alpha)$ for some constant $\alpha < 1$, and choose generators to satisfy (4.5.3) for $M, s(M)$. Under this $s(M)$, for large enough M we have the inequality

$$(\log M)^s < M^{\left[\left(\frac{\log\log M}{\log M}\right)^{1-\alpha}\right]} < M^\beta \tag{4.5.6}$$

for an arbitrary constant $\beta \in (0,1)$. Thus for this choice of s, the right side in (4.5.5) decays to 0 with rate $M^{-1+\epsilon}$. Now for arbitrary fixed dimension $\tilde{s}$, the sequence of $\tilde{s}$-tuples

$$(u^{(1)}_{M-1}, \dots, u^{(\tilde{s})}_{M-1}), \dots, (u^{(M-\tilde{s})}_{M-1}, \dots, u^{(M-1)}_{M-1}) \tag{4.5.7}$$

formed by the generators above have discrepancy that decays to 0 at rate $M^{-1+\epsilon}$ by Lemmas 4.2.1 and 4.2.2. □

It should be noted that, in the above proof, the implied constant in the rates of discrepancy decay for different dimensions $\tilde{s}$ is different, and there is no notion of uniform discrepancy decay in all dimensions. For s that grows with M at least as quickly as $\log M/\log\log M$, the discrepancy bound in (4.5.5) is useless, as the $(\log M)^s$ term grows too quickly. For our Metropolis-Hastings sampler, we need a generator of period $N = M - 1$ to run the chain for N steps, and for practical sample sizes, the resulting value of s for which the above bound is useful is actually quite small. It should be pointed out that this bound is usually quite conservative. Still, for purposes of establishing consistency via the array-CUD property, it is sufficient.

To say that a function is $o([\log M/\log\log M]^{\alpha})$ for some $\alpha < 1$ is equivalent to saying that the function is $o([\log M]^{\beta})$ for some $\beta < 1$; this latter expression may seem simpler, but in the calculation above and in future calculations, it is easier to work with the former.

The choice generator of a desired size for MCQMC is motivated by minimizing discrepancy for all dimensions up to some tolerable $s_{\max}$. The actual computation of star discrepancy of a point set is laborious and becomes far more difficult as the dimension of the point set grows. Alternatively, one can compute the L_2 norm of the anchored local discrepancy rather than the star discrepancy (which is the L_∞ norm). This mean square discrepancy has a simple calculation due to [43]; in higher dimensions, the calculation time is accelerated due to a recursive formula in [13]. While the generator with optimal mean square discrepancy is not necessarily the one with the best star discrepancy, it is safe to expect that the rank of sequences based on this criterion is not much different from the true rank. In independence sampling, alternate notions of discrepancy can also be used to bound integration error (see [14]), although in the Markov chain case, these notions may be difficult to use in an analog of Theorem 3.3.3.

As previously discussed, a more popular criterion used to evaluate the quality of an integration lattice is one of several functions of the lattice called a "figure of merit." The figures of merit discussed in [33] and [40] are easier to compute and relate to an upper bound on the discrepancy. A table of Korobov lattices for a series of primes (close to 2^n for different n) whose 8-blocks and 32-blocks have optimal figures of merit (among Korobov lattices of the same size) is given in [18].

4.5.2 Linear Feedback Shift Register Generators

The intuitive value of a series of lattice points in approximating uniformity is the homogenous spacing of the points. Another intuitive approach to approximating uniformity by a point set is that of equidistribution, in which the unit hypercube is partitioned into subcubes of equal size, and the point set puts the same number of points into each subcube. If the subcubes of $[0,1)^s$ have side length 2^{-k}, the placement of a point in a subcube is uniquely determined by the k leading binary digits of the s coordinates of the point. The goal of random number generators based on bit recursion is the even distribution of the leading binary digits of successive points.

The linear feedback shift register generator constructs its variates from an underlying sequence $b_1, b_2, \ldots$ of zeroes and ones. For some choice of integers $a_1 < a_2 < \ldots < a_k$, the sequence is advanced by the recursive formula

$$b_i = \left(\sum_{j=1}^{k} b_{i-a_j} \right) (\text{mod } 2). \tag{4.5.8}$$

As the future of the sequence is completely determined by the last $m = a_k$ values and there are only 2^m possible choices for these values, the sequence has period at most 2^m. As a set of m zeroes yields an all-zero sequence, the maximal period is only

$2^m - 1$.

The existence of parameters $a_1, \dots, a_k$ that create a sequence of period $2^m - 1$ is guaranteed through the following well-known theorem (see [38], e.g.):

Theorem 4.5.2. *The resulting sequence from the recurrence relation (4.5.8) has period $2^m - 1$ (for $m = a_k$) if and only if the polynomial*

$$1 + \sum_{j=1}^{k} z^{a_j} \tag{4.5.9}$$

is a primitive polynomial over the Galois field with two elements. There are $Z_m \triangleq m^{-1}\phi(2^m - 1)$ such primitive polynomials of degree m over GF(2), and so there are Z_m degree m recurrence relations whose corresponding sequence has period $2^m - 1$.

Any sequence which achieves this maximal period has every m-block of bits in the set $\{0,1\}^m \setminus \{\mathbf{0}\}$. Thus for any integer g such that $\mathrm{gcf}(g, 2^m - 1) = 1$, the sequence defined by

$$u^{(i)} = \sum_{j=1}^{B} b_{(i-1)g+j} 2^{-j} \tag{4.5.10}$$

has $2^m - 1$ distinct values which each lie in a different interval in the partition of $[0,1)$ into intervals of length 2^{-m}. The lowest interval has no entries, as the leading m bits are never all 0. B is the total number of bits in the number and is usually taken to be 32 or 64. We call this sequence a linear feedback shift register (LFSR) sequence. In the special case where $k = 2$ and the generator corresponds to a primitive trinomial, this generator is also known as a Tausworthe generator.

The relationship between successive points is not necessarily one that approximates uniformity well. For example, if the offset parameter g is 1, then the resulting pairs $(u^{(i)}, u^{(i+1)})$ lie in one of four rectangles, each with volume 1/8. (The second bit of $u^{(i)}$ is the first bit of $u^{(i+1)}$.) We would like to choose g such that there is a better cover of the hypercube by consecutive s-tuples. Recall the notion of equidistribution discussed at the beginning of this section. A more formal definition as it applies to these LFSR sequences is the following.

Definition 4.5.3. An LFSR sequence is 2^{-l}-equidistributed in s dimensions if, upon the partition of $[0,1)^s$ into subcubes of side length 2^{-l}, the number of s-tuples from blocks of s consecutive outputs in each subcube is the same, with the exception of the subcube containing the origin, whose count is one fewer.

Clearly 2^{-l}-equidistribution can only hold for $l \leq \lfloor m/s \rfloor$. Given s, a significant fraction of offsets g relatively prime to $2^m - 1$ seem to satisfy equidistribution for $l = \lfloor m/s \rfloor$.

Given an LFSR sequence of length $N = 2^m - 1$ and a 2^{-l}-equidistribution property in s dimensions for $l = \lfloor m/s \rfloor$, the local discrepancy is $1/N$ on every box whose corner opposite the origin has coordinates which are integer multiples of 2^{-l} (call this collection of boxes $\mathcal{B}$). The volume of any box differs by at most $s2^{-l}$ from that of a set in $\mathcal{B}$ which is either a superset or subset of the box. Thus the star discrepancy satisfies

$$D_N^* \leq \frac{1}{N} + s2^{-m/s+1} < \frac{1}{N} + 2sN^{-1/s}. \tag{4.5.11}$$

This bound is not strong, but we can verify an array-CUD property for a collection of LFSR sequences from this.

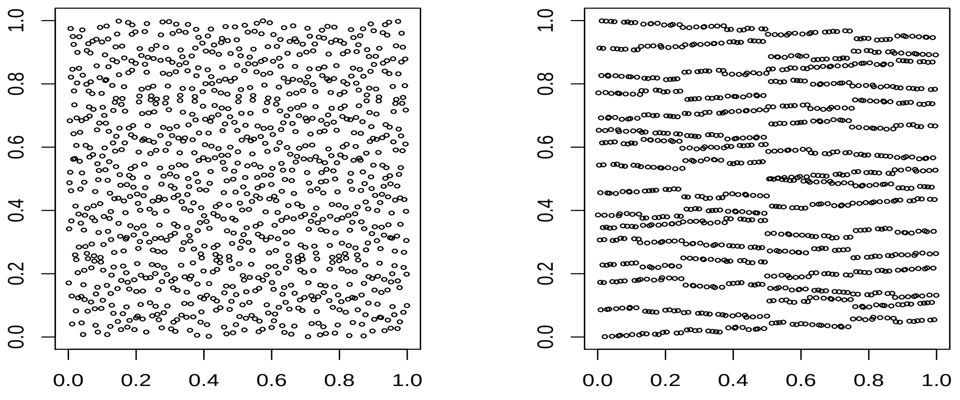
Theorem 4.5.4. *Let $s(N)$ be an integer function with growth $o(\log N/\log\log N)$. For each integer m, define an LFSR sequence of size $N = 2^m - 1$ such that the sequence is $2^{-\lfloor m/s(N)\rfloor}$-equidistributed. The collection of LFSR sequences is array-CUD.*

Proof. For $s(N)$ above and N sufficiently large, the right side of (4.5.11) is bounded above by $(\log\log N)^{-1}$, and so it decays to 0 as N and $s(N)$ grow to ∞. For a specific s, the s-dimensional discrepancy decays to 0 by the above result and Lemmas 4.2.1 and 4.2.2, and so the collection is array-CUD. $\square$

There are many choices of primitive polynomial and offset which satisfy the equidistribution condition. Because finding star discrepancy is tedious for high dimensions, an exhaustive search for optimal sequence of a certain size in terms of discrepancy in a certain dimension becomes far too computationally expensive quickly. It is again easier to look at mean square discrepancy, and still, a nonexhaustive search for good mean square discrepancy in a few choices of dimension yielded the sequences used in the examples in Chapter 6.

The decision of whether to use an LFSR sequence or an MCG sequence is not clear. Beyond the specific dimension by which the MCG is selected, the s-blocks still form a lattice, although a lattice may still have large gaps (recall Figure 2.1). The equidistribution property of the LFSR can only hold for a small set of dimensions and cube sizes; beyond this, the LFSR may also have large gaps, as seen in Figure 4.1. The discrepancy of a sequence and the integration error resulting from its use, even in an independent sampling scheme, are not always well-correlated. A result in [27] states that a lattice of size N on the s-dimensional hypercube lies in at most $(s!N)^{1/s}$ parallel hyperplanes. For a function f with large variability in the transverse direction, the MCG points would not provide substantial improvements in estimate accuracy over random sampling. The results of the search for good MCG sequences have been well-documented, but there is little literature to endorse specific LFSR sequences in terms of optimal discrepancy; however, results that are at least comparable to, and in some cases substantially better than those attained using MCG sequences emerge using the

Figure 4.1: Projections of successive values from an LFSR generator, with equidistribution in 2-dimensions holding on the left. From an LFSR generator with multipliers $(3, 10)$ and offset 52, these are the plots of $(u^{(i)}, u^{(i+k)})$ for $k = 2$ on the left and $k = 47$ on the right.



best LFSR sequences from a nonexhaustive search of a group of sequences.

Chapter 5

Algorithm Implementation

The results of the previous chapter indicate that an MCQMC algorithm which replaces IID sampling with points drawn from a multiplicative congruential generator or a linear feedback shift register generator is valid in an array-consistency sense. This chapter will include details of a general strategy for populating the variate matrix used in the simulation, randomizing the variates, and beginning the sampler. Throughout, the notion of consistency through an array-WCUD property will be preserved.

5.1 Populating the Variate Matrix

The discussion in Section 4.5 suggests a strategy for the inclusion of a full-period generator output (with length N) into the variate matrix (3.1.1), where the sequence is repeated d times. (Recall that this d is the dimension of the algorithm itself and is fixed.) This strategy assumes that d and N are relatively prime, such that each value of the generator appears in each column of the variate matrix exactly once. In the instance where $\mathrm{gcf}(d, N) > 1$, some adjustment of this sequence repetition is necessary so that the balance in the columns of the variate matrix is preserved.

For $b = \mathrm{gcf}(d, N)$, the method used in simulations in [37] and [42] uses a series of $b-1$ skips that occur after every N/b rows, such that for generator sequence $u^{(1)}, u^{(2)}, \ldots, u^{(N)}$, the variate matrix appears as such:

$$\begin{bmatrix} u^{(1)} & u^{(2)} & \ldots & u^{(d)} \\ u^{(d+1)} & u^{(d+2)} & \ldots & u^{(2d)} \\ \vdots & \vdots & \ddots & \vdots \\ u^{((N/b)d-d+1)} & u^{((N/b)d-d+2)} & \ldots & u^{((N/b)d)} \\ u^{(2)} & u^{(3)} & \ldots & u^{(d+1)} \\ u^{(d+2)} & u^{(d+3)} & \ldots & u^{(2d+1)} \\ \vdots & \vdots & \ddots & \vdots \\ u^{((N/b)d-d+2)} & u^{((N/b)d-d+3)} & \ldots & u^{((N/b)d+1)} \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ u^{(b)} & u^{(b+1)} & \ldots & u^{(d+b-1)} \\ u^{(d+b)} & u^{(d+b+1)} & \ldots & u^{(2d+b-1)} \\ \vdots & \vdots & \ddots & \vdots \\ u^{((N/b)d-d+b)} & u^{((N/b)d-d+b+1)} & \ldots & u^{((N/b)d+b-1)} \end{bmatrix} \tag{5.1.1}$$

In the above notation and in similar expressions in this chapter, we define $u^{(i)} = u^{(j)}$ for $i \equiv j \bmod(N)$. Although this scheme places every output of the generator in each column exactly once, the s-tuples formed by consecutive values in the columns of the variate matrix do not correspond to $(u^{(i+k_1)}, u^{(i+k_2)}, \ldots, u^{(i+k_s)})$ for fixed values $k_1, \ldots, k_s$. For example, in the MCG case, the consecutive s-tuples do not form a lattice, as shown in Figure 5.1. To preserve some notion of approximate uniformity among the s-tuples that govern s successive updates to a component, the skips in the sequence must be the same between every pair of rows. An improved strategy is to find the smallest integer $y \geq d$ such that $\mathrm{gcf}(y, N) = 1$, and form the variate matrix in the following fashion:

$$\begin{bmatrix} u^{(1)} & u^{(2)} & \dots & u^{(d)} \\ u^{(y+1)} & u^{(y+2)} & \dots & u^{(y+d)} \\ u^{(2y+1)} & u^{(2y+2)} & \dots & u^{(2y+d)} \\ \vdots & \vdots & \ddots & \vdots \\ u^{((N-1)y+1)} & u^{((N-1)y+2)} & \dots & u^{((N-1)y+d)} \end{bmatrix} \tag{5.1.2}$$

This strategy maintains a balance among the s-tuples in the columns. As the random number generator sequence is selected for its optimal properties in small dimensions, $y - d$ (the number of skips between rows) should remain small. To keep $y - d$ small such that the array-CUD property is preserved, nothing need be done for a collection of array-CUD LFSR sequences, but some primes should be avoided in the MCG case. These cases will be treated separately below.

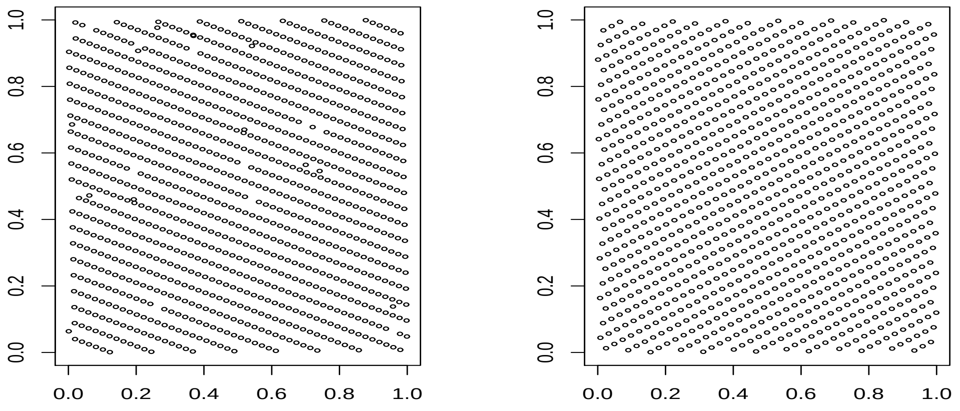
5.1.1 The LFSR Case

The preservation of the array-CUD property with generator skips is verified through the following theorem.

Theorem 5.1.1. *For a fixed positive integer d, let y_i be the smallest value $\geq d$ such that gcf$(y_i, N_i) = 1$. If for all sequence lengths N_i in a CUD triangular array, the value y_i is bounded above by some constant K, then the sequences*

$$u_{N_i}^{(1)}, u_{N_i}^{(2)}, \dots, u_{N_i}^{(d)}, u_{N_i}^{(y_i+1)}, u_{N_i}^{(y_i+2)}, \dots, u_{N_i}^{(y_i+d)}, u_{N_i}^{(2y_i+1)}, \dots, \dots, u_{N_i}^{((N_i-1)y_i+d)}$$

Figure 5.1: Left is lag plot of successive updates using (5.1.1) on MCG with $M = 1021, a = 65, d = 12$. Right is the same, using (5.1.2). 12 is a factor of 1020, and so 12 points are out of place in the lattice on the left.



form a CUD array.

Proof. For arbitrary dimension s, the s-dimensional discrepancy of the above is bounded above by the $\lceil sK/d \rceil$-dimensional discrepancy of the corresponding generator sequence repeated d times without skips, plus an error term for end values. This bound is achieved by an analogous proof to that of Lemma 4.2.1, with the error term of size $O(1/N)$ by Lemma 4.2.2. So as the original sequence collection is array-CUD, the s-dimensional discrepancies of these sequences decay to 0. As s is arbitrary, the sequences with skips are still array-CUD. $\square$

The period of each LFSR generator sequence is odd, and so any power of 2 is relatively prime to the sequence length. Thus for an algorithm dimension d, the skip $y - d$ is less than d for any LFSR sequence. Thus the array-CUD property is preserved for the collection of LFSR sequences by Theorem 5.1.1 with $K = 2d$.

5.1.2 The MCG Case

For a prime M, the period of an MCG sequence with base M is $M-1$, which is an even composite number for any $M > 3$. As N increases (algorithm dimension d is fixed), the smallest value $y \geq d$ relatively prime to N has $\limsup_{N\to\infty}(y/\log N) > 0$. So some thinning of the prime numbers is necessary. For a specific dimension d, we select a "threshold" prime number y_0 at least as large as d. For $M > y_0$, the smallest value $y \geq d$ relatively prime to $M-1$ is at most y_0 if M is not congruent to 1 (mod y_0); thus y_0 can serve as the constant K in Theorem 5.1.1 if each MCG sequence with prime base M congruent to 1 (mod y_0) is discarded from the CUD array.

More rigorously, we can partition the set of primes bigger than y_0 into $y_0 - 1$ subsets $A_1, \dots, A_{y_0-1}$ such that

$$M \in A_i \qquad \text{iff} \qquad M \equiv i (\text{mod } y_0). \tag{5.1.3}$$

An array-CUD sequence based on MCG generator outputs is still array-CUD with skips added if the sequences based on primes in the set A_1 are discarded, by application of 5.1.1 with $K = y_0$. The sparseness of the subsets in the above partition is not a problem, as a result of Vallée Poussin [8] says that, for all i:

$$\lim_{N\to\infty} \frac{\#\big|A_i \cap \{1, 2, \dots, N\}\big|}{N/\log N} = \frac{1}{y_0 - 1}. \tag{5.1.4}$$

In practice, the application of a generator sequence of period $M-1$ for which y is large should be avoided in favor of a sequence of roughly the same length for which y is small.

5.2 Randomization

The randomization of the values in the variate matrix is an important tool to minimize the bias of the estimation procedure. In creating a good randomization scheme, the goal is to make points marginally uniform while preserving the low-discrepancy property; specifically, we want to preserve an array-WCUD property. It seems natural to apply the same randomization to each row in the variate matrix (3.1.1), such that the d-dimensional point in each row is marginally distributed $U[0,1)^d$, but the balance in the columns is still preserved. This can be achieved by applying a common randomization to every univariate value in a single column of the variate matrix, with the condition that the randomizations applied to different columns are independent.

5.2.1 The LFSR Case

The verification of an array-CUD property of a collection of optimally equidistributed LFSR generators relied on this equidistribution. Consequently, a randomization that preserved the notion of equidistribution would be a good choice. Such a randomization that does this is an additive bit scramble, defined below.

Definition 5.2.1. For arbitrary $x \in [0,1)$, take the unique binary representation $x = 0.x_1x_2x_3\ldots$ such that $x_i = 0$ infinitely often. The additive bit scramble creates a random binary sequence $(a_1, a_2, \ldots)$ whose ith coordinate is 0 or 1 with probability $1/2$ (independently of the values of other coordinates), and thus maps x to the number whose ith binary digit is $x_i + a_i \pmod 2$. Equivalently, each digit of x is flipped independently with probability $1/2$. The distribution of x under this random map is uniform on $[0,1]$.

A common additive bit scramble is applied to each column of the variate matrix, with the scrambles on separate columns independent. Now each row is marginally uniform, and an array-WCUD property still holds.

Theorem 5.2.2. *For an array-CUD collection of LFSR generator sequences defined in Theorem 4.5.4 repeated with regular skips as in (5.1.2), define the random array by a series of d independent additive bit scrambles applied to each sequence (such that the nth value in the sequence gets the jth randomization, where $n \equiv j \pmod d$). This array is WCUD.*

Proof. Take $s(N) = o([\log N/\log\log N]^{\alpha})$ for some constant $\alpha < 1$. For such s, $s^2 N^{-1/s} \to 0$. For the original generator sequence of length $N = 2^m - 1$, take $k(N)$ the largest integer power of 2 such that $s(N) \geq 2k(N)d$. (Recall that there are at most d skips per row in the construction (5.1.2).) For $l = \lfloor m/s \rfloor$ the nonoverlapping $k(N)d$-tuples in the sequence are 2^{-l} equidistributed in the sense that every subcube in a partition of $[0, 1)^{k(N)d}$ into subcubes of side length 2^{-l} has the same number of points, except one random cube which has one fewer point. By similar reasoning to that in Theorem 4.5.4, the nonoverlapping discrepancy of the point set of size N in dimension $k(N)d$ goes to 0.

For arbitrary constant k_0 which is an integer power of 2, when N is sufficiently large, $k_0 < k(N)$. For some N sufficiently large, the set of nonoverlapping $(k_0 d)$-tuples can be partitioned into subsets such that the ith point is mapped to a subset indexed by the residue of i mod $(k(N)/k_0)$. From this decomposition, the discrepancy of nonoverlapping $(k_0 d)$-tuples is bounded above by the sum of $k(N)/k_0$ terms at most equal to the $k(N)d$-discrepancy, and so the discrepancy of the nonoverlapping $(k_0 d)$-tuples is at most $s(N)k(N)N^{-1/s(N)}$, which goes to 0 as specified above. By Lemma 4.3.3, the array is WCUD. □

In practice, only the first 32 or 64 bits of the number are recorded, and so the randomization is only carried out to the same number of bits.

5.2.2 The MCG Case

As we want to preserve the regular spacing between points, the natural randomization here is the Cranley-Patterson rotation where every row is translated by a common variable U uniformly distributed on $[0,1)^d$. This is equivalent to independent univariate Cranley-Patterson rotations on each column. To show that this preserves the array-WCUD property, we first start with a Lemma relating types of discrepancy.

Definition 5.2.3. For $0 \le a < b \le 1$, define the wrap-around interval [b,a] to be $[0,a] \cup [b,1]$. A wrap-around box B takes the form $\prod [a_i, b_i]$, where the interval is wrap-around if $a_i > b_i$ and traditional otherwise. The wrap-around discrepancy D_n^W takes the supremum of the absolute difference between empirical measure and Jordan measure over all wrap-around boxes.

Lemma 5.2.4. *For the same point set in* $[0,1)^d$,

$$D_n^* \le D_n^W \le 4^d D_n^*. \tag{5.2.1}$$

Proof. The first inequality is clear, as all anchored boxes are wrap-around boxes. All simple unanchored boxes have local discrepancy at most $2^d D_n^*$ by (A.2.6). All wrap-around boxes are the union of at most 2^d simple unanchored boxes, and so every wrap around box has local discrepancy at most $2^d \cdot 2^d D_n^*$. The result follows. $\square$

Note that the wrap-around discrepancy of a point set with a common Cranley-Patterson rotation applied to every point does not change. We use this fact to verify an array-WCUD property.

Theorem 5.2.5. *For an array-CUD collection of MCG generator sequences defined in Theorem 4.5.1 repeated with regular skips as in (5.1.2) and thinned to primes not congruent to 1 mod y_0, define the random array by a series of d independent Cranley-Patterson rotations, where the nth value in the sequence undergoes the jth rotation if $n \equiv j \pmod d$. Then the array is WCUD.*

Proof. Take $s(N) = o([\log N/\log\log N]^{\alpha})$ as before, where $\alpha < 1$. Define $k(N)$ the largest integer power of 2 such that $s(N) > k(N)y_0$. The nonoverlapping $k(N)d$-tuples have, by (4.5.5) and (5.2.1), discrepancy at most $As(N)N^{-1}(\log\log N)(4\log N)^{s(N)}$. Note that $4^{s(N)}$ is $o(N^{\log 4/\log\log N})$, and so this discrepancy is still $O(N^{-1+\epsilon})$ as N (and $s(N)$) grow to ∞. The remainder of the argument follows as in the proof of Theorem 5.2.2 (note $k(N)O(N^{-1+\epsilon}) = O(N^{-1+2\epsilon})$ for arbitrary ϵ), verifying that the array is WCUD. $\square$

5.2.3 The Issue of Bias

It should be noted that even if we assume the starting value $X^{(0)}$ of the Metropolis Algorithm is π-distributed (i.e., we are already in a stationary distribution), the randomizations above do not make the resulting estimate unbiased. The distribution of successive variates in the same column of the matrix is not uniform under the randomization, and so the resulting chance that the path moves in any direction given its previous move is not the same as with independent sampling.

For example, take the s-tuple of the first s variates used to update the first component. In the MCG case with the Cranley-Patterson rotation, this s-tuple is uniformly distributed on a finite set of line segments. In the LFSR case with the additive bit scramble, the s-tuple can lie in only 2^l of the 2^{sl} subcubes in a partition of the cube. Yet overall, the set of directions each particle goes in each block of s steps is still well-balanced after the simulation is complete, and so we do not expect a problematic bias.

One idea to make bias less worrisome is to apply independent transformations to each element in a block of m rows, where m is some small number bigger than 1. If m is relatively prime to the sequence period N, by running through the generator dm times, with the same block of dm independent transformations applied to all N nonoverlapping (dm)-blocks in the variate matrix, the bias is contained in the approximation error of (4.1.1). This involves the same work as the use of a generator of sequence length $\approx Nm$, which may have much nicer distribution among its successive values, and so the marginal return of this step to reduce bias may not be worthwhile.

Specifically in the LFSR case, one can generalize the additive bit scramble to a linear bit scramble, where we define a matrix A whose entries are 1 on the diagonal, 0 above the diagonal, and 0 or 1 independently with probability 1/2 below the diagonal; and B an additive bit vector as before. For two values $x = 0.x_1x_2x_3\ldots$ and $y = 0.y_1y_2y_3\ldots$, if $j = \min\{n \geq 1 : x_n \neq y_n\}$, then the range of the random map $(Ax + B, Ay + B)$ is a set of 2-dimensional measure 2^{-j}, as (x_i, y_i) is uniformly distributed over $\{0, 0\}, \{0, 1\}, \{1, 0\}, \{1, 1\}$ for all $i > j$. Under the additive bit scramble alone, the range of $(x + B, y + B)$ had 2-dimensional measure 0. Equidistribution is also preserved under this map. The marginal distribution of s-tuples with the same randomization applied componentwise could approach uniformity even more if the upper diagonal were not restricted to 0, but in this case, equidistribution is not preserved, and so this should certainly be avoided.

5.3 Acceptance/Rejection Sampling

The strategy for use of CUD and array-CUD sequences in a Metropolis-Hastings algorithm has assumed that at most d variates are needed to generate the next step in the chain, where d is determined beforehand. In the case where acceptance/rejection sampling (as described in Table 2.1) is necessary to draw from proposal distributions in the Metropolis-Hastings algorithm (or conditional distributions in the Gibbs sampler), it is not possible to put a finite bound on the number of variates needed to generate a sample point. (The number of variates needed is twice a geometrically distributed variable.)

In the experiment of randomizing QMC points to run a Gibbs sampler in [23], Liao drew from a Gamma distribution using acceptance/rejection sampling. His strategy was to run two iterations of an acceptance/rejection algorithm using four coordinates of a d-dimensional QMC point, and if two rejections occurred, a sequence assumed to be IID was used to finish the acceptance/rejection algorithm. It is not easy to prove an analogous theorem to Theorem 3.3.3 without some sort of regular inclusion of IID sampling after a fixed number of rejections using a CUD sequence.

A Metropolis-Hastings algorithm that runs up to k steps of an acceptance/rejection sampler with points from a CUD sequence before using IID points is still weakly consistent, as shown in [42]. The reversion to IID sampling to generate from some distribution is equivalent to drawing a single uniform that corresponds to the CDF of the distribution evaluated at the drawn value. Hence the entire algorithm, assuming k acceptance/rejection steps are necessary, is equivalent to using the variate matrix

$$\begin{bmatrix} u^{(1)} & u^{(2)} & \dots & u^{(d-1)} & v^{(1)} & \dots & v^{(k)} & u^{(d)} \\ u^{(d+1)} & u^{(d+2)} & \dots & u^{(2d-1)} & v^{(k+1)} & \dots & v^{(2k)} & u^{(2d)} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ u^{((N-1)d+1)} & u^{((N-1)d+2)} & \dots & u^{(Nd-1)} & v^{((N-1)k+1} & \dots & v^{(Nk)} & u^{(Nd)} \end{bmatrix} \tag{5.3.1}$$

where the $\{u^{(i)}\}$ sequence is WCUD (or from an array-WCUD collection) and the $\{v^{(i)}\}$ are assumed to be IID and independent of the $\{u^{(i)}\}$ sequence. The proof that the sequence formed across the rows is WCUD or array-WCUD appears in [42]. This proof formalizes the intuitive idea that the nonoverlapping $m(d+k)$-tuples must have discrepancy that decays to 0. Lemma 4.3.3 completes the proof.

5.4 The Final Variate Matrix Expression

We assume the following things:

1. We wish to use a length N sequence from a CUD array
2. The Metropolis-Hastings sampler requires d variates and up to k reversions to IID sampling for acceptance/rejection
3. The smallest integer $\geq d$ relatively prime to N is y
4. The randomizations $\psi_1, \dots, \psi_d$ are independent

From this, the general format of the variate matrix to be used for MCQMC is given below. Note that for both generators studied, the point set is incomplete in the sense that the leading bits in an LFSR sequence point are never all 0, and the origin of the integration lattice from an MCG sequence is missing. The inclusion of the origin at the beginning of the sampling scheme makes the one-dimensional balance in the columns more complete and is recommended. As it is one point, its inclusion does not affect results pertaining to the CUD nature of the arrays. In the following, the sequence $\{v^{(i)}\}$ is an IID uniform sequence used only when an acceptance-rejection algorithm needs further iterations.

$$\begin{bmatrix} \psi_1(0) & \dots & \psi_{d-1}(0) & v^{(1)} & \dots & v^{(k)} & \psi_d(0) \\ \psi_1(u^{(1)}) & \dots & \psi_{d-1}(u^{(d-1)}) & v^{(k+1)} & \dots & v^{(2k)} & \psi_d(u^{(d)}) \\ \psi_1(u^{(y+1)}) & \dots & \psi_{d-1}(u^{(y+d-1)}) & v^{(2k+1)} & \dots & v^{(3k)} & \psi_d(u^{(y+d)}) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \psi_1(u^{((N-1)y+1)}) & \dots & \psi_{d-1}(u^{((N-1)y+d-1)}) & v^{(Nk+1)} & \dots & v^{((N+1)k)} & \psi_d(u^{((N-1)y+d)}) \end{bmatrix} \tag{5.4.1}$$

This final form of the variate matrix is the one used in the simulations whose results appear in the next chapter, unless otherwise indicated in studies that examine the marginal benefit of the variate matrix adjustments described here.

Chapter 6

MCQMC Examples

Throughout this section, we denote the stationary distribution by π and the transition distribution from state x by Q_x. The densities or mass functions of these at state y are $\pi(y)$ and $q(x, y)$. For the Gibbs sampler, the conditional distributions are denoted by $Q(\theta_k|\theta_1, \ldots, \theta_{k-1}, \theta_{k+1}, \ldots, \theta_d)$; for simplicity of notation, we denote the vector of theta except for the kth component by θ_{-k}.

6.1 Toy Problems

Before examining the performance of MCQMC in comparison to regular MCMC in substantial problems, it is interesting to explore various aspects of the process in simple toy problems where the effects of various aspects of the algorithm can be easily seen.

A few simple examples shown here will have a Gaussian target distribution π with known parameters, such that $E_\pi[f(X)]$ is easily computed. The first example is a simple univariate Metropolis-Hastings sampler with proposal distributions either

Table 6.1: Mean Square Error, Random Walk

		$\sigma = 2.4$	$\sigma = 1.2$	$\sigma = 0.5$
$f(x) = x$	IID	4.54e-03	6.28e-03	5.65e-02
	MCG	1.96e-03	1.64e-03	2.03e-03
	LFSR	1.77e-03	2.15e-03	1.78e-03
$f(x) = 1_{\{x>0\}}$	IID	1.11e-03	1.02e-03	2.65e-01
	MCG	6.36e-04	3.71e-04	2.57e-01
	LFSR	5.77e-04	4.53e-04	2.44e-01
$f(x) = x^2$	IID	8.77e-03	1.35e-02	9.98e-01
	MCG	6.01e-03	5.62e-03	1.02e+00
	LFSR	5.74e-03	5.75e-03	9.22e-01

symmetric about the current state or independent of it:

$$\pi = N(0,1), \qquad Q_x = N(x, \sigma^2). \tag{6.1.1}$$

$$\pi = N(0,1), \qquad Q_x = N(0, \sigma^2). \tag{6.1.2}$$

The first sampler is known as the random walk sampler, the second is known as the independence sampler. The parameter σ^2 will affect the rate of proposal acceptance and the rate of decay of dependence on past values. In terms of the mean square error of the resulting estimates, the performance of MCQMC using MCG or LFSR sequences can be compared to MCMC with IID sampling for the two samplers, with a variety of σ values for several functions. The values of σ were chosen to provide a wide range of mixing speeds of the chain. From here onward, we will denote an MCG with prime modulus M and primitive root multiplier a as the (M, a) MCG, and an LFSR with recursion sequence $(a_1, \ldots, a_k)$ and offset g as the $[(a_1, \ldots, a_k), g]$ LFSR. Tables 6.1 and 6.2 compare the (1021,65) MCG and the [(3,10),52] LFSR to IID sampling with 1024 steps.

Table 6.2: Mean Square Error, Independence

		$\sigma = 2.4$	$\sigma = 1.2$	$\sigma = 0.5$
$f(x) = x$	IID	2.75e-03	7.42e-04	9.81e-01
	MCG	5.13e-04	1.09e-04	1.21e+00
	LFSR	5.16e-04	1.35e-04	1.13e+00
$f(x) = 1_{\{x>0\}}$	IID	7.35e-04	2.84e-04	2.66e-01
	MCG	1.44e-04	3.22e-05	2.60e-01
	LFSR	1.01e-04	5.45e-05	2.33e-01
$f(x) = x^2$	IID	3.87e-03	2.47e-02	1.71e+00
	MCG	1.09e-03	1.91e-04	1.36e+00
	LFSR	1.45e-03	1.89e-04	1.11e+00

These results show that the MSE reduction using MCQMC is best in the random walk case for $\sigma = 1.2$. An explanation on the diminished performance of the $\sigma = 2.4$ case is the high number of rejections, such that the autocorrelation of the sequence is high.

The independence sampler also favors MCQMC in the $\sigma = 1.2$ case; note that for $\sigma = 1$ the sampler is a simple Monte Carlo sampler, and so any σ quite close to 1 will behave similarly to this. The MSE reductions for the $\sigma = 2.4$ and $\sigma = 1.2$ samplers range from 1.4 to 3.9 in the random walk case and from 2.7 to 13 in the independence case. The dependence on the past is lower in the independence sampler: note that two chains at different starting values with the same driving sequence will match as soon as an acceptance occurs under the independence sampler, whereas the random walk sampler on these two chains can only move closer together when one chain accepts and the other rejects.

The $\sigma = 0.5$ sampler is highly unstable, rarely reaching the tails of the target density and remaining for long epochs in values of higher magnitude once they are reached. MCQMC does not help this sampler.

Somewhat surprisingly, the advantage of MCQMC over IID sampling does not seem to change much for different functions, including the one with discontinuity. In a Gaussian setting, the mean and variances are the expectations of unbounded functions, but the normal tails decay rapidly, such that bounded functions provide an excellent approximation to these functions. So it is reasonable to expect that MCQMC does best in estimating the mean and worst in estimating the probability of positivity, but the results do not differ significantly. In terms of looking at the image of f over the uniform variates used to generate the sample, the decision to accept or reject proposals in the Metropolis-Hastings sampler introduces discontinuity anyway.

The next toy example is small Gibbs sampler on a joint Gaussian distribution (with Gaussian conditional distributions):

$$\pi = N(\mu, \Sigma),\ Q(\theta_k|\theta_{-k}) = N(\mu_k + \Sigma_{k,-k}\Sigma^{-1}_{-k,-k}(\theta_{-k} - \mu_{-k}), \Sigma_{k,k} - \Sigma_{k,-k}\Sigma^{-1}_{-k,-k}\Sigma_{-k,k}). \tag{6.1.3}$$

As the correlations σ_{jk} for $j \neq k$ increase in magnitude, the autocorrelation of the Markov chain increases as well, and so it is reasonable to expect the advantage of MCQMC to be the best for target distributions with low correlations. The performance of the sampler for a few functions and choices of Σ is shown in Table 6.3 (we keep $\mu = 0$ as the performance of the simulation is not affected by μ). The same MCG and LFSR as in the previous example are used. Each Gaussian distribution is trivariate with marginal variances 1 and covariance terms $(\rho_{12}, \rho_{13}, \rho_{23})$ specified in Table 6.3. For ease of interpretation, the MSE reduction factors of the MCQMC methods versus IID MCMC are given in Table 6.4.

The sampler estimates the mean and variance of θ_1 and the covariance of θ_1 and θ_2 with much greater accuracy by MCQMC in all cases. The cases with lowest

Table 6.3: Mean Square Error, Gaussian Gibbs Sampler

		$(0.7, 0.4, 0.6)$	$(0.3, -0.2, 0.5)$	$(0.95, 0.7, 0.75)$
$f(\theta) = \theta_1$	IID	4.03e-03	1.67e-03	2.04e-02
	MCG	1.81e-05	1.14e-05	4.12e-04
	LFSR	1.74e-05	4.43e-06	2.61e-03
$f(\theta) = \theta_1 \cdot \theta_2$	IID	3.76e-03	1.30e-03	2.48e-02
	MCG	8.50e-04	3.07e-04	1.09e-02
	LFSR	7.20e-05	1.25e-05	1.63e-02
$f(\theta) = \theta_1^2$	IID	4.23e-03	1.90e-03	2.44e-02
	MCG	6.45-04	2.10e-04	1.04e-02
	LFSR	6.39e-05	2.36e-05	1.82e-02

Table 6.4: MSE Reduction Factors, Gaussian Gibbs Sampler

		$(0.7, 0.4, 0.6)$	$(0.3, -0.2, 0.5)$	$(0.95, 0.7, 0.75)$
$f(\theta) = \theta_1$	MCG	22	146	50
	LFSR	24	375	7.8
$f(\theta) = \theta_1 \cdot \theta_2$	MCG	4.4	4.2	2.2
	LFSR	52	104	1.5
$f(\theta) = \theta^2$	MCG	6.6	9.0	2.3
	LFSR	66	79	1.3

correlation see the greatest advantage of MCQMC, and those with highest correlation see the lowest advantage. Here the estimation of the means seems to show the best improvement; interestingly the MCG performance deteriorates much more than the LFSR performance for the covariance and variance estimates. Overall, except in the case of $\rho_{12} = 0.95$, the error reductions in the Gibbs sampler are far more impressive than those in the Metropolis-Hastings samplers above.

6.2 A Bayes Model

An example explored in [23] exhibits promising results for MCQMC in dimensions far larger than the conservative theoretical bounds support. The problem and data come from [11]. Ten pumps experience failures according to independent Poisson processes with rates $\lambda_1, \ldots, \lambda_{10}$. Each λ_i is assumed to have a Gamma distribution with shape parameter $\alpha = 1.802$ and scale parameter β with Gamma prior distribution (shape parameter $\gamma = 0.1$, scale parameter $\delta = 1$). The data recorded are the number of failures s_i of each pump and times t_i over which the number of failures of the pump was monitored (see Table 6.5).

For a rate λ, the number of failures in time t has a Poisson(λt) distribution, and so the distribution of λ_i given β and the data is indepedent of the other λ values and has a Gamma$(\alpha + s_i, \beta + t_i)$ distribution. The distribution of β given all the λ values is independent of the data and has a Gamma$(\gamma + 10\alpha, \delta + \sum \lambda_i)$ distribution. We use these conditional distributions to run a Gibbs sampler whose values converge to the joint posterior distribution.

For Bayesian modeling, the posterior distributions of these parameters are of interest, as well as the construction of estimates of these parameters. The value a that

Table 6.5: Pump failure data

Pump	Failures	Time
1	5	94.32
2	1	15.72
3	5	62.88
4	14	125.76
5	3	5.24
6	19	31.44
7	1	1.048
8	1	1.048
9	4	2.096
10	22	10.48

minimizes $E[(\theta - a)^2]$ over the distribution of θ is the mean, so for the joint posterior distribution π, the Monte Carlo estimates of $E_\pi[\lambda_i]$ and $E_\pi[\beta]$ (by the sample means) will be the parameter estimates. We wish to investigate the square error of the sample means from the Gibbs sampler in estimating the true means of the parameters. The posterior mean is not obtainable in closed form; the variance of the estimates will be explored with a cautious eye towards the potential bias of the MCQMC estimates.

These simulations were run in the program language R. The following MCGs were used: (1021,65), (4093,209), (16381,665). These were taken from [18]. The following LFSRs were used: [(3,10),52], [(1,3,7,9,11,12),29], [(1,2,6,10,11,14),35]. The primitive polynomial was chosen at random from the full list of primitive polynomials of given degree, and then the offset was chosen to minimize mean square discrepancy.

100 replications were conducted of simulations of size $\approx 2^{10}, 2^{12}$ and 2^{14} using pseudorandom Mersenne Twister outputs, randomly permuted lattice points ("Liao method"), MCG sequence points and LFSR sequence values. The sample variances of these 100 estimates are shown in Table 6.6. From these results, the minimum and

maximum (over the eleven parameters) variance reduction factor of each MCQMC method over IID sampling is shown in Table 6.7. Note that the ratio of sample variances, if the sampling distributions are identical, follows an $F_{99,99}$ distribution for two independent samples of size 100. The .95 quantile of the $F_{99,99}$ distribution is roughly 1.4, and so variance reduction factors of size larger than 1.4 are considered statistically significant. All MCQMC estimates yield statistically significant variance reductions in this example.

The tables indicate that the LFSR method yields the lowest variance, with each method exhibiting larger reductions in variance over IID sampling as the sample size increases (indicating an improved empirical error rate decay). The bias of the methods is uncertain, but, assuming that the true mean is somewhere near the mean of the 100 unbiased estimates via IID sampling, a look at the boxplots of estimates shows that bias is likely far smaller than the variance of the estimates by IID sampling. Sensitivity to the quality of sequences in more "important" dimensions makes inference on error rate volatile from a small set of sequences.

Table 6.6 shows that the variance reduction is the largest for the Poisson parameters with larger monitoring periods ($\lambda_1, \lambda_3, \lambda_4$) and smallest for those with smaller monitoring periods (λ_7, λ_8). This is not surprising, as the dependence of the conditional distribution on β is stronger for smaller periods. The multiplicative differences in variance reduction between λ_7 and λ_8 (which have identical data) under the MCG and LFSR methods are consistently around 2, indicating that the quality of the sequences used has an effect on the improvement in performance.

MCQMC is not expected to perform as well in determing other aspects of the target distribution not related to expectation. Appendix B contains histograms of four samples of β obtained by separate Gibbs samplers using each of the IID, MCG and LFSR methods. Not much discernible difference exists between the samples constructed by the various methods; we might expect the CUD samplers to provide an "even" histogram with greater frequency. Still, when the medians of the samples are used as estimates of the medians of the marginal posterior distributions, the variances of these estimates in the $N \approx 2^{10}$ case drop by factors between 4 and 140 for the MCG case and 6 and 170 for the LFSR case. So MCQMC does seem beneficial in determining quantiles as well. Liao already noted the benefits of his method in determining quantiles in [23].

6.3 Probit Regression Model

This model is due to [2] on data from [10]. There are 39 measurements of patient respiration, each of which recorded an indicator Y_i of vasoconstriction and measurements of the volume $X_{i,1}$ of air inspired and the rate $X_{i,2}$ of inspiration. The probit regression model says that

$$P(Y_i = 1) = \Phi^{-1}(\beta_0 + \beta_1 X_{i,1} + \beta_2 X_{i,2}), \tag{6.3.1}$$

where Φ^{-1} is the inverse CDF of the standard Gaussian distribution. To fit this model, latent data values Z_i are introduced where Z_i has Gaussian distribution with mean $\beta_0 + \beta_1 X_{i,1} + \beta_2 X_{i,2}$ and variance 1, and Y_i is the indicator of whether Z_i is positive. The prior distribution on β is noninformative.

Given this setup, the conditional distribution of the β variables given the Z_i values is independent of the response data, and it has a multivariate Gaussian distribution with mean $(X^TX)^{-1}X^TZ$ and covariance $(X^TX)^{-1}$. The Z_i, given Y_i and β, have a truncated distribution which is the Gaussian distribution above restricted to $[0, \infty)$

Table 6.6: Variances of posterior mean estimates, Bayes model

$N \approx 2^{10}$

Parameter	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7	λ_8	λ_9	λ_{10}	β
IID	6.21e-07	9.21e-06	1.89e-06	1.22e-06	9.00e-05	1.63e-05	3.19e-04	4.14e-04	3.74e-04	1.61e-04	9.00e-04
Liao	3.72e-09	4.88e-08	8.23e-09	4.13e-09	9.02e-07	1.05e-07	1.17e-05	1.37e-05	9.34e-06	1.35e-06	1.37e-05
MCG	3.79e-09	4.86e-08	7.86e-09	5.52e-09	7.69e-07	7.76e-08	9.34e-06	1.99e-05	3.92e-06	9.99e-07	1.04e-05
LFSR	1.03e-09	1.36e-08	1.62e-09	7.93e-10	1.80e-07	2.71e-08	7.04e-07	1.32e-06	9.90e-07	3.15e-07	3.14e-06

$N \approx 2^{12}$

Parameter	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7	λ_8	λ_9	λ_{10}	β
IID	1.67e-07	1.90e-06	3.45e-07	3.29e-07	2.79e-05	4.97e-06	7.12e-05	8.88e-05	9.98e-05	4.77e-05	1.64e-04
Liao	2.27e-10	8.53e-09	7.29e-10	3.05e-10	2.14e-07	8.31e-09	2.46e-06	4.14e-06	1.96e-06	1.98e-07	2.00e-06
MCG	2.93e-10	6.64e-09	5.33e-10	3.42e-10	5.06e-08	5.98e-09	9.25e-07	4.81e-07	4.22e-07	8.81e-08	1.90e-06
LFSR	4.53e-11	1.52e-09	1.25e-10	6.58e-11	1.14e-08	1.18e-09	1.01e-07	5.77e-08	4.68e-08	1.48e-08	5.40e-07

$N \approx 2^{14}$

Parameter	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7	λ_8	λ_9	λ_{10}	β
IID	3.96e-08	4.62e-07	8.46e-08	6.95e-08	5.44e-06	1.02e-06	2.18e-05	2.65e-05	3.13e-05	1.07e-05	7.04e-05
Liao	2.48e-11	1.01e-09	5.81e-11	2.30e-11	3.34e-08	1.02e-09	7.57e-07	7.46e-07	4.63e-07	2.52e-08	8.58e-07
MCG	2.20e-11	1.37e-09	4.67e-11	2.67e-11	6.35e-09	6.96e-10	3.73e-08	5.33e-08	2.89e-08	1.09e-08	5.79e-07
LFSR	3.51e-12	4.45e-11	8.06e-12	4.32e-12	8.55e-10	9.12e-11	3.80e-09	2.24e-08	5.22e-09	1.27e-09	9.69e-09

Table 6.7: Minimum and maximum variance reduction factors, Bayes model

	$N \approx 2^{10}$		$N \approx 2^{12}$		$N \approx 2^{14}$	
Method	min VRF	max VRF	min VRF	max VRF	min VRF	max VRF
Liao	27	296	21	1078	29	3016
MCG	21	241	77	961	121	2603
LFSR	286	1543	304	5003	1186	16089

if $Y_i = 1$ and $(-\infty, 0]$ if $Y_i = 0$. These conditional distributions are used to run a Gibbs sampler. The parameters of interest are the regression parameters β_0, β_1, and β_2. Again we look at the estimation of posterior means by the sample means of each parameter, hoping to minimize the square error of these estimates.

These simulations were performed in JAVA with the same MCG and LFSR sequences as were used in the Bayes model. The Colt Package Mersenne Twister [15] was implemented in place of JAVA's insufficient random number generator for the IID sequences.

Reported in Table 6.8 are the variance reduction factors of 300 estimates of the posterior means using the various methods versus using IID sampling. Here our significance threshold is roughly 1.2, taken from the $F_{299,299}$ distribution. The specific choices of sequence seem to have a sizeable effect on the performances of the MCQMC methods. For the LFSR case, the search for a good sequence in terms of discrepancy is far from exhaustive, and so the lower improvement in accuracy for sample size $\approx 2^{10}$ may be ameliorated by a better sequence choice. None of the recommended Korobov lattices in [18] for prime base $M = 4093$ seem to perform as well here as might be expected. Again the MSE reduction is likely not as high as the variance reduction due to possible bias, although boxplots of estimates still indicate that bias is likely far smaller than variability under IID sampling. For the 2^{14} sample sizes, the boxplots of the estimates under IID, MCG and LFSR sampling are contained in Figures 6.1, 6.2 and 6.3.

A more concrete justification of the minor effects of bias comes from a simulation of this same problem conducted over a much longer time frame, where 1000 posterior mean estimates using chains of length 100,000 each following a heavy burn-in period were used to create a small 95% confidence interval for the means of β_0, β_1 and β_2. These intervals are bounded by the horizontal lines in Figures 6.1, 6.2 and 6.3. The range of MSE reductions taken from the assumption that each value in this interval is the true mean yields a 95% confidence interval for the true reduction in MSE. In the 2^{14} case, the MSE reduction confidence intervals are in Table 6.9.

The same simulation with the same MCG sequences was performed in [42], but the skipping of generator values (as discussed in Section 5.1) was done according to (5.1.1), while the simulations here skipped according to (5.1.2). Variance is reduced up to an additional 60% by the new method, which is algorithmically simpler and computationally comparable.

6.4 A Larger Metropolis-Hastings Algorithm

An example discussed in [5] from quantum physics is used in the attempt to calculate the ground-state energy of a helium atom. This model assumes that the nucleus of the atom is at the origin, and the electrons exist at positions ρ_1 and ρ_2 in $\mathbb{R}^3$. There is a true ground-state wavefunction of the electron positions that is unknown, and so the quality of a trial wavefunction is evaluated. Assuming this trial function is the true function, one estimates the ground-state energy by estimating the mean of a local energy function of the electron positions. The distribution on the electron positions is proportional to the squared modulus of the wavefunction (which can be

Table 6.8: VRFs of posterior mean estimates, probit model

$N \approx 2^{10}$			
Parameter	β_0	β_1	β_2
Liao	20	19	21
MCG	20	18	24
LFSR	14	15	14

$N \approx 2^{12}$			
Parameter	β_0	β_1	β_2
Liao	23	22	24
MCG	24	24	24
LFSR	64	56	76

$N \approx 2^{14}$			
Parameter	β_0	β_1	β_2
Liao	19	20	18
MCG	55	62	47
LFSR	114	108	124

Table 6.9: Confidence Intervals for True MSE Reduction

Parameter	β_0	β_1	β_2
MCG	[44,54]	[45,60]	[41,47]
LFSR	[70,110]	[66,102]	[83,123]

Figure 6.1: Boxplots for, from left to right, IID, MCG and LFSR estimates of $E[\beta_0]$. The horizontal lines bound 95% confidence intervals for the true mean, obtained by much larger simulations.

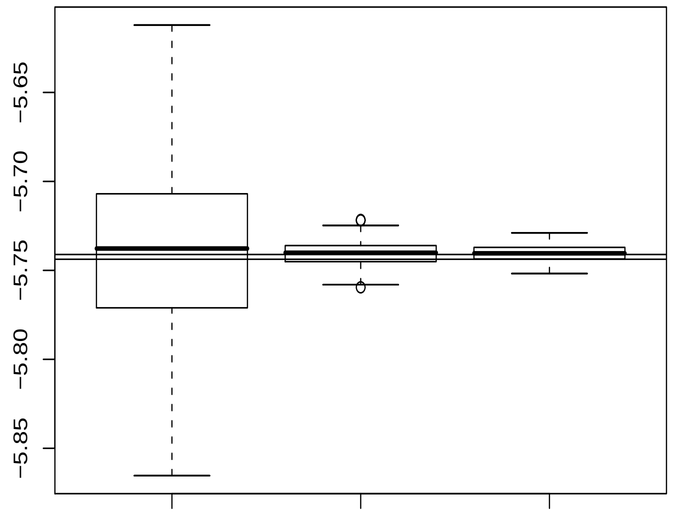


Figure 6.2: Boxplots for, from left to right, IID, MCG and LFSR estimates of $E[\beta_1]$. The horizontal lines bound 95% confidence intervals for the true mean, obtained by much larger simulations.

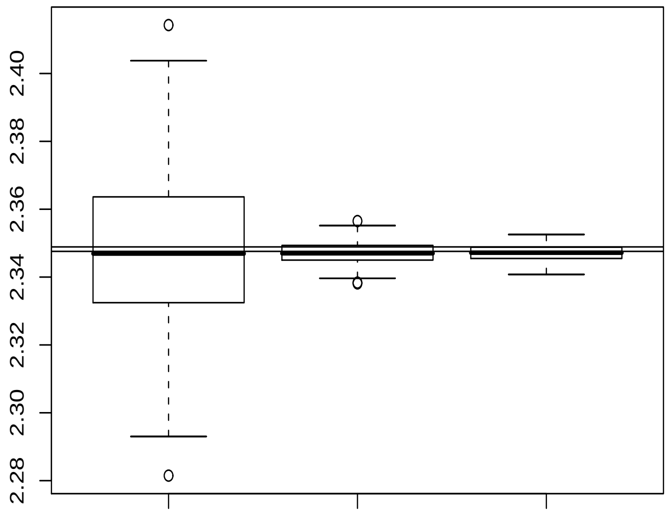
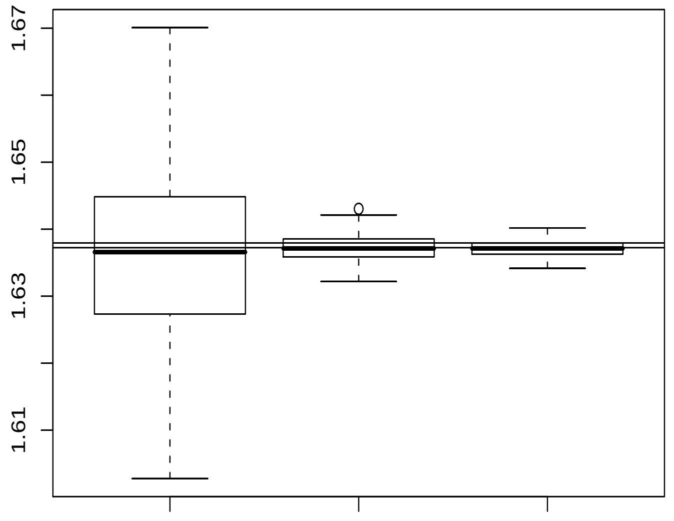


Figure 6.3: Boxplots for, from left to right, IID, MCG and LFSR estimates of $E[\beta_2]$. The horizontal lines bound 95% confidence intervals for the true mean, obtained by much larger simulations.



complex). The example in [5] evaluates the trial wavefunction

$$\Phi(\rho_1, \rho_2) = e^{-2|\rho_1|-2|\rho_2|+(1/2)|\rho_2-\rho_1|} \tag{6.4.1}$$

and, from [26], the local energy function reduces to

$$f(\rho_1, \rho_2) = -\frac{17}{4} - \frac{\rho_1 \cdot (\rho_2 - \rho_1)}{|\rho_1||\rho_2 - \rho_1|} + \frac{\rho_2 \cdot (\rho_2 - \rho_1)}{|\rho_2||\rho_2 - \rho_1|}. \tag{6.4.2}$$

Thus the task is the integration of f over distribution π proportional to Φ^2. This is a 7-dimensional Metropolis-Hastings algorithm if the proposals are uniform moves of ρ_1 and ρ_2 over cubes of side length 2δ centered at their current value. The sample variances of 300 estimates of $E_\pi[f(\rho_1, \rho_2)]$ using IID sampling and MCQMC sampling with an MCG sequence, using sample size 2^{14}, did not differ significantly. (The sample variance of the estimates under IID sampling was 1.3e-04, and the sample variance of estimates under MCG sampling was 1.1e-04.) As is evident from these variances, the improvements of MCQMC in the Gibbs sampler are not seen in this case. There are several explanations for the diminished benefits of MCQMC here. One is the sharper fluctuation of the function f. The acceptance/rejection step in the Metropolis-Hastings sampler also creates a discontinuity in the estimand as a function of the uniform variates used to generate it. A third difficulty is the strong dependence of the sequence on the past, as was seen in the smaller random walk sampler (6.1.1).

The simulations here were performed in JAVA using the Colt Distribution Mersenne Twister and the (16381,665) MCG.

Chapter 7

Conclusion

7.1 Future Directions

A framework that gives sufficient confidence in the acceptability of MCQMC with CUD driving sequences has been outlined here. As it has been written and implemented with the sequence classes discussed, no theoretical derivation of an improved error rate is yet available. The qualitative explanation of the inherent difficulty in obtaining an error rate is that the growth of the discrepancy bounds for the sequence classes is too quick as the dimension of the point set grows. As $s = \log N$ grows too rapidly to make the discrepancy bounds meaningful, the geometric decay of the marginal dependence of values in the chain on the past is still too large to ignore entirely. It is worth noting that the implied infinite dimensional integrals corresponding many Monte Carlo sampling schemes are of a relatively low effective dimension (as defined in [4] in the functional ANOVA sense of [41]) and that the low-dimensional projections of large s-blocks from a small random number generator output sequence tend to look more uniform than is theoretically guaranteed. These observations corroborate the encouraging results seen in the Gibbs sampler examples above, despite the slow asymptotics of the theory.

The improved accuracy of MCQMC estimation is best in scenarios where the function estimated is "nice", the transitions are continuous, and the dependence on the past decays quickly. Efforts to augment the benefits of MCQMC in cases that are not as nice are of importance in the further development the field. Although the greatest benefits are seen in cases where the traditional method is thought to work well already, the added benefits are quite important, as the ability to perform MCMC simulations quickly to a desired accuracy is essential for its widespread use.

7.1.1 Functional ANOVA

This brief outline follows [25]. We have a function f on $[0,1]^d$ with $\int_{[0,1]^d} f \ \delta u = I$. We are interested in looking at the effects on $f(u_1, \ldots, u_d)$ of each set of arguments $u_{a_1}, \ldots, u_{a_k}$ for some subset $a = (a_1, \ldots, a_k) \subset \{1, \ldots, d\}$. The method of functional ANOVA performs the decomposition

$$f(u) = \sum_{a \subset \{1,\ldots,d\}} f_a(u), \tag{7.1.1}$$

where $f_a(u)$ is independent of u_m for all $m \notin a$. This unique decomposition sets $f_\emptyset(u) = I$ constant and then recursively defines, with $u^{(-a)}$ the components of u whose indices are not in a:

$$f_a(u) = \int \Big(f(u) - \sum_{v \subsetneq a} f_v(u) \Big) \ \delta u^{(-a)} = \int f(u) \ \delta u^{(-a)} - \sum_{v \subsetneq a} f_v(x). \tag{7.1.2}$$

Since $\int f_a(u) f_v(u) = 0$ when the sets a and v are not equal, the following equality holds for all $f \in L^2[0,1]^d$:

$$\sigma^2 = \sum_{a \subseteq \{1,\ldots,d\}} \sigma_a^2, \tag{7.1.3}$$

where $\sigma^2 = \int f(u)\ \delta u$ and $\sigma_a^2 = \int f_a(u)\ \delta u$.

For $s < d$ and some predetermined tolerance ϵ, a d-dimensional function f can be thought of as having "effective dimension" s (due to [4]) in a superposition sense if

$$\sum_{|a| \leq s} \sigma_a^2 \geq \sigma^2(1-\epsilon) \tag{7.1.4}$$

and in a truncation sense if

$$\sum_{a \subseteq \{1,\ldots,s\}} \sigma_a^2 \geq \sigma^2(1-\epsilon). \tag{7.1.5}$$

The superposition sense is often used to say that a large-dimensional function is still a suitable candidate for independent QMC sampling.

The Markov transition function (3.1.2) applied recursively becomes

$$X_i = g(u^{(1)}, u^{(2)}, \ldots), \tag{7.1.6}$$

an infinite-dimensional function of the uniform variates used to arrive at the value from the infinite past. Heuristic arguments contained in this paper have said that at time m in the past, the values before then have negligible effect on the current states. This is equivalent to the notion that the infinite-dimensional function is of effective dimension dm in the truncation sense. (The functional ANOVA above has a natural extension to infinite-dimensional functions, although only finite subsets are included in this decomposition.)

A look at the Bayes example and Probit example finds that many of the parameters tend to be independently updated; this independence implies that the higher cardinality terms in the ANOVA will tend to have lower variance as well. This is another explanation for the immense reductions in variance from MCQMC. Analysis of the relationship between effective dimension and the MCQMC advantage is worth future study.

It remains to be seen if an error bound for MCQMC estimates can be obtained from conditions on the functional ANOVA that are generally applicable to MCMC samplers of interest. Again, a simple relation of the decay on the past to the discrepancy bounds on the sequences used is not sufficient, and so this line of inquiry for this application is still in an inchoate stage.

7.1.2 Smoothing the Metropolis Algorithm

While the scenario of estimating parameter means via a quickly mixing continuous Gibbs sampler shows the strongest advantage of using CUD arrays, the fact remains that MCQMC is not clearly outperformed by regular MCMC sampling in the Metropolis-Hastings examples above. One way to reduce the effects of discontinuities in Metropolis-Hastings samplers, initially suggested by Chaudary [5], is by a modified algorithm that runs a chain as normal, but replaces each sample point by a weighted average of the point and a nearby point. As written Chaudary's algorithm had an error that led to inconsistent estimates, and so the algorithm here is offered both as a correction and as an incorporation of this algorithm into the MCQMC framework above, such that a CUD sequence can supply the variates which drive all parts of the algorithm.

Table 7.1: Smoothed Metropolis-Hastings

```
The Smoothed Metropolis-Hastings Algorithm
1 | Begin at $X^{(0)} \in \mathbf{S}$
2 | Given $X^{(i)}$, generate $Y^{(i+1)}$
  |     Transition proposal density $q(X^{(i)}, \cdot)$
3 | Generate $U^{(i+1)} \sim U[0, 1)$
4 | For $A(x, y) = \min\left(\frac{\pi(y)q(y,x)}{\pi(x)q(x,y)}, 1\right)$
  |     **If** $U^{(i+1)} < A(X^{(i)}, Y^{(i+1)})$
  |         Set $X^{(i+1)}$ to $Y^{(i+1)}$
  |     **Else**
  |         Set $X^{(i+1)}$ to $X^{(i)}$
5 | Given $X^{(i+1)}$, generate $Z^{(i+1)}$
  |     Transition density $\tilde{q}(X^{(i)}, \cdot)$
6 | For $\tilde{A}(x, z) = \min\left(\frac{\pi(z)\tilde{q}(z,x)}{\pi(x)\tilde{q}(x,z)}, 1\right)$
  |     Define $g(x, z) = \tilde{A}(x, z)f(z) + (1 - \tilde{A}(x, z))f(x)$
7 | Repeat steps 2-6 $K + N$ times
8 | Return $\frac{1}{N}\sum_{j=K+1}^{K+N} g(X^{(j)}, Z^{(j)})$
```

The goal is the estimate of $E_\pi[f(X)]$, and the algorithm as written in Table 7.1 returns the estimate constructed from a sample, rather than the sample itself.

The steps 5 and 6 look similar to standard Metropolis-Hastings, except the acceptance/rejection decision is replaced by a continuous weighting of the two points. The branch points $Z^{(i)}$ do not influence future steps of the chain. The correction to Chaudary is the inclusion of the latter term in the definition of g; this latter term can be viewed as the continuous analog of repeating a sample point upon rejection of a proposal.

Theorem 7.1.1. *The smoothed Metropolis-Hastings algorithm is consistent under a CUD sampling scheme if the underlying chain is a valid ergodic Metropolis-Hastings chain on a finite state space under IID sampling.*

Proof. We examine the distribution of $(X^{(i)}, Z^{(i)}) \in \mathbf{S} \times \mathbf{S}$. The distribution $\phi(x, y) \triangleq \pi(x)\tilde{q}(x, y)$ satisfies the reversibility condition for all $(x_1, y_1), (x_2, y_2) \in \mathbf{S} \times \mathbf{S}$, and so it is the stationary distribution of the chain $(X^{(i)}, Z^{(i)})$. Under IID sampling, by ergodicity, the algorithm returns a value which converges to $E_\phi[g(X, Z)]$. This is the same as the target value of the estimation procedure:

$$\begin{aligned} E_\phi g &= \sum_x \sum_z \pi(x)\tilde{q}(x,z)[\tilde{A}(x,z)f(z) + (1 - \tilde{A}(x,z))f(x)] \\ &= \sum_x \sum_z \min\left(\tilde{q}(x,z)\pi(x), \tilde{q}(z,x)\pi(z)\right)(f(z) - f(x)) + \sum_x \pi(x)f(x) \sum_z \tilde{q}(x,z) \\ &= \sum_x \pi(x)f(x) \;=\; E_\pi f. \end{aligned} \tag{7.1.7}$$

The algorithm is consistent under IID sampling; CUD consistency follows by completely similar reasoning to that in Theorems 3.3.3 and 4.3.2. □

For cases where the transitions are homogenous ($\tilde{q} = q$), the benefit of this smoothed algorithm is apparent in the small random walk sampler (6.1.1), with an additional 6-fold variance reduction beyond the use of an MCG on regular Metropolis-Hastings, but no significant improvements come in the variational Monte Carlo example of Section 6.4. Beyond the heuristic improvement of smoothing the acceptance/rejection step to make a QMC-theme approach more worthwhile, the benefit of this method may be the application of a different proposal distribution set for generating the branches Z from that used to drive the chain. An application of Slutsky's Theorem allows for the use of antithetics in creating multiple branches with the same marginal distribution but a joint antithetic property, with consistency preserved. Like many of the results here, this result establishes a relatively broad set of conditions under which the algorithm works, such that there is freedom to adjust the algorithm to find further variance reduction techniques. Many extensions and generalizations of the Metropolis-Hastings algorithm are gaining popularity ([24] includes an interesting survey of these). The extension of CUD consistency to these generalizations is likely possible, although those algorithms were not designed to exploit the advantages of QMC sequences.

7.1.3 General implementation

The expression (5.4.1) of the variate matrix used for MCQMC is one that is readily implemented in a general sense. For the MCG and LFSR cases, the computational cost of constructing a variate and applying the appropriate randomization is not any worse than that of the complex procedure that generates the next value in a Mersenne Twister. As sample size and sequence type can be options of the user, the greatest task in creating software that takes in an algorithm and returns an MCQMC estimate is the specific selection of sequences among those of a certain size and type in response to the dimension of the problem. A simple method that chooses a good sequence based on figures of merit or mean square discrepancy in some moderate dimension would not be difficult to implement. It would be more complicated to develop a method for sequence choice based on the algorithm dimension, chain autocorrelation and component interaction together.

Appendix A

Jordan Measurability

A.1 Construction

The condition of regularity in the proposals of the Metropolis-Hastings sampler is necessary for the proof of Theorem 3.3.3; this measurability condition perhaps merits more attention than could be afforded in [37]. Its use in the theorem relates the "volume" of sets to the fraction of points in a sequence contained in that set. This volume is the Jordan measure of the set in question, and to see how we can take the relevant steps in the proof of Theorem 3.3.3, a brief background on the construction of Jordan measure is useful. This construction and the relevant definition of measurability is due to 19th century mathematician Camille Jordan.

We define a semi-open box $[a, b)$ in $[0, 1)^d$ to be the Cartesian product $\prod_{i=1}^{d} [a_i, b_i)$, where $b_i > a_i$ for all i. We assign Jordan measure $V([a, b)) \triangleq \prod_{i=1}^{d} (b_i - a_i)$ to this set. We include the empty set (with measure 0) as a semi-open box. From here we move to expand the measure to increasingly complex sets such that the measure is valid.

Definition A.1.1. A simple set is a set in $[0, 1)^d$ which can be expressed as the finite union of semi-open boxes.

The collection of semi-open boxes which comprise a simple set is not unique. But by chopping the simple set along any $(d-1)$-dimensional plane on which the boundary of the simple set has positive $(d-1)$-dimensional volume, we can divide the simple set into a collection of disjoint semi-open boxes. The measure of the simple set must therefore be the sum of the measures of the semi-open boxes in this disjoint collection.

For an arbitrary set $Y \subset [0, 1)^d$, there is at least one simple set X such that $X \subset Y$, and at least one simple set Z such that $Y \subset Z$. So for the collection $\mathcal{S}$ of simple sets in $[0, 1)^d$, one can define the internal and external volumes of Y:

$$V_{\text{int}}(Y) \triangleq \sup_{X \in \mathcal{S}, X \subset Y} V(X), \tag{A.1.1}$$

$$V_{\text{ext}}(Y) \triangleq \inf_{Z \in \mathcal{S}, Y \subset Z} V(Z). \tag{A.1.2}$$

Definition A.1.2. A set Y is Jordan measurable if $V_{\text{int}}(Y) = V_{\text{ext}}(Y)$. Its Jordan measure $V(Y)$ is equal to this common internal and external volume.

Note that any Jordan measurable set is Lebesgue measurable but some Lebesgue measurable sets are not Jordan measurable. The rational points are not Jordan measurable, for example, as they have internal volume 0 and external volume 1. Clearly when a set is Jordan measurable, the Jordan measure and the Lebesgue measure are identical.

The proof of Theorem 3.3.3 requires that finite unions and tensor products of Jordan measurable sets are also Jordan measurable. These results are shown below.

Lemma A.1.3. *The collection of simple sets in* $[0, 1)^d$ *is closed under finite unions,* *finite intersections and complements.*

Proof. The finite union of simple sets is also a simple set by definition. As the intersection of two semi-open boxes is also a semi-open box, the finite intersection of simple sets is the finite union of semi-open boxes and is also a simple set. As the complement of a semi-open box is a simple set, the complement of a simple set is the finite intersection of simple sets, and so it is also a simple set. □

Theorem A.1.4. *The collection of Jordan measurable sets in* $[0,1)^d$ *is closed under finite unions, finite intersections and complements.*

Proof. For any subset $A \subset [0,1)^d$, $V_{\text{int}}(A) = 1 - V_{\text{ext}}(A^C)$ and $V_{\text{ext}}(A) = 1 - V_{\text{int}}(A^C)$. Thus if A is Jordan measurable, so is A^C.

To verify closure under unions, take arbitrary Jordan measurable sets A and B. For any $\epsilon > 0$, there are simple sets A_1, A_2, B_1 and B_2 such that $A_1 \subseteq A \subseteq A_2$, $B_1 \subseteq B \subseteq B_2$ and $V(A) - \epsilon/4 < V(A_1) \leq V(A_2) < V(A) + \epsilon/4$, $V(B) - \epsilon/4 < V(B_1) \leq V(B_2) < V(B) + \epsilon/4$. $A_2 \setminus A_1$ and $B_2 \setminus B_1$ have internal volume less than $\epsilon/2$; by the above lemma, these are simple sets with measure less than $\epsilon/2$ each.

From the above lemma, the sets $A_1 \cup B_1$ and $A_2 \cup B_2$ are simple sets, and $A_1 \cup B_1 \subseteq A \cup B \subseteq A_2 \cup B_2$. The simple set $(A_2 \cup B_2) \setminus (A_1 \cup B_1) \subseteq (A_2 \setminus A_1) \cup (B_2 \setminus B_1)$, and so

$$V(A_2 \cup B_2) - V(A_1 \cup B_1) \leq V(A_2 \setminus A_1) + V(B_2 \setminus B_1) < \epsilon, \tag{A.1.3}$$

and so the internal and external volumes of $A \cup B$ differ by an amount less than ϵ. As ϵ is arbitrary, the internal and external volumes agree, so $A \cup B$ is Jordan measurable. This result naturally extends to closure under finite unions. Closure under finite unions and complements yields closure under finite intersections. □

Theorem A.1.5. *For Jordan measurable sets $A \in [0,1)^{s_1}$ and $B \in [0,1)^{s_2}$, the Cartesian product $A \times B \in [0,1)^{(s_1+s_2)}$ is Jordan measurable.*

Proof. The Cartesian product of two semi-open boxes is clearly a semi-open box whose measure is the product of the box measures, and so (using the decomposition of a simple set into a finite union of disjoint semi-open boxes) the Cartesian product of two simple sets is also a simple set whose measure is the product of the measures of the simple sets.

It follows by definition that for two arbitrary sets, the internal volume of the Cartesian product is the product of the internal volumes, and the external volume of the Cartesian product is the product of the external volumes. Thus for A, B Jordan measurable, the internal and external volumes of $A \times B$ agree, and $A \times B$ is Jordan measurable. $\square$

A.2 Empirical Measure

For a sequence $x^{(1)}, x^{(2)}, \ldots$, we can define the empirical measure of a set Y on the first n values of the sequence as

$$\hat{V}_n(Y) \triangleq \frac{1}{n} \sum_{i=1}^{n} 1_{\{x^{(i)} \in Y\}}. \tag{A.2.1}$$

Suppose our sequence $x^{(1)}, x^{(2)}, \ldots$ has $D_n^* \to 0$. Then we have the following:

Lemma A.2.1. *For $x^{(1)}, x^{(2)}, \ldots$ with $D_n^* \to 0$ and arbitrary semi-open box $[a, b)$,*

$$\hat{V}_n([a,b)) \to V([a,b)). \tag{A.2.2}$$

The analogous weak law holds for random sequences.

Proof. Since $D_n^* \to 0$, the empirical measure of the missing boundary $[a,b] \setminus [a,b)$ converges to 0, as $\lim_{n\to\infty} \hat{V}_n([\mathbf{0},b)) = \lim_{n\to\infty} \hat{V}_n([\mathbf{0},b])$ So it suffices to prove

$$\lim_{n\to\infty} \hat{V}_n([a,b]) \to \prod_{i=1}^{n}(b_i - a_i). \tag{A.2.3}$$

We define signed local discrepancy $\delta_n^{\pm}([a,b]) = \hat{V}_n([a,b]) - \prod_{i=1}^{n}(b_i - a_i)$ and note that

$$D_n^* = \sup_{b\in[0,1)^d} |\delta_n^{\pm}([\mathbf{0},b])| \tag{A.2.4}$$

For the collection $\mathcal{C}$ of sets $[\mathbf{0},c]$ such that $c_i \in \{a_i, b_i\}$, partition $\mathcal{C}$ into $\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_d$ where $[\mathbf{0},c] \in \mathcal{C}_d$ iff exactly d of the c_i are equal to a_i. The following inclusion-exclusion formula holds:

$$\delta_n^{\pm}([a,b]) = \sum_{j=0}^{d} \sum_{C\in\mathcal{C}_j} (-1)^j \delta_n^{\pm}(C) \tag{A.2.5}$$

And by the triangle inequality,

$$|\delta_n^{\pm}([a,b])| \leq 2^d D_n^* \tag{A.2.6}$$

and so the result follows from $D_n^* \to 0$.

The analogous weak law for random sequences, where $D_n^* \xrightarrow{P} 0$, is verified by the same logic with little modification.

□

As empirical measure converges to Jordan measure of a semi-open box, the same is true for simple sets. The key lemma emerges from this fact.

Lemma A.2.2. *For a Jordan measurable set Y and set $x^{(1)}, x^{(2)}, \ldots$ with $D_n^* \to 0$,*

$$\lim_{n\to\infty} \hat{V}_n(Y) \to V(Y), \tag{A.2.7}$$

and an analogous weak convergence holds for weak discrepancy decay.

Proof. Fix $\epsilon > 0$. There is a simple set X contained in Y such that $V(X) > V(Y) - \epsilon$. Since $\hat{V}_n(X) \to V(X)$ by the above results and $\hat{V}_n(Y) \geq \hat{V}_n(X)$ for all n, $\liminf_{n\to\infty} \hat{V}_n(Y) > V(Y) - \epsilon$. Similarly using a simple set Z containing Y, we get $\limsup_{n\to\infty} \hat{V}_n(Y) < V(Y) + \epsilon$. As ϵ is arbitrary, (A.2.7) holds. For the weak law, note that $P(|\hat{V}_n(Y) - V(Y)| - \epsilon) \to 0$ for arbitrary ϵ from the above results, and so the weak law holds as well. $\square$

Appendix B

Auxiliary Graphs

Figure B.1: Four samples from the marginal posterior of β in the Bayes model, under a Gibbs sampler of size 2^{10} with IID sampling.

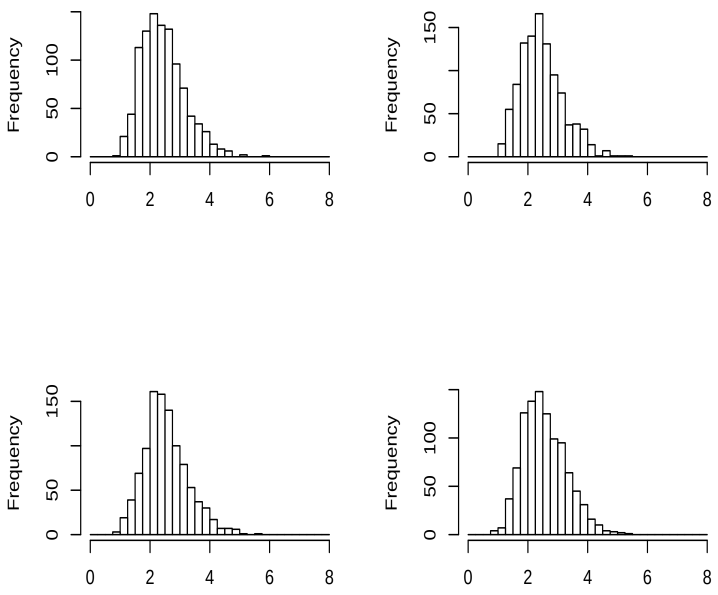


Figure B.2: Four samples from the marginal posterior of β in the Bayes model, under a Gibbs sampler of size $\approx 2^{10}$ with MCG sampling.

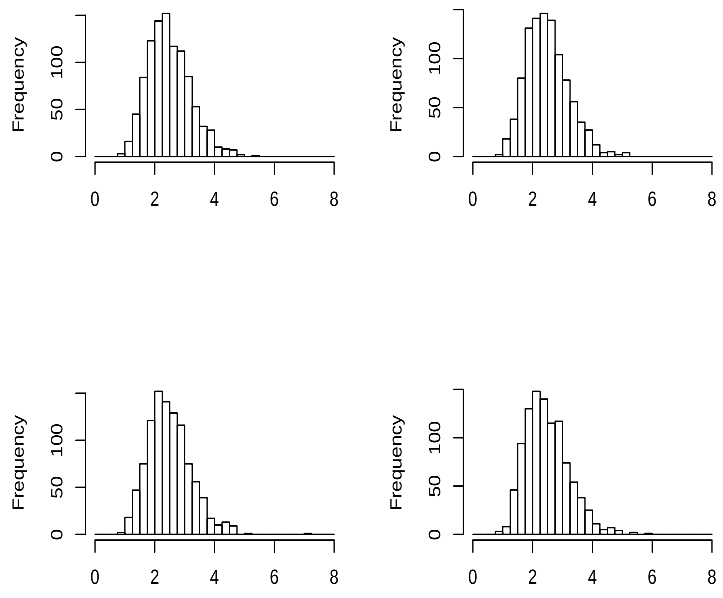
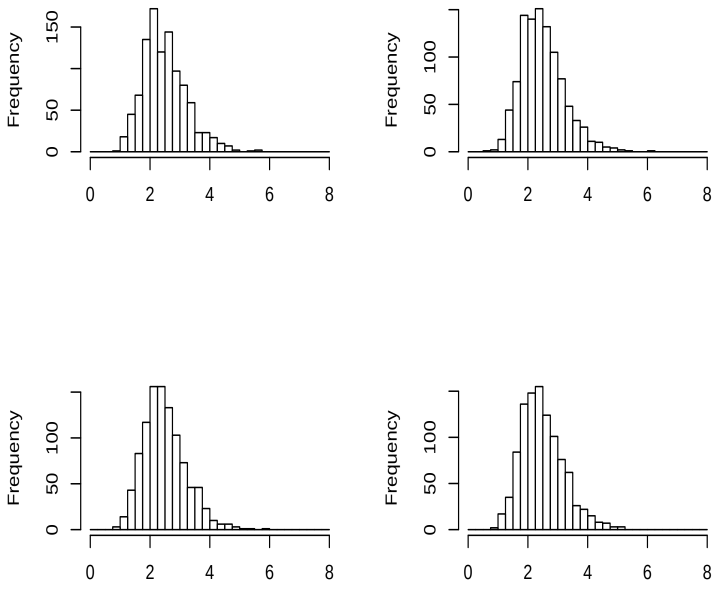


Figure B.3: Four samples from the marginal posterior of β in the Bayes model, under a Gibbs sampler of size $\approx 2^{10}$ with LFSR sampling.



Bibliography

[1] J.H. Ahrens and U. Dieter. Generating gamma variates by a modified rejection technique. *Communications of the ACM*, 25:47–54, 1982.

[2] J. Albert and S. Chib. Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, 88:669–679, 1993.

[3] D.M. Burton. *Elementary Number Theory, 4th ed.*, pages 184–205. William C. Brown Publishers, 1989.

[4] R. Caflisch, W. Morokoff, and A.B. Owen. Valuation of mortgage-backed securities using the quasi-Monte Carlo method. *Journal of Computational Finance*, 1:27–46, 1997.

[5] S. Chaudary. *Acceleration of Monte Carlo methods using low discrepancy sequences*. PhD thesis, UCLA, 2004.

[6] N. Chentsov. Pseudorandom numbers for modelling Markov chains. *Computational Mathematics and Mathematical Physics*, 7:218–232, 1967.

[7] R. Cranley and T. Patterson. Randomization of number theoretic methods for multiple integration. *SIAM Journal of Numerical Analysis*, 13:904–914, 1976.

[8] C.J. de la Vallée Poussin. Recherches analytiques la théorie des nombres premiers. *Ann. Soc. scient. Bruxelles*, 20:183–256, 1896.

[9] L. Devroye. *Non-uniform Random Variate Generation.* Springer, 1986.

[10] D.J. Finney. The estimation from individual records of the relationship between dose and quantal response. *Biometrika*, 34:320–334, 1947.

[11] A. Gelfand and A.F.M. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85:398–409, 1990.

[12] W.K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109, 1970.

[13] S. Heinrich. Efficient algorithms for computing the l_2 discrepancy. *Mathematics of Computation*, 216:1621–1633, 1996.

[14] F. Hickernell. Quadrature error bounds with applications to lattice rules. *SIAM Journal of Numerical Analysis*, 33:1995–2016, 1996.

[15] W. Hoschek. http://dsd.lbl.gov/ hoschek/colt/.

[16] D.E. Knuth. *The Art of Computer Programming*, volume 2. Addison-Wesley, 1998.

[17] N. Korobov. On functions with uniformly distributed fractional parts. *Dokl. Akad. Nauk SSSR*, 62:21–22, 1948.

[18] P. L'Écuyer. Tables of linear congruential generators of different sizes and good lattice structure. *Mathematics of Computation*, 68:249–260, 1999.

[19] P. L'Écuyer, C. Lécot, and B. Tuffin. Randomized quasi-Monte Carlo simulation of Markov chains with an ordered state space. In *Monte Carlo and Quasi-Monte Carlo Methods 2004*. Springer, 2005.

[20] P. L'Écuyer and C. Lemieux. Quasi-Monte Carlo via linear shift-register sequences. In *Proceedings of the 1999 Winter Simulation Conference*, 1999.

[21] C. Lemieux and P. L'Écuyer. Lattice rules for the simulation of ruin problems. In *Proceedings of the 1999 European Simulation Multiconference*, 1999.

[22] M.B. Levin. Discrepancy estimates of completely uniformly distributed and pseudo-random number sequences. *International Mathematics Research Notices*, pages 1231–1251, 1999.

[23] L.G. Liao. Variance reduction in Gibbs sampler using quasi random numbers. *Journal of Computational and Graphical Statistics*, 7:253–266, 1998.

[24] J. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer, 2001.

[25] R. Liu and A.B. Owen. Estimating mean dimensionality of ANOVA decompositions. *Journal of the American Statistical Association*, 101(474):712–721, 2006.

[26] A. MacKinnon. http://www.cmth.ph.ic.ac.uk/angus/lectures/compphys/.

[27] G. Marsaglia. Random numbers fall mainly in the planes. *Proceedings of the National Academy of Sciences*, 61(1):25–28, 1968.

[28] M. Matsumoto and T. Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Trans. Model. Comput. Simul.*, 8(1):3–30, 1998.

[29] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

[30] W. Morokoff and R. Caflisch. A quasi-Monte Carlo approach to particle simulation of the heat equation. *SIAM Journal of Numerical Analysis*, 30:1558–1573, 1993.

[31] H. Niederreiter. Pseudo-random numbers and optimal coefficients. *Advances in Mathematics*, 26:99–181, 1977.

[32] H. Niederreiter. Multidimensional numerical integration using pseudorandom numbers. *Mathematical Programming Study*, 27:17–38, 1986.

[33] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods.* SIAM, 1992.

[34] D. Ormoneit, C. Lemieux, and D.J. Fleet. Lattice particle filters. In *Conference on Uncertainty in Artificial Intelligence*, pages 395–402. Morgan Kaufmann Press, 2001.

[35] M. Ostland and B. Yu. An adaptive quasi-Monte Carlo alternative to metropolis. *Statistics and Computing*, 7:217–228, 1997.

[36] A.B. Owen. Multidimensional variation for quasi-Monte Carlo. In *International Conference on Statistics in honour of Professor Kai-Tai Fang's 65th birthday*, 2005.

[37] A.B. Owen and S. Tribble. A quasi-Monte Carlo Metropolis algorithm. *Proceedings of the National Academy of Sciences*, 102(25):8844–8849, 2005.

[38] W.W. Peterson and E.J. Weldon. *Error-Correcting Codes, 2nd ed.* MIT Press, 1972.

[39] J. Propp and D. Wilson. Exact sampling with coupled Markov chains. *Random Structures and Algorithms*, 9:223–252, 1996.

[40] I. Sloan and S. Joe. *Lattice Methods for Multiple Integration.* Oxford Science Publications, 1994.

[41] I.M. Sobol'. *Multidimensional Quadrature Formulas and Haar Functions.* Nauka, 1969.

[42] S. Tribble and A.B. Owen. Constructions of weakly CUD sequences for MCMC. Technical report, Stanford University, 2005.

[43] T.T. Warnock. Computational investigations of low-discrepancy point sets. In S.K. Zaremba, editor, *Applications of Number Theory to Numerical Analysis*, pages 319–344. Academic Press, 1971.

[44] M.J. Wichura. Algorithm as 241: The percentage points of the normal distribution. *Applied Statistics*, 37:477–484, 1988.